Modelling automated industrial experiments as incremental diverse AI planning problems

Antonio Lomuscio[1], Franci Rrapi[1], Pierfrancesco Esposito[1], Sofia Santilli[1], Alessandro Trapasso[1], Luca Iocchi[1], Fabio Patrizi[1] and Fabio Zonfrilli[2]

[1]*Sapienza University of Rome, Italy*
[2]*Procter&Gamble, Brussels, Belgium*

Abstract

In this paper, we describe an industrial use case, a planning-based solution and the use of different planning formalisms to model different aspects of the use case that results in different solutions with different performances. The paper shows that more sophisticated planning formalisms are necessary to improve performance, as well as the presence of other features that are desirable and that are not fully supported by the current planning technology.

Keywords

AI planning and scheduling, robot manipulation, human-robot interaction,

1. Introduction

AI Planning (or action/task planning) is a fundamental research area in AI [1] that is based on exploiting some knowledge about models of a dynamic system in order to extract plans (solutions) with some guarantee to achieve the desired goals. The research in the field is very active and has produced languages to represent planning domains and problems (e.g., PDDL [2])), as well as algorithms for solving the planning problems in many different contexts. Successful applications of planning techniques can be found in many diverse application domains. However, although most of the planning techniques are mature for industrial use, several obstacles hinder their adoption, thus preventing a large diffusion in industrial applications.

In this context, AIPlan4EU project[1] aims at improving usability and effectiveness of AI planning technology in relevant industrial use cases through a unified framework, libraries, and planning engines that allow practitioners to easily define and solve planning problems and to integrate solutions based on planning technology into the overall system.

In this paper, we describe an industrial scenario in Procter&Gamble in which robots are employed to support people in the development of automated quality tests, considering in particular testing laundry detergent soluble pouches (see Appendix A for details).

AIRO'22 Workshop
totototo96@gmail.com (A. Lomuscio); rrapi.franci@gmail.com (F. Rrapi); esposito.pierfrancesco@gmail.com (P. Esposito); sofiasantilli1998@gmail.com (S. Santilli); trapasso@diag.uniroma1.it (A. Trapasso); iocchi@diag.uniroma1.it (L. Iocchi); patrizi@diag.uniroma1.it (F. Patrizi); zonfrilli.f@pg.com (F. Zonfrilli)
0000-0002-0877-???? (L. Iocchi); 0000-0002-0877-???? (F. Patrizi)

[1]https://www.aiplan4eu-project.eu/

The main goal of this paper is to show how this industrial use case can generate diverse interesting AI planning problems and how the AIPlan4EU *unified planning framework* can be used for incremental development of planning-based solutions and for effective integration of such solutions in the overall industrial system.

2. Automated Planning

In this section, we describe diverse planning models for the considered industrial use case. The use of the Unified Planning (*UP*) framework[2] facilitates access to the planning technology and incremental development, in that it allows not only for enriching a domain with additional actions, fluents, etc., but also for switching from a variant of planning to another one at virtually no cost, i.e., without the need for rewriting a specification for each variant. For instance, one can easily take a problem originally intended to be solved by a deterministic planner and change it into a nondeterministic, or numeric or temporal one, without having to rewrite the entire input, but by simply making those changes needed to enrich the domain model. This is very important when real scenarios have to be modelled, since it allows for incrementally taking into account features of increasing complexity, with the advantage of controlling both model correctness and the impact of each feature on the solution.

Classical planning. A Classical Planning problem assumes perfect knowledge about the initial state, as well as full observability on all domain fluents. Solutions are represented as sequences of actions, whose sequential execution, from the initial state, guarantees reaching a goal state. In the UP framework, classical planning problems can be solved with planners like *Pyperplan* [3] and *Fast Downward* [4].

A basic classical planning problem has been defined to model the P&G use case. In particular, we consider basic manipulation actions to pick and place objects (i.e., pouches) on different measurement devices that perform quality tests. The plan is a sequence of robot pick-and-place actions that reaches a goal state in which all the pouches have been correctly tested.

