Simplifying the A.I. Planning modeling 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— **For an effective deployment in manufacturing, Collaborative Robots should be capable of adapting their behavior to the state of the environment and to keep the user safe and engaged during the interaction. Artificial Intelligence (AI) enables robots to autonomously operate understanding the environment, planning their tasks and acting to achieve some given goals. However, the effective deployment of AI technologies in real industrial environments is not straightforward. There is a need for *engineering tools* facilitating communication and interaction between AI engineers and Domain experts. This paper proposes a novel software tool, called TENANT (Tool fostEriNg Ai plaNning in roboTics) whose aim is to facilitate the use of AI planning technologies by providing domain experts like e.g., production engineers, with a graphical software framework to synthesize AI planning models abstracting from syntactic features of the underlying planning formalism.**

I. 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 during 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 [1]. Several approaches aim to achieve robust action selection via Artificial Intelligence (AI) Planning, e.g., [2], [3], [4] or robust execution via some form of finite state machine (FSM), e.g., [5], [6]. The development of tools that facilitate the integration (and interaction) of AI planning and robotics entails different skills that are all necessary to effectively address the underlying variety of control issues, spanning from low-level control to decisional (and behavioral) autonomy [7]. 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 [2] has been proposed as a unique integrated solution for PDDL-based planners to be smoothly deployed in ROS architectures [8] also with planning techniques specifically used for human-robot interaction [9]. 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 left alone in managing a knowledge planning modelling. Several timeline-based planning frameworks such as, e.g., EUROPA [10] and APSI-KEEN [11], 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 [7] to facilitate the interaction between AI and Robotics experts. 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 a robotic system, while other actors have responsibility on more specific aspects, i.e., a *planning expert*, with models to provide robust A.I. planning features, and a *robotic expert*, responsible for 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 [7] and Planning and Scheduling software framework, e.g., PLATINUm [12]. TENANT specifically focuses on Human-Robot Collaboration and allows domain experts to specify *collaborative models* and thus describe *tasks* needed to achieve desired (productive) goals. Tasks can be either compound or simple (i.e., further structured in other sub-tasks) and are characterized by specific configurations/properties relevant for their execution, like e.g., collaborative modalities, or assignment preferences specifying who is actually in charge of performing a certain (sub)task. Furthermore, TENANT supports the definition of *operational constraints* such as, e.g., temporal synchronizations or precedence constraints in order to characterize the correct execution of the resulting production/control flow. Given a complete (and correct) collaborative model, TENANT can automatically generate a suitable *planning model* that can be used to feed a Planning & Scheduling system thus enabling intelligent (collaborative) robot behaviours.

The developed tool has been validated on a concrete (and realistic) HRC production process derived from an EU-funded research project called Sharework[1]. The validation shows the feasibility of the tool and the underlying engi-

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

neering methodology. TENANT indeed supports the correct design of the considered collaborative process and the automatic synthesis of a valid (timeline-based) planning specification, completely abstracting from the syntactic features of the underlying planning formalism.

The paper is organized as follows. Sec. II gives some backgrounds on AI Task Planning. TENANT is presented in Sec. III. Then, a HRC scenario is presented in Sec. IV to demonstrate how TENANT can be used to easily generate several models (Sec. V). Some conclusions end the paper.

II. Automated Planning & Scheduling for HRC

The development of cobots capable of autonomously performing complex tasks while interacting with a dynamic production environment and dealing with (unpredictable) worker behaviors entails the integration of different technologies. From a planning point of view, efficiency and safety in HRC scenarios are related to the solution of different intertwined problems. First, the robot control system should be able to compute the sequence of 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. In this regard, AI Planning and its integration with Motion Planning and Robot Control have shown promising results in terms of improved robot flexibility [13], [14] and efficiency of resulting collaborative processes [15], [16], [17]. In this work we rely on the *timeline-based paradigm* which represents a well established AI planning technology. This is a particular temporal planning paradigm which has been introduced in early 90s [18] taking inspiration from the classical Control Theory, and successfully applied in many real-world scenarios [18], [19]. 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 consists 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* representing 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 duration of 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 ("global") constraints are usually necessary to coordinate different state variables and realize complex functionalities or achieve complex goals. A set of rules, called *synchronizations*, model such "global" constraints.

A task planner synthesizes a set of timelines representing envelopes of temporal behaviors of state variables that satisfy local and global constraints of the planning specification. An executive system carries out timelines by deciding the *start time* of the valued temporal intervals (*tokens*) composing the timelines of a plan. Tokens of a timeline may not be fully controlled by the executive which must dynamically adapt the plan according to the *feedback* received during execution and thus the observed state of the environment.

