A Mixed-Integer Linear Programming Formulation for Human Multi-Robot Task Allocation

Martina Lippi, Alessandro Marino

Abstract— **In this work, we address a task allocation problem for human multi-robot settings. Given a set of tasks to perform, we formulate a general Mixed-Integer Linear Programming (MILP) problem aiming at minimizing the overall execution time while optimizing the quality of the executed tasks as well as human and robotic workload. Different skills of the agents, both human and robotic, are taken into account and human operators are enabled to either directly execute tasks or play supervisory roles; moreover, multiple manipulators can tightly collaborate if required to carry out a task. Finally, as realistic in human contexts, human parameters are assumed to vary over time, e.g., due to increasing human level of fatigue. Therefore, online monitoring is required and re-allocation is performed if needed. Simulations in a realistic scenario with two manipulators and a human operator performing an assembly task validate the effectiveness of the approach.**

I. Introduction

In the last decade, Human–Robot collaboration (HRC) has received increasing interest of the research community, as well as of the the industrial world. The benefits of having humans and robots cooperating in the execution of complex tasks lay in the possibility of achieving flexible and highly reconfigurable production systems and of easily changing task execution to accommodate different product families while meeting high quality standards [1].

Antithetical yet complementary capabilities characterize human and robot entities: reasoning and manipulation skills for the human component, strength and endurance capabilities for the robotic one. The different abilities make humans and robots better suited to different sets of tasks, e.g., the former are more apt to handle objects of small size or with particular shapes and/or materials, while the latter are more appropriate for manipulating heavy objects with regular shapes or for performing repetitive tasks [2].

Indeed, a core component of HRC is how to optimally distribute tasks between robotic and human counterparts; despite the potentialities of human-robot collaborative scenarios, task allocation in these setups is far from trivial since there are a variety of tasks that can be performed by both and for which it must be decided to whom to assign execution while maximizing certain criteria. Furthermore, several tasks also exist that benefit from the simultaneous collaboration of humans and robots, e.g., robots holding large parts for assembly and humans performing operations on these parts [3]. Concerning the optimization criteria, different factors should be taken into account such as overall completion time, human fatigue, and specific expertise. In addition, human parameters are generally challenging to be quantified and can vary over time.

In this work, we propose a novel flexible task allocation formulation in which multiple manipulators and human operators are involved. In the proposed framework, each agent, either human or robotic, can be differently skilled at a given task. Multiple manipulators can tightly collaborate if required to carry out a task, while human operators can either directly execute tasks or play a *supervisory* role, allowing to guarantee a minimum task execution accuracy. Indeed, the supervisory role can be useful, for example, in case where the robotic component is not fully capable at a task and the person monitors its activity to prevent errors from occurring. Moreover, precedence as well as spatial constraints are taken into account by the devised framework with the latter taking into consideration that, due to workspace limitations, some tasks might no be carried on simultaneously.

The devised solution aims at minimizing the overall required time while optimizing the quality of the executed tasks as well as human and robotic workload. Moreover, as realistic in human contexts, we consider that human parameters can vary over time. Therefore, the proposed framework also comprises online monitoring and possible re-allocation whenever required to meet future constraints as well as to preserve the solution optimality to a certain extent.

II. Related work

Task allocation and scheduling for multi-robot as well for human-robot teams are important problems with applications to manufacturing, warehouse automation, and pickup-and-delivery [4]. Human-Robot and Multi-Robot Task Allocation (HRTA and MRTA, respectively) share many common points and methodologies but also have inherent differences due to peculiarities of human agents. Indeed, the problem of MRTA is formally addressed in [5] by combining operations research and combinatorial optimization, and how this formalism can be used for the synthesis of new approaches is shown. Since then, several approaches have been proposed addressing many aspects and problems concerning both MRTA and HRTA as, for example, distributed implementation; this is the case in [6] where authors focus on a distributed multi-robot setting with tasks having time window and ordering constraints.

The work in [7] formulates a MILP problem in which temporal and spatial constraints are taken into account in the

[1]Roma Tre University, Italy, martina.lippi@uniroma3.it
[2]University of Cassino and Southern Lazio, Italy al.marino@unicas.it

This work was supported by Dipartimento di Eccellenza granted to DIEI Department, University of Cassino and Southern Lazio and by H2020-ICT project CANOPIES (Grant Agreement N. 101016906).
10.1109/RO-MAN50785.2021.9515362

framework of HRTA and re-planning is envisaged as well. In detail, the solution is a multi-agent task sequencer that is inspired by real-time processor scheduling techniques and is adapted to leverage hierarchical problem structure. The same authors in [8] conduct experiments to establish the effect on the task allocation solution of the team composition and of the level of influence authority. In this perspective, the allocation could be completely made by human, semi-autonomous, with the human deciding which task to allocate to himself/herself, or autonomous, i.e. the robot allocates all tasks. The authors found that an autonomous robot can outperform a human worker in the execution of part or all of the process of task allocation, and that people preferred to cede their control authority to the robot.

A closed-loop planning framework which adapts to varying human parameters and task sets is presented in [9]. Highly coupled tasks between humans and robots are considered and human parameters like task execution time, workload and performance are adopted to update in real-time parameters of human model with the purpose or replanning whenever required.

The work in [10] considers a scenario requiring processing multiple parts, each consisting of a series of tasks constrained by precedence relationships between them. Tasks are executed by human workers and robots which are resources required for processing tasks, and the problem is identified as a simplification of the Multi-Mode-Resource-Constrained-Project-Scheduling problem. Differently from previous works, additional constraints, such as minimum distance requirements and waiting times for agents, are added in order to enhance the fluency of the mixed human-robot team. Finally, heuristics are proposed by the authors to solve the problem and mitigate the computational burden required by the proposed model.

