A Tool to Model Task Planning Domain for Human-Robot Collaboration

Elisa Foderaro, Amedeo Cesta, Alessandro Umbrico, Andrea Orlandini
Institute of Cognitive Science and Technology, National Research Council of Italy, Rome
Email: {name.surname}@istc.cnr.it

Abstract

Temporal flexible planning enables Collaborative Robots to autonomously operate understanding the environment, dynamically planning their tasks and acting to achieve some given goals. The introduction of tools to facilitate the integration of Planning and Robotics is crucial but entails different kind of expertise to address a wide variety of control issues and the design of integrated solutions. Namely, Domain experts, Planning specialists and Roboticists are to configure a set of proper control solutions to enable an effective and safe production. This paper presents a novel software tool, called TENANT (Tool fostEriNg Ai plaNning in roboTics), that facilitates domain experts in defining goals, tasks and a set operational constraints enabling the automatic generation of planning models for robot control. TENANT is validated in an industrial collaborative scenario derived from a EU-funded research project demonstrating its effectiveness in generating automated planning models to control a collaborative robot.

Introduction

The deployment of Human-Robot Collaborative (HRC) Robots (i.e., cobots) in manufacturing scenarios requires to address multiple challenges. It is of paramount importance to endow cobots with the ability of quickly adapting behaviours to actual state of the environment and to keep the user safe and engaged in the interaction. Long-term research activities are ongoing to enable robots to autonomously operate in environments, i.e., understanding the actual situation, planning their tasks and acting to safely and effectively achieve some given goals (Rajan and Saffiotti 2017). Several approaches aim to achieve robust action selection via Artificial Intelligence Planning, e.g., (Cashmore et al. 2015; Chanel, Lesire, and Teichteil-Königsbuch 2014; Lacerda, Parker, and Hawes 2014) or robust execution via some form of finite state machine (FSM), e.g., (Bohren and Cousins 2010; Ziparo et al. 2011). Nevertheless, state-of-the-art software tools are usually not developed to support the design of such applications as a collaborative process between different human experts making also strong assumptions on mutual knowledge (Viola et al. 2019). Therefore, the introduction of tools to facilitate the integration of A.I. planning and robotics and the design of well-integrated solutions entails different kind of expertise to address a wide variety of control issues, spanning from low-level control to decisional autonomy configuration. Among others, a crucial knowledge engineering problem is the lack of a generally accepted modeling methodology entailing many potential back-and-forth (re)work over models and control parameters before defining the proper control configuration.

Some attempts to connect AI and Robotics environments have been made. For instance, ROSPlan (Cashmore et al. 2015) has been proposed as a unique integrated solution for PDDL-based planners to be smoothly deployed in ROS architectures (Quigley et al. 2009; Cashmore et al. 2014) also with planning techniques specifically used for human-robot interaction (Sanelli et al. 2017). Robotics experts can easily connect their ROS-based modules to any (supported) PDDL-planner but there is no support to define planning specifications. So, roboticists are leaft alone in managing a knowledge planning modelling. Several timeline-based planning frameworks such as, e.g., EUROPA (Barreiro et al. 2012) and APSI-KEEN (Orlandini et al. 2014), provide knowledge engineering support for planning. But none of them provides structured support for deployment of robotic applications. In general, all those solutions require robotic experts to have some expertise in planning specification.

A first solution addressing this issue has been proposed in (Viola et al. 2019). Yet, a domain expert may have difficulties in approaching this kind of solutions. A *Domain expert* is usually responsible for the definition of the tasks and overall goals of robotic system, while other actors should have responsibility on more specific aspects, i.e., a *planning expert*, with knowledge on P&S to provide robust A.I. planning features, and a *robot expert*, responsible of actually implementing robot operations.

This paper proposes a novel software tool, called "Tool fostEriNg Ai plaNning in roboTics" (TENANT), addressing the needs of Domain Experts to set goals, defining tasks and set operational constraints notwithstanding the intrinsic complexity required at planning and robotics level. TENANT is a general purpose tool that can be deployed for addressing multiple applications/domains and can be easily integrated with other knowledge engineering tools such as, e.g., ROS-TiPlEx (Viola et al. 2019) and Planning and Scheduling software framework, e.g., PLATINUm (Umbrico et al. 2017, 2018). TENANT allows domain experts

to specify sequence of tasks that has to be performed to accomplish a specific mission in Human-Robot Collaborative scenarios. Each tasks can be structured by subtasks and characterized by specific configurations/properties relevant for their execution, e.g., specifying collaborative modalities, or with a predefined assignment to a human user or a robot, e.g., declaring who is actually in charge of performing a certain (sub)task. TENANT allows also to define operational constraints such as, e.g., temporal synchronizations or precedence constraints in order to properly model the nominal execution of flows of the tasks. Then, leveraging planning specification templates, TENANT can automatically generate a planning model that can be used to feed a P&S system enabling intelligent robot behaviors.

TENANT has been validated in an industrial HRC scenario derived from a EU-funded research project[1] whose goal is the development of innovative technologies to endow collaborative robots with suitable intelligent features to perceive, understand and properly act to support human workers in industrial collaborative cells.