As a concrete planning and scheduling tool, we consider a timeline-based software framework, called PLATINUm [12]. PLATINUm complies with the formal characterization of the timeline-based approach proposed in [20] which takes into account also *temporal uncertainty*. This is crucial to capture the temporal unpredictability entailed by the presence of human operators working besides collaborative robots.

III. 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. The 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.

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).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 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 design and 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 database 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 generated 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 of 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. A structured modeling processes indeed prevent/limit misunderstandings and errors and facilitates *knowledge sharing*.

A. Planning Template Definition

A planning model template is defined by a planning engineer 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 depicted in Figure 1 is an example of a planning model for a generic collaborative process to be performed by a human operator and a cobot.

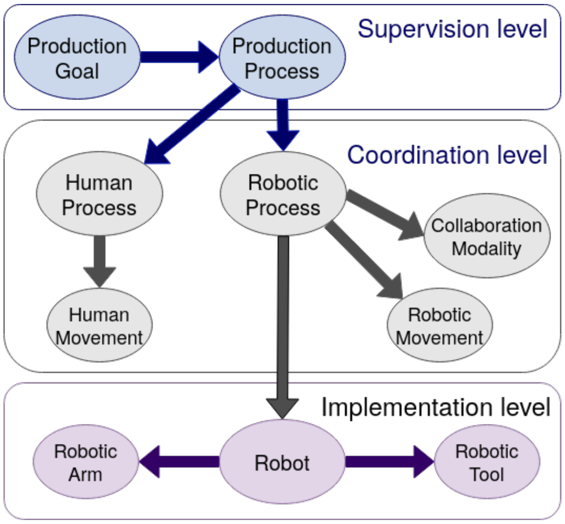


Fig. 1. An template of a (timeline-based) planning model specification showing the set of state variables (circles) and synchronizations (arrows).

The template is structured according to a hierarchical methodology decomposing high-level production goals into increasingly simple tasks [21] and considering three main abstraction levels [22], [23]: (i) supervision; (ii) coordination; (iii) implementation. Broadly speaking, the resulting timeline-based model consists of state variables characterizing these hierarchical levels and synchronization rules characterizing possible decomposition of “complex tasks” into “simpler tasks” until defining the concrete behaviors of the human and the robot.

The *Supervision* level models the general production process. A state variable represents the general goals to be accomplished (e.g., an assembly operation) while other state variables describe production processes in terms of high-level tasks needed to achieve associated goals. 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 a concrete operation the robot or the worker shall perform.

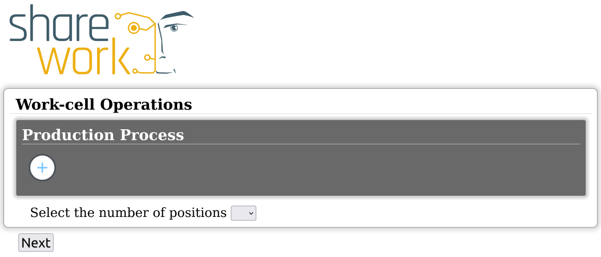
The *Coordination* level models the behaviours of both the human operator and the collaborative robot through two state variables referring to the subtask they can perform (e.g., a manipulation subtask or a pick and place) and their movements. In the planning model, all the values of state variables related to the human operator are considered as uncontrollable [20] in order to capture her “unpredictable” (temporal) dynamics. This information is crucial to synthesize reliable robot behaviours and effective human-robot collaborations [15]. In addition, the Coordination level models the interaction between the human operator and the cobot through *collaboration modalities*. According to [24], four types of (physical) interaction 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 operations. Implementation constraints are modeled by means of three state variables modelling: (i) the configuration of the robot; (ii) the robotic arm; (iii) the working tool (e.g., a gripper). During a manipulation subtasks for example the robot assumes the manipulation pose, which implies the lowering of the arm. During pick and place tasks, instead, it first assumes the pick pose, 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 assumes the idle pose. This hierarchical representation clearly shows that an assembly process for each product differs only in the sequence of tasks and subtasks, i.e. the Supervision level.

B. User Interface for Process and Tasks Definition

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.

1) Home Page: On a first page of the GUI, shown in Fig. 2, the user can first select the number of positions on the worktable. Subsequently, the domain expert can define the production process by inserting the tasks, one at a time, through the input form.



