Human-Robot Collaboration for Assembling IKEA Furniture through Reinforcement Learning-Based Planning

Giulio Giacomuzzo*, Diego Romeres‡

***Abstract*— In this paper we consider the task planning problem of a Human-Robot collaborative scenario to assemble IKEA furniture. The human is considered an uncontrollable agent and the task planner schedules the optimal actions of the robot in order to complete the full assembly task in the least amount of time. We formalize the problem as a Discrete Event Markov Decision Problem (DE-MDP) which allows to model stochastic events such as human change of mind. While the problem could be solved by creating a graph of all the possible actions, this solution would have computational limitations. However, the proposed formulation allows a solution with Reinforcement Learning to compute an optimal policy for the robot.**

I. Introduction

In recent years, the area of human-robot collaboration has gained increasing attention because of the advancement of robotics and artificial intelligence. The synergistic integration of human dexterity and robotic precision holds great potential especially for small- and medium-sized enterprises with high-mix and low-volume productions. One particularly challenging yet promising test-bed application is the assembly of IKEA furniture which involves dexterous manipulations, precise alignments, transportation of cumbersome objects and tasks that require more than two hands.

Central to the realization of this H-R collaborative vision is the development of robust planning algorithms that combines the complementary strengths of both humans and robots. This problem has recently been studied in literature by several works. In [1] a Robust Plan Recognition and Trajectory Prediction (RPR-TP) is proposed, where the human's intentions and plan are recognized using a perception module. The robot recognize the human plan and perform his actions accordingly, but its contribution is limited to assistive operations, without the possibility actively contribute to the task advancement. In [2] the same authors propose a collaborative framework where the robot plan is obtained in order to minimize the total task completion time and promote workspace separation. However, within this framework the the possibility to collaborate together on the same joint task is not considered. In [3] a robust planning algorithm based a a Partially Observable Markow Decision Process (POMDP) is developed. Therein the focus is on understanding how to model the inherent coupling between the two agents decision making processes, which makes determining both the human and robot policies very complex. The proposed solution introduces the ability for the two agents to perform both individual and joint actions. However, they do not consider actions of prolonged and different duration. A decentralized multi-agent framework was proposed in [4], where teams of humans and robots are considered to perform together a collaborative task. This work focuses on planning with actions of extended time duration, also considering the problem of handling failures. However, the possibility of performing joint actions is not considered. In general, to the best of the authors' knowledge, a framework for collaborative assembly scheduling which combines both the ability to work on independent tasks and to cooperate together on joint actions, tacking into consideration realistic problems such as the human change of mind or the human action detection delay, still lacks in literature.

Fig. 1: On the left-hand-side the components of the disassembled IKEA Ivar chair are displayed. On the right-hand-side the fully assembled chair is displayed (Image is downloaded from IKEA website `https://www.ikea.com/us/en`)

In this paper we bridge this gap by presenting a novel framework for the scheduling problem in collaborative assembly tasks. First, we assume the human behavior to be inherently uncontrollable, with its decision making only being observed by the robot after an action detection time. This defines a general framework with no assumptions on the human behavior and a reactive robot to all the actions observed. Consequently, the problem is a single agent task where the robot plans the actions based on the environment state and on the decisions made by the human. Then, the collaborative assembly scheduling is modeled as a Discrete Event Markov Decision Process (DE-MDP), where the state dynamics is driven by the occurrence of significant events such as action completion, action detection or human change of mind. We consider two methods to solve the DE-MDP, the first one based on a deterministic decision graph, the second one based on Reinforcement Learning (RL). Results on both toy problems and on a simulated chair example show that RL converges to the optimal policy when the interaction is not affected by uncertainty, and provides a

[1]Department of Information Engineering, Università di Padova, Italy, giacomuzzo@dei.unipd.it
[2]MERL, romeres@merl.com

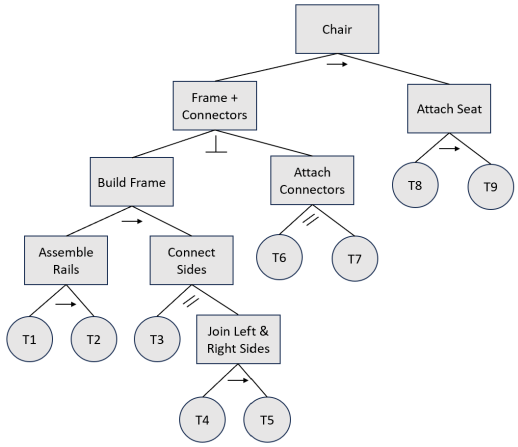

Fig. 2: A possible HTM model for Ivar chair assembly task.

viable solution when stochasticity is taken into consideration.

