2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Behavior Tree Based Decentralized Multi-agent Coordination for Balanced Servicing of Time Varying Task Queues

Niklas Dahlquist[1], Akshit Saradagi and George Nikolakopoulos

Abstract— **In this article, we present a reactive multi-agent coordination architecture for the management of material flows between production/pickup stages and delivery/drop-off stages, in scenarios such as underground mines and automated factory floors. The pickup and delivery stages are modelled as variable task queues, with no a priori information about the inflow into the production queues. The proposed solution coordinates the movement of a group of mobile agents operating between the two stages in a reactive and scalable manner, so that the material is transported from multiple production queues to multiple delivery queues in a balanced/equalized manner. In such a scenario, centralized planners suffer from low reactivity and poor scaling, as the number of agents and number of queues increases. To overcome this problem, we propose a decentralized approach comprising of two separate auction-based task distribution systems for the production and delivery stages, along with behavior-tree based management of agent autonomy and task bidding. Each auction system tracks the length of production/delivery queues and solves the optimal task assignment, based on the bids submitted by the agents. The agents participate in one of the two auction systems at any given time, based on the status of the behavior tree executing the two-stage tasks. We analytically show that the proposed decentralized auctioning approach along with agent autonomy and bidding managed by behavior trees, offers better scalability and reactiveness compared to the centralized approach. The proposed methodology is experimentally validated in a lab environment, in three illustrative material flow management scenarios, using TurtleBot3 robots as agents.**

I. Introduction

In the recent years, large-scale autonomy in challenging environments such as underground mines [1] and automated factory floors [2] has been inspiring and driving developments in robotics research. Safe and effective multi-agent coordination is central in many applications, where the overarching task can be thought of as material flow management, in the sense that teams of robots/autonomous vehicles are expected to move and manipulate goods/material along the production-processing-shipping pipeline. Challenges introduced by agent and task heterogeneity, multi time-scale operations and need for reactivity have to be tackled in providing solutions to material flow management in large-scale applications.

The need for transporting material between production sites, intermediary storage places and process plants is well motivated in multiple industrial settings, including gravel pits, production industries and mining [3]. The management of flows in a material-handling center involves picking up material from the inflow sites and delivering it at outflow centers. Designing agent autonomy in such scenarios introduces significant complexities and a recent approach for reactively handling unpredictable situations is to utilize behavior trees [4]. This article introduces the novel concept of combining auction-based task allocation and behavior trees for local management in material flow scenarios. The work in [5] proposed a reactive market-based optimal task allocation approach for generic two-stage costs with random arrival times and locations, where different stages are coupled to local behavior trees for task execution. This work builds on [5] and presents a specialized decentralized solution for reactive and balanced material flow management, by modelling the inflow and outflow of material as task queues. A multi-agent coordination solution is proposed for balanced servicing of multiple production and delivery task queues.

A. Related Work

A crucial aspect in efficient deployment of multi-agent systems is the problem of optimal allocation of available tasks among the participating agents. A recent survey in [6] identifies multi-agent deployment in scenarios with environmental variability and high dynamic change, as an open challenge for the theme of coalition forming and task allocation. Additionally, the survey highlights the need of dynamic and flexible task allocation algorithms for coping with such operating conditions and points to the lack of real-world multi-agent experiments. Another key problem involved in large-scale deployment of multi-agent systems is the requirement of large and reliable communication resources. It is often infeasible to assume perfect communication and multi-agent solutions are expected to restrict the bandwidth to a minimum. In the literature, there exists studies, such as [7], where a solution is proposed for multi-agent coordination of second-order agents in scenarios involving unreliable communication, including packet losses and time delays.

In this article, the scenario where no a priori information is available is being investigated. In such a variable and dynamic scenario, the coordinating algorithm is expected to be reactive, as the current optimal assignments may no longer be optimal due to variable inflows. On-Demand Transport is a comparable scenario, where a dynamic environment is a major factor; in this theme the requirement of highly reactive solutions is crucial and it is infeasible to meet the requirements using one centralized unit. This problem has been tackled in [8] where the authors proposes a combination of two parallel coordination processes for solving this problem. However, that work focuses only on optimal task assignment and does not consider other important complexities involved in a material flow management scenario. This article

[1] Niklas Dahlquist is the corresponding author of the article, nikdah@ltu.se. The authors are with Luleå University of Technology (LTU), Sweden.

proposes a coordination architecture taking into account several operational aspects from material flow management, such as lengths of the production and delivery queues, to provide a feasible and optimal solution to the problem of balanced servicing of task queue while considering the global operations in a material management center.