Work-cell Operations

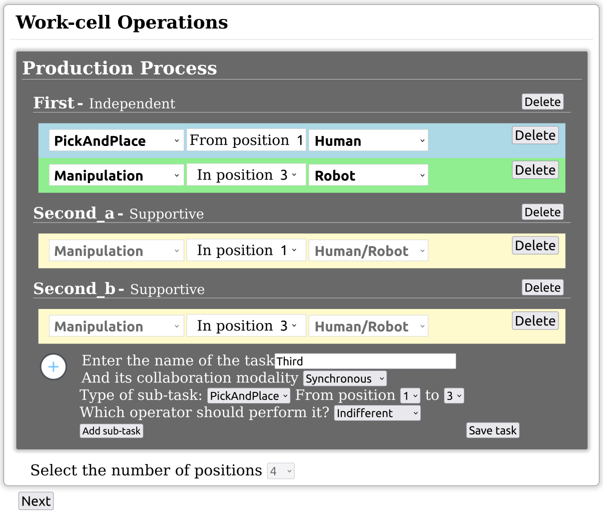
Production Process

Select the number of positions

Next

Fig. 2. The home page of TENANT GUI

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 must specify its type (e.g., object manipulation or pick and place), which agent is supposed to perform it (i.e., robot, human or no preference) and the involved positions in the working area.



Work-cell Operations

Production Process

First - Independent — Delete

PickAndPlace | From position 1 | Human — Delete

Manipulation | In position 3 | Robot — Delete

Second_a - Supportive — Delete

Manipulation | In position 1 | Human/Robot — Delete

Second_b - Supportive — Delete

Manipulation | In position 3 | Human/Robot — Delete

Enter the name of the task Third
And its collaboration modality Synchronous
Type of sub-task: PickAndPlace From position 1 to 3
Which operator should perform it? Indifferent
Add sub-task — Save task

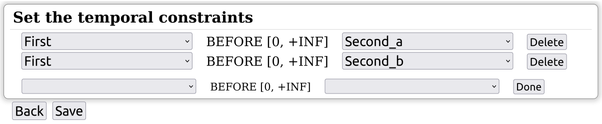
Select the number of positions 4

Next

Fig. 3. Intermediate definition of a process with tasks and subtasks

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 agents at the same time. Fig. 3 shows an example of an almost completed assembly process consisting of four tasks.

2) 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. 4, the user can select the desired constraints and then confirm them, or delete them in case of mistakes.



Set the temporal constraints

First | BEFORE [0, +INF] | Second_a | Delete
First | BEFORE [0, +INF] | Second_b | Delete
BEFORE [0, +INF] | Done

Back Save

Fig. 4. Definition of ordering constraints between tasks

3) Confirmation Page: Finally, the user can save the model as a domain description file (e.g., DDL used by PLATINUm) by clicking the ”Save” button. After successful saving, a new view is loaded. The user can also download a problem description file (e.g., PDL for PLATINUm), which sets the production goal defined with TENANT. Giving these two files as input to a planning tool, e.g., PLATINUm, the user can generate valid plans for the defined process.

C. Encoding Process and Implementation Details

TENANT is implemented in Python. The web-based GUI was developed using Flask, Jinja2 and Jquery. The communication between the GUI and the tool is achieved through JSON messages.

In order to generate the domain description file, TENANT iterates over the list of tasks defined by the user and adds each element as a value of *Production Process* i.e. the Supervision level state variable of the model presented in III-A which represents the sequence of high level tasks. As already mentioned, the state variables of the lower levels are the same for all the possible processes because they are defined within the template. Then the synchronization rules for the two state variables in the Supervision level are generated.

For most of the values, a conversion to a domain description language is implemented and the new information is merged with the template presented in Section III-A. The synchronization rules for *Production Goal* (i.e. the Supervision level state variable which represents the general production process) contains simpler tasks and related temporal constraints.

The synchronization rules for *Production Process* define, for each task, the sequence of subtasks needed to realize it, specifying the constraints 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 decision of assigning the (sub)task to the human operator or to the robot is left to the planner, thus, the domain description 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.

IV. An industrial HRC scenario

A real industrial HRC scenario is considered (See Figure 5) to validate TENANT. The working cell is constituted by a logistic station in which a cobot and a human operator cooperate to properly assembly parts to be machined on a device called pallet.

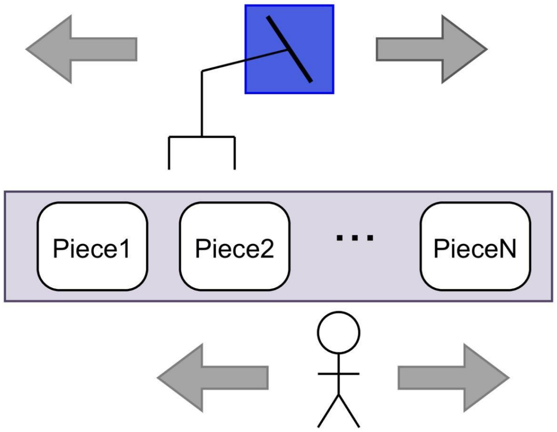


Fig. 5. A schematic description of the collaborative scenario. A Robot (on the upper part) and a Human (on the bottom) work together to pick and place some workpieces

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.