The paper is organized as follows. First, we describe an industrial HRC scenario that will be considered as a use case. Then, we present some background knowledge on Planning and Scheduling. Then, TENANT is presented and validated in the considered HRC scenario. Finally, some conclusions close the paper.

An industrial HRC scenario

A real industrial HRC scenario is considered (See Figure 1) where the working cell is constituted by a logistic station in which a collaborative robot (cobot) and a human operator cooperate and work together to properly assembly parts to be machined on a device called pallet.

The logistic station consists of a central worktable, where the workpieces are placed, with a collaborative robot and a human operator at its opposite sides. The cobot consists of three different components: the robotic arm itself, which can assume two positions: idle or lowered on the workpiece; a mobile base for its movement; a gripper to grasp or release a workpiece. Due to high variability in the production, the assembly process (i. e. the sequence of tasks to be performed) is different for each different product, and it has to be defined by a domain expert. A specific feature of the production work is that each task in the assembly process can be performed by the human operator, by the collaborative robot, or by both in a cooperative manner. The human operator and the robot can perform two types of operation: manipulating a workpiece (e.g. to screw it on the pallet), or a ”pick and place” task. To perform a manipulation task, it is required to stand in front of the goal position (i.e., the location where the workpiece is placed), whereas to perform a pick and place task the following operations are required: stand in front of the initial position, pick up the workpiece, move to the final position, place the workpiece down.

From a planning point of view, efficiency and safety in such HRC scenarios are related to the solution of different intertwined problems. First, the system should be able

[1] http://wwww.sharework-project.eu

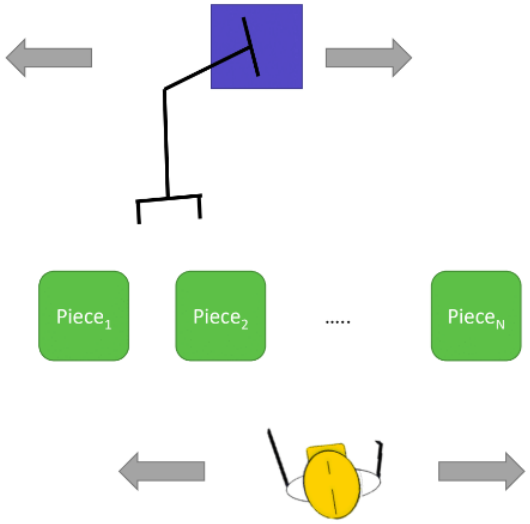


Figure 1: A schematic description of the considered collaborative scenario. A Robot (on the upper part) and a Human (on the bottom) work together to pick and place some workpieces (green boxes). Both the human and the robot can move towards right/left to pick and place the pieces.

.

to compute the sequence of the operations (*task planning*). Then, the execution of the operations should be adapted at run-time based on the human and robot's state (*motion planning and re-planning*). Furthermore, task assignment and synchronization between humans and robots should consider duration variability and good (or bad) synergies of simultaneous collaborative operations. The complexity of the overall planning problem has limited the effectiveness of existing methodologies in real-world scenarios. As a matter of fact, considering geometric reasoning (*e.g.*, collision checking) at a high level of abstraction is computationally expensive and not tractable for complex scenarios. Tiered approaches that interleave task planning and motion planning are therefore preferred, but this field of study is still not suitable for HRC in realistic manufacturing scenarios (Tsarouchi, Makris, and Chryssolouris 2016). In this regard, researches in task planning of HRC processes and human-aware motion planning are underway, and shared standard approaches are still to come.

In this paper, the focus is on task assignment and human-robot coordination proposing to use TENANT to support the definition of task planning models to endow cobots with human-aware task planning features. Therefore, in this scenario, the details on the geometric shape of the pieces which are object of the collaborative process are not considered. A symbolic association is defined to create a correspondence between the actual position of the workpieces on the table and the ”names” of such positions. Geometric reasoning is out of the scope of our analysis.

Automated Planning & Scheduling

The development of cobots capable of performing autonomous task planning as well as tasks assignment and coordination entails the integration of Artificial Intelligence technologies. In this work, we propose a specific well established planning approach and technology. The *timeline-based planning and scheduling* approach is a particular temporal planning paradigm which has been introduced in early 90s (Muscettola 1994) taking inspiration from the classical Control Theory, and successfully applied in many real-world scenarios (Muscettola 1994; Cesta et al. 2011). This planning paradigm aims at controlling a system by synthesizing temporal behaviours for a set of identified domain features that must be controlled over time.

A timeline-based model is composed of a set of *state variables* describing the possible temporal behaviours of the domain features that are relevant from the control perspective. Each state variable specifies a set of *values* that represent the states or actions the related feature may assume or perform over time. Each *value* is associated with a *flexible duration* and a *controllability tag* which specifies whether the value is controllable or not from the system. A *state transition function* specifies the valid temporal behaviours of a state variable by modelling the allowed sequences of values (i.e., the transitions between the values of a state variable). State variables model "local" constraints a planner must satisfy to generate valid temporal behaviours of single features of the domain i.e., valid *timelines*. Further constraints may be necessary for the behaviours of state variables in order to coordinate different domain features and realize complex functionalities or achieve complex goals. A set of rules, called *synchronizations*, model "global" constraints that a planner must satisfy to build a valid plan. Such rules are also to specify *planning goals*. A task planner synthesizes a set of timelines representing an envelope of valid temporal behaviours of state variables. An executive system can carry out timelines by temporally instantiating the associated sequences of values, called *tokens*. Namely, an executive decides the exact *start time* of tokens composing the timelines in the plan. The actual tokens execution may not be fully controlled by the executive which must dynamically adapt the plan according to the *feedbacks* received during execution.