The concept of market-based task allocation has been extensively studied, with classic studies including [9], where an auction based approach was introduced and evaluated. However, there is limited work involving combinatorial auction-based architectures where unpredictable environments or operational aspects such as material queues is considered.

Other existing approaches for tackling similar problems include, all with their own advantages and disadvantages, methods based directly on mixed integer optimization, genetic algorithms, colony optimization and decentralized auction based methods.

B. Contributions

Considering the previously mentioned state-of-the-art, the following are the main contributions of this work. We present the novel modelling of an automated material flow management scenario, where multiple production and delivery centers are serviced by a team of autonomous ground vehicles. The production/delivery centers are modelled as variable queues, whose size is proportional to the material available at the centers, with no a priori information of the inflow. By coupling balanced queue management and optimal multi-agent coordination, we present a solution to the problem of equalized servicing of queues by autonomous robots. We present both centralized and decentralized multi-agent coordination algorithms for material flow management and present an analysis of their scalability and reactivity. We analytically show that the proposed decentralized two-stage auction, with separate auctions for pickup and drop-off tasks, offer better reactivity and scaling.

We introduce the concept of utilizing behavior trees in the task allocation, where the agents are locally managed by behavior trees and have the autonomy to estimate costs for available tasks and to bid in a particular relevant auction depending on its current stage of execution. This reduces the burden on the auction systems to keep track of the task execution status of the agents. The decoupling of task assignment and task execution results in no entity requiring global knowledge of the system, while still keeping the benefits of optimal central coordination. We present a full experimental validation of the proposed multi-agent coordination architecture in a laboratory environment using ground robots and demonstrate optimal multi-agent assignments for equalized servicing of production/delivery queues.

II. Problem Formulation

In this section, we define the various entities involved in a typical material flow management scenario.

The set of pickup locations at the material production centres/queues is denoted by $\mathfrak{S} = \{S_1, \ldots, S_{n_p}\}$, where $n_p \in \mathbb{N}$ is the number of pickup locations. The set of delivery locations/queues is denoted by $\mathfrak{D} = \{D_1, \ldots, D_{n_d}\}$, where $n_d \in \mathbb{N}$ is the number of delivery locations. Every pickup location S_i is associated with a set of currently available pickup tasks $\mathfrak{Q}_i$, where $i \in \{1, \ldots, n_p^i\}$ and $n_p^i \in \mathbb{N}$ represents the number of tasks. The set of all tasks $\mathscr{T} = \mathfrak{Q}_1 \cup \cdots \cup \mathfrak{Q}_{n_p} = \{T_1, \ldots, T_{n_t}\}$, where n_t is the total number of pickup tasks currently available and T_j, $j \in \{1, \ldots, n_t\}$, represents the individual tasks. Similarly, as agents visit drop-off locations, the lengths of the delivery queues increases. At any time instant, the variables q_p^i and q_d^j are used to denote the lengths of the production and delivery queues respectively. A team of mobile agents $\mathfrak{R} = \{R_1, \ldots, R_{n_a}\}$, where $n_a \in \mathbb{N}$ is the number of agents, is available for completing the tasks in $\mathscr{T}$. A task for any agent is said to be completed after an agent first moves to a pickup location and then to any of the drop-off locations. The action of picking up an object is considered irreversible, in the sense that once an agent has visited the pickup location, it cannot be rerouted to another pickup location and must deliver the item before picking up another item. To accurately consider the material flow management scenario, the following assumptions are made.

Assumptions: **a)** The set of tasks $\mathscr{T}$ is assumed to be homogeneous, with no consideration for their time of arrival of waiting time. The queues at the production and delivery sites essentially capture the state of the production/delivery centers. **b)** Abandoning a task before visiting a delivery location is not considered. However an agent can be rerouted to a different pickup locations before a pick up has occurred or rerouted to a different delivery location after pickup. **c)** Every agent $R \in \mathfrak{R}$ has the computational resources, and a map of the environment, to estimate the cost for completing each task given the pickup and drop-off locations. **d)** There exists a reliable two-way communication between every agent and a centralized system. This communication link is used by the agents to send bids/costs to the task allocator and by the task allocator to send task assignments to the agents.

