Explainable Planning

Maria Fox, Derek Long, Daniele Magazzeni
King's College London
firstname.lastname@kcl.ac.uk

Abstract

As AI is increasingly being adopted into application solutions, the challenge of supporting interaction with humans is becoming more apparent. Partly this is to support integrated working styles, in which humans and intelligent systems cooperate in problem-solving, but also it is a necessary step in the process of building trust as humans migrate greater responsibility to such systems. The challenge is to find effective ways to communicate the foundations of AI-driven behaviour, when the algorithms that drive it are far from transparent to humans. In this paper we consider the opportunities that arise in AI planning, exploiting the model-based representations that form a familiar and common basis for communication with users, while acknowledging the gap between planning algorithms and human problem-solving.

1 Introduction

DARPA recently launched the *Explainable AI (XAI) program*[1] that aims to create a suite of AI systems able to explain their own behaviour. This program is mainly concerned with machine/deep learning techniques, as they are currently treated almost as a black box. For example, it is not possible to fully understand why alphaGo selected a specific move at each turn, or on what basis a neural network recognises an image as an "image of a cat".

The need for explainable AI is motivated mainly by three reasons:

- the need for trust;
- the need for interaction;
- the need for transparency.

If doctors want to use a neural network to make a diagnosis, they need to be confident that there is a clear rationale for the NN to diagnose a cancer, in order to build *trust*. As autonomy gathers traction, in many scenarios, instead of full autonomy, Human-Autonomy Teaming (HAT) is required, where humans interact with the AI systems, and for this humans need to understand why the AI system is suggesting something that the human would not do: this requires *interaction*. There are growing legal implications in the use of AI, and in the cases where the AI system makes the wrong decision, or simply disagrees with the human, it is important to understand why a wrong or different decision was made: this is *transparency*.

Explainable AI is harder to achieve than the good decision-making that underlies it. The need to explain decisions forces them to be made in ways that can be subsequently justified in human terms. Entirely trustworthy and theoretically well-understood algorithms can still yield decisions that are hard to explain. For example, linear programming is a well-established tool, but explaining the results it generates without simply 'appealing to authority' remains hard. Part of the difficulty lies in understanding what an explanation should actually contain.

On one hand it is evident that there has been amazing progress in machine/deep learning research and there is a huge proliferation of ML and DL learning. On the other hand, Deep Neural Networks are still far away from being explainable.

In contrast, AI Planning is potentially well placed to be able to address the challenges that motived the DARPA project on AI: planners can eventually be trusted; planners can allow an easy interaction with humans; planners are transparent (at least, the process by which the decisions are made are understood by their programmers).

This paper presents Explainable Planning (XAIP), describing some initial results, and proposing a roadmap for making XAIP more effective and efficient.

Of course the challenge of Explainable AI and the need of making machine/deep learning explicable remain of critical importance. At the same time, we think that XAIP is an important contribution in this direction, as Planning is an important area of AI with applications in domains where learning is not an option.

The paper is structured as follows. We give an overview of related work in the next section. In Section 3 we list some important questions that XAIP should address and in Section 4 we discuss the features of planning that facilitate explanations. In Section 5 we present initial results and suggest future directions. In Section 6 we show two illustrative examples. Section 7 concludes the paper.

[1] http://www.darpa.mil/attachments/DARPA-BAA-16-53.pdf

2 Related Work

For a survey of recent works in the broader area of Explainable AI, we refer to the IJCAI-17 XAI workshop website[2]. Here we briefly highlight some recent works that are related and can contribute to Explainable Planning. *Plan Explanation* is an area of Planning where the main goal is to help humans to understand the plans produced by the planners (e.g., [Sohrabi *et al.*, 2011]. This involves the translation of the planner outputs (e.g., PDDL plans) in forms that humans can easily understand; the design of interfaces that help this understanding (e.g., spoken language dialog systems [Bidot *et al.*, 2010]); and the description of causal and temporal relations for plan steps (e.g., [Seegebarth *et al.*, 2012]). Note that making sense of a plan (plan explanation) is different from explaining why a planner made decisions (XAIP).

Plan Explicability [Zhang *et al.*, 2017] focuses on human's interpretation of plans. Learning is used to create a model of the interpretations, which is then used to measure the explicability and predictability of plans.

Veloso and her team look at the problem of generating narrations for autonomous mobile robot navigations. They contribute with *verbalization*, where the robot experience is described via natural language [Rosenthal *et al.*, 2016].

