Opportunistic Planning for Increased Plan Utility

Michael Cashmore, Maria Fox, Derek Long, Daniele Magazzeni, and Bram Ridder*

Abstract

This paper explores the execution of planned missions in situations in which opportunities to achieve additional utility can arise during execution. The missions are represented as temporal planning problems, with hard goals and time constraints. Opportunities are soft goals with high utility. The probability distributions for the occurrences of these opportunities are not known, but it is known that they are unlikely so it is not worth trying to anticipate their occurrence prior to plan execution. However, as they are high utility, it is worth trying to address them dynamically when they are encountered, as long as this can be done without sacrificing the achievement of the hard goals of the problem. We formally characterise the opportunistic planning problem, introduce a novel approach to opportunistic planning and compare it to an on-board replanning approach in an example domain involving autonomous underwater vehicles.

1 Introduction

There are many examples of long-horizon control problems in which the goal is to complete specific tasks under time and resource constraints. To do so requires goal-achieving activities to be *planned*. An executive system then executes the resulting plan in the physical world to bring about the desired goals. This picture is complicated by the fact that most physical environments are dynamic, leading to uncertainty about the effects of actions (Paulos et al. 2015). One way to handle this uncertainty is to build a plan as a *policy* (a mapping from states to actions), allowing reactive control during execution, but the current Reinforcement Learning-based approaches to policy construction (Kaelbling, Littman, and Cassandra 1995a; Pineau, Gordon, and Thrun 2006a; Sanner and Boutilier 2009; Ong et al. 2009) do not scale to handle long-horizon tasks. It is computationally most efficient to plan without taking uncertainty into account. When large parts of a plan can be expected to execute without incident, it is more efficient to exploit the strategy of *replanning on failure*, rather than to try to plan ahead for contingencies. During the execution of a plan, execution activity will divert from the original plan when failures occur and actions to achieve the original goals are replanned from the resulting unexpected state.

Another motivation for diverting from an original plan arises when unforeseen *opportunities* to achieve additional utility present themselves. Replanning on failure is a widely recognised technique, but responding to opportunities demands a different behaviour. In contexts in which these opportunities are unlikely, and might arise without warning during execution of a plan, the construction of a policy or a contingent plan that can exploit them is generally impractical. An example of a domain in which opportunities can arise is in the pursuit of planetary space science, where an unexpected high-value science phenomenon might occur during the execution of a long traverse. Instead of missing the phenomenon and having to be directed back by human experts (as was the case when the Mars rover, Opportunity, missed Block Island in 2009), the intelligent vehicle should autonomously detect the phenomenon and determine, without recourse to human advice, whether there are resources available to devote to it.

The domain we consider in this paper is the autonomous inspection and maintenance of underwater installations. We begin, in Section 2, by briefly introducing the field of Automated Planning, the technology we have exploited to address opportunistic planning. In Section 3, we then describe the operational context, introducing the relevant concepts and explaining the planning problem. In Section 4 we explain what we mean by *opportunistic planning*, and in Section 5 we formalise this problem. In Section 6 we discuss probabilistic approaches to similar problems. In Section 7 we explain how we have addressed opportunistic planning within a deterministic planning framework. We then present results for a number of experiments and discuss future work directions.

2 Automated Planning

Planning is the process of considering and organising actions to achieve goals, before starting to execute them. In planning, the actions that must be performed are not predetermined by the goals, but are selected, from amongst a typically large number of alternative actions. The choice is guided by an effort to achieve the goals whilst optimising various metrics. Ordering choices and resource allocations are made, and evaluated, as part of the selection process.

*M. Cashmore, M. Fox, D. Long, D. Magazzeni and B. Ridder are with the Dept. of Informatics, King's College London, Strand, WC2R 2LS, UK.

The selection of a particular action affects choices that can be made subsequently, so has an important impact on the quality of the eventual plan. The consequence of this approach is that neither the number of actions in a plan, nor the makespan or resource allocation of the plan, are predetermined. This distinguishes planning from scheduling, where the actions to be performed are predetermined but the timing of actions, and the allocation of resources to them, are not.