The problem of task allocation in a framework where single human is involved is presented in [11]. The assembly scheduling problem is adaptive in the sense that it is modulated according to human capabilities: the latter is constantly monitored and re-allocation is carried out if its variation is above a pre-defined threshold; a genetic algorithm is then used to solve the optimization problem.

In [12], the task distribution between humans and robots is addressed by considering a quantitative classification of task complexity. The latter considers physical features of components to be assembled and tasks are differentiated in high-complexity tasks due to the inherent complexity of handling and mounting components and in low-complexity tasks. It is shown that such a classification lowers deployment and changeover times.

Human factors are considered also in [13], where task allocation in human-robot assembly applications is addressed by considering human capabilities and workload as well as ergonomics aspects based on the human body posture.

Similarly to previous work, in [14] factors to be considered in allocating tasks among a set of agents made of humans and robots in industrial manufacturing scenarios are considered. Tasks are decomposed in atomic actions and are offline allocated based on indexes like complexity, agent dexterity, required effort, and experiments are carried out to evaluate system's performance.

Finally, work in [15] focuses on large-scale instances in which algorithmic solutions can be not suitable and proposes a graph attention network-based scheduler which leverages imitation learning to learn from expert demonstrations on small scale problems.

With respect to previous works, the following novelties are introduced:

- the offline task allocation takes into account several features and constraints simultaneously, like time and spatial constraints, and it allows to assign more than one agent to the same task; moreover, the concept of human supervision is introduced in order to increase the quality of task execution;
- a closed-loop re-scheduling framework is devised that, based on real-time monitoring of parameters performance, is able to adapt to dynamic and stochastic agent parameters.

III. Problem setting

TABLE I: Main notations introduced in the paper.

Variable	Meaning
m	Number of tasks
n_a	Number of agents
n_r	Number of robot agents
n_h	Number of human agents
$M \in \mathbb{R}$	Arbitrary large positive constant
$\mathcal{T}$	Set of tasks $\mathcal{T} = \{\tau_1, \tau_2, \cdots, \tau_m\}$
$\mathcal{A}$	Set of agents $\mathcal{A} = \{a_1, a_2, \cdots, a_{n_a}\}$ comprising humans and robots
$\mathcal{A}_r$	Set of robot agents $\mathcal{A}_r = \{a_{r,1}, a_{r,2}, \cdots, a_{r,n_r}\}$
$\mathcal{A}_h$	Set of human agents $\mathcal{A}_h = \{a_{h,1}, a_{h,2}, \cdots, a_{h,n_h}\}$
$g(\tau_i)$	Task category to which τ_i is associated
$X_{i,j} \in \{0,1\}$	Binary decision variable for the assignment of task τ_i to agent a_j
$S_{i,j} \in \{0,1\}$	Binary decision variable for the assignment of human supervision of task $\tau_i \in \mathcal{T}$ to human agent $a_{h,j} \in \mathcal{A}_h$
$P_{i,k} \in \{0,1\}$	Binary variable denoting a precedence constraint, i.e. if $P_{i,k} = 1$ task τ_i must end before task τ_k, otherwise no order is assigned
$D_{i,k} \in \{0,1\}$	Binary variable denoting if there is a spatial constraint, i.e. if $D_{i,k} = 1$ task τ_i and τ_k require to occupy the same location.
$C_i \in \{0,1\}$	Binary variable which is 1 if task τ_i requires two agents to be executed (collaborative task), 0 otherwise
$\underline{t}_i, \overline{t}_i$	Start and end times of task τ_i
$\Delta_{i,j}$	Execution time required by agent a_j to perform task τ_i
T_M	Maximum overall execution time
$q_{i,j}$	Execution quality index for executing task τ_i by agent a_j
$q^s_{i,j}$	Supervision quality provided by human agent $a_j \in \mathcal{A}_h$ for task $\tau_i \in \mathcal{T}$
$w_{i,j}$	Workload of agent a_j in performing task τ_i
$w^s_{i,j}$	Workload of human agent $a_{h,j}$ in supervising task τ_i
$V_{i,k,j} \in \{0,\ 1\}$	Auxiliary variable for allocating at most one task to each agent at each time
$Z_{i,k} \in \{0,\ 1\}$	Auxiliary variable for spatial constraints
δ	Re-allocation parameter

In this section, we describe the problem setting and introduce the main variables of the paper, which are summarized in Table I. We consider a human-multi-robot collaborative scenario in which there are several tasks to accomplish, with different requirements and constraints, and there are several agents to which these tasks might be allocated. In detail, agents are divided into human and robot agents. We denote the set of agents by $\mathcal{A} = \{a_1, a_2, \cdots, a_{n_a}\} = \mathcal{A}_h \cup \mathcal{A}_r$, with $\mathcal{A}_h = \{a_{h,1}, a_{h,2}, \cdots, a_{h,n_h}\}$ the set of human agents with cardinality n_h, $\mathcal{A}_r = \{a_{r,1}, a_{r,2}, \cdots, a_{r,n_r}\}$ the set of robotic agents with cardinality n_r and $n_a = n_h + n_r$. A set $\mathcal{T} = \{\tau_1, \tau_2, \cdots, \tau_m\}$ of m tasks with cardinality m is defined in which:

- precedence constraints are defined, i.e., a binary variable $P_{i,k}$ is introduced for each couple of tasks τ_i, τ_k and is 1 if task τ_i must end before task τ_k, 0 otherwise. This allows to express sequentiality of tasks;
- estimated execution times by each agent are defined, i.e., $\Delta_{i,j}$ is the execution time required by agent a_j to perform task τ_i;
- estimated workloads for each agent are defined, i.e., we denote the workload for agent a_j to execute task τ_i by $w_{i,j}$;
- spatial locations are assigned, i.e., for each task τ_i the location $\boldsymbol{p}_i \in \mathbb{R}^3$ where it should be carried out is defined. Based on this, we introduce the binary variable $D_{i,k}$ which is equal to 1 if the execution locations of task τ_i and τ_k are too close to be executed simultaneously, i.e., $D_{i,k} = 1$ if $\|\boldsymbol{p}_i - \boldsymbol{p}_k\| < \varepsilon$ with ε a positive threshold, 0 otherwise. These variables are symmetric by construction, i.e., $D_{i,k} = v$ implies $D_{k,i} = v$ with $v \in \{0, 1\}$, and enable to guarantee that no tasks are simultaneously carried out in the same location. Note that any criterion to define volume occupation for executing each task can be adopted to define $D_{i,k}$.