Problem statement: The problem considered in this work can now be stated as follows. We consider two scenarios:

1) The production queues $\mathfrak{S}$ are initialized at $q_p^i(0)$, $i \in \{1, \ldots, n_p\}$ and the delivery queues $\mathfrak{D}$ are initialized at $q_d^j(0)$, $j \in \{1, \ldots, n_d\}$. The problem is to design a multi-agent coordination algorithms that optimally routes the agents such that $q_p^i(k) \to q_p^j(k)$ for all $i, j \in \{1, \ldots, n_p\}$ and $i \neq j$ and $q_d^i(k) \to q_d^j(k)$ for all $i, j \in \{1, \ldots, n_d\}$ and $i \neq j$. This is termed balancing of the queues or equalized servicing of queues by a multi-agent robotic system.
2) Balancing or equalized servicing as defined previously has to be accomplished when the production queues are not statically initialized, but when there is persistent random inflow of material into the queues.

The above two objectives must be accomplished in such a way so that no single unit needs to have full global knowledge of the complete system and while minimizing the overall costs for the agents to complete the tasks.

III. Centralized Approach

In this Section, we first present a centralized optimization problem that captures the problem formulation presented in Section II. We then identify the shortcomings of the centralized approach and present a decentralized auction-based version of the problem involving two separate optimization problems for production and delivery queues respectively.

A. Centralized Task Allocation and Queue Balancing

Here we derive an optimization-based multi-agent coordination algorithm that takes into consideration the assumptions and expectations set earlier. Let $c_{k,i,j}$ be the cost for agent R_k to pick up an item from a production queue $S_i \in \mathfrak{S}$ and drop it at delivery queue $D_j \in \mathfrak{D}$. The path costs of the agent is denoted as $c^p_{k,i,j}$. To incentivize the agents to move to large production queues for picking and small production queues for delivery, the cost associated with the i^{th} production queue size c^p_i and the cost associated with the j^{th} delivery queue size c^d_i are incorporated into the cost $c_{k,i,j}$ in the following manner $C_{k,i,j} = c^p_{k,p,d} - c^p_i + c^d_j$. The scalar $x_{k,i,j}$ is the assignment variable and takes the value 1 if agent R_k is assigned to production queue S_i and delivery queue D_j.

Next, we present a centralized approach to find the optimal assignment $x^\star_{k,i,j} \in \{0,1\}^{n_a} \times \{0,1\}^{n_p} \times \{0,1\}^{n_d}$.

$$x^* = \underset{x_{k,i,j}\in\{0,1\}}{\arg\min} \sum_{k,i,j\in E} C_{k,i,j} \cdot x_{k,i,j} \tag{1}$$

$$\text{s.t.} \sum_{(i,j)\in E_{i,j}} x_{k,i,j} \leq 1, \ \forall\, k \in \{1,\ldots,n_a\} \tag{2}$$

$$\sum_{(k,j)\in E_{k,j}} x_{k,i,j} \leq n_p, \ \forall\, i \in \{1,\ldots,n_p\} \tag{3}$$

$$\sum_{(k,i)\in E_{k,i}} x_{k,i,j} \leq n_d, \ \forall\, j \in \{1,\ldots,n_d\}. \tag{4}$$

where $E_{i,j} = \{1,\ldots,N_p\} \times \{1,\ldots,N_d\}$, $E_{k,j} = \{1,\ldots,N_a\} \times \{1,\ldots,N_p\}$ and $E_{k,i} = \{1,\ldots,N_a\} \times \{1,\ldots,N_p\}$. The constraint (2) enforces that one agent is assigned only one route going from the production center to the delivery center. The constraint (3) enforces that no more than n_p agents are assigned to any of the production queues. The constraint (4) enforces that no more than n_d agents are assigned to any of the delivery queues. Constraints (3) and (4) are necessary to avoid overcrowding of agents of the queues, which could be narrow places around a mining frontier or a pickup location in a logistics/factory setting.