In this paper, the focus is on task assignment and human-robot coordination. TENANT is to support the definition of task planning models to endow cobots with human-aware task planning features. Therefore, 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.

V. Experiments and discussion

TENANT has been developed in Python and 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. 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.

We have used TENANT to create several instances of the described HRC domain generating the domain and problem description in DDL and PDL to work with PLATINUm. We have considered several versions of the process with an increasing number of tasks (3-5-7), each composed of: 2 manipulation subtasks; the same collaborative modality and; different required ordering constraints of the tasks (100% full tasks ordering - 50% half of tasks ordered - 0% no ordered task). We have 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). An initial model was defined with all the tasks set as ”Independent”. Then, leveraging TENANT ability to modify domains, we set different versions considering an increasing number of collaborative modalities (the more complex scenario being the one with all tasks requiring a ”Supportive” collaboration). Defining models by hand would have required an error-prone and long revision of planning specification. With TENANT, the planning models were generated after a simple revision of tasks specifications, i.e., by just modifying the collaboration modality required for each task. 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 fully ordered set of tasks order is the simpler case as the task planner must only assign tasks either to human or robot.

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

In this way, a total of 36 planning models have been defined and run with PLATINUm considering 2 different solution search strategies: priority to balancing human and robot effort and priority to increasing the throughput. The generated plans proved the validity of the generated models highlighting how different collaborative modalities correspond to different complexities, and therefore different times of resolution.[3] PLATINUm was able to find a plan for 64% of planning models, with an average planning time of 273 seconds (timeout at 1200 sec). As expected, the most difficult instances (i.e. those for which the planner was not always able to find a plan within the timeout) are those with the greatest number of tasks and the least rigid ordering. In addition, the tests showed that the complexity introduced by a less rigid ordering is greater than that introduced by a higher number of tasks. One interesting evidence concerns the use of TENANT as a tool to perform so-called *what-if* analyses and support the design of HRC processes. Its easiness-of-use and the integrated task planning engine would indeed provide production engineers with a powerful tool to compare alternative implementations of production processes and select the "structure" that best fit desired requirements.

VI. Conclusions

This paper presented TENANT, a novel software tool that facilitates domain experts in defining goals, tasks and operational constraints for the automatic generation of AI models for robot control. It has been validated in an industrial collaborative scenario derived from an research project demonstrating its effectiveness in generating automated planning models to control a cobot. As future works, we plan to integrate TENANT in ROS-TiPlEx [7] to provide a comprehensive knowledge engineering tool for fostering A.I. and Robotics solutions and perform an analysis with domain experts to assess its usability and the required cognitive workload. Connecting this tool to suitable ontologies (see, e.g., [25]) can further facilitate the definition of models over well designed and formally defined set of concepts for HRC.

Acknowledgement

Authors are partially supported by European Commission within Sharework project (H2020 DT-FOF-02-2018 GA No. 820807).

References

[1] K. Rajan and A. Saffiotti, "Towards a science of integrated ai and robotics," *Artificial Intelligence*, vol. 247, pp. 1 – 9, 2017.

[2] M. Cashmore, M. Fox, D. Long, D. Magazzeni, B. Ridder, A. Carrera, N. Palomeras, N. Hurtos, and M. Carreras, "Rosplan: Planning in the robot operating system," in *Proceedings of ICAPS*, vol. 2015-January. AAAI press, 2015, pp. 333–341.

[3] C. P. C. Chanel, C. Lesire, and F. Teichteil-Königsbuch, "A robotic execution framework for online probabilistic (re)planning," in *Proceedings of ICAPS*. AAAI press, 2014.

[4] B. Lacerda, D. Parker, and N. Hawes, "Optimal and dynamic planning for markov decision processes with co-safe ltl specifications," in *IEEE International Conference on Intelligent Robots and Systems*. Institute of Electrical and Electronics Engineers Inc., 2014, pp. 1511–1516.

[5] J. Bohren and S. Cousins, "The smach high-level executive [ros news]," *IEEE Robotics Automation Magazine*, vol. 17, no. 4, 2010.

[6] V. A. Ziparo, L. Iocchi, P. U. Lima, D. Nardi, and P. F. Palamara, "Petri net plans," *Autonomous Agents and Multi-Agent Systems*, vol. 23, no. 3, pp. 344–383, 2011.