Based on the estimated execution times, it is possible to define the maximum time T_M to execute all the tasks. We assume that tasks are partitioned into groups or clusters according to common features, e.g., two pick-and-place operations of the same kind of object reasonably belong to the same cluster. We denote the group that is associated with task τ_i by $g(\tau_i)$. Note that this does not undermine the generality of the approach as m different groups can be defined, i.e., each task can be associated with a different group. In addition, we consider the following features for the tasks:

F.1 a task must be executed with a certain level of accuracy and may be supervised by a human operator to guarantee its correctness;

F.2 a task may be required to be carried out in a collaborative fashion, e.g., transporting heavy/large objects;

F.3 a task can be suitable for humans only, e.g., manipulating objects with difficult geometry or which are highly deformable;

F.4 a task can be suitable for robots only, e.g., carrying very heavy objects or manipulating objects that can be dangerous for human operators.

To formalize F.1, we introduce an *execution quality index* $q_{i,j} \in [0, 1]$, $\forall \tau_i \in \mathcal{T}$, $a_j \in \mathcal{A}$ which, for each pair (agent a_j, task τ_i), assesses the quality of the execution of task τ_i by agent a_j. We assume that tasks in the same group are associated with same execution quality index for each agent a_j, i.e., $q_{i,j} = q_{k,j}$ if $g(\tau_i) = g(\tau_k)$, $\forall \tau_i, \tau_k \in \mathcal{T}$. A minimum quality $\underline{q}$ is required to guarantee a minimum task execution accuracy. If an agent does not meet the minimum quality requirement for a certain task, a human operator can be assigned for its supervision. In this way, the human can monitor if the task execution is correct or not, and possibly intervene if necessary. We thus introduce a *supervision quality* $q^s_{i,j} \in [0, 1]$, $\forall \tau_i \in \mathcal{T}$, $a_j \in \mathcal{A}$ which quantifies for each human agent a_j the achievable quality for task τ_i under his/her supervision. Human intervention is allowed during supervision, and supervision quality is equal to 0 for robotic agents, i.e., only human operators can play supervision role. As for the execution quality index, we consider that tasks in the same group are associated with same supervision quality index for each human agent $a_{h,j}$, i.e., $q^s_{i,j} = q^s_{k,j}$ if $g(\tau_i) = g(\tau_k)$, $\forall \tau_i, \tau_k \in \mathcal{T}$. We consider that quality indices are additive, i.e., the overall quality of a task is given by the sum of the qualities of the agents that execute and supervise it. Note that this is only a possible choice made in the paper but the proposed framework could be easily extended to tackle different models for the quality, e.g., considering minimum or maximum quality of the involved agents as overall quality of the task. Note that the quality indices can generally vary over time and we update them at the end of the execution of each task as detailed in Section V.

Concerning the remaining features, collaborative tasks in F.2 are denoted by the binary variable C_i, $\forall \tau_i \in \mathcal{T}$ which is equal to 1 if task τ_i requires two agents to be accomplished, 0 otherwise. With regard to features F.3 and F.4, let $\mathcal{T}_j$ be the set of tasks that can be carried out by agent a_j (either robotic or human); these features can be easily expressed by setting $\Delta_{i,j} = M$, with M an arbitrary high constant, or $w_{i,j} = M$ for tasks $\tau_i \in \mathcal{T} \setminus \mathcal{T}_j$.

IV. Human multi-robot task allocation problem

Let $X_{i,j}$ be the binary decision variable for the assignment of task τ_i to agent a_j, i.e., $X_{i,j} = 1$ if agent a_j has to execute task τ_i, $X_{i,j} = 0$ otherwise, let $S_{i,j}$ be the binary decision variable for the supervision of task τ_i by the human agent a_j, i.e., $S_{i,j} = 1$ if human a_j has to supervise the execution of task τ_i, $S_{i,j} = 0$ otherwise, and let $\underline{t}_i, \overline{t}_i$ be the starting and end time of each task $\tau_i \in \mathcal{T}$. Our objective is to define the tasks assigned to each agent and the respective starting and end times as well as the supervision assignments by minimizing a cost function while complying with system constraints. In detail, the following MILP problem is formulated:

$$\min_{X_{i,j},S_{i,j},\underline{t}_i,\overline{t}_i} \underbrace{\max_{\tau_i\in\mathcal{T}} \frac{\overline{t}_i}{T_M}}_{\text{makespan}} - \sum_{\tau_i\in\mathcal{T}}\sum_{a_j\in\mathcal{A}} \big(\underbrace{q_{i,j}X_{i,j} + q^s_{i,j}S_{i,j}}_{\text{overall quality}} - \underbrace{w_{i,j}X_{i,j} - w^s_{i,j}S_{i,j}}_{\text{overall workload}}\big) \tag{1a}$$

s.t.