B. Shortcomings of the Centralized Approach

The centralized formulation, although capturing all the aspects of a material flow scenario, suffers from several problems when the scale of the problem grows large. Firstly, as the assignment of both a production site and a delivery site to every agent happens together, the number of decision variables $n_a \times n_p \times n_d$ grows large as the number of different entities in the problem grows large. To incorporate possibility of rerouting in changing ground scenarios, the optimization problem has to be solved repeatedly. When the scales of the distances between production and delivery locations and times to complete tasks grows large, as in a mining scenario, solving a large optimization problem at a certain rate leads to loss of reactivity. Moreover, assignment related to the second stage is not subject to change. For the above reasons, we look to provide a decentralized version of the problem coupled with behavior trees for robot management, that overcomes all the aforementioned drawbacks of the centralized method.

IV. Decentralized Two-stage Auctioning With Behavior Trees

To incorporate reactivity and to improve the scaling of the problem, we here propose a two-stage decentralized counterpart to problem (1), where two separate auction systems, each of size $n_a \times n_p$ and $n_a \times n_d$ are used for the two stages respectively. This separation of the problem has the advantage of reducing the computational complexity of the resulting architecture from $n_a \times n_p \times n_d$ in the original problem (1) to $n_a \times \max\{n_p, n_d\} \times 2$. The optimization problem (1) is separated into two smaller problems: a) first optimizing which agents should move to which pickup location through (5), and b) optimizing to where the agent should deliver the already picked up item (6). This is possible since we are considering the scenario where the picking up of an object is considered irreversible, thus enabling the decoupling of the two parts of the problem. Behavior trees (detailed in Subsection IV-C) are used for agent autonomy, which provides the autonomy to switch between the auction systems, thus enhancing the agent's participation in the overall system.

A. Auction-based Routing for Balancing Production Stages

The first part, considering how to allocate the agents among the available pickup tasks is solved using an auction based approach, where the agents themselves are responsible for estimating the current cost to move to the available pickup locations and a central unit is responsible of collecting the estimation from all the agents and decide on the optimal task allocation. This part of the task allocation can be summarized in three steps, 1) Announcement stage: The available tasks in the set $\mathscr{T}$ are announced to the agents. 2) Bidding stage: costs $c_{k,j}$, the cost of agent R_k performing task T_j, are calculated by the agents for the available tasks. The costs $c_{k,j}$ for all $(k,j) \in E_{k,j}\{1,\ldots,N_a\} \times \{1,\ldots,N_p\}$, are then sent as bids to the centralized task allocator. 3) Task allocation stage: The central unit collects the bids from the agents and the status of the tasks and queues from the pickup stations, optimizes and announces the allocations $x_{k,j}$. The boolean variable $x_{k,j} = 1$ indicates if agent R_k has been assigned to task T_j. These three stages are executed periodically, as long as there are available tasks, either for picking or delivering.

The auction system treats the task of balancing as the equivalent of problems encountered in consensus- and rendezvous literature. If the pickup task is from a queue S_j, the following cost, termed the consensus cost, is associated with the queue $C^j_q = w_p \cdot \frac{1}{N} \sum_{k=1}^{N} q^p_k - q^p_j$ where $w_p > 0$ is a parameter to decide the priority between balancing the queues and minimize the cost $c_{k,j}$ and N is the total number of queues. If the above cost is minimized, every queue length is pushed towards a common average value. If the consensus

cost is consistently minimized in each run of the auction system, it can be proved that the lengths of all the queues achieve consensus or get equalized/balanced with time. $x^\star$ is derived through the following optimization problem.

$$\begin{aligned} x^* = &\arg\min_{x_{k,j}} \sum_{(k,j)\in E} (c_{k,j} + C_q^j)\cdot x_{k,j} \\ \text{s.t.} \quad &\sum_{\{k|R_k\in\mathfrak{R}\}} x_{k,j} \le 1,\ \forall j \in \{1,\dots,n_t\} \\ &\sum_{\{j|T_j\in\mathfrak{T}_t\}} x_{k,j} \le 1,\ \forall k \in \{1,\dots,n_a\} \\ &\sum_{\{j|T_j\in\mathfrak{Q}_i\}} \sum_{\{k|R_k\in\mathfrak{R}\}} x_{k,j} \le m_p,\ \forall i \in \{1,\dots,n_p\} \end{aligned} \tag{5}$$

where m_p denotes the maximum number of agents that can be assigned to one picking station at any time.

As in the central auction system, the three constraints impose that only one agent is assigned to one task, one task is assigned to one agent and no more than m agents are assigned to each queue respectively.