In *Model reconciliation* [Chakraborti *et al.*, 2017], the focus is on the agent and the human having two different models, hence the explanations must identify and reconcile the relevant differences between the models.

David Smith in his AAAI invited talk presented *Planning as an Iterative Process* [Smith, 2012], and he discussed the broad problem of users interacting with the planning process, which also includes questions about choices made by the planner. Pat Langley et al. more recently used *Explainable Agency* to refer to the ability of autonomous agents to explain their decisions, and in [Langley *et al.*, 2017] they discuss some functions that agents should exhibit.

In this paper, we go beyond a discussion of the questions that need to be answered, and by focusing on AI planning we provide initial results on how to address some of the questions and we point to concrete works in the community to address the others.

3 Things to Be Explained

As mentioned in the introduction, one of the challenges of XAI is to understand what constitutes an explanation. In general, rewriting the steps of the decision-making algorithm in natural language is not what is required. For example, despite it being the case that many planners select actions in their plan-construction process in order to minimize a heuristic distance to goal, based on a relaxed plan, even these terms are inappropriate vocabulary to explain the process to a human. In any case, a real danger in XAI is to reduce an explanation to the statement of the obvious. It is clearly not the answer to the question 'why did you do that?' to say 'because it got me closer to the goal'. A request for an explanation is really an attempt to uncover a piece of knowledge that the questioner believes must be available to the system and that the questioner does not have.

In this section we list some of the questions that characterise what it means for the behaviour of a planner to be *explainable*, and we discuss what constitutes a response to these questions.

- Q1: Why did you do that?

  This is one of the most fundamental questions that can be asked about a plan. It is also an excellent example of how complex the intention behind the question can be. In a sufficiently long and complex plan, it is plausible that the questioner is unable to immediately see which later action in the plan is supported by the target action. Thus, the answer could be as simple as 'action A is in the plan to allow this application of action B'. However, for shorter and more easily assimilated plans, the question is far more likely to be an implicit question: 'why did you do action A? *I would have done action B*'
- Q2: And why didn't you do *something else* (that I would have done)?

  This question is similar to the intention in Q1, but makes the alternative action explicit. An answer to this question would normally be a demonstration of a flaw in a plan that uses the proposed alternative action compared with the plan actually produced. It would usually be acceptable to demonstrate that the plan actually produced was no worse than a plan using the proposed alternative action. In order to respond with either a flaw or else a demonstration of neutral cost, it is necessary to infer by what metric the alternatives are to be compared (one plan might be longer but cheaper than a second — depending on the relative values of time and money, either plan might be considered better).
- Q3: Why is what you propose to do more efficient/safe/cheap than something else (that I would have done)?

  This question refines Q2 by being explicit about the metric being used to evaluate the plans. If the metric is different to the one used in constructing the original plan, then the answer might be to point out the different basis for evaluation of plans. This is a valid explanation provided the original plan is better under the original metric than the rival proposal.
- Q4: Why can't you do that?

  Here we consider the form of this question arising when a planner fails to find a plan for a problem. Planners are typically not very effective at proving unsolvability of planning problems, but model-checking techniques can be applied to planning domain models in order to attempt to prove the non-existence of plans. Unfortunately, converting the exhaustive search of a space (albeit supplemented with careful relaxation-based approaches that bundle parts of the search space) into a transparent and succinct argument is extremely challenging.
- Q5: Why do I need to replan at this point?

  During execution, plan failure will be caused by a deviation between the expected behaviour and the observed

[2]http://home.earthlink.net/∼dwaha/research/meetings/ijcai17-xai/

behaviour of the world. This question can be directed at discovering what has diverged from expectation, or it might be that the deviation is understood, but the significance of the deviation is not. Thus, this question seeks to know what is it that the executing plan was depending on being true that has been observed not to be.

- Q6: Why do I not need to replan at this point?

  There are two reasons possible for this question. One is that the observer has seen a divergence in expected behaviour and does not understand why it should not cause plan failure and the other is that the observer has seen non-diverging behaviour that is not what was expected by the observer. In other words, the divergence could be observable for the executive (and then the explanation is to show why it does not cause plan failure) or else there might be no divergence observable by the executive, in which case the explanation (assuming there is a valid explanation) is to show why the observed behaviour was expected at this point.

Of course there are other questions related to Explainable Planning, including those arising when we consider probabilistic planning or planning under uncertainty, as well as anytime planning (e.g., *will I get a significantly better plan if I give the planner 10 more minutes?*).

4 Unique Features of Planning

AI Planning exploits a collection of techniques that have the potential to make it easier to understand the decision process (even if it is very complex and requires sophisticated algorithms and heuristics).