$$\sum_{a_j\in A} X_{i,j} = 1 + C_i, \qquad \forall \tau_i \in \mathcal{T} \tag{1b}$$

$$S_{i,j} + X_{i,j} \leq 1, \qquad \forall \tau_i \in \mathcal{T}, a_{h,j} \in \mathcal{A}_h \tag{1c}$$

$$\sum_{\tau_i\in\mathcal{T}}\sum_{a_{r,j}\in\mathcal{A}_r} S_{i,j} = 0 \tag{1d}$$

$$\sum_{a_j\in\mathcal{A}} \big(q_{i,j}X_{i,j} + q^s_{i,j}S_{i,j}\big) \geq \underline{q}, \qquad \forall \tau_i \in \mathcal{T} \tag{1e}$$

$$\overline{t}_i - \underline{t}_i \geq X_{i,j}\mathit{\Delta}_{i,j} \qquad \forall \tau_i \in \mathcal{T}, a_j \in \mathcal{A} \tag{1f}$$

$$\underline{t}_k - P_{i,k}\overline{t}_i \geq 0 \qquad \forall \tau_i, \tau_k \in \mathcal{T} \tag{1g}$$

$$\underline{t}_k - \overline{t}_i \geq -M(2 - X_{i,j} - X_{k,j} - S_{i,j} - S_{k,j}) - M(1 - V_{i,k,j}) \qquad \forall \tau_i, \tau_k \in \mathcal{T}, a_j \in \mathcal{A} \tag{1h}$$

$$\underline{t}_i - \overline{t}_k \geq -M(2 - X_{i,j} - X_{k,j} - S_{i,j} - S_{k,j}) - M\,V_{i,k,j} \qquad \forall \tau_i, \tau_k \in \mathcal{T}, a_j \in \mathcal{A} \tag{1i}$$

$$\underline{t}_k - \overline{t}_i \geq -M(1 - D_{i,k}) - M(1 - Z_{i,k}) \qquad \forall \tau_i, \tau_k \in \mathcal{T} \tag{1j}$$

$$\underline{t}_i - \overline{t}_k \geq -M(1 - D_{i,k}) - M\,Z_{i,k} \qquad \forall \tau_i, \tau_k \in \mathcal{T} \tag{1k}$$

According to the objective function in (1a), we aim to minimize the system makespan, i.e., the overall execution time, normalized with respect to the maximum execution time T_M while maximizing the overall process quality, given by the cumulative quality of each task, and minimizing the overall agents workload. The quality values at the allocation time are considered, and supervision quality and workload by human operators are also taken into account. The following constraints are considered:

- equation (1b) ensures that the exact number of agents is allocated to each task, i.e., for each task $\tau_i \in \mathcal{T}$ a total of one or two agents must execute it depending on whether the task is collaborative (one agent case) or not (two agents case);
- equation (1c) imposes the mutual exclusivity for supervision and execution by human operators, i.e., for each task $\tau_i \in \mathcal{T}$ a human agent cannot simultaneously execute ($X_{i,j} = 1$) and supervise it ($S_{i,j} = 1$);
- equation (1d) imposes that no supervision can be performed by robotic agents $a_{r,j} \in \mathcal{A}_r$;
- equation (1e) implies that a minimum quality $\underline{q}$ is guaranteed for each task $\tau_i \in \mathcal{T}$. The overall quality takes into account both the execution quality indices $q_{i,j}$ of the assigned agents with $X_{i,j} = 1$ and the supervision quality indices $q^s_{i,j}$ for the assigned human supervisors with $S_{i,j} = 1$;
- equation (1f) defines the minimum duration of each task, i.e., if task τ_i is assigned to agent a_j ($X_{i,j} = 1$), then the allocation time $\overline{t}_i - \underline{t}_i$ must be at least equal to $\mathit{\Delta}_{i,j}$;
- equation (1g) allows to guarantee the required tasks sequentiality, i.e., for each pair of tasks $\tau_i, \tau_k \in \mathcal{T}$ if $P_{i,k} = 1$ then τ_k has to start after τ_i ends, while if $P_{i,k} = 0$ no precedence is imposed;
- equations (1h)-(1i) ensure that each agent $a_j \in \mathcal{A}$ cannot simultaneously execute or supervise more than one task. To this aim, the auxiliary decision binary variable $V_{i,k,j}$ $\forall a_j \in \mathcal{A}, \tau_i, \tau_k \in \mathcal{T}$ is introduced. Let us consider the case in which both tasks τ_i, τ_k are assigned to agent a_j, i.e., $X_{i,j} = X_{k,j} = 1$. By virtue of (1c) and (1d), it follows that $S_{i,j} = S_{k,j} = 0$. Equations (1h)-(1i) thus lead to
$$\underline{t}_k - \overline{t}_i \geq -M(1 - V_{i,k,j})$$
$$\underline{t}_i - \overline{t}_k \geq -M\,V_{i,k,j}$$
implying that either $\underline{t}_k \geq \overline{t}_i$ (if $V_{i,k,j} = 1$) or $\underline{t}_i \geq \overline{t}_k$ (if $V_{i,k,j} = 0$), but no simultaneous execution can occur. Similar reasoning applies for the case in which $S_{i,j} = S_{k,j} = 1$. Finally, when the tasks are not assigned to the same robots, no constraints are imposed by (1h)-(1i);
- equations (1j)-(1k) allow to specify that tasks that occupy the same spatial location are not executed simultaneously. Similarly to the constraints in (1h)-(1i), we introduce an auxiliary decision binary variable $Z_{i,k}$ $\forall \tau_i, \tau_k \in \mathcal{T}$. Based on this, if a spatial limitation exists between tasks τ_i, τ_k, then equations (1j)-(1k) lead to specify that either τ_k starts after τ_i, i.e., $\underline{t}_k \geq \overline{t}_i$ when $Z_{i,k} = 1$, or the opposite holds true, i.e., $\underline{t}_i \geq \overline{t}_k$ when $Z_{i,k} = 0$.