As a concrete planning and scheduling tool, we consider a timeline-based software framework, called PLATINUm (Umbrico et al. 2017, 2018). PLATINUm complies with the formal characterization of the timeline-based approach proposed in (Cialdea Mayer, Orlandini, and Umbrico 2016) which takes into account *temporal uncertainty*. This is crucial to capture the temporal unpredictability entailed by the presence of human operators working besides collaborative robots. Indeed, PLATINUm has been successfully applied in real-world manufacturing scenarios (Pellegrinelli et al. 2017) effectively and safely controlling a robot while collaborating with a human worker.

A Tool fostEriNg Ai plaNning in roboTics

To endow a cobot with the ability of dynamically choosing the proper task to be executed, a planning software should be integrated in the cobot control architecture. A suitable task planning model must be defined to capture the significant elements of cooperative operations (e.g., assembly and pick-and-place operations) as well as the desired operational requirements.

This paper proposes a novel software tool, called "Tool fostEriNg Ai plaNning in roboTics" (TENANT), addressing the needs of Domain Experts to set goals, defining tasks and set operational constraints notwithstanding the intrinsic complexity required at planning and robotics level. TENANT is a general purpose tool that can be deployed for addressing multiple applications/domains and can be easily integrated with other knowledge engineering tools such as, e.g., ROS-TiPlEx (Viola et al. 2019) and Planning and Scheduling software framework, e.g., PLATINUm (Umbrico et al. 2017). The proposed methodology behind TENANT can be described as follows: (i) a domain expert defines a collaborative process in terms of tasks, operational requirements and possible *assignments* (some tasks should be performed by the human only, some by the robot only and some by both); (ii) a planning model *template* for the task planning specification is defined by a planning expert to model generic tasks and operational requirements as an abstraction of task planning (e.g., a set of state variables and temporal constraints) and; (iii) an automatic tool *instantiates* the template of the task planning specification into a concrete planning model according to the described process.

TENANT implements such methodology constituting an automatic tool to encode the given production description in a fully instantiated (timeline-based) planning specification. An automated planning software, like e.g., PLATINUm, is then able to use such specification as input, control the cobot and guarantee the achievement of the production objective (defined by a domain expert) producing as output a sequence of cooperative tasks that, if properly dispatched and correctly executed by the robot, will ensure the achievement of the production goals. Without such automatic tool, this process requires a non-trivial effort by domain experts and planning engineers to build a valid and correct planning model properly representing the production process and the tasks a human and/or a robot can perform.

In this sense, TENANT defines a shared modelling process supporting both domain experts and planning engineers during the definition of the collaborative process, guiding the planning specification generation process and fostering production quality as well as improving its productivity. Three main steps are supported to facilitate intelligent human-robot coordination through the deployment of task planning technologies that are: (i) *template definition*; (ii) *process and task definition* and; (iii) *planning model generation*.

The proposed framework is implemented as a web-based application providing domain experts with an interactive graphical environment to incrementally define process descriptions. Input information is persistently stored into a local data-base for successive refinement and updates of process descriptions. As soon as a process description is complete, TENANT allows a domain expert to generate (as output) a complete planning specification, coherent with the

given process description. A *planning expert* can then use the obtained model as input for a planning and scheduling software.

TENANT promotes process definition reusability and thus facilitates task planning model adaptation in case of changes in production needs (tasks and/or operational requirements). It offers an access point to task planning technologies for domain experts, presenting a specific vision on the overall system and allowing them to define information closer to their expertise. In this way, domain experts and planning experts do not need to build strong cross-competencies or to have long iterative interaction to build a shared model. The use of structured information and modeling processes indeed prevent/limit misunderstandings and errors by fostering *knowledge sharing*.

Planning Template Definition

Firstly, a planning model template is defined by a planning engineers to model the behaviours of all the invariant components of a domain. In this way, a generic planning template is obtained and ready to be adapted to any compatible production process, instantiating process and tasks according to the type and the ordering of the tasks. The template in depicted Figure 2 is an example of a planning model for the considered collaborative process. The model is structured according to a hierarchical approach (supervision, coordination and implementation levels) that are pursued also in the subsequent steps.