First of all, AI Planning is based on *models*. Models are used to create plans, and can also be used during plan execution as well as after a plan has been executed. One of the driving principles for a large part of the work carried out in planning is that planners should encapsulate the machinery of planning independently of the domain of application. This means that models capture the dynamics of domains. However, a second guiding principle has been that domain models should not attempt to direct the decision process in the planner (this is not a universally adopted principle, but it has been a consequence of the international planning competition series that domain models have been developed to be 'pure' descriptions of what can be done, not how to do it). McDermott articulated this as the maxim that domain models should contain 'axioms, not advice'. The implication of this for modellers is that they do not need to understand how a planner works, but only the behaviour of their domain. This leads to more intuitive and accessible models for users and facilitates their use in explanation.

Second, plan execution provides its *execution trace*, as a set of pairs (observation, action) which can be used to explore the reasons behind the choices of actions and allows explanations to focus on aspects of state or of action choice, depending on the question.

Third, some progress has been made in explaining plans, in order to help humans to understand the meaning of plans, with a long history based on mixed-initiative planning.

Most planners are based on transparent algorithms, where the planner choice at each decision point is deterministic, repeatable and based on a specific choice mechanism. The fact that the reason for the choice of an action is transparent to the programmer, at least, makes it plausible that we can construct an articulation of parts of that reason in a form a human user might appreciate.

5 Providing Explanations

In this section we address the questions presented in Section 3. For each question, we highlight the main challenges, present some preliminary results and propose a roadmap for achieving the goal of providing reasonable answers and explanations.

5.1 Explaining why the planner chose an action

This explanation introduces two main challenges, that are also inherited by all the following explanations. First, an explanation needs to show *causality* among actions. While this is obvious in many cases (e.g., I get the key first so that I can open the door later), there are examples where action A early in the plan is needed to support action B much later in the plan. This causal relationship might not be evident.

As a practical example, in the electricity domain [Piacentini *et al.*, 2015], the planner reverses current through a transformer *early* in the afternoon in order to support the achievement of supply within thermal constraints in the peak demand period *later* in the evening. This was performed early because there was more flexibility to reconfigure the flows when the load was lighter and by the time the supply was required, demand was so high that the network had no longer got flexibility to change the configuration.

The second issue is that the plan must be understandable to humans, who are not supposed to be planning researchers or experts. Hence, planning formalisms such as PDDL need to be presented to humans in a more natural-language fashion, and the works on plan explanation go in this direction (e.g., [Seegebarth *et al.*, 2012]).

5.2 Explaining why the planner did not choose an action

When a planner decision is confronted with an alternative suggested by the human, an explanation should be a demonstration that the alternative action would prevent from finding a valid plan or would lead to a plan that is no better than the one found by the planner. However, just showing the different heuristic value is not a valid explanation, unless it is translated into a value which is more informative for the user than for example the RPG heuristic value.

What is needed, instead, is an algorithm that executes the plan up to the point where the human suggests the alternative, then injects the human decision, and finally replans from the state obtained after applying the action suggested by the human.

Here there is a first issue, though, as one possible behaviour of the planner could be to just do an *undo* of the human action and produce the original plan (see Figure 1 part (a)).

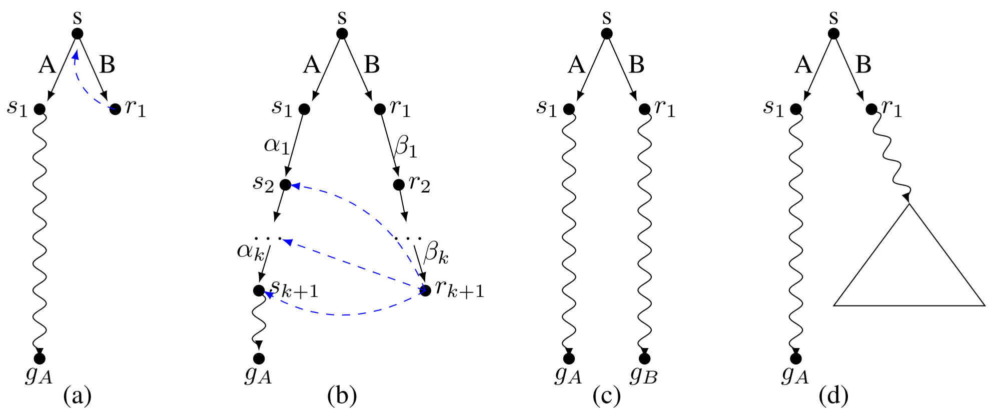

Figure 1: Possible plan behaviours after human-decision injection.