II. Problem Formulation

Assembly tasks can be seen as a hierarchical set of atomic tasks, hereafter denoted as macro-actions, which must be executed complying with a set of ordering constraints. In this work, we assume macro-actions to be either individual or joint actions. Individual actions can be executed by a single agent either the human a^h or the robot a. Joint actions must be executed by both the agents at the same time. Moreover, we consider the macro-actions to have different duration, which depends on the action itself and on the agent executing it. The scenario includes two agents but we consider the human to be an uncontrolled agent. Therefore, the assembly is modeled as a single agent problem in which the goal is to learn the robot policy $\pi_\theta(\cdot)$ which minimize the execution time and reacts to the human choices.

The aforementioned setting introduces interesting challenges, which we aim to address within our framework. **Synchronization:** the human and the robot can work in parallel on different macro-actions but have to perform joint actions together. Since macro-actions have different duration, agent synchronization is required to perform joint actions.
Human detection: as the human behave uncontrolled and the robot has to adapt its actions to the human choices, human action detection is required. This inevitably introduces a delay into the robot action scheduling, which needs to be considered by the planner.
Human change of mind: the human could begin a task, and then suddenly change its mind and starting performing a new task. In that case, the robot should reschedule its plan.

To address the aforementioned challenges, we consider the following assumptions. i) Joint action can be chosen only by the human. In case the human chose a joint action, the robot joins them as soon as it finishes the ongoing action. ii) We introduce an additional action called *idle*, which can be performed by both the agents. When *idle*, the agent is waiting, without contributing to the task advancement. We use the *idle* as a synchronization before a joint action: when the human chooses a joint action, they remain idle until the robot has completed its action. When the robot finishes, the two agents perform the joint action together. iii) If the human action has not been detected, the robot must remain *idle*. iv) The change of mind can occur only if the human action has been detected. After a change of mind, another detection is required.

A. Hierarchical Task Model for Assembly task description

As proposed in [2], we rely sequential/parallel Hierarchical Task Model (HTM) for task representation. An example of HTM for the assembly of a chair is reported in Fig. 2. Within the HTM representation, the root node represents the entire assembly, while all the other nodes represent assembly subtasks. The leafs represent the macro-actions to be executed. Each node can be categorized as parallel ($\|$), sequential ($\rightarrow$) or independent ($\perp$). Children of parallel nodes can be executed at the same time by the two agents, in any order. Children of sequential nodes, instead, must be individually executed in the specified order (from left to right). Finally, children of independent nodes must be individually executed, but can be performed in any order.

In our framework, the HTM provides the sequential constraints on action executions, which are encoded in the DE-MDP described in the next Section.

III. Robot Task Scheduler

In this section we describe the proposed method to model and solve the collaborative assembly scheduling. Our method describe the system as a Discrete Event Markov Decision Process (DE-MDP). A DE-MDP is an MDP in which the state transitions do not happen on a fixed time basis, but are determined by the occurrence of events. In the case of collaborative assembly tasks, the tasks advancement is related to the completion of macro-actions which have different, possibly stochastic, duration. Moreover, the cooperation state is also modified by events such as the human action recognition or the human change of mind.

This modeling choice allows to unify under the same framework many desirable properties for cooperative assembly that were scattered in several different works in literature.

A. Discrete Event Markov Decision Process

Consider an assembly task composed by N macro-actions. The collaborative scheduling problem is modeled as a DE-MDP (S, A, $\mathscr{E}$, Γ, ℓ, T, R, γ), where S is the state space of the system; A is the action space; $\mathscr{E}$ represents the set of possible events; $\Gamma(s)$ is the set of feasible events at state $s \in S$; $\ell(s,a,e)$ is the event lifespan function, which gives the probability distribution over the time after which the event e is likely to occur at the current state $s \in S$ and robot action $a \in A$; $T(s,a,s',e)$ is the transition probability function which gives the probability of transitioning to state s' given the current state s the current robot action a and the event e; $R(s,a,e)$ is the reward function and γ the discount factor.

Differing from the standard MDP framework where the state update occurs at every time step, the DE-MDP state transition happens only in presence of a significant event.