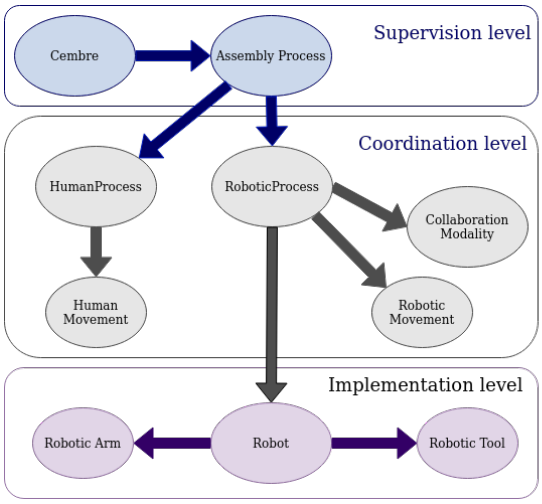


Figure 2: An example of template of a (timeline-based) planning model specification showing the set of state variables (circles) and synchronizations (arrows).

Following a hierarchical approach (Pellegrinelli et al. 2017), the components of the timeline-based planning model consist of state variables and synchronization rules characterizing possible decomposition of tasks and behaviors of the human and the robot. The template specifically considers three hierarchical levels, shown in Fig. 2. The *Supervision* level models the assembly process considering a state variable representing the general production process (e.g., the assembly of workpieces on a pallet), which is the general goal to be accomplished, and another state variable representing the sequence of high level tasks needed to realize that process. In order to complete the process (i.e. to accomplish the goal) all its tasks must be executed and, similarly, to complete each task all the subtasks in which it is decomposed must be completed. Each subtask is actually an operation that the robot or the human operator have to perform.

The *Coordination* level models the behaviours of both the human operator and the collaborative robot through two state variables referring to the subtask they can perform (i.e. a manipulating subtask or a pick and place) and their movements. All the values of the state variables related to the human operator are considered as uncontrollable (Cialdea Mayer, Orlandini, and Umbrico 2016) in order to capture "unpredictable" (temporal) dynamics. This information is crucial to synthesize reliable robot behaviors and effective human-robot collaborations (Pellegrinelli et al. 2017).

This level describes the "logical operations" (i.e., tasks) and constraints that characterize a collaborative process. Different tasks entail different constraints the human and the robot should fulfill in order to correctly carry out production processes. For example manipulation tasks require that the operators stand in front of the workpiece (i.e., the variable that models their movements must be in that position throughout the task). Pick and place tasks instead require the operators to position themselves in front of the target workpiece, pick it up, move, and place the workpiece down.

In addition, the Coordination level models the interaction between the human operator and the collaborative robot because a different collaborative modality corresponds to different constraints in the process. According to (Helms, Schraft, and Hagele 2002), four different degrees of interaction between the human operator and the robot are considered: Independent, the human operator and the robot perform different tasks on different workpieces without collaboration; Synchronous, the human operator and the robot have to complete sequential different tasks on the same workpiece, i.e. they operate consecutively without physical contact; Simultaneous, the human operator and the robot perform distinct tasks on the same workpiece at the same time, still without physical contact. Supportive, the human operator and the robot perform the same task on a single workpiece, i.e. they work cooperatively and simultaneously on the same task.

Tasks of the coordination level should be further specified in order to be effectively executed by a robot. To this aim, the Implementation level models the specific physical constraints and skills that allow the robot to perform all the operations needed to accomplish tasks. These constraints are modeled by means of three state variables: (i) one modelling the configuration of the robot; (ii) one the robotic arm and; (iii) the last one the gripper. During a manipulation subtasks for example the robot takes the manipulation configuration, which implies the lowering of the arm. During pick and place tasks, instead, it first assumes the pick configuration, which implies the arm lowered and the closing of the gripper, and then the place configuration, which implies

again the lowering of the arm, which had been raised during the movement, and the opening of the gripper. During the movement in fact, the robotic arm takes the idle configuration. This hierarchical representation clearly shows that the assembly process for each product differs only in the sequence of tasks and subtasks, i.e. the Supervision level.

User Interface for Process and Tasks Defintion

TENANT provides a Graphical User Interface (GUI) to domain experts so that they can easily specify the assembly processes and tasks as well as generate a coherent planning specification. We here show the structure of the developed GUI following the proposed modeling methodology.

Home Page On the first page, shown in Fig. 3, the user has first to select the number of positions on the worktable. Subsequently, the domain expert can define the assembly process by inserting the tasks, one at a time, through the input form.

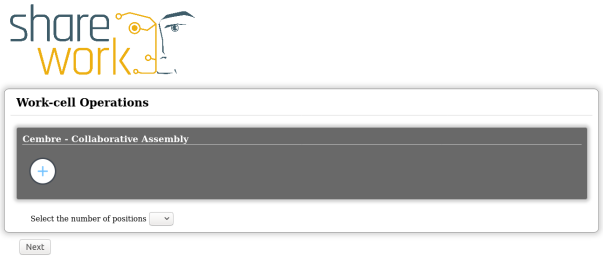


Figure 3: The home page of TENANT GUI to allow domain experts defining process and tasks.

For each task, the domain expert can initially define the task's name and its collaborative modality. Then, a sequence of subtasks can be defined to build its actual composition. For each subtask, the user has to specify its type (e.g., object manipulation or pick and place), which agent is supposed to perform it (i.e., robot, human, both, or no preference) and the involved positions in the working area.

Furthermore, within this domain, some constraints have to be taken into account when defining a subtask. For instance, a pick and place subtask can only be performed in Independent or Synchronous collaboration modality. On the other hand, a task in Simultaneous or Supportive collaboration modality has to be performed by both operators at the same time.