Planning relies on the use of a model of the available actions to support both prediction of their effects on a state and the identification of states from which the actions are applicable. A standard modelling language used to represent actions for this purpose is the *Planning Domain Description Language* (PDDL), originally developed in 1998 by a committee led by Drew McDermott (McDermott et al. 1998), but later extended through several variants, including PDDL2.1 (Fox and Long 2003), PDDL2.2 (Edelkamp and Hoffmann 2004), PDDL3 (Gerevini et al. 2009) and PDDL+ (Fox and Long 2006). The extensions of most relevance to us here are PDDL2.1 and PDDL2.2, which introduced actions with duration and the opportunity for concurrency and management of deadlines. In this language, a planning problem is formally described by providing two files: the *domain* and the *problem*. The problem file consists of two parts: the *initial state* and the *goals*.

Definition 1 *A* **state** *is a set of known true facts consisting of boolean and numeric variables. A Boolean variable is expressed as a proposition consisting of a predicate and a vector of typed arguments, which is assigned the value True or False. A numeric variable is expressed as a function applied to a vector of typed variables, which is assigned to a numeric value.*

Definition 2 *An* **action** *is a tuple* $\langle P, A, D \rangle$*, representing a function from state to state, described in terms of its preconditions,* P*, and effects,* $A \cup D$*. The Boolean effects in* A *are the facts that are added by the action, while the Boolean effects in* D *are the facts that are removed by the action. The numeric effects in* $A \cup D$ *are to* increase *or* decrease *a numeric variable by some numeric quantity, or to* assign *a value to a numeric variable. An action may be applied in any state in which the preconditions are true, and it produces a state in which the effects are true.*

Definition 3 *A* **planning problem** *is a tuple,* $\langle D, I, G \rangle$*, where* D *is the domain file specifying the types, functions and predicates required to describe the problem, and containing the set of action schemas available to the planner.* I *is the initial state, consisting of all the facts that are known to be true when planning begins.* G *is the goal state, consisting of the hard goal conditions that must be achieved by the planner. The problem instance description varies, depending on the problem to be solved, while the domain is a fixed description of what the planner can do to change the state of the world.*

A temporal planning problem is an extension of a planning problem in which actions have *duration*. An action, A, is specified as having a $start$ and an end, and the temporal constraint, that the start precedes the end ($A_{start} < A_{end}$), is always enforced. The duration of an action might be flexible, so that the planner can choose it dynamically. Durative actions can specify invariant conditions that must hold over their entire interval. When durative actions are present, the planner must maintain a *simple temporal network* (Dechter, Meiri, and Pearl 1991) to enable the enforcement of temporal consistency during planning.

The actions used to model a domain usually encapsulate a behaviour that is managed, in execution, by one or more controllers, handling sensing and actuation to achieve a specific effect. The planner is concerned not with the execution of actions, but their organisation into larger collections in order to efficiently achieve a collection of goals. Thus, an action to navigate between waypoints will be implemented by controllers that attempt to use motors and localise via sensing, while the planner is concerned with deciding which locations to visit, for what purpose and in what order.

3 The Operational Context: Underwater Maintenance and Inspection Tasks

In this paper, we focus on long-term maintenance and inspection of underwater installations, using an Autonomous Underwater Vehicle (AUV). This work was carried out in the EU FP7 project, PANDORA[1]. The PANDORA project explored the achievement of *persistent autonomy*, through planning, task learning, plan execution within resource limits and adaptive response to unanticipated events.

The PANDORA project considers an underwater oil installation, consisting of manifolds, pipelines, valves and welds, requiring regular inspection and maintenance. The installation must be maintained over long periods, such as days or weeks, without human intervention. Because of energy and time constraints, mission plans must ensure that the best use is made of limited resources such as on-board energy. The situation is complicated by the fact that environmental conditions (such as currents and marine life) might affect how long tasks take to complete, and when they are available for completion. There is also uncertainty, both in the layout of the installation and the condition of its components (both of which might have changed since the construction of the installation).

The overall objective of the PANDORA project is for a suitably equipped AUV to: (i) construct long-term mission plans to ensure an effective monitoring of the site over time, and (ii) to execute the operations in these mission plans whilst managing uncertainty and responding to unexpected events. The AUV is equipped with a retractable gripper for turning valves, and a water jet for cleaning.

The daily operations to be performed by the AUV include: inspecting pillars, manifolds, welds and pipelines, reading valve-sensors, turning valves, cleaning components exposed to bio-fouling, and updating the mapped layout of the site. This latter task involves investigating objects that appear in unexpected locations, such as collapsed pillars, buried chains and pipeline segments, and other phenomena that could affect the welfare of the installation.