Note that the proposed formulation is particularly versatile and can be easily adapted to different collaborative production processes and tasks involving an arbitrary number of humans and robots.

V. Online Re-allocation

Due to uncertainty in realistic scenarios, online re-allocation may be necessary to meet future plan constraints, as well as to preserve the optimality of the plans to a certain extent. For this reason, we consider that at the end of the execution of each task, first a *monitoring* step is performed to update the respective parameters, i.e., quality, execution time, and workload, next, the evaluation of a *re-allocation condition* establishes whether re-allocation is necessary or not for future plans on the basis of the updated parameters. Note that re-allocation should only be performed when necessary to avoid excessive computational burden of solving problem (1) and the mental overload caused by frequent task switching [16].

1) Parameters monitoring and update: The first step for re-allocation is to measure the quality and workload indices and execution times of both human and robotic agents at the end of the execution of each task τ_i. We refer to the values of the parameters before the update as *nominal* values in the following. Based on these measures, the parameters are updated as follows.

Concerning the quality update, a distinction is made depending on whether the task τ_i is supervised or not. In the case no supervision is foreseen, i.e., $S_{i,j} = 0 \ \forall a_{h,j} \in \mathcal{A}_h$, the measured quality becomes the current execution quality for the assigned agents. When the task is collaborative, i.e., $C_i = 1$, the measured quality is equally distributed between the two agents. In addition, when the quality index $q_{i,j}$ is updated, also the quality indices of agent a_j for tasks belonging to the same group $g(\tau_i)$ are updated accordingly, i.e., $q_{k,j} = q_{i,j} \ \forall \tau_k \in \mathcal{T}$ such that $g(\tau_i) = g(\tau_k)$. In the case supervision is foreseen, i.e., it exists $a_{h,j} \in \mathcal{A}_h$ such that $S_{i,j} = 1$, we further distinguish the cases in which the human does and does not intervene during the supervision. This is motivated by the fact that when human supervision is required, the human does not need to necessarily intervene during the task execution if the assigned agents are able to carry it out autonomously. Hence, if the human does not intervene, the measured quality becomes the execution quality of the executing agents, while if the human intervenes, the measured quality becomes his/her supervision quality and no update is made on the execution quality of the assigned agents.

As far as the workload and the execution times are concerned, they are updated to the measured values. Moreover, workload and the execution times for the tasks of the same group $g(\tau_i)$ are updated according to the same proportion, e.g., if the measured execution time is increased by 10% compared to the nominal one, then also the execution times of the tasks belonging to the same group are increased by 10%. Clearly, only the parameters associated with the involved agents, either supervising or executing the task, are updated. Furthermore, any other update policy is possible depending on the particular scenario.

2) Re-allocation strategy: After the parameters are updated, it is necessary to establish whether to online re-allocate the remaining tasks or not. We propose to perform re-allocation *i)* firstly, to ensure the feasibility of the allocation, i.e., re-allocation is performed if constraints of future tasks are violated considering the updated parameters. As instance, if the execution quality of an agent for a group $g(\tau_i)$ is decreased to a value lower than $\underline{q}$ during the monitoring and update step, and a task from this group must be executed by the same agent in the future, then supervision may be required to guarantee the minimum quality constraint (1e); *ii)* secondly, re-allocation depends on the change in performance. More specifically, let $\mathcal{T}^+ \subset \mathcal{T}$ be the subset of tasks which still need to be executed and let $\hat{C}^+$ be the cost function related to tasks in $\mathcal{T}^+$ and evaluated with the parameters available at current planning time, i.e.,

$$\hat{C}^+ = \max_{\tau_i \in \mathcal{T}^+} \frac{\bar{t}_i}{T_M} - \sum_{\tau_i \in \mathcal{T}^+} \sum_{a_j \in \mathcal{A}} \Big(\hat{q}_{i,j} X_{i,j} + \hat{q}^s_{i,j} S_{i,j} - \hat{w}_{i,j} X_{i,j} - \hat{w}^s_{i,j} S_{i,j} \Big)$$

where the notation $\hat{(\cdot)}$ is used to denote the best estimate of the parameters available so far. Similarly, let C^+ be the cost function related to future tasks and evaluated with the *updated* parameters. Re-allocation is carried out when

$$\delta \triangleq \frac{|\hat{C}^+ - C^+|}{\hat{C}^+} > \delta_t \tag{2}$$

with δ_t a positive constant. The rationale behind (2) is that re-allocation is also performed to preserve the allocation optimality with a certain tolerance. In particular, the higher the re-allocation parameter δ, the more likely the allocation made with parameters at planning time is not optimal for the updated parameters. Finally, online re-allocation is performed also *iii)* when a new batch of operations to execute is available or *iv)* when the available resources, either robotic or human, change with respect to the planned ones [17], e.g., a new human operator is available or a fault on a robot occurs.

VI. Simulation results

In this section, the proposed approach is evaluated in a realistic simulation environment.

A. Simulation setup

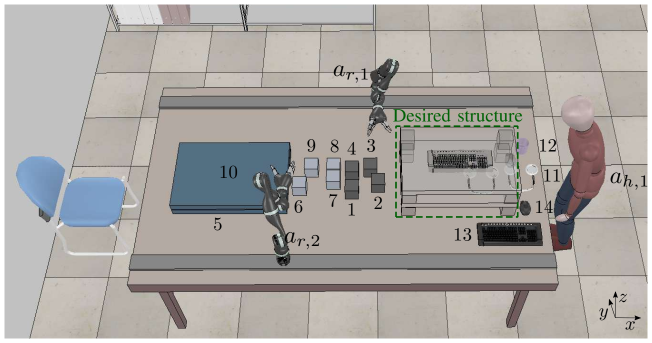

Fig. 1: Simulation setup composed of two manipulators ($a_{r,1}$,$a_{r,2}$) and a human operator ($a_{h,1}$). Objects to assemble are numbered and the desired structure is shown in transparency in the dashed box. The reference frame is shown in the bottom right corner.