One possible fix is to forbid the planner to revisit the state where the human decision was injected. This, however, does not prevent the planner to get back to the original plan after k steps, as shown in Figure 1 part (b). In this case, the explanation can provide the different costs of the two alternatives, that is $C_A = c(A) + c(\alpha_1) + \ldots + c(\alpha_k)$ and $C_B = c(B) + c(\beta_1) + \ldots + c(\beta_k)$. More in general, the human may want to inject a longer plan, and in this sense the explanation must enable the interaction with the human, by allowing the human to inject more than one action, and provide an explanation after each injection.

The two remaining possible outcomes after the human-decision injections are either that the planner finds a plan for a different goal, or it fails to find a plan, as shown in Figure 1 parts (c) and (d), respectively.

5.3 Explaining why the planner decisions are *better*

The third question we listed was: *why is the planner decision more efficient/safe/cheap than what I would do?* The focus here is that different *metrics* can be used to evaluate a plan. The more interesting case is when one wants to evaluate a plan using a metric which is different from the one used when searching for the plan. This is of practical importance, given that when dealing with complex domains (temporal/numeric domains) there are a number of planners able to minimise the makespan, while almost none are able to optimise other metrics (e.g., total cost). Hence, for example one would like to understand whether a plan found using POPF or FastDownward is actually more efficient (in terms of total cost), than an alternative plan suggested by the human.

This explanation is a refinement of previous explanations, and our proposed solution is to integrate the previous explanation approach with the validator VAL [Fox *et al.*, 2005], where different metrics can be specified to evaluate plans. After each injection of human decision, the validator is used to execute the alternative plan against the new metric (total cost for example).

5.4 Explaining why things can *not* be done

There are two reasons why one action can not be applied in a given state. Either because the current state does not satisfy the action precondition; or because the application of that action would prevent achieving the goal from the resulting state. To provide an explanation for the first case is straightforward, and the validator VAL already provides this. Explanations for the second case are more challenging, and anyway would also be used for providing more general justifications for why the goal cannot be achieved at all (i.e., the planning problem is unsolvable). To this end, we suggest that a promising direction is given by the work being done in proving plan non-existence (see, among others, [Bäckström *et al.*, 2013], [Steinmetz and Hoffmann, 2016], [Hoffmann *et al.*, 2014]).

Model-checking algorithms and tools can prove very suitable [Clarke *et al.*, 2001]. Indeed, in the planning-as-model-checking paradigm [Giunchiglia and Traverso, 1999], a planning problem is cast as a verification problem, where the safety property to be verified is set as the negation of the goal of the planning problem. In this way, if the model checker returns an error trace, that would correspond to the plan. On the contrary, if the model checker states that the property (not goal) is satisfied for all the reachable states, this is a proof that there is no plan. Recently, this paradigm has been applied to complex planning problems with temporal and numeric features [Bogomolov *et al.*, 2014; 2015].

Another very relevant topic is the research around Simple Temporal Networks [Dechter *et al.*, 1991]. STN can be used, for example, to show why an action taking longer than expected can invalidate the plan. There is a large amount of research around STN, STNU [Vidal and Ghallab, 1996], and controllability of STN. For a survey on this field, we refer to [Micheli, 2016].

While proving plan-non-existence (or STN inconsistency) is not yet explaining why the problem is unsolvable, we highlight here that a roadmap for this question should build on the cited works.

5.5 Explaining why one needs to replan

This kind of explanation is needed at plan-execution time. In many real-world scenarios, it is not obvious that the plan being executed will fail for some changes in the environment and/or for mismatching about the model of the environment and the real environment. In most of the cases, plan failure is discovered only when it is too late for replanning in an efficient way.

Explanations for when replanning is needed must take into account the whole plan being executed and check its validity when monitoring the environment. To this end, one possible approach is to use the *filter violation* techniques, as described in [Cashmore *et al.*, 2015] and implemented in ROSPlan.