The main limitations of this formulation are that: 1) it requires a full specification of the initial state, which may require *guessing*, when not available, the initial value of some variables, possibly resulting to plan failures and need for replanning at execution time; 2) the sequential execution of actions does not guarantee an optimal throughput, since sensors may take time to perform measurements, during which the robot remains idle, while it could operate on other pouches; 3) it does not support multiple robots to speed up the overall process; 4) it does not model and support humans that can be present in the environment and involved in the task. These limitations are addressed by extending the problem to other forms of planning, as discussed in the next paragraphs.

Contingent planning. In Contingent planning, the assumption of full knowledge about the initial state is dropped, and sensing actions are assumed available to acquire new knowledge at execution time. Contingent plans thus contain *if-then-else* structures allowing to choose the proper branch according to the value of a variable sensed at run-time (without the need for replanning). Support for contingent planning in the UP is under development.

[2] https://github.com/aiplan4eu/unified-planning/

The formulation of the P&G use case based on classical planning will be extended by adding sensing actions and incomplete knowledge about the initial state. For example, the number of pouches to be processed and their presence in the drawers is not known at planning time and must be captured during execution. Additional sensing actions are used to check the correct functionality of the devices. The resulting plans will be more robust (with respect to classical plan solutions), since the planner can insert sensing actions and conditional branches in the plan to cope with some sources of uncertainty.

Temporal planning. In temporal planning, states include information about the current absolute time and how far each active action has been running. In the UP, temporal planners, such as TAMER [5], introduce durative actions, timed effects and timed goals. Plans generated by temporal planners usually include parallel execution of actions.

In the P&G use case, we can transform the instantaneous actions representing movements of the robot and measuring from the devices into durative actions. In this way, a temporal planner can determine a plan including parallel execution of actions, allowing for solutions that minimize the overall execution time by performing robot pick-and-place actions of some pouch and measuring actions of some other pouch in parallel.

Multi-agent planning. In Multi-agent planning (MAP), two or more agents operate in the same environment, taking actions that can influence each other, as well as the environment. In the UP framework, we are considering at the moment *cooperative MAP*, in which all the agents share a common goal to be achieved and they all contribute only to such goal. *FMAP* planner[6] is integrated in the UP to generate multi-agent plans that can then be distributed for execution to all the agents in the team.

In the multi-agent extension of the P&G use case, several robots can simultaneously pick and place any pouch and use any measurement device. The plans generated in this scenario allow for a synchronization of the robots' actions in order to execute effective multi-robot behaviours to achieve the desired goals in a more efficient way with respect to the single-robot case.

Human-aware planning. Human-aware planning can be seen as multi-agent planning in which one or more agents are humans and humans are explicitly modelled in the domain. The main difference with respect to a multi-agent team formed by only artificial agents is that we cannot assume complete knowledge and complete control of human agents. In other words, humans cannot be modelled as artificial agents whose behaviour will be defined by a planner. Research in human-aware planning is still on-going and the integration of planners in UP is a future goal for the AIPlan4EU project.

In the P&G use case, humans are usually involved in the execution of experiments, since some operations cannot be performed by robots and some others must be carefully monitored by humans. Moreover, human operators can help the robot when some situation that is not solvable by the robot occurs (for example grasp failure which makes the pouch fall outside the robot's operation range). Finally, humans can ask the robot to explain its status, its actions and its goals and to change its behaviour. Synthesising robot plans (behaviours) that take into account the available and needed interactions with people would be an important factor not only to further increase overall performance and robustness in executing the task, but also to increase acceptability and trustworthiness of the use of AI Planning technology in the application scenario.

Planning features integration. The limitations observed in the classical planning formulation have been individually addressed by resorting to advanced forms of planning described above. However, the current technology is not yet able to address the combination of all the features required to faithfully capture all the relevant aspects of the domain of interest. Indeed, no currently available planning engine provides support for contingent, temporal, multi-agent, and human-aware planning at the same time.

3. Robot development

3.1. Plan execution and robot control

The robot used in the P&G laboratory is a Universal Robot UR5e manipulator with a Robotiq 2F_85 gripper (Fig. 1). ROS Melodic and *MoveIt* libraries have been used to implement the robotic actions and perceptions. Plan execution is performed with the *Petri Net Plans (PNP)* library[3]. Plans generated by the planners are converted into PNPs and executed by the ROS-based PNP executor. An interface between action and fluent symbols defined in the planning domain and their actual implementations has been implemented using the PLEXI interface[4].