[1] http://persistentautonomy.com/

4 Opportunistic Planning

During the execution of a plan by an AUV, unexpected events might occur that provide *opportunities* for the vehicle to increase the overall utility of its operations. An example is that a part-submerged section of an anchor chain, or other structure, might be spotted during the execution of a mission. This event provides an opportunity to perform an unplanned inspection, or chain-following activity, provided that resources permit the execution of the necessary extra actions. Opportunities are not modelled or anticipated by the planner, and they can be managed without requiring the planner to reason with probabilities. They can be treated as dynamically occurring soft goals. These are distinguished from the goals specified in the problem instance description, which are treated as hard goals that must be satisfied.

To manage unexpected opportunities within a deterministic planning framework, we use a *conservative* planning approach. Conservative planning is a method that seeks to exploit the classical planning framework, while simultaneously recognising the underlying, but unknown, stochastic behaviour of the execution environment. This means that well-researched methods in temporal-metric planning can be exploited. In this approach, rather than seeking to produce a plan with optimal utility, we seek to produce a robust plan, in which we can have very high confidence that the goals will be achieved within the time and energy budget of the vehicle. We then use opportunities to increase the utility of the plan during its execution.

To construct the mission plans, we use the POPF planner (Coles et al. 2010), which takes planning domain models written in the temporal planning language PDDL2.2 (Edelkamp and Hoffmann 2004). A temporal planner is required because the valve-turning tasks impose temporal constraints. They are constrained to be turned within specified time windows. For example, in a given mission it might be necessary to reset a valve within a one-hour window timed to occur six hours into the future from the start of the plan. These constraints necessitate reasoning with deadlines and synchronisation of activities. Thus, although we consider only a single AUV executing actions in sequence, and therefore no concurrent activity, these deadlines raise synchronisation issues which make online methods such as the online receding horizon approach (Burns et al. 2012; Ross et al. 2008) impractical.

We assume that opportunities are rare, but offer high utility gain when they are spotted and exploited. Thus, opportunities in this framework are somewhat similar to *high impact, low probability* events (HILPs) (Lee, Preston, and Green 2011), although in this setting we are considering rare events with a positive value, while HILPs are typically treated as risks that threaten execution. We further assume that the probability density function governing the distribution of these opportunities in the physical space is unknown, so we cannot plan to anticipate them or determine their expected utility.

Our conservative planning strategy is based on the assumption that, when executed, the actions in a plan will have durations that are normally distributed around their means, and that actions will in fact take much longer than their mean durations. To build a robust plan we therefore use estimated durations for the actions that are longer than the means. For example, to have 95% confidence, we use 1.65 standard deviations from their means as the estimated durations of the actions. 1.65 standard deviations from the mean is the 95th percentile of the Gaussian distribution.

As a plan containing multiple actions is executed, the use of the 95th percentile as an estimate for the nominal execution time of each action leads to an accumulating expected error. So, if k actions all with independently distributed mean execution times m and standard deviations s are executed in sequence, the sum of the estimated durations will yield a total time for execution of $k(m + 1.65s)$. The time actually required to achieve the 95th percentile for the combined sequence of actions is only $km+1.65s\sqrt{k}$, showing that the estimate based on individual 95th percentiles yields a $1.65s(k - \sqrt{k})$ over-estimate of the time required for 95% confidence in execution of the entire sequence. Our proposed opportunistic planning method is designed to exploit this over-estimate for other tasks that arise opportunistically.

As a practical example, suppose that navigating the traverse between two waypoints on the installation has a mean time of 11,507 seconds (3.2 hours), and a standard deviation of 925 seconds (about 15 minutes). If 5 successive traverses between waypoints are to be executed, the use of the nominal time estimates will yield an estimated duration of 65,166 seconds (about 18 hours). The 95th percentile for the estimated duration of the combined sequence is 60,948 seconds, so using the 95th percentile for nominalisation will lead to an expected overestimate for the execution time of 4,218 seconds: just over an hour, which is about 7% of the 95th percentile time for the execution of the complete sequence.