ROSPlan uses a Knowledge Base to store information about the environment and the plan being executed. The Knowledge Base is updated as soon as new information becomes available. Each change to the Knowledge Base is checked against a filter that is created as follows. Once the plan is generated, the filter is constructed by taking the intersection of static facts in the problem instance with the union of all preconditions of actions in the plan. In addition, each object instance involved in these facts is added to the filter. For example, suppose we have a PDDL domain with the object type `waypoint`, the static fact `(connected ?from ?to - waypoint)` and the action `navigate ( ?v vehicle ?from ?to  waypoint)` whose preconditions include `(connected ?from ?to)`. For each `navigate` action scheduled in the plan, the waypoint instances bound to `from` and `to` and the ground fact `(connected ?from ?to)` are included in the filter. If these objects are removed, or altered in the Knowledge Base, a notification will be sent to the Planning System. An example of this type of explanation is provided in Section 6.

Another promising approach in this direction is Discover-History, described in [Molineaux *et al.*, 2012].

5.6 Explaining why one does *not* have to replan

This explanation is of practical importance, as it is concerned with the situation where the environment being observed (including the plan execution) is different from what was anticipated. In this very common case, it is important to avoid the naive approach of continuous replanning. Rather, it would be ideal to have a way to understand why the plan is still valid, despite the differences between what expected and what being observed. The most common scenario is when actions are taking longer than expected (again, one can think of an underwater mission, where an unexpected current is slowing down the AUV, hence navigate actions are taking longer than anticipated). We highlight here that a promising research direction is represented by the work on dynamic controllability of STN (e.g., [Vidal and Fargier, 1999], [Morris *et al.*, 2001]).

6 Illustrative Examples

In this section we provide two examples of how the approach described in the previous section can be used to explain a plan and to explain why replanning is needed.

6.1 The Rover Domain

We consider the `rover time` domain from IPC-4 and problem 3. Here is the plan found by POP-F [Coles *et al.*, 2010].

```
0.000: (navigate r1 wp3 wp0)  [5.0]
0.000: (navigate r0 wp1 wp0)  [5.0]
5.001: (calibrate r1 camera1 obj0 wp0)  [5.0]
5.001: (sample_rock r0 r0store wp0)  [8.0]
10.002: (take_image r1 wp0 obj0 camera1 col)  [7.0]
13.001: (navigate r0 wp0 wp1)  [5.0]
17.002: (navigate r1 wp0 wp3)  [5.0]
18.001: (comm_rock_data r0 general wp0 wp1 wp0)  [10.0]
22.003: (navigate r1 wp3 wp2)  [5.0]
27.003: (sample_soil r1 r1store wp2)  [10.0]
28.002: (comm_image_data r1 general obj0 col wp2 wp0) [15.0]
43.003: (comm_soil_data r1 general wp2 wp2 wp0)  [10.0]

[Duration = 53.003]
```

An instance of Q1 could be: *"Why did you use Rover0 to take the rock sample at waypoint0?"*

A naive answer could be: *so that I can* `communicate_rock_data` *from Rover0 later in the plan (at 18.001).* This is naive because the plan is so short that the user can easily see that this is performed and, presumably, will realise that the data can only be communicated by the rover that has it. A better way to interpret the question would be to consider alternative ways to achieve the goal this action supports: to communicate the rock data from Waypoint0. It turns out the only way to communicate the rock data is to first have the rock analysis from Waypoint0. And there are only two ways to do this, either to sample rock with Rover0 and or with Rover1.

Hence, an instance of Q2 could be: *Why didn't Rover1 take the rock sample at waypoint0?* In order to provide an answer to this question, we need to force the planner to second the human input. To this end, we remove the ground action instance for Rover0 from those available to the planner and ask it to replan, and here is the new plan:

```
0.000: (navigate r1 wp3 wp0)  [5.0]
5.001: (calibrate r1 camera1 obj0 wp0)  [5.0]
10.002: (take_image r1 wp0 obj0 camera1 col)  [7.0]
10.003: (sample_rock r1 r1store wp0)  [8.0]
18.003: (navigate r1 wp0 wp3)  [5.0]
18.004: (drop r1 r1store)  [1.0]
23.004: (navigate r1 wp3 wp2)  [5.0]
28.004: (comm_image_data r1 general obj0 col wp2 wp0) [15.0]
28.005: (sample_soil r1 r1store wp2)  [10.0]
43.005: (comm_soil_data r1 general wp2 wp2 wp0)  [10.0]
53.006: (comm_rock_data r1 general wp0 wp2 wp0)  [10.0]

[Duration = 63.006]
```

Clearly this is far worse quality than the first plan (the metric is specified as makespan for these plans). So the answer could be: *Because not using Rover0 for this action leads to a worse plan.* It could be argued that this is not a very satisfactory answer, although it is better than the naive answer above, because it does not seem to explain why Rover1 does everything.