Note that if all terms in the cost to be minimized in (5) are positive, a trivial choice of $x_{k,j} = 0$ for all $(k,j) \in E_{k,j}$ exists, that is consistent with the constraints. Moreover, there is a possibility of the consensus cost C_q^j taking negative values and dominating the cost $c_{k,j}$, thus tampering the minimization problem. To overcome this ill-posedness, we perform two transformations that ensure the invariance of the optimizer $x^\star$: a) Add a large positive value $\bar{C}$ to $c_{k,j} + C_q^j$ to ensure their positive and then b) negate the values $\bar{C} + c_{k,j} + C_q^j$, denoted as profits $\rho_{k,j} = -(\bar{C} + c_{k,j} + C_q^j)$, to construct an equivalent maximization problem, with $x^\star$ remaining the optimizer and preserving the relative distance between the values in $\rho_{k,j}$.

B. Auction-based Routing for Balancing Delivery Locations

The second problem, that of optimally routing agents that have picked up an object to delivery locations, is also solved in a similar manner, where the agents estimate the cost of delivering to the available delivery locations. The estimates are then used by an auction system to optimize and assign where the agents should move. But, in contrast to the previous optimization problem, the constraints are adjusted since the current problem is to allocate agents to the available locations. This results in the following optimization problem

$$\begin{aligned} x^* = &\arg\min_{x_{k,j}} \sum_{(k,j)\in E} (c_{k,j} - C_d^j)\cdot x_{k,j} \\ \text{s.t.} \quad &\sum_{\{k|R_k\in\mathfrak{R}\}} x_{k,j} \le 1,\ \forall j \in \{1,\dots,n_t\} \\ &\sum_{\{j|T_j\in\mathfrak{T}_t\}} x_{k,j} \le 1,\ \forall k \in \{1,\dots,n_a\} \\ &\sum_{\{j|T_j\in\mathfrak{D}_i\}} \sum_{\{k|R_k\in\mathfrak{R}\}} x_{k,j} \le m_d,\ \forall i \in \{1,\dots,n_d\} \end{aligned} \tag{6}$$

where the cost assigned to each delivery queue D_j is $C_d^j = w_d \cdot \frac{1}{N}\sum_{k=1}^{N} q_k^d - q_j^d$ and $w_d > 0$ is a parameter to decide the priority between balancing the queues and minimize the cost

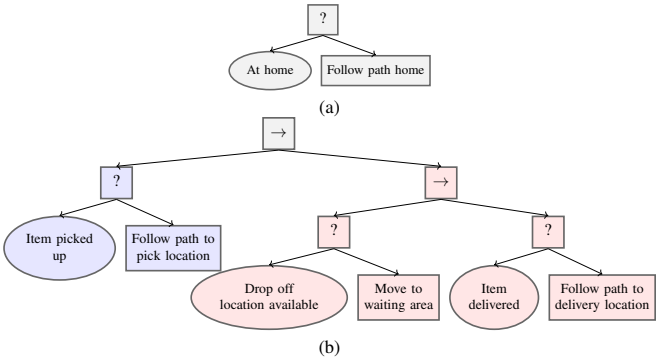


Fig. 1: Behavior trees used. (a) is executed when no task is allocated to the agent and (b) when the pickup and deliver task is allocated to the agent. The condition "Item delivered" is used to switch between the auction systems.

$c_{k,j}$ and which is similar to the production consensus cost (IV-A), but the essential difference being the negative sign associated with C_d^j, which incentives agents to favour small delivery queues. If the cost C_d^j is consistently minimized in each run of the auction system, it can be proved that the lengths of all the queues achieve consensus or are equalized/balanced. As with the optimization problem (5), the two stages of conditioning are adopted to ensure the well-posedness of the problem.

C. Behavior Trees for Bidding and Task Execution

The behavior trees approach [10] provides the agents the ability to switch between behaviors to perform complex actions. The conditions in the behavior tree are used to keep track of the current stage of task execution. This gives the agents the ability to switch participation in the two auction systems. Moreover, this also reduces the burden on the centralized agent to keep track of all the agents and their progress in completing the tasks. The costs $c_{k,j}$ are locally estimated by all agents and submitted to a centralized unit to perform the optimization (5) and (6) and the optimal assignments are then broadcast to all participants.