Fig. 4 shows an example of an almost completed assembly process consisting of four tasks.

Temporal constraints Once finished entering the sequence of tasks, the user can proceed and set the temporal constraints among the defined tasks.

Through a second view, shown in Fig. 5, the user can select the desired constraints and then confirm them, or delete them in case of mistakes.

Confirmation Page Finally, the user can save the DDL file by clicking the "Save" button that presents the user with a popup window to save the file. After successful saving, a

Figure 4: The GUI showing an intermediate definition of a process (Collaborative Assembly) with some tasks and subtasks characterized with specific features.

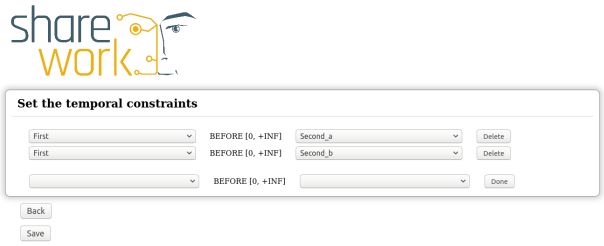


Figure 5

new view is loaded, as shown in Fig. 6. Through this page, the user can also download the PDL file, which is common to all domains created with TENANT.

Using these two files as PLATINUm inputs, the user can then generate valid plans for the assembly process.

Encoding Process and Implementation Details

TENANT is implemented in Python. The GUI allows the definition of the sequence of tasks, subtasks and the temporal contraints. This information are locally stored and the module produce as output a DDL file representing the do-

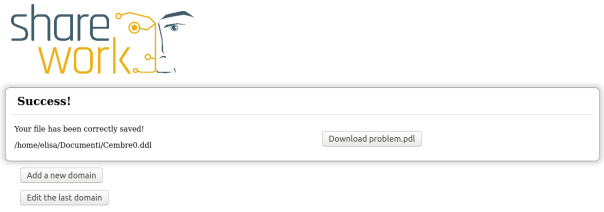


Figure 6

main specifications, i.e. a domain description in the format required by PLATINUm. The web-based GUI was developed using Flask, Jinja2 and Jquery. The communication between the GUI and the tool is achieved through JSON formatted messages.

In order to generate the DDL file, TENANT iterates over the list of tasks defined by the user and adds each element as a value of *Assembly Process* i.e. the Supervision level state variable of the model presented above which represents the sequence of high level tasks. As already mentioned, the state variables of the below levels are the same for all the possible processes because they're defined within the template.

Then the synchronization rules for the two state variables of the Supervision level are generated.

For most of the values, a conversion to DDL language is implemented and the new information is merged with the template presented above.

The synchronization rules for *Cembre* (i.e. the Supervision level state variable which represents the general production process) contains the sequence of tasks and the temporal contraints between them.

The syncronization rules for *Assembly Process* define, for each task, the sequence of subtasks needed to realize it, specifying the contraints due to its collaborative modality. The Supportive collaboration modality implies that for each subtask two manipulation operations are simultaneously performed, one by the robot and the other by the human operator. Even for Simultaneous modality for each subtask two operations must be performed, but there are no particular time constraints between them. The Synchronous collaboration modality, instead, implies that the subtasks must be performed in a specific order, therefore temporal constraints of precedence are introduced among the subtasks of the task. Lastly, the Independent modality does not require any particular constraints.

In the latter two collaborative modalities, it is possible to choose the operator as "Indifferent". In this case, TENANT does not generate the synchronization rule directly but it will take an intermediate step. This is because if the operator is "Indifferent", the choice of assigning the (sub)task to the human operator or to the robot is left to the planner, thus, the domain in the DDL file must contain both the synchronization rules. Therefore, for each task containing at least one subtask assigned to "Indifferent", a list containing all possible combinations of the operators is created. Iterating on this list, synchronization rules for the task will be added several times, replacing each time the "Indifferent" operator with the corresponding list item.

Experiments and discussion

Although TENANT is a general purpose tool, it has been validated in an industrial collaborative scenario derived from a EU-funded research project demonstrating its effectiveness in generating automated planning models to control a collaborative robot.

Specifically, to assess the feasibilty and effectiveness of our approach, we tested TENANTfor creating several instances of the same domain: for the tests have been considered instances of the process with an increasing number of tasks (3-5-7), each composed of 2 manipulation subtasks, all with the same collaborative modality and with less and less rigid temporal ordering constraints in the sequence of tasks (100% full tasks ordering - 50% half of tasks ordered - 0% no ordered task). We also considered tasks with different collaborative modalities, i.e., Independent, Synchronous, Simultaneous and Supportive, entailing a decreasing complexity (Independent tasks do not entail any ordering, while Supportive tasks entail full temporal overlap). It is worth underscoring that domains with no ordered tasks are the more complex for task planning as the planner is supposed to choose with the maximum branching factor in the search space. A full tasks order is the simpler case as the task planner is only to choose tasks assignment (i.e., either to human or robot).