Actions and fluents implemented for this use case are summarized in Appendix B, while the system architecture of the implemented solution is described in Appendix C.

Figure 1: Application scenario

3.2. Perception about the environment

The P&G scenario is composed by static objects (the measurement devices) and the pouches to be manipulated. Static objects are assigned to known positions, while the pouches have to be perceived by the on-board camera to determine their pose with respect to the robot. Pouch pose estimation has been implemented by using a custom-trained version of a YoloV5 Deep Neural Network[5] for pouch pose detection and pixel-to-real coordinate transformation to determine coordinates to be used in the grasping phase. Moreover, ArUco Markers and some textual labels that are relevant for the experiments are present in the drawers. ArUco marker detector[6] and Tesseract OCR[7] have been integrated to acquire information needed to complete the task.

[3] pnp.diag.uniroma1.it
[4] https://github.com/iocchi/PLEXI
[5] https://github.com/ultralytics/yolov5
[6] https://docs.opencv.org/4.x/d5/dae/tutorial-aruco-detection.html
[7] https://github.com/tesseract-ocr/tesseract

3.3. Human-Robot Interaction

Interaction with the robot performing the experiment is a critical requirement for this use case to achieve both improved performance and acceptability. A first implementation of an HRI component for the use case is given by a GUI showing the plan execution status and allowing the user to interact with such execution, in the following forms: 1) robot asking for help to humans if it detects an unsolvable planning or execution problem; 2) operator monitoring and supervising plan execution.

Moreover, we implemented people perception by using an environmental camera to detect the presence of people close by and to estimate their role in the laboratory and their current activities. For some functionality, we can assume that people in the lab wear unique ID markers that can be recognized by the robot to assign a unique identifier (possibly anonymous) to each individual person. With such information, the robot can perform personalized interactions: 1) robot asks for help when people are present and in different ways with respect to the person who is in its proximity; 2) current plan execution is explained with a GUI that is customized for the user role and his/her level of experience with the robot; 3) users can set or change current goals for the robot, also using verbal communication.

3.4. Preliminary results

The current implementation is obtained as a mix of planning-based solutions and manual coding. The executor of plans generated with classical planning is integrated with manually coded procedures that allow the robot to perform the task in an efficient way. Our main KPI is the cycle time to run measurements on one sample, which is currently in the order of 2.5 minutes. However, we currently experience some unexpected events and errors which increase the overall time.

Our goal for the next development is to reduce testing time to 1.5 minutes per sample and to remove (or drastically reduce) any unexpected event. We aim at obtaining this result by optimizing the analysis sequence, running multiple operations in parallel, and eventually leverage forms of collaboration with human operators. Moreover, we also want to scale up the system using multiple collaborative robots.

4. Conclusion

The system described in this paper has potential for demonstrating the effectiveness of AI planning technologies in relevant industrial scenarios, as well as for highlighting the features of the Unified Planning formalism to build and test incrementally planning-based solutions. In this paper, we have shown the main design concepts from the planning point of view and very preliminary implementation and results. Future work includes 1) implementing and evaluating all the planning features described in this paper, measuring the increase of task performance when advanced planning features are used; 2) moving more decisions from the manually written code to the planner domain, including in particular the actions that are executed for human-robot interactions to increase the flexibility of the system, further improve task performance, user acceptability and overall trustworthiness.

Acknowledgments

This work has been carried out within AIPlan4EU project funded by the European Commission – H2020 research and innovation programme under grant agreement No 101016442.

References

[1] M. Ghallab, D. Nau, P. Traverso, Automated Planning: Theory and Practice, The Morgan Kaufmann Series in Artificial Intelligence, Morgan Kaufmann, Amsterdam, 2004.

[2] D. McDermott, M. Ghallab, A. C. Howe, PDDL-the planning domain definition language, Technical Report, Yale Center for Computational Vision and Control, 1998.

[3] Y. Alkhazraji, M. Frorath, M. Grützner, M. Helmert, T. Liebetraut, R. Mattmüller, M. Ortlieb, J. Seipp, T. Springenberg, P. Stahl, J. Wülfing, Pyperplan, https://doi.org/10.5281/zenodo.3700819, 2020. URL: https://doi.org/10.5281/zenodo.3700819. doi:10.5281/zenodo.3700819.