Considering the pick and deliver task behavior tree in Fig. 1b. The agent will participate in the delivery problem when the red nodes are executed, only after it has already picked up an item i.e. the condition "Item picked up" has succeeded. When the blue nodes in the behavior trees are executed, the agent participates in the auctioning for picking up an object. The execution of the behavior tree in Fig. 1b can be described as follows. First the agent follows its planned path to the pickup location, then it participates in the auctioning for finding a suitable drop-off location. If there are no drop-off locations currently available the agent moves to a designated waiting are and if it gets assigned to a drop-off location it can find a safe path to move to the drop-off location and thereby completing the task. The basic node types used in behavior trees are described in [11] and the behavior trees used in here can be seen in Fig. 1.

V. Local Agent Management

A. Path Planning

Considering tasks where the main part of execution consists of moving between different locations the main cost

to consider is the distance that needs to be traversed in order to complete the task. Additionally, the shortest path can be different depending on the size of the agent, the reliability of the localization, etc. that determines how close to obstacles it is considered safe to traverse. The grid-search algorithm DSP [12], based on D* Lite, is a path planner that incorporates a risk-layer to avoid entering high-risk areas, unless it is absolutely necessary. DSP plans a path P from robot position $\hat{p}$ to the goal position p_g such that the cost $\Gamma = \Gamma_{\text{dist}} + \Gamma_{\text{risk}}$, where Γ_{risk} represents the risk of moving close to obstacles and Γ_{dist} is the total distance to the goal position, is minimized.

In the following scenarios, DSP is used both for generating paths and for estimating distances between positions.

B. Cost Estimation

The estimated cost, $c_{k,j}$, associated with completing the available tasks $\mathscr{T}$ is based on the planned distance, as described earlier, multiplied with a factor k_{bt} derived from the current stage of execution giving $c_{k,j} = \Gamma_{\text{dist}} \cdot k_{\text{bt},j}$ where Γ_{dist} is the distance for agent R_k to move to the pickup location of task T_j and $k_{\text{bt},j} \in \{0,1\}$ to indicate if the item is picked up or not, and thus reduce the cost of the currently allocated task to zero if an agent already picked up an item to guarantee that there is no reallocation of that task.

C. Nonlinear Model Predictive Control

To enable multi-robot systems operating in a shared environment it is a must to have a method of avoiding collisions. In this article we use a nonlinear model predictive controller (NMPC) based on the previous work in [5], applied to ground robots. Using NMPC allows for formulating collision avoidance for a wide range of robot types considering multiple constraints. In this work we utilize the nonlinear kinematic model of a differential-drive robot: $(\dot{p}_x(t), \dot{p}_y(t), \dot{\psi}(t)) = (\cos\psi(t)u_v(t), \sin\psi(t)u_v(t), u_\omega(t))$, since it is appropriate for the platform used in the evaluation.

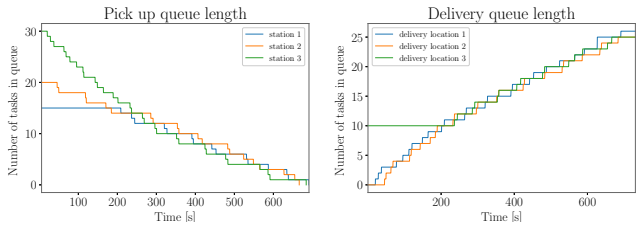
The resulting NMPC is used in the experiments for tracking global reference paths from DSP, while proactively preventing collisions among agents. The NMPC runs with a sampling time of 0.1s and a horizon of $N = 50$, implying a 5s prediction horizon.

VI. Experimental Validation

A. Experimental Setup

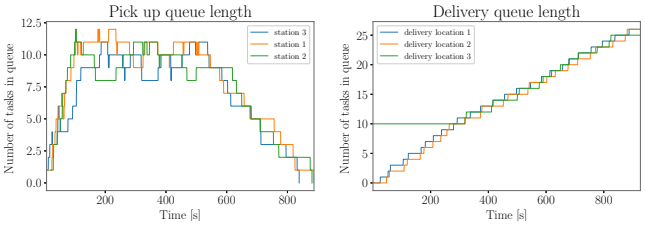
The proposed architecture is evaluated in a realistic large scale laboratory environment inspired from automated factory floors and underground mines, at Luleå University of Technology, Sweden. Four TurtleBot3 are used as mobile agents and a MoCap system is used to get the current pose of all agents. During the scenarios a static 2D map, shared among all agents, is used both for path planning and cost estimation. The scenarios are executed on a computer with an AMD Ryzen 7 PRO 5850U, 32 GB RAM with Ubuntu 20.04. The architecture is implemented within ROS using C++ and the optimization problems are solved using SCIP.