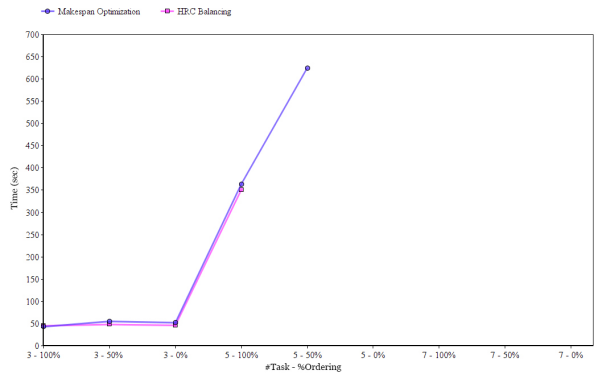
A total of 36 instances were obtained, each of which was tested 3 times with a timeout of 20 minutes and 2 different solution search strategies for PLATINUm: `HRCBalancing`, i.e., trying to balance as much as possible the number of tasks assigned to the two agents (main priority is balancing human and robot effort), and `MakespanOptimization`, i.e., trying to minimize the makespan (priority is increasing the throughput).

The results presented in Tables 1 and 2 and in Figure 7 represent the average planning costs (i.e., timings for generating task plans) of these experiments. The cost for planning specification resulted to be negligible (few milliseconds). The workstation used for the tests is an ASUS Vivobook S15 with an Intel core i5-8250u processor (1.6 Ghz) and 8 GB of ram with Ubuntu 18.04.5 (64 bit) operating system. However, TENANT can be used on both Linux OS and Windows OS, by downloading the repository available on GitHub [2]. In order to use TENANT, it is also necessary to install Flask. TENANT can be started by running "app.py" that will allow the GUI to be accessible to "localhost:5000" on any web browser.

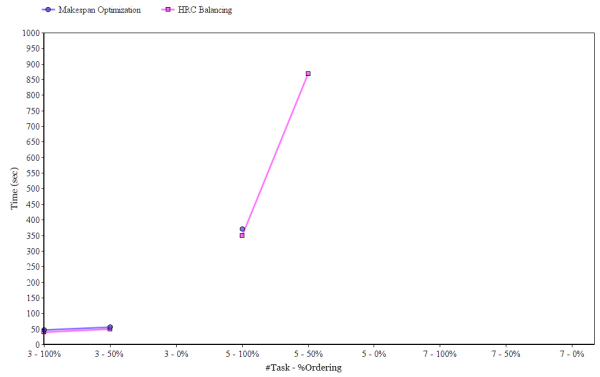
All the generated models have been proved with PLATINUm. The generated plans have proved the validity of the model highlighting how different collaborative modalities correspond to different complexities, and therefore different times of resolution, particularly for the more difficult instances. As can be seen from Tables, and as was expected, the most difficult instances (i.e. those for which the planner was not always able to find a plan within the time limit) are those with the greatest number of tasks and the least rigid ordering. This is because in these domains the search space is the greatest. In addition, the tests showed that the complexity introduced by a less rigid ordering is greater than that introduced by an increase in the number of tasks. Thus, processes in Synchronous modality are simplest because they include more ordering time constraints between their subtasks and reduce the search space, which allows valid plans to be obtained in a shorter time.

The difference between strategies was evident in the instances with least rigid collaboration modalities (i.e. Synchronous and Independent) in which it was possible to

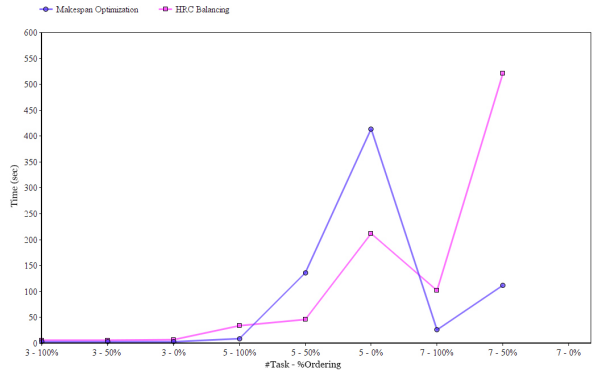
[2] https://github.com/Berenice02/InterfacciaGeneratoreDomini.git



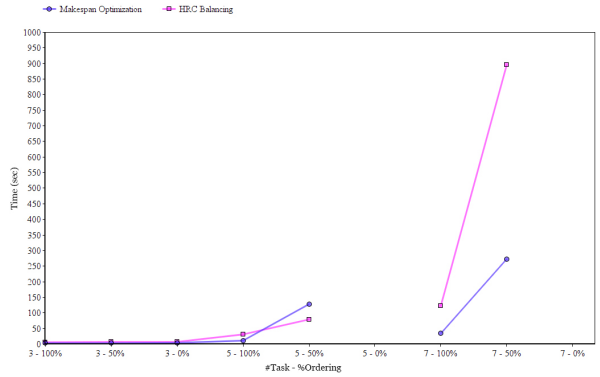
(a)



(b)



(c)



(d)

Figure 7: Comparisons of strategies divided by collaborative modality (a) Supportive (b) Simultaneous (c) Synchronous (d) Independent.

choose "Indifferent" as operator, leaving decision-making capacity to the planner. With *HRCBalancing*, a perfectly balanced division of tasks between the two operators was obtained.