[4] M. Helmert, The fast downward planning system, Journal of Artificial Intelligence Research 26 (2006) 191–246. URL: https://doi.org/10.1613%2Fjair.1705. doi:10.1613/jair.1705.

[5] A. Valentini, A. Micheli, A. Cimatti, Temporal planning with intermediate conditions and effects, Proceedings of the AAAI Conference on Artificial Intelligence 34 (2020) 9975–9982. URL: https://ojs.aaai.org/index.php/AAAI/article/view/6553. doi:10.1609/aaai.v34i06.6553.

[6] A. Torreño, O. Sapena, E. Onaindia, Fmap: A platform for the development of distributed multi-agent planning systems, Know.-Based Syst. 145 (2018) 166–168. URL: https://doi.org/10.1016/j.knosys.2018.01.013. doi:10.1016/j.knosys.2018.01.013.

A. Automated Industrial Experiments

This work is in collaboration with Procter&Gamble which provides a set of real-world industrial problems, in the field of R&D consumer product testing. There are thousands of test methods used in consumer products, which generate vital data to guide scientists in the innovation process. Cleaning surfaces or dishes, testing washing cycles, testing the usability or durability of a package, the list is endless. The outcome of these tests guide scientists in the selection and evolution of new products, which will be later launched on the market. The usage of robotics and online sensors offer an attractive opportunity to standardize the testing process and improve dramatically the quality and reliability of the generated data, but with the current ”traditional” programming it takes a significant amount of time to develop and adopt robotics procedures, and today it can only be done by specialists in the field. Our dream would be to deliver a high number of robotics procedures for the various use cases with a fraction of the effort of today, particularly in the area of planning; where today we have a very long list of if-then-else to cope with all the various tasks, errors and unexpected events, we would like an easy to use automated planner that could quickly adapt to different situations and objectives, so that we will be able to develop new applications in hours. Another challenge is the interaction

with human operators, which in R&D labs can be very frequent; this is by definition a dynamic unpredictable problem, very difficult to encode in a set of fixed instructions, another situation where an automated planner would be an ideal solution. Finally we would like to empower non-robotics-experts, our lab users, to be able to customize, adapt, change the course of the robotics procedures, another unsolved problem of today.

Quality testing of unit dose pouches. An interesting case study addressed in our work is the quality testing of laundry detergent soluble pouches. There are several test methods on this type of product, such as weight, dimensions, elasticity, strength and tightness of the pouch, and for statistical significance they need to be run on a high number of samples (thousands). The incoming stream of samples is highly dynamic, it happens through human operators. Our current test unit is made up of a collaborative robot (Univeral Robot UR5) which interacts with various instruments and the queue of incoming pouch samples to be analyzed. The objective of the planner is to control the operations of the robot, optimize the testing strategy to deliver the hightest possible throughput of results in the unit of time, and react to errors and unexpected events in the process. Today we have a fixed procedures manually encoded, which is hard to scale-up and we know is not optimal. In the future we would like to expand it to more instruments in parallel and more robotic units, and the complexity of a manual programming would be unbearable.

B. P&G use case implemented actions and fluents

The **actions** implemented for plan execution in the P&G use case are the following.

- *goto*: this action performs complex movements of the robot for reaching various joint states or pose goals that are reachable in the environment.
- *box*: this action performs lots of movements of the robot for opening or closing a certain drawer.
- *movegripper*: this action is useful for controlling the gripper, such as the activation, reset, full gripper open/close, open/close with a certain middle value or change the open/close speed/force.
- *percept*: this action performs the perception of the pouches with the extraction of their exact position with respect to the robot and also the number recognition of the current drawer.
- *scale*: this action performs the interaction with the scale (through *USB* connection) during weight test.
- *tightness*: this action performs the interaction with the Mark-10 (through *socat* connection) during tightness test.
- *strength*: this action performs the interaction with the Mark-10 (through *socat* connection) during strength test.
- *say*: this action performs a simple label print on screen.
- *wait*: this action performs a time delay during some action.

The **fluents** implemented in the P&G use case are described below.