One option is for the human to follow up with another question: *Why does Rover1 do everything?* In order to answer this question, we can require the plan to contain at least one action that has Rover0 as an argument. This could be encoded in the domain automatically, by adding a dummy effect to all actions using Rover0 and then adding this as a goal, but here we use a plan generated by remodelling the domain manually, and this is the new plan found by the planner:

```
0.000: (navigate r0 wp1 wp0)  [5.0]
0.000: (navigate r1 wp3 wp0)  [5.0]
5.001: (calibrate r1 camera1 obj0 wp0)  [5.0]
10.002: (take_image r1 wp0 obj0 camera1 col)  [7.0]
10.003: (sample_rock r1 r1store wp0)  [8.0]
18.003: (navigate r1 wp0 wp3)  [5.0]
18.004: (drop r1 r1store)  [1.0]
23.004: (navigate r1 wp3 wp2)  [5.0]
28.004: (comm_image_data r1 general obj0 col wp2 wp0)  [15.0]
28.005: (sample_soil r1 r1store wp2)  [10.0]
43.005: (comm_soil_data r1 general wp2 wp2 wp0)  [10.0]
53.006: (comm_rock_data r1 general wp0 wp2 wp0)  [10.0]
```

Hence, an explanation is that this plan, while not being any longer, contains more actions, so is even worse than the last plan (and in fact contains all the actions of the last plan, so is actually a simple extension of that plan).

However, this is also not entirely satisfactory, because it only shows us that the planner cannot find a useful way to incorporate Rover0 into the plan, but not why. If we restrict the actions that can be used to achieve the dummy condition (that Rover0 acted in the plan) to the set of actions that achieve goals, then the planner cannot find a plan. So, the answer to the question could be slightly improved to: *I cannot find a plan in which Rover0 does not sample the rock at Waypoint0, but achieves a goal in the plan*.

In fact, it turns out that the problem specification prevents Rover0 from reaching Waypoint2, so the soil data there cannot be collected by Rover0, while only Camera1 can be calibrated for Objective0, so only Rover1 that carries Camera1 can be used to take that image. Therefore, the only task that Rover0 can perform is the rock sample mission at Waypoint0.

6.2 The AUV Domain

We consider the AUV domain from the PANDORA project [Cashmore *et al.*, 2014; Palomeras *et al.*, 2016], where an AUV has to complete an inspection mission, by navigating (`do_hover`) between waypoints and making observation of a set of inspection points. Here is a fragment of a plan for this scenario:

```
0.000: (observe auv wp1 ip3)  [10.000]
10.001: (correct_position auv wp1)  [10.000]
20.002: (do_hover auv wp1 wp2)  [71.696]
91.699: (observe auv wp2 ip4)  [10.000]
101.700: (correct_position auv wp2)  [10.000]
111.701: (do_hover auv wp2 wp23)  [16.710]
128.412: (observe auv wp23 ip5)  [10.000]
138.413: (correct_position auv wp23)  [10.000]
148.414: (observe auv wp23 ip1)  [10.000]
158.415: (correct_position auv wp23)  [10.000]
168.416: (do_hover auv wp23 wp22)  [16.710]
185.127: (do_hover auv wp22 wp26)  [30.201]
215.329: (observe auv wp26 ip7)  [10.000]
225.330: (correct_position auv wp26)  [10.000]
235.331: (do_hover auv wp26 wp21)  [23.177]
258.509: (observe auv wp21 ip2)  [10.000]
268.510: (correct_position auv wp21)  [10.000]
278.511: (do_hover auv wp21 wp27)  [21.255]
299.767: (observe auv wp27 ip8)  [10.000]
309.768: (correct_position auv wp27)  [10.000]
319.769: (observe auv wp27 ip6)  [10.000]
329.770: (correct_position auv wp27)  [10.000]
339.771: (do_hover auv wp27 wp17)  [23.597]
363.369: (do_hover auv wp17 wp25)  [21.413]
384.783: (do_hover auv wp25 wp32)  [16.710]
401.494: (do_hover auv wp32 wp36)  [21.451]
422.946: (observe auv wp36 ip9)  [10.000]
432.947: (correct_position auv wp36)  [10.000]
442.948: (observe auv wp36 ip15)  [10.000]
```

In the PANDORA project, the plans are generated and dispatched through the ROSPlan framework [Cashmore *et al.*, 2015] which is also used to monitor plan execution. ROSPlan implements the filter violation described in Section 5.5.

Figure 2 shows the execution of the plan above.