Conclusions

This paper presented a novel software tool, called TENANT (Tool fostEriNg Ai plaNning in roboTics) that facilitates domain experts in defining goals, tasks and a set operational constraints enabling the automatic generation of A.I. models for robot control. TENANT has been validated in an industrial collaborative scenario derived from a EU-funded research project demonstrating its effectiveness in generating automated planning models to control a collaborative robot. As future works, we plan to integrate TENANT in ROS-TiPlEx (Viola et al. 2019) to provide a comprehensive knowledge engineering tool for fostering A.I. and Robotics solutions. Also, we plan to perform some analysis to investigate usability, engagement and cognitive workload for domain experts. Our long term goal is to connect this tool also to suitable ontologies (see, e.g., (Umbrico, Orlandini, and Cesta 2020)) to build models over well designed and formally defined set of concepts for HRC.

Acknowledgement

Authors are partially supported by the European Commission within the Sharework project (H2020 Factories of the Future GA No. 820807).

References

Barreiro, J.; Boyce, M.; Do, M.; Frank, J.; Iatauro, M.; Kichkaylo, T.; Morris, P.; Ong, J.; Remolina, E.; Smith, T.; and Smith, D. 2012. EUROPA: A Platform for AI Planning, Scheduling, Constraint Programming, and Optimization. In *ICKEPS 2012: the 4th Int. Competition on Knowledge Engineering for Planning and Scheduling*.

Bohren, J.; and Cousins, S. 2010. The SMACH High-Level Executive [ROS News]. *IEEE Robotics Automation Magazine* 17(4): 18–20.

Cashmore, M.; Fox, M.; Larkworthy, T.; Long, D.; and Magazzeni, D. 2014. AUV mission control via temporal planning. In *Proceedings - IEEE International Conference on Robotics and Automation*, 6535–6541. Institute of Electrical and Electronics Engineers Inc. ISSN 10504729. doi: 10.1109/ICRA.2014.6907823.

Cashmore, M.; Fox, M.; Long, D.; Magazzeni, D.; Ridder, B.; Carrera, A.; Palomeras, N.; Hurtos, N.; and Carreras, M. 2015. Rosplan: Planning in the robot operating system. In *Proceedings of ICAPS*, volume 2015-January, 333–341. AAAI press. ISBN 9781577357315. ISSN 23340835.

Cesta, A.; Cortellessa, G.; Fratini, S.; and Oddi, A. 2011. MRSPOCK: Steps in Developing an End-to-End Space Application. *Computational Intelligence* 27(1).

Chanel, C. P. C.; Lesire, C.; and Teichteil-Königsbuch, F. 2014. A Robotic Execution Framework for Online Probabilistic (Re)Planning. In *Proceedings of ICAPS*. AAAI press.

Table 1: Results of `MakespanOptimization` strategy for: (a) Supportive; (b) Simultaneous; (c) Synchronous and; (d) Independent

#task	%constraints	makespan	H%R	time (sec)
3	100%	31	50% - 50%	43 s
3	50%	31	50% - 50%	55 s
3	0%	31	50% - 50%	52 s
5	100%	51	50% - 50%	364 s
5	50%	51	50% - 50%	625 s
5	0%			TIMEOUT
7	100%			TIMEOUT
7	50%			TIMEOUT
7	0%			TIMEOUT

(a)

#task	%constraints	makespan	H%R	time (sec)
3	100%	31	50% - 50%	47 s
3	50%	31	50% - 50%	56 s
3	0%			TIMEOUT
5	100%	51	50% - 50%	371 s
5	50%			TIMEOUT
5	0%			TIMEOUT
7	100%			TIMEOUT
7	50%			TIMEOUT
7	0%			TIMEOUT

(b)

#task	%constraints	makespan	H%R	time (sec)
3	100%	31	100% - 0%	3 s
3	50%	31	100% - 0%	3 s
3	0%	31	100% - 0%	3 s
5	100%	51	100% - 0%	9 s
5	50%	51	100% - 0%	136 s
5	0%	51	100% - 0%	41 s
7	100%	71	100% - 0%	26 s
7	50%	71	100% - 0%	112 s
7	0%			TIMEOUT

(c)

#task	%constraints	makespan	H%R	time (sec)
3	100%	31	100% - 0%	3 s
3	50%	31	100% - 0%	3 s
3	0%	31	100% - 0%	4 s
5	100%	51	100% - 0%	11 s
5	50%	51	100% - 0%	128 s
5	0%			TIMEOUT
7	100%	71	100% - 0%	34 s
7	50%	71	100% - 0%	273 s
7	0%			TIMEOUT

(d)

Table 2: Results of `HRCBalancing` strategy for: (a) Supportive; (b) Simultaneous; (c) Synchronous and; (d) Independent

#task	%constraints	makespan	H%R	time (sec)
3	100%	31	50% - 50%	45 s
3	50%	31	50% - 50%	48 s
3	0%	31	50% - 50%	46 s
5	100%	51	50% - 50%	351 s
5	50%			TIMEOUT
5	0%			TIMEOUT
7	100%			TIMEOUT
7	50%			TIMEOUT
7	0%			TIMEOUT