Opportunities can only be spotted and exploited during the execution of *preemptible* actions, or at points between the execution of actions. In our application, the only preemptible actions are the navigation actions (of different types, corresponding to different modes of movement). We consider opportunities that are physically located in space, so it is generally the case that they will arise during movement between locations, when large areas are scanned as part of the navigation action.

With these points in mind, the opportunistic planning problem is as follows:

- The problem is a temporal planning problem, with deterministic actions and a collection of hard goals specified in the problem instance description.
- The problem exists in a 3-dimensional space, with tasks requiring the executive to perform actions at particular locations and actions allowing the executive to move between locations (possibly in more than one way).
- The initial state is uncertain in a limited way: there is a possibility that, at random locations, instances of objects exist that offer high reward if certain actions are performed at their locations, but their existence and locations are not known to the planner. There is also uncertainty

about whether these objects will be observed, even if the executive passes close to them.

- Although the probability distribution of opportunities is unknown, it is assumed that they are rare and it is therefore entirely likely that the plan for the original goals will be completely executed without an opportunity ever being encountered.
- The executive is required to satisfy the hard goals of the original problem, and to collect as much reward as possible from opportunities, given that the hard goals are achieved.

Since the durations of actions can be longer than expected, the goals might not be achieved when executing a plan that is expected to satisfy them, due to failure to meet deadlines. In fact, during execution actions can fail for various reasons and the goals might become unachievable as a consequence. In this paper we do not focus on what happens when actions fail. Instead, we are interested in the possibility that a conservative assumption about the time required to execute actions used in the original plan might lead to slack time that can be used to pursue opportunities.

In this paper we formalise the opportunistic planning problem, we propose a way to obtain good quality solutions to it and we compare the proposed approach with the simple alternative to replan whenever the observed state diverges from the predicted state during execution.

5 The Opportunistic Planning Model

We present a formal description of the opportunistic planning problem. We assume that P is a temporal planning problem, consisting of a *domain* and a *problem instance*, expressible in PDDL2.2 (Edelkamp and Hoffmann 2004). The domain provides a finite, enumerated type representing locations, W, in a 3-dimensional space. In the PDDL family of languages, the members of this type are all explicitly named in the definition of the planning problem instance. We suppose that P represents a problem in which goals are associated with locations (for example, pillars are located at waypoints), so that the executive must visit those locations in order to complete the achievement of the goals. We further suppose that the domain file of P contains at least one action schema that allows an executive to move between locations (possibly subject to accessibility constraints, restricting which pairs of locations are directly connected).

Definition 4 *An* **opportunity** *is a tuple, $\langle T, Og, U\rangle$, where T is the name of a* PDDL *enumerated type in P, (x, y, z); Og is a goal, with one free variable, v of type T; and U is a utility value in $\mathbb{R}$. The goal $Og[v]$ is called an opportunistic goal.*

An opportunity is a soft goal schema that is associated with objects of a particular type, T, appearing in the domain of P. The idea is that instances of T can be discovered and added to the world during plan execution, each leading to the creation of a new soft goal by instantiation of free variable v in the opportunistic goal, Og. For example, Og might be an inspection goal, and v might be instantiated by the object "pillarA" of type Pillar, resulting in a new soft goal to have inspected pillarA. In this work, we assume that soft goals always correspond to performing operations on single objects. An opportunistic planning domain consists of the original domain, P, and a collection of opportunities.

In a real world situation, opportunities are distributed in some way around the physical area being explored. In a simulation, they can be placed randomly around in the space. In both cases, they exist to be discovered, but are not modelled by the planner. They arise when new objects are identified at locations that may have been previously unmapped and inaccessible. If an opportunity is present, it can only be discovered if the executive passes within sensing distance of its location (a distance dependent on the type and effectiveness of sensors available) and with some associated probability, which is unknown.

The modelling language PDDL2.2 provides a feature called Timed initial literals (TILs) which record, in the problem instance description, time windows during which goals are achievable.