- *presentpouch*: returns *true* if at least one pouch has been detected, *false* otherwise.
- *switchdrawer*: returns *true* if at least one drawer is available to be open, *false* otherwise.
- *sensepouch*: returns *true* if the pouch has been grasped, *false* otherwise. To know if pouch has been grasped another pouch perception is performed; if the number of pouches detected is different from the previous perception, it means that the pouch with high probably has been grasped.
- *failure*: returns *true*[8] if the robot failed to finalize a *goto* action, *false* otherwise.
- *sensearruco*: returns *true* if the number of ArUcos detected is equal to 0 or 4, *false* otherwise.
- *pause*: returns *true* if a *pause* request has been received from the GUI, *false* otherwise.
- *senseopenbox*: returns *true* if all the ArUco markers has been detected and the drawer number has been perceived, *false* otherwise.
- *sensestrength*: returns *true* if the value perceived during the strength test exceeds a threshold, *false* otherwise.
- *sensetightness*: returns *true* if the value perceived during the tightness test exceeds a threshold, *false* otherwise.
- *senseweight*: returns *true* if the value perceived during the weight test exceeds a threshold, *false* otherwise.

C. System Architecture

The system architecture for the proposed solution is illustrated in Figure C, highlighting the roles of different types of human operators interacting and collaborating with the system: *Designer*, who is a planning expert interacting with the system to provide high-level specifications of the problem; *Supervisor*, who is a domain expert interacting with the system in order to monitor and orchestrate the overall execution; *User*, who is a bystander user that can interact with the system to contribute to achieving the system goals.

Notice that each user profile has a different interface to interact with the system: 1) programming of planning specification language interface for the *Designer*, 2) a specific GUI for the *Supervisor*, 3) natural HRI functionalities for the *User*.

The modules of the architecture have the following functions. The *Domain Encoder* creates and manages domain specifications used to define the planning problem. The *Problem Encoder* creates and manages planning problems to be sent to the planners for computing the corresponding solutions (i.e., the plans to be executed). This component is activated by signals indicating when the planning problem should be composed (from the currently available inputs) and the planning engines should be invoked. The *Solution Decoder* transforms plans into specific structures used by the executor component to execute the plan. The Online exec module executes the plan. It will include monitoring rules, allowing the plan to proceed also in some non-nominal situations (i.e., to recover from some local failures in action execution).

[8]The reasons of the failure could be the robot *bringup* disconnection or collision with scene object or human.

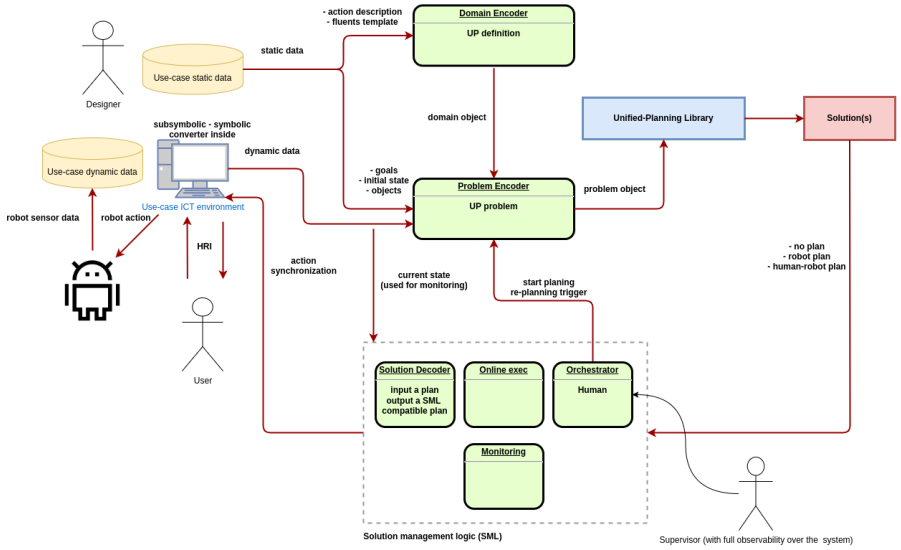

Figure 2: System architecture

The *Orchestrator* provides a user interface for an operator that monitors the plan execution and can send signals to: stop/interrupt the plan, start re-planning, etc. Finally, the *Monitoring* is responsible to track the plan execution and to update the information made available to the Orchestrator, including explanation of the internal state of the robot, the goals, the actions under execution, and the remaining part of the plan.