A simulation collaborative setup, shown in Fig. 1, composed of two Kinova Jaco2 ($n_r = 2$), with 7-DOFs, mounted on sliding tracks (1-DOF) and a human operator ($n_h = 1$) is employed to validate the proposed approach. The left human hand of the simulated human operator is teleoperated in real-time by a person through a Microsoft Xbox controller. The developed architecture is reported in Fig. 2. Specific buttons are used to grasp/release objects as well as to move along x,y and z axes as detailed in Fig. 2 (bottom left). All software components are developed in Matlab interfaced with *i)* the controller, to receive human inputs, *ii)* Gurobi solver[1], to solve the optimization problem, and *iii)* Coppelia V-REP[2],

[1]https://www.gurobi.com
[2]https://www.coppeliarobotics.com

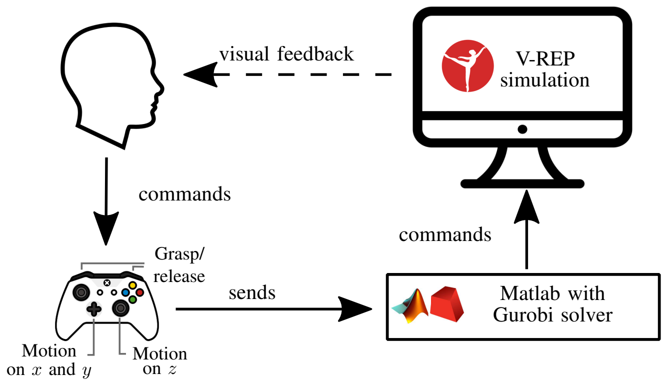

Fig. 2: Architecture of the validation setup. The human operator sends commands through a controller which is interfaced with Matlab. The latter resorts to Gurobi solver to find a solution for the optimization problem and communicates with V-REP simulation environment. Visual feedback is provided to the human through V-REP.

to simulate the operating environment. The latter provides visual feedback to the human operator. An illustrative video showing the effectiveness of the developed framework is available at the link[3].

A collaborative assembly process is considered in which a structure with two levels is built, as depicted in transparency in Fig. 1 and highlighted by the green dashed box. In particular, the following objects, divided in three groups, are involved: *i)* 8 cubes that make up the bases of the levels (four cubes for each level), *ii)* two planar surfaces with size 0.75 m $\times$ 0.5 m $\times$ 0.05 m (one for each level), *iii)* four office items, namely a cup, headphones, keyboard and mouse, which are added on the planar surfaces (two on each level).

Objects are numbered as shown in Fig. 1 and a pick-and-place task τ_i is defined for each object i. Due to the size of the planar surfaces (objects 5 and 10), their transport cannot be carried out by a single agent. Therefore, the respective tasks are set as collaborative, i.e., $C_5 = C_{10} = 1$.

Precedence constraints are introduced to properly build the two-level structure: tasks $1-4$, which position the cubic bases for the bottom level, need to be completed in order to start task 10, which positions the bottom planar surface; the latter task then needs to be finished to position the cubic bases for the top level (tasks $6-9$) as well as the office items $11, 12$. After tasks $6-9$ and $11, 12$ are accomplished, the top planar surface can be placed (task 5). Finally, when the planar surface is mounted, the remaining two office items (tasks $13-14$) can be placed.

Spatial constraints are defined for the cubic bases $7, 8$, i.e., $D_{7,8} = D_{8,7} = 1$, since they are placed close together in the initial configuration as shown in Fig. 1. For the same reason, spatial constraints are also introduced for the office items $13, 14$ (keyboard and mouse), i.e., $D_{13,14} = D_{14,13} = 1$.

Execution times for the robotic agents $a_{r,1}, a_{r,2}$ are computed by considering average linear velocity equal to 0.05 m/s and average angular velocity equal to 1.3 rad/s. Execution times for the human agent are set by recording once the required times by the human operator to perform the tasks and multiplying these times by a factor greater than 1 (1.1 in our case) in order to obtain more conservative estimates. Moreover, $\Delta_{i,j} = M$ is set if object i is outside the reachable workspace of agent j. The resulting maximum execution time in (1a) is $T_M = 668.37$ s.

Finally, workload of the robotic agents ($a_1, a_2 \in \mathcal{A}$) is initialized to 0.5 for tasks associated with the first group of objects, i.e., for the cubic bases of the structure, while it is set equal to 1 for the remaining tasks. Concerning the human agent, unit workload is considered for executing and supervising tasks associated with groups 1 and 3, i.e., $w_{i,3} = 1$, $w^s_{i,3} = 1$ for each τ_i such that $g(\tau_i) = 1$ or $g(\tau_i) = 3$, with 3 the index of the human agent (i.e., $a_3 \in \mathcal{A}$). High execution workload is considered instead to perform the collaborative tasks 5 and 10 for which it holds $w_{5,3} = w_{10,3} = M$, thus preventing the assignment of these heavy tasks to the human operator.

The following set of parameters is used: $\underline{q} = 0.8$ and $M = 1000$ in the problem formulation (1) and $\underline{\delta}_t = 0.15$ for the re-allocation condition in (2).

In the following, first we discuss the allocation and results obtained to carry out the described assembly process with the proposed framework. Next, we perform a simulation campaign to prove the effectiveness of the re-allocation strategy. Quality values are detailed in the respective case studies.