[7] C. L. Viola, A. Orlandini, A. Umbrico, and A. Cesta, "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)*, 2019, pp. 1–6.

[8] M. Cashmore, M. Fox, T. Larkworthy, D. Long, and D. Magazzeni, "Auv mission control via temporal planning," in *Proceedings - IEEE International Conference on Robotics and Automation*. Institute of Electrical and Electronics Engineers Inc., 2014, pp. 6535–6541.

[9] V. Sanelli, M. Cashmore, D. Magazzeni, and L. Iocchi, "Short-term human-robot interaction through conditional planning and execution," in *Proceedings of ICAPS*. AAAI press, 2017.

[10] J. Barreiro, M. Boyce, M. Do, J. Frank, M. Iatauro, T. Kichkaylo, P. Morris, J. Ong, E. Remolina, T. Smith, and D. Smith, "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*, 2012.

[11] A. Orlandini, G. Bernardi, A. Cesta, and A. Finzi, "Planning meets verification and validation in a knowledge engineering environment," *Intelligenza Artificiale*, vol. 8, no. 1, pp. 87–100, 2014.

[12] A. Umbrico, A. Cesta, M. Cialdea Mayer, and A. Orlandini, "Platinum: A new framework for planning and acting," in *AI*IA 2017 Advances in Artificial Intelligence*, 2017, pp. 498–512.

[13] S. Srivastava, E. Fang, L. Riano, R. Chitnis, S. Russell, and P. Abbeel, "Combined task and motion planning through an extensible planner-independent interface layer," in *2014 IEEE International Conference on Robotics and Automation*, 2014, pp. 639–646.

[14] J. Wolfe, B. Marthi, and S. J. Russell, "Combined task and motion planning for mobile manipulation," in *Proceedings of the 20th International Conference on Automated Planning and Scheduling, ICAPS 2010, Toronto, Ontario, Canada, May 12-16, 2010*, 2010, pp. 254–258.

[15] S. Pellegrinelli, A. Orlandini, N. Pedrocchi, A. Umbrico, and T. Tolio, "Motion planning and scheduling for human and industrial-robot collaboration," *CIRP Annals - Manufacturing Technology*, vol. 66, pp. 1–4, 2017.

[16] C. Zhang and J. A. Shah, "Co-optimizing task and motion planning," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 2016-Novem. IEEE, oct 2016, pp. 4750–4756.

[17] M. Faroni, M. Beschi, S. Ghidini, N. Pedrocchi, A. Umbrico, A. Orlandini, and A. Cesta, "A layered control approach to human-aware task and motion planning for human-robot collaboration," in *2020 29th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, 2020, pp. 1204–1210.

[18] N. Muscettola, "HSTS: Integrating Planning and Scheduling," in *Intelligent Scheduling*, Zweben, M. and Fox, M.S., Ed. Morgan Kauffmann, 1994.

[19] A. Cesta, G. Cortellessa, S. Fratini, and A. Oddi, "MrSPOCK: Steps in Developing an End-to-End Space Application," *Computational Intelligence*, vol. 27, no. 1, 2011.

[20] M. Cialdea Mayer, A. Orlandini, and A. Umbrico, "Planning and execution with flexible timelines: a formal account," *Acta Inf.*, vol. 53, no. 6-8, pp. 649–680, 2016.

[21] J. T. C. Tan, F. Duan, Y. Zhang, and T. Arai, "Task decomposition of cell production assembly operation for man-machine collaboration by hta," in *2008 IEEE International Conference on Automation and Logistics*, 2008, pp. 1066–1071.

[22] A. Cesta, A. Orlandini, and A. Umbrico, "Fostering robust human-robot collaboration through ai task planning," *Procedia CIRP*, vol. 72, 2018, 51st CIRP Conference on Manufacturing Systems.

[23] A. Cesta, A. Orlandini, G. Bernardi, and A. Umbrico, "Towards a planning-based framework for symbiotic human-robot collaboration," in *2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2016, pp. 1–8.

[24] E. Helms, R. D. Schraft, and M. Hagele, "Rob@work: Robot assistant in industrial environments," in *Proceedings. 11th IEEE International Workshop on Robot and Human Interactive Communication*, 2002, pp. 399 – 404.

[25] A. Umbrico, A. Orlandini, and A. Cesta, "An ontology for human-robot collaboration," *Procedia CIRP*, vol. 93, pp. 1097 – 1102, 2020.

[3] The workstation used for the tests is endowed with an Intel core i5 (1.6 Ghz), 8GB RAM on Ubuntu 64-bit.