The blue lines represent the Probabilistic Road Map used to determine accessibility for the AUV, while the yellow lines represent the AUV trajectory. At time-point 215.329, while the AUV is observing the inspection point `ip7` it also observes that waypoints `wp32` and `wp36` are actually not connected (this is represented by the red line in Figure 5.5). Given that `(connected wp32 wp36)` was in the filter and after this observation the predicate is removed, a filter violation is triggered, which provides a justification for why replanning is needed, explaining that an action later in the plan (precisely at time-point 401.494) will no longer be executable, and also highlighting which condition has changed from its expected value to one that prevents the plan from being executable.

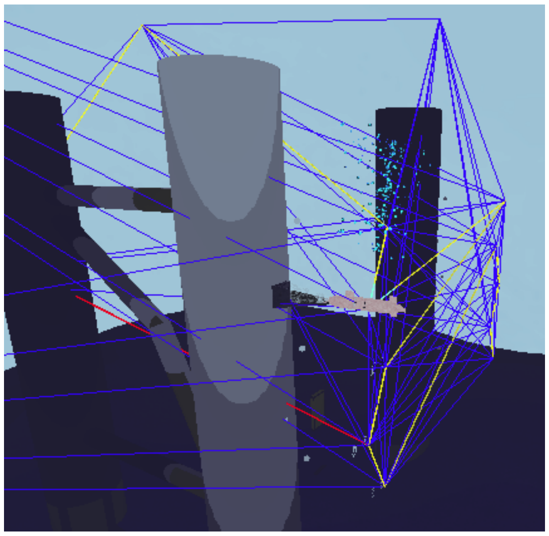
Figure 2: Plan Execution in the AUV Domain.

7 Conclusion

We have introduced Explainable Planning (XAIP), as a promising contribution to the Explainable AI (XAI) challenge. We characterised some of the questions that need to be explained, and provided initial results and a roadmap for achieving the objective of providing effective explanations. The next steps include a full formalisation of the XAIP problem, and a formulation of the user/planner interaction in terms of new constraints to add and alternatives to explore.

This work opens up a number of future directions in explanations for plans as well as plan execution. For example temporal planning introduces interesting planning choices about the order in which (sub)goals are achieved. Another interesting problem is to understand whether to expect improvement in giving the planner a given additional amount time for planning. For plan execution, especially with probabilistic planning and planning under uncertainty, one of the problems is to explain what has been observed at execution time that made the planner make a particular choice.

There is no clear way to define what constitutes a good explanation. As we argued in the paper, XAIP should not focus on explaining the obvious. However, defining a good metric for explanation is an important issue.

More in general, the literature in planning contains many works that could contribute to Explainable Planning. On the other hand, nowadays plans are much more complex than before, and are also used in many new critical domains. Existing works should be revisited and leveraged in order to make XAIP more effective and efficient.

References

[Bäckström *et al.*, 2013] Christer Bäckström, Peter Jonsson, and Simon Ståhlberg. Fast detection of unsolvable planning instances using local consistency. In *Proceedings of SOCS*, 2013.

[Bidot *et al.*, 2010] Julien Bidot, Susanne Biundo, Tobias Heinroth, Wolfgang Minker, Florian Nothdurft, and Bernd Schattenberg. Verbal plan explanations for hybrid planning. In *Proceedings MKWI*, 2010.

[Bogomolov *et al.*, 2014] Sergiy Bogomolov, Daniele Magazzeni, Andreas Podelski, and Martin Wehrle. Planning as model checking in hybrid domains. In *Proceedings of AAAI*, 2014.

[Bogomolov *et al.*, 2015] Sergiy Bogomolov, Daniele Magazzeni, Stefano Minopoli, and Martin Wehrle. PDDL+ planning with hybrid automata: Foundations of translating must behavior. In *Proceedings of ICAPS*, 2015.

[Cashmore *et al.*, 2014] Michael Cashmore, Maria Fox, Tom Larkworthy, Derek Long, and Daniele Magazzeni. AUV mission control via temporal planning. In *Proceedings of ICRA*, 2014.

[Cashmore *et al.*, 2015] Michael Cashmore, Maria Fox, Derek Long, Daniele Magazzeni, Bram Ridder, Arnau Carrera, Narcis Palomeras, Natalia Hurtos, and Marc Carreras. ROSPlan: Planning in the robot operating system. In *Proceedings of ICAPS*, 2015.

[Chakraborti *et al.*, 2017] T. Chakraborti, S. Sreedharan, Y. Zhang, and S. Kambhampati. Plan explanations as model reconciliation: Moving beyond explanation as soliloquy. In *Proceedings of IJCAI*, 2017.