We will use the index variable k to represent the time instant of the state evolution. Moreover, Δt^k denotes the time spent by the system to transition from state s^k to s^{k+1} and, for convenience, it is multiple of a discrete time step $\bar{T}$. Therefore, the variable time step, Δt^k, at which the DE-MDP state updates depends on the event lifespan $\ell(s^k, a^k, e^k)$, with e^k being the event causing the system transition from s^k to s^{k+1}. In the following, for notation simplicity we will explicit the dependence on the index k only when necessary.

In the following the elements of the DE-MDP are detailed.

a) States: the system state is $s = (s_a, a^h, t^h, t^r, d)$ where $s_a \in \{0,1\}^N$ is the macro-actions execution indicator, the i-th component representing the i-th macro-action being executed or not; $a^h \in \{0,\dots,N\}$ is the current human macro-action which is part of the state because the agent cannot be controlled; $t^h \in \mathbb{N}$ is time elapsed since the current human macro-action started; $t^r \in \mathbb{N}$ is the time since the current robot macro-action started; $d \in \{0,1\}$ indicates if the current human action has been detected.

b) Actions: the actions set A includes all the macro-actions that can be executed by the robot. Let $\bar{A} = \{a_1,\dots,a_N\}$ be the set of all possible macro-actions to complete the task. Moreover, let us denote with a_{N+1} the *idle* macro-action. We model each macro-action as a tuple (o_j, δ_j), where $o_j \in \{0,1,2,3\}$ denotes if the action can be performed by the human (0), by the robot (1), by both the agents (2) or is a joint action (3); $\delta_j = [\delta_j^h, \delta_j^r]$ represents the nominal action duration when performed by the human (δ_j^h) or by the robot (δ_j^r). Note that in case of joint actions, δ_j^h and δ_j^r are equal and the *idle* macro-action lasts until a triggering events happens. For the sake of simplicity, in this work the duration of the macro-actions is deterministic and known. Within this formalism, we define the action space as $A = \{a_j \in \bar{A} | o_j \neq 0\} \cup \{a_{N+1}\}$.

Feasible Actions: A_f is the set of all feasible actions for the two agents at the current state. The non feasible actions are: the the macro-actions already completed, the current action being executed by the other agent and the *idle* action for the robot if the human is already *idle*, to avoid infinite cycles.

c) Events: the set $\mathscr{E} = \{H, R, D, C\}$ contains the events triggering the state transitions, where H represents the end of a human action, R represents the end of a robot action, D represents the human action detection and C represents a human change of mind. Note that, the probability of an event happening $p(e|s,a)$ can be fully determined by the current state $s \in S$ and robot action $a \in A$. In particular, if $e = C$, we denote with $p_c \in [0,1]$ the probability of a human change of mind to happen. Then, $p(C|s,a)$ represents the probability of having a change of mind before the next H or R event. The probability of having a change of mind before the next H event is p_c, as, if the change of mind happens, it will for sure happen before the end of the current human action. The probability of having a change of mind before the next R event, instead, depends on the time at which the next C and R will happen. Let Δ_C and Δ_R be the lifespans of C and R, respectively. Then, C will happen before R with probability $p_c p_{cr}$, with p_{cr} being the probability $p(\Delta_C < \Delta_R)$. See the paragraph regarding the lifespan function for the description on how the lifespan distributions are defined. Note that $p_{cr} = 1$ if $\Delta_R > \Delta_H$. We can in general conclude that $p(C|s,a) = p_c p_{cr}$. Then, if $e = H$, the probability $p(H|s,a)$ is 0 if the robot finishes its action before the human i.e., $\delta_a^r - t^r < \delta_{a^h}^h - t^h$, while it is $1 - p(C|s,a)$ if the human finishes before the robot. Analogous reasoning applies for the event $e = R$. Finally, if $e = D$, $p(D|s,a) = 1$ if $d = 0$, otherwise $p(D|s,a) = 0$.

d) Feasible event set: $\Gamma(s) : S \to E \subseteq \mathscr{E}$ provides the set of feasible events given the current state $s \in S$. In particular we have: $\Gamma(s|d=0) = \{D\}$, namely if the human action has not been detected, the only feasible next event is the detection, while if the human action has been detected all the other events are feasible, namely $\Gamma(s|d=1) = \{H, R, C\}$.