Three scenarios are created to assess the functionality of the proposed architecture. In the first scenario the pickup queues are initialized with different amount of items to show how the proposed architecture handles a basic scenario where all tasks are introduced at the start. In scenario 2 a dynamic inflow of tasks is introduced to show how the architecture handles a typical steady state situation where pickup tasks are stochastically introduced and how the overall system reacts over time. In the final scenario, scenario 3, the inflow of tasks is adjusted to be uneven and shows how balancing occurs even if one pickup location has a greater inflow of items. During all scenarios the weighting parameters for the queue lengths are set $w_p = 1000$ and $w_d = 1000$ and the constraints for how many agents can serve a specific pick and drop-off location simultaneously are set to $m_p = 2$ and $m_d = 2$ respectively. This means that, since the evaluation scenarios contains four agents and three pickup locations/drop-off locations, there always exists a possibility for an agent to either drop off or pick up material. This means that the behavior tree for task execution, illustrated in Fig. 1b, never needs to execute the 'move to waiting area' behavior. A video illustrating the experimental results from the scenarios is included here: https://youtu.be/ogEQ-oP5Tjk



(a) (b)

Fig. 2: Results for scenario 1. (a) showing the queue for the pickup location and (b) showing the evolution of the number of delivered items to the delivery locations.



(a) (b)

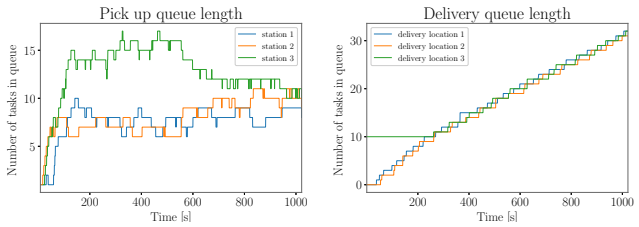
Fig. 3: Results for scenario 2. (a) showing the queue for the pickup location and (b) showing the evolution of the number of delivered items to the delivery locations.

B. Scenario 1

In the first scenario, the pickup locations are initialized with a preset amount of available items. The queue state of the delivery locations is initialized to 0, 0, 10 items respectively. The results, showing the evolution of both queues over time, can be seen in Fig. 2. This scenario, with a static amount of tasks added at the start, shows the desired behavior of the system. Fig. 2a shows that all queues first converge to the same value and then to zero in a balanced way, the same can be seen for the pickup queues in Fig 2b.

C. Scenario 2

In this scenario, the three pickup queues start empty and every second there is a probability of 5% to add a new task



(a) (b)

Fig. 4: Results for scenario 3. (a) showing the queue for the pickup location and (b) showing the evolution of the number of delivered items to the delivery locations.

to each queue, the inflow of new pickup tasks continues until a total of 65 tasks have been added. There is also a global maximum of 30 available tasks, meaning that no more than 30 pickup tasks are available at any given time. The results, showing the evolution of both the pickup queue and the delivery queue, can be seen in Fig. 3. In this scenario the delivery queue converges and becomes very balanced, as seen in Fig. 3b, and even considering a random, but equally distributed, inflow of pickup tasks the queue for the pickup location stays balanced with a maximum difference b between the longest and shortest queue being $b = 4$ during the steady state behavior.

D. Scenario 3

In the third scenario the pickup queues start empty and there is a probability of 5%, 5% and 10% to add a new task to each queue respectively. In the same way as in scenario 2 there is a global maximum of of 30 available tasks, meaning that no more than 30 pickup tasks are available at any given time. The results can be seen in Fig. 4. In the last evaluation scenario since an uneven inflow of new tasks is introduced, Fig. 4a shows that there is a big unbalance in the pickup queues during the transient behavior before all queues converge to approximately the same value after about 950 seconds. Fig. 4b on the other hand, shows that the delivery queue is not affected by the uneven inflow.