[Clarke *et al.*, 2001] Edmund M. Clarke, Orna Grumberg, and Doron A. Peled. *Model checking*. MIT Press, 2001.

[Coles *et al.*, 2010] Amanda Jane Coles, Andrew Coles, Maria Fox, and Derek Long. Forward-chaining partial-order planning. In *Proceedings of ICAPS*, 2010.

[Dechter *et al.*, 1991] Rina Dechter, Itay Meiri, and Judea Pearl. Temporal constraint networks. *Artif. Intell.*, 49(1-3):61–95, 1991.

[Fox *et al.*, 2005] Maria Fox, Richard Howey, and Derek Long. Validating plans in the context of processes and exogenous events. In *Proceedings of IAAI*, 2005.

[Giunchiglia and Traverso, 1999] Fausto Giunchiglia and Paolo Traverso. Planning as model checking. In *Proceedings of ECP*, 1999.

[Hoffmann *et al.*, 2014] Jörg Hoffmann, Peter Kissmann, and Álvaro Torralba. Distance? who cares? tailoring merge-and-shrink heuristics to detect unsolvability. In *Proceedings of ECAI*, 2014.

[Langley *et al.*, 2017] Pat Langley, Ben Meadows, Mohan Sridharan, and Dongkyu Choi. Explainable agency for intelligent autonomous systems. In *Proceedings of AAAI*, 2017.

[Micheli, 2016] Andrea Micheli. *Planning and Scheduling in Temporally Uncertain Domains*. PhD thesis, University of Trento, Italy, 2016.

[Molineaux *et al.*, 2012] Matthew Molineaux, Ugur Kuter, and Matthew Klenk. Discoverhistory: understanding the past in planning and execution. In *Proceedings of AAMAS*, 2012.

[Morris *et al.*, 2001] Paul H. Morris, Nicola Muscettola, and Thierry Vidal. Dynamic control of plans with temporal uncertainty. In *Proceedings of IJCAI*, 2001.

[Palomeras *et al.*, 2016] Narcís Palomeras, Arnau Carrera, Natàlia Hurtós, George C. Karras, Charalampos P. Bechlioulis, Michael Cashmore, Daniele Magazzeni, Derek Long, Maria Fox, Kostas J. Kyriakopoulos, Petar Kormushev, Joaquim Salvi, and Marc Carreras. Toward persistent autonomous intervention in a subsea panel. *Autonomous Robots*, 40(7):1279–1306, 2016.

[Piacentini *et al.*, 2015] Chiara Piacentini, Varvara Alimisis, Maria Fox, and Derek Long. An extension of metric temporal planning with application to AC voltage control. *Artif. Intell.*, 229:210–245, 2015.

[Rosenthal *et al.*, 2016] Stephanie Rosenthal, Sai P. Selvaraj, and Manuela M. Veloso. Verbalization: Narration of autonomous robot experience. In *Proceedings of IJCAI*, 2016.

[Seegebarth *et al.*, 2012] Bastian Seegebarth, Felix Müller, Bernd Schattenberg, and Susanne Biundo. Making hybrid plans more clear to human users - A formal approach for generating sound explanations. In *Proceedings of ICAPS*, 2012.

[Smith, 2012] David Smith. Planning as an iterative process. In *Proceedings of AAAI*, pages 2180–2185, 2012.

[Sohrabi *et al.*, 2011] Shirin Sohrabi, Jorge A. Baier, and Sheila A. McIlraith. Preferred explanations: Theory and generation via planning. In *Proceedings of AAAI*, 2011.

[Steinmetz and Hoffmann, 2016] Marcel Steinmetz and Jörg Hoffmann. Towards clause-learning state space search: Learning to recognize dead-ends. In *Proceedings of AAAI*, 2016.

[Vidal and Fargier, 1999] Thierry Vidal and Hélène Fargier. Handling contingency in temporal constraint networks: from consistency to controllabilities. *J. Exp. Theor. Artif. Intell.*, 11(1):23–45, 1999.

[Vidal and Ghallab, 1996] Thierry Vidal and Malik Ghallab. Dealing with uncertain durations in temporal constraint networks dedicated to planning. In *Proceedings of ECAI*, 1996.

[Zhang *et al.*, 2017] Y. Zhang, S. Sreedharan, A. Kulkarni, T. Chakraborti, H. Zhuo, and S. Kambhampati. Plan explicability and predictability for robot task planning. In *Proceedings of ICRA*, 2017.