e) Event lifespan: the event lifespan function $\ell(s,a,e)$ provides a probability distribution over the time spent by the system in the current state $s \in S$ with the current robot action $a \in A$, before transitioning to the next state due to the event $e \in \mathscr{E}$, namely $\ell(s,a,e) = p(\Delta_e|s,a)$, with Δ_e being the time after which the event e will occur given s and a. In the assembly scenario we are considering, if $e = D$ then $\Delta_D = \bar{\Delta}_D$ with probability 1, where $\bar{\Delta}_D$ represents the human action detection time, which we assume to be deterministic and known. If $e = H$, $\Delta_H = \delta_{a^h} - t^h$. If $e = R$, $\Delta_R = \delta_{a_r} - t^r$. Finally, we model the lifespan Δ_C of the event $e = C$ as a discretized truncated exponential distribution with support spanning the time interval from the detection time to the end of the human macro-action.

f) Transition: The transition function $T(s,a,s',e)$ provides the probability of transitioning to the next state $s' \in S$ given the current state s and the current robot action a due to the occurrence of the event e. We define in the following the transition probability for each possible event in the collaborative assembly scenario we are considering. If $e = D$, namely the event is a human action detection, the only effect on the state is the deterministic transition of d from 0 to 1. If $e = H$, namely the event is a human action end, the transition will affect s_a, a^h, d and t^h. In particular, s_a will transition to $s'_a = s_a + 1_{a^h}$, where $1_{a_j} \in \{0,1\}^{N+1}$ is the vector containing all zeros except for the component corresponding to a_j. a^h will transit to $a^{h'}$ according to the human policy, which we consider to be known. Finally, $d' = 0$ and $t^{h'} = 0$. If $e = R$, the transition will affect s_a and t^r, in particular $s'_a = s_a + 1_a$ and $t^{r'} = 0$. Finally, a change of mind $e = C$ will cause the human and consequently the robot to change action, namely $a^{h'}$ will be chosen according to the human policy, while $t^{h'} = 0$ and $t^{r'} = 0$.

g) Reward: as we aim to minimize the execution time, we model the reward as the negative transition time, namely $R(s,a,e) = -\Delta_t$.

B. DE-MDP Solutions

Solutions to the DE-MDP provide a robot policy $\pi_\theta(\cdot)$ mapping the current state s to the next robot action a. In this work we consider two solutions, one based on a deterministic decision graph and the other one obtained using RL.

a) Decision Graph: given the DE-MDP description, a decision graph can be built assuming a deterministic setting, namely considering the nominal values for macro-actions duration and disregarding stochastic events such as the human change of mind. A decision graph is a directed graph in which each node represents a state, while each edge represent a robot action. It can be built starting from the initial state and computing, for each state, all the possible transitions, until the final state is reached. Each edge of the graph is weighted with the cost of the transition (e.g. the transition time). Then, any graph search algorithm (e.g. Dijkstra [5]) can compute, for each state, the minimum cost path to the final node. The union of all the paths is the optimal policy and it can be computed offline and stored in a look-up table. Nonetheless, the generation of the decision graph can be particularly inefficient or even infeasible both in terms of computational time and storage requirements, when the number of macro-actions and possible transition increase. Moreover, stochasticity is not taken into consideration and the policy can only react to stochastic events.

b) Reinforcement Learning: Given the problem formalization as an MDP, RL algorithms are well suited to overcome the limitation of the approach based on the decision graph. In particular, in presence of predictable stochastic events, RL can learn the stochastic dynamics and exploit the information to take preventive actions. Moreover, parametric function approximators can be exploited in presence of high dimensional state-action spaces to reduce the computational and storage burden.

IV. Simulations

In this section we show that RL can be effective in solving the planning problem in two different cases: i) a deterministic setting, in which the human policy $\pi_h(s,a)$ is deterministic and the human does not change action, and ii) a stochastic setting with a random human policy and the RL performance are evaluated as the amount of change of mind increases. In both settings the RL policy is trained with the Proximal Policy Optimization (PPO) algorithm [6]. To this aim, we implemented the DE-MDP described in Section III-A as a custom Gym environment [7] and used the Stable Baselines library [8] for policy training. The training process is sped up using action masking to limit the explored actions only to A_f, as explained in [9].