VII. Discussion

During all scenarios it is clear that the pickup- and deliver queues are converging in a bounded fashion. The tuning parameters, w_p and w_d, for the two auction systems respectively, is set to a high value to show the optimal, from the queues perspective, behavior. These two parameters in the scalarized optimization problems could be tuned to strike another balance between bounded/balanced queues and optimal traversed distance for the agents. In certain scenarios, considering very large environments, it might be more desirable to allow for some difference between the queue lengths and prioritize the distance traversed by the agents. Since the auction system is run at a much higher frequency compared to the time needed to execute a task, and that the tuning weights are high, the queue lengths are, from a consensus perspective, kept bounded. It is also worth noting that the proposed framework could be generalized to accommodate any type of multi-stage task, not only the specific pick- and deliver tasks, with minimal changes to the auctioning framework. The necessary addition would be the logic, here stated as behavior trees, for executing the task and estimating the cost of execution.

VIII. Conclusions and Future Work

In this article we presented an architecture suitable for solving the problem of material flow management between multiple inflow locations and multiple delivery locations in situations where there is no a priori information on when material is available to be picked up and therefore there is a big need for having a reactive coordination layer. To enable efficient execution we considered multiple aspects, including the queue length at both the pickup and the delivery locations and the distance traveled, and these aspects were demonstrated in multiple illustrative scenarios. The improved scalability and reactivity of the proposed decentralized system, compared to a centralized, is motivated by analyzing the computational complexity of the optimization problems.

Future work includes analysis of the convergence rate and how bounded the queues will stay, from a discrete consensus theory perspective, and extension to include a longer horizon to enable a more efficient long term scheduling.

References

[1] S. Ge, F.-Y. Wang, J. Yang, Z. Ding, X. Wang, Y. Li, S. Teng, Z. Liu, Y. Ai, and L. Chen, "Making standards for smart mining operations: Intelligent vehicles for autonomous mining transportation," *IEEE Transactions on Intelligent Vehicles*, vol. 7, no. 3, pp. 413–416, 2022.

[2] T. T. Mezgebe, H. B. E. Haouzi, G. Demesure, R. Pannequin, and A. Thomas, "Multi-agent systems negotiation to deal with dynamic scheduling in disturbed industrial context," *Journal of Intelligent Manufacturing*, vol. 31, pp. 1367–1382, dec 2019.

[3] C. Lieberoth-Leden, J. Fischer, J. Fottner, and B. Vogel–Heuser, "Control architecture and transport coordination for autonomous logistics modules in flexible automated material flow systems," in *2018 IEEE 14th International Conference on Automation Science and Engineering (CASE)*, pp. 736–743, 2018.

[4] M. Iovino, J. Förster, P. Falco, J. J. Chung, R. Siegwart, and C. Smith, "On the programming effort required to generate behavior trees and finite state machines for robotic applications," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5807–5813, 2023.

[5] N. Dahlquist, B. Lindqvist, A. Saradagi, and G. Nikolakopoulos, "Reactive multi-agent coordination using auction-based task allocation and behavior trees," in *2023 IEEE Conference on Control Technology and Applications (CCTA)*, pp. 829–834, 2023.

[6] Y. Rizk, M. Awad, and E. W. Tunstel, "Cooperative heterogeneous multi-robot systems," *ACM Computing Surveys*, vol. 52, pp. 1–31, apr 2019.

[7] A. Mannucci, L. Pallottino, and F. Pecora, "Provably safe multi-robot coordination with unreliable communication," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3232–3239, 2019.

[8] A. Daoud, F. Balbo, P. Gianessi, G. Picard, M. Lujak, I. Dusparic, F. Klügl, and G. Vizzari, "Ornina: A decentralized, auction-based multi-agent coordination in odt systems," *AI Commun.*, vol. 34, p. 37–53, jan 2021.

[9] M. B. Dias, *TraderBots: A New Paradigm for Robust and Efficient Multirobot Coordination in Dynamic Environments*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, January 2004.

[10] P. Ögren and C. I. Sprague, "Behavior trees in robot control systems," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 5, no. 1, pp. 81–107, 2022.

[11] M. Colledanchise and P. Ögren, "Behavior trees in robotics and ai," Jul 2018.

[12] S. Karlsson, A. Koval, C. Kanellakis, A.-a. Agha-mohammadi, and G. Nikolakopoulos, "D^*_+: A generic platform-agnostic and risk-aware path planing framework with an expandable grid," *arXiv preprint arXiv:2112.05563*, 2021.