(a)

#task	%constraints	makespan	H%R	time (sec)
3	100%	31	50% - 50%	40 s
3	50%	31	50% - 50%	50 s
3	0%			TIMEOUT
5	100%	51	50% - 50%	349 s
5	50%	51	50% - 50%	869 s
5	0%			TIMEOUT
7	100%			TIMEOUT
7	50%			TIMEOUT
7	0%			TIMEOUT

(b)

#task	%constraints	makespan	H%R	time (sec)
3	100%	16	50% - 50%	6 s
3	50%	16	50% - 50%	6 s
3	0%	16	50% - 50%	7 s
5	100%	26	50% - 50%	34 s
5	50%	26	50% - 50%	46 s
5	0%	26	50% - 50%	212 s
7	100%	36	50% - 50%	102 s
7	50%	36	50% - 50%	521 s
7	0%			TIMEOUT

(c)

#task	%constraints	makespan	H%R	time (sec)
3	100%	16	50% - 50%	6 s
3	50%	16	50% - 50%	7 s
3	0%	16	50% - 50%	7 s
5	100%	26	50% - 50%	31 s
5	50%	26	50% - 50%	79 s
5	0%			TIMEOUT
7	100%	36	50% - 50%	123 s
7	50%	36	50% - 50%	896 s
7	0%			TIMEOUT

(d)

Cialdea Mayer, M.; Orlandini, A.; and Umbrico, A. 2016. Planning and execution with flexible timelines: a formal account. *Acta Inf.* 53(6-8): 649–680.

Helms, E.; Schraft, R. D.; and Hagele, M. 2002. Rob@work: Robot Assistant in Industrial Environments. In *Proceedings. 11th IEEE International Workshop on Robot and Human Interactive Communication*, 399 – 404. doi:10.1109/ROMAN.2002.1045655.

Lacerda, B.; Parker, D.; and Hawes, N. 2014. Optimal and dynamic planning for Markov decision processes with co-safe LTL specifications. In *IEEE International Conference on Intelligent Robots and Systems*, 1511–1516. Institute of Electrical and Electronics Engineers Inc. ISBN 9781479969340. ISSN 21530858.

Muscettola, N. 1994. HSTS: Integrating Planning and Scheduling. In Zweben, M. and Fox, M.S., ed., *Intelligent Scheduling*. Morgan Kauffmann.

Orlandini, A.; Bernardi, G.; Cesta, A.; and Finzi, A. 2014. Planning meets verification and validation in a knowledge engineering environment. *Intelligenza Artificiale* 8(1): 87–100.

Pellegrinelli, S.; Orlandini, A.; Pedrocchi, N.; Umbrico, A.; and Tolio, T. 2017. Motion planning and scheduling for human and industrial-robot collaboration. *CIRP Annals - Manufacturing Technology* 66: 1–4. ISSN 0007-8506.

Quigley, M.; Gerkey, B. P.; Conley, K.; Faust, J.; Foote, T.; Leibs, J.; Berger, E.; Wheeler, R.; and Ng, A. 2009. ROS : an open-source Robot Operating System. In *ICRA Workshop on Open Source Software*.

Rajan, K.; and Saffiotti, A. 2017. Towards a science of integrated AI and Robotics. *Artificial Intelligence* 247: 1 – 9. ISSN 0004-3702. doi:https://doi.org/10.1016/j.artint.2017.03.003.

Sanelli, V.; Cashmore, M.; Magazzeni, D.; and Iocchi, L. 2017. Short-Term Human-Robot Interaction through Conditional Planning and Execution. In *Proceedings of ICAPS*. AAAI press.

Tsarouchi, P.; Makris, S.; and Chryssolouris, G. 2016. Human–robot interaction review and challenges on task planning and programming. *International Journal of Computer Integrated Manufacturing* 29(8): 916–931. doi:10.1080/0951192X.2015.1130251. URL https://doi.org/10.1080/0951192X.2015.1130251.

Umbrico, A.; Cesta, A.; Cialdea Mayer, M.; and Orlandini, A. 2017. PLATINUm: A New Framework for Planning and Acting. In *AI*IA 2017 Advances in Artificial Intelligence*, 498–512. ISBN 978-3-319-70169-1.

Umbrico, A.; Cesta, A.; Cialdea Mayer, M.; and Orlandini, A. 2018. Integrating Resource Management and Timeline-based Planning. In *The 28th International Conference on Automated Planning and Scheduling (ICAPS)*.

Umbrico, A.; Orlandini, A.; and Cesta, A. 2020. An Ontology for Human-Robot Collaboration. *Procedia CIRP* 93: 1097 – 1102. ISSN 2212-8271.

Viola, C. L.; Orlandini, A.; Umbrico, A.; and Cesta, A. 2019. ROS-TiPlEx: How to make experts in A.I. Planning and Robotics talk together and be happy. In *2019 28th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, 1–6.

Ziparo, V. A.; Iocchi, L.; Lima, P. U.; Nardi, D.; and Palamara, P. F. 2011. Petri Net Plans. *Autonomous Agents and Multi-Agent Systems* 23(3): 344–383. ISSN 1573-7454.