B. Experimental results

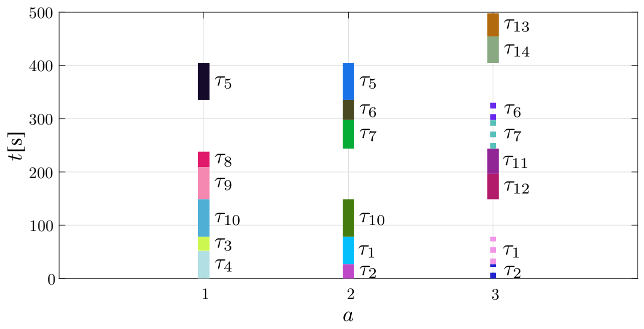

Fig. 3: Optimal allocation for the three agents to perform the assembly process. Each task is highlighted with a different color and start and end times are represented as well as assigned agents. Dashed thin lines are used to denote supervision tasks by the human operator a_3.

The experiment consists in performing the assembly process according to the optimal allocation obtained as solution of (1). Quality indices for the two robotic agents are initialized to 0.8 and 0.7, respectively, for tasks belonging to group 1 (i.e., cubic bases), to 0.4 and 0.4 for the collaborative tasks belonging to group 2 (i.e., planar surfaces), and to 0.5 and 0.5 for tasks belonging to group 3 (i.e., office items). The latter low values are motivated by the fact that the office items have more complex shapes to be manipulated by the robots than the objects in the other groups. With regard to the

[3]http://webuser.unicas.it/lai/robotica/video/HRC-MILP.mp4

human agent, unit execution and supervision quality indices are considered for all tasks, i.e., $q_{i,3} = 1$, $q^s_{i,3} = 1 \ \forall \tau_i \in \mathcal{T}$.

Before starting the execution of the tasks or when re-allocation is performed, the human operator is informed of the planned allocation through a visual information as shown, for example, in Fig. 3. At execution time, whenever the person has to start executing or supervising a task, an appropriate message is displayed in the V-REP simulator console as reported in the attached video together with a complete execution of the experiment[3]. In addition, an audio signal is generated to inform the human operator. When a task τ_i is required to be supervised, i.e., $S_{i,3} = 1$, the human operator can intervene and possibly reposition the object involved in the task. Let $\mathcal{N}(\mu, \sigma^2)$ denote a Gaussian distribution with mean μ and variance σ^2. To simulate realistic scenarios, a perturbation of the final objects position generated according to a Gaussian distribution $\mathcal{N}(0, 0.02)$ is introduced during the pick-and-place operations by the robotic agents. During the monitoring phase, the quality of a completed task is assessed by computing the difference between the desired configuration of the object and its measured one.

Figure 3 shows the planned optimal allocation to carry out the assembly process. In detail, tasks (highlighted with different colors) assigned to each agent a_j with $j = 1, 2, 3$ are represented along with their start and final times. Dashed thin lines are used to denote supervision tasks by the human operator a_3. A solution with makespan equal to 497.97 s is found in which all precedence, quality, simultaneity and spatial constraints are fulfilled. In particular, the execution of all tasks belonging to group 1 is distributed between the two robotic agents: tasks $3, 4, 8, 9$ are assigned to agent a_1, while tasks $1, 2, 6, 7$ are assigned to agent a_2. Among the latter tasks, task 8 starts after task 9 is accomplished in order to meet the spatial constraints. Moreover, supervision of the tasks $1, 2, 6, 7$, executed by the robotic agent a_2, is obtained in order to ensure minimum quality $\underline{q}$. In this regard, as visible from the video, the human operator only intervenes during task τ_7 to reposition the respective cubic base. Collaborative tasks 5 and 10 are assigned to the robotic agents to position the planar surfaces, while office items (i.e., tasks $11, 12, 13, 14$) are assigned to the human operator, achieving the best compromise quality/effort among the involved agents. Online monitoring of the parameters is made during the execution but no constraints are violated nor the performance parameter δ exceeds the re-allocation threshold δ_t, implying that no re-allocation is made in this case study. However, its validation is extensively carried out in the following.

C. Re-allocation results

To prove the effectiveness of the online re-allocation procedure, we perform a simulation campaign in which we monitor the values of the performance parameter δ, quantifying the similarity between the planned cost and the measured one, as well as of the cost function obtained with and without re-allocation. In particular, this case study focuses on evaluating the optimality provided by the re-allocation strategy compared to the static allocation. Random initial conditions as well as random online perturbations of the performance are considered for the simulation campaign. Initial values of the quality indices (both for execution and supervision) are set by adding a Gaussian random noise $\mathcal{N}(0, 0.2)$ to the minimum quality $\underline{q}$. The actual execution time of each task is generated by perturbing the value at planning time with a Gaussian random noise $\mathcal{N}(0, 0.02)$. Similarly, at the end of the execution of each task, measured quality and workload for execution and supervision are generated by perturbing the values at planning time with a Gaussian random noise $\mathcal{N}(0, 0.1)$. Perturbations are generated in such a way to always guarantee feasibility of the allocation, thus enabling the evaluation of the re-allocation impact on the solution optimality. Moreover, possible human intervention during supervised tasks is established according to a uniform discrete distribution in the range $\{0, 1\}$.

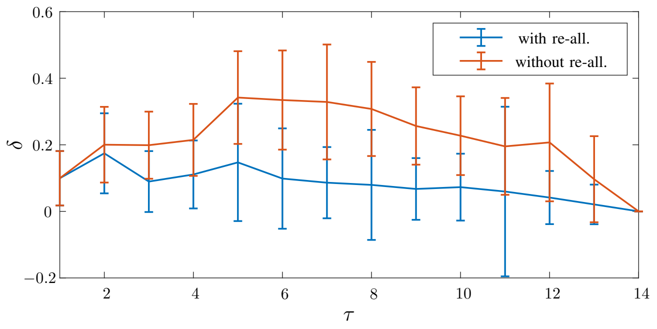

Fig. 4: Comparison of the values of the parameter δ at the end of each task obtained when re-allocation is enabled (in blue) and when it is not (in red). Average and standard deviation values are computed over 50 trials.