A. Deterministic setting

In the deterministic case we designed 4 toy experiments with number of macro-action varying from $n = 8$, to $n = 32$. For each experiment, we generated a random HTM, composed of $\frac{n}{4}$ joint macro-actions and $\frac{n}{2}$ only robot actions. The remaining macro-actions can be performed by either the human and the robot. The duration of the macro-action duration is randomly generated between $4\bar{T}$ and $16\bar{T}$. Moreover, we considered also the realistic example of the IKEA Ivar chair assembly (see Fig. 1 and 2). The performances of the RL policy are compared with those of the graph based policy, which in this setting represents the optimal policy. Results

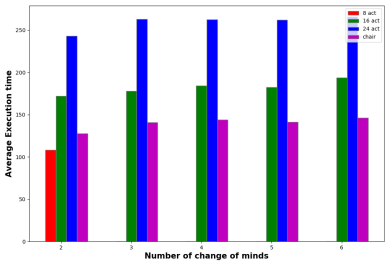

Fig. 3: Average completion time obtained with the Monte Carlo experiments presented in Section IV-B, at the increase of the number of changes of mind. Results are reported as number of discrete time steps.

are reported in Tab. I in terms of number of steps $\bar{T}$ required to complete the task. In all the 4 considered experiments, the RL policy produces the same results of the Graph policy, which confirms that RL is able to learn the optimal policy.

	8 Actions	16 Actions	24 Actions	32 Actions	Chair
PPO	91	157	240	305	73
Graph	91	157	240	305	73

TABLE I: Completion time on the deterministic experiments. Results are reported in terms of number of time steps $\bar{T}$.

B. Stochastic setting

In the stochastic setting, we assumed the human chooses its action randomly among the set of feasible actions, and we considered different levels of change of mind probability p_c, from 0.1 to 0.9. For each value of p_c, we trained a policy with PPO and tested it with a Monte Carlo experiment composed by 1000 simulations. Fig. 3 plots the the mean completion time against the number of changes of mind occured, for the tasks considered in the previous section. As one can expect, at the increase of the change of mind probability, the average completion time increases. However, the RL policy is able to deal with the change of mind and compute a viable solution. Further experiments and comparisons against different methods will be performed to assess the optimality of the RL solution.

V. Discussion and Conclusions

We have formalized the collaborative assembly of IKEA furniture between a Human and a Robot as a Discrete Event Markov Decision Problem. The proposed framework models at the same time all the properties of Parallel-, Sequential-, Joint- Independent-, Asynchronous- and with Variable Duration- macro actions, and includes Human intent detection and change of mind. To the best of our knowledge, none of the previous literature has proposed a model for all of these properties at the same time. The formulation as DE-MDP also makes the problem treatable with RL which solves the problem of computational complexity of graph based solutions. We solved in simulations both toy problems and an HTM of a real IKEA Ivor Chair. In the future we will also consider more stochastic events such as failure cases and will also deploy the algorithm to assemble a real IKEA chair.

REFERENCES

[1] Y. Cheng, L. Sun, C. Liu, and M. Tomizuka, "Towards efficient human-robot collaboration with robust plan recognition and trajectory prediction," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 2602–2609, 2020.

[2] Y. Cheng, L. Sun, and M. Tomizuka, "Human-aware robot task planning based on a hierarchical task model," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1136–1143, 2021.

[3] Y. You, V. Thomas, F. Colas, R. Alami, and O. Buffet, "Robust robot planning for human-robot collaboration," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023.

[4] N. Dhanaraj, S. V. Narayan, S. Nikolaidis, and S. K. Gupta, "Contingency-aware task assignment and scheduling for human-robot teams," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 5765–5771.

[5] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[6] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017.

[7] M. Towers, J. K. Terry, A. Kwiatkowski, J. U. Balis, G. d. Cola, T. Deleu, M. Goulão, A. Kallinteris, A. KG, M. Krimmel, R. Perez-Vicente, A. Pierré, S. Schulhoff, J. J. Tai, A. T. J. Shen, and O. G. Younis, "Gymnasium," Mar. 2023. [Online]. Available: https://zenodo.org/record/8127025

[8] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, "Stable-baselines3: Reliable reinforcement learning implementations," *Journal of Machine Learning Research*, vol. 22, no. 268, pp. 1–8, 2021. [Online]. Available: http://jmlr.org/papers/v22/20-1364.html

[9] S. Huang and S. Ontañ ón, "A closer look at invalid action masking in policy gradient algorithms," *The International FLAIRS Conference Proceedings*, vol. 35, may 2022. [Online]. Available: https://doi.org/10.32473%2Fflairs.v35i.130584