Figure 4 reports the obtained values of the parameter δ at the end of each task when re-allocation is enabled (in blue) and when it is not (in red), i.e., in the latter case the initial allocation is maintained throughout the execution. Average and standard deviation values over 50 trials are shown. The figure makes evident the general improvement (i.e., lower values) given by the re-allocation procedure on the parameter δ compared to the case of static allocation. More specifically, average δ equal to 0.08 is achieved with reallocation, while average δ equal to 0.21 is reached without reallocation. Note that same results for the initial and final tasks are obtained since at the beginning the same initial allocation is considered by the two solutions (with and without re-allocation), while at the end no further tasks need to be executed meaning that $\mathcal{T}^+ = \emptyset$ and $\hat{C}^+ = C^+ = 0$.

Finally, the optimality improvement is also confirmed by the evaluation of the cost function achieved by the solutions with and without reallocation. In particular, the former achieves 0.91 ± 1.15, while the latter achieves 2.62 ± 0.68, thus leading to an average improvement on the obtained cost function equal to $\approx 65\%$ compared to the case of static allocation (i.e., without re-planning).

VII. Conclusion

In this work, the task allocation problem in a human multi-robot collaborative scenario was addressed. A general framework was proposed that allows to obtain optimal allocation considering several aspects, like execution quality and workload, cooperative tasks, human supervision and spatial constraints. The problem is formulated as a Mixed-Integer Linear Programming problem and an optimal allocation is found; re-allocation is obtained via real-time monitoring of execution parameters and when performance falls below a given threshold a new plan is computed. Future work aims at validating the solution in a real scenario and at considering also plan switching cost in the optimization problem. Furthermore, we plan to define an adaptive procedure to determine the threshold δ_t for online re-allocation according to human preferences.

References

[1] P. Tsarouchi, S. Makris, and G. Chryssolouris, “Human–robot interaction review and challenges on task planning and programming,” *International Journal of Computer Integrated Manufacturing*, vol. 29, no. 8, pp. 916–931, 2016.

[2] S. E. Hashemi-Petroodi, S-Thevenin, S. Kovalev, and A. Dolgui, “Operations management issues in design and control of hybrid human-robot collaborative manufacturing systems: a survey,” *Annual Reviews in Control*, vol. 49, pp. 264–276, 2020.

[3] S. El Zaatari, M. Marei, W. Li, and Z. Usman, “Cobot programming for collaborative industrial tasks: An overview,” *Robotics and Autonomous Systems*, vol. 116, pp. 162–180, 2019.

[4] E. Nunes, H. Mitiche, and M. Gini, “A taxonomy for task allocation problems with temporal and ordering constraints,” *Robot. Auton. Syst.*, vol. 90, p. 55–70, Apr 2017.

[5] B. P. Gerkey and M. J. Matarić, “A formal analysis and taxonomy of task allocation in multi-robot systems,” *The International Journal of Robotics Research*, vol. 23, no. 9, pp. 939–954, 2004.

[6] E. Suslova and P. Fazli, “Multi-robot task allocation with time window and ordering constraints,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 6909–6916, 2020.

[7] M. Gombolay, R. Wilcox, and J. Shah, “Fast scheduling of multi-robot teams with temporospatial constraints,” 2013.

[8] M. C. Gombolay, R. A. Gutierrez, S. G. Clarke, G. F. Sturla, and J. A. Shah, “Decision-making authority, team efficiency and human worker satisfaction in mixed human—robot teams,” vol. 39, p. 293–312, Oct. 2015.

[9] C. J. Shannon, L. B. Johnson, K. F. Jackson, and J. P. How, “Adaptive mission planning for coupled human-robot teams,” in *American Control Conference*, pp. 6164–6169, 2016.

[10] K. Bogner, U. Pferschy, R. Unterberger, and H. Zeiner, “Optimised scheduling in human–robot collaboration – a use case in the assembly of printed circuit boards,” *International Journal of Production Research*, vol. 56, no. 16, pp. 5522–5540, 2018.

[11] S. Zhang, Y. Chen, J. Zhang, and Y. Jia, “Real-time adaptive assembly scheduling in human-multi-robot collaboration according to human capability,” in *IEEE International Conference on Robotics and Automation*, pp. 3860–3866, 2020.

[12] A. A. Malik and A. Bilberg, “Complexity-based task allocation in human-robot collaborative assembly,” *Ind. Robot*, vol. 46, pp. 471–480, 2019.

[13] E. Makrini, K. Merckaert, J. de Winter, D. Lefeber, and B. Vanderborght, “Task allocation for improved ergonomics in human-robot collaborative assembly,” *Interaction Studies*, vol. 20, no. 1, pp. 102–133, 2019.

[14] E. Lamon, A. De Franco, L. Peternel, and A. Ajoudani, “A capability-aware role allocation approach to industrial assembly tasks,” *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3378–3385, 2019.

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

[16] “Discrete task switching in overload: A meta-analyses and a model,” *International Journal of Human-Computer Studies*, vol. 79, pp. 79–84, 2015. Integrating Knowledge of Multitasking and Interruptions Across Different Perspectives and Research Methods.

[17] N. Nikolakis, K. Sipsas, P. Tsarouchi, and S. Makris, “On a shared human-robot task scheduling and online re-scheduling,” *Procedia CIRP*, vol. 78, pp. 237–242, 2018.