Definition 5 *An* **opportunistic planning problem** *is a tuple $\langle P, I, G, A, Opps, R\rangle$, where P is a temporal planning problem (as described above), I is the initial state (including timed initial literals that determine deadlines for goals), G is a set of hard goals (they must all be achieved in any goal state), A is a distinguished subset of actions in P that are preemptible, $Opps$ is a set of opportunities, and R is a function giving the mean and standard deviation of the duration of any grounded instance of an action in P. The durations of action instances are specified at the 95th percentile of the distributions whose parameters R reports.*

If an opportunity is discovered, replanning is initiated and an extended initial state is constructed. If the opportunity is discovered at time t, the TILs in the extended initial state must be displaced by t (to allow for the time that has passed since the start of execution). Any TIL with time earlier than t are discarded and those later than t have their times reduced by t. We call these *time-corrected* TILs. For example, consider a TIL with an original time of t, when replanning is initiated 30 minutes into the execution of the plan. The TIL t occurs at $(t - 30)$ minutes from the extended initial state. Hence, the time of the *time-corrected* TIL is $(t - 30)$.

Definition 6 *A* **monotonic extension** *of an initial state description, I, extends I with a new collection of objects and waypoints, $O = \{o : T\} \cup \{w : W\}$ and facts F, such that each $f \in F$ includes at least one object in O, and new soft goals Og, formed by grounding the opportunities associated with type T using objects in O. Connectivity is added, linking the new waypoints so that the newly added opportunity can be reached. The extended initial state, I', adds O to the objects in I and records the opportunity utility as a reward for actions achieving goals in Og. I' contains all the facts and time-corrected TILs in I as well as facts F.*

The extended initial state will locate discovered opportunities at new locations and new paths will be available by which they can be accessed. According to the topography of the space, some paths might require additional intermediate locations to have been added to the state.

Definition 7 *An* **opportunistic plan fragment** *in state S is a plan constructed to achieve a grounded opportunistic goal from state S.*

Our approach integrates opportunistic plan fragments with the original plan, in order to exploit an opportunity within the context of achieving the hard goal set. Opportunistic plan fragments, once integrated with a plan, can be visualised as sub-plans (which might be long chains of actions) that branch off from the main plan trajectory, finally returning to the main plan at a point enabling its continued execution to result in the achievement of the hard goal set. The means by which this integration is achieved is discussed in Section 7.

6 Relationship to other Probabilistic Models

Earlier work (Fox and Long 2002; Gough, Fox, and Long 2004) explores a different model of opportunities in which the opportunities and their locations are known in advance of starting the execution of the plan. Opportunistic plan fragments are computed offline, and executed online if their resource requirements are met. Woods et al. (Woods et al. 2009) assume that the *types* of opportunities that can arise are known, and that all opportunities of the same type can be exploited by the same opportunistic plan fragment. Plan fragments are precomputed and stored in a plan library. The relevant plan fragment is then inserted into the plan whenever an opportunity of its type is identified, and resources allow.

The opportunistic planning problem can be seen as a special case of a Partially-Observable Markov Decision Problem (POMDP), with an infinite state space (due to the continuous 3-dimensional distribution of locations of possible opportunities). A general solution to such a problem is a policy, mapping each possible state to an action. If the probability distribution over the opportunity space were known, the problem could be modelled as an explicit POMDP (Kaelbling, Littman, and Cassandra 1995b). Whether or not to pursue an opportunity in a certain belief state amounts to whether the expected utility of pursuing the opportunity, in addition to achieving the hard goal set, all within the resource envelope available, exceeds the expected utility of completing the current plan under execution with lower resource pressure. The problem can be modelled but, even with recent work on improving efficiency (Ong et al. 2009; Pineau, Gordon, and Thrun 2006b; Ross et al. 2008), the decision-theoretic approach will not scale to the sizes of problems that arise in practical applications. Moreover, the offline decision-theoretic reasoning cannot be done at all in the absence of knowledge about the probability distribution over the opportunity space. A further problem in creating a POMDP model is that states must record histories in order to capture the fact that repeated observations of a part of the physical space do not have independent probabilities of leading to discovery of an opportunity: if nothing is seen on one observation, then the probability that there is anything there to be seen is much lower. Finally, the continuous space presents a very significant challenge in representing the state space, since we cannot know in advance which locations are of interest, or, therefore, which areas of the space might be observed or even become accessible.

Although a POMDP model appears very difficult to realise and a full policy impossible to achieve with current approaches, a partial policy might be more tractable. One possible partial policy structure is a *contingent plan* in which alternative branches are built explicitly into the plan structure (Pryor and Collins 1996; Drummond, Bresina, and Swanson 1994). Contingent planning is very expensive, so various methods have attempted to limit the number of contingent branches constructed. In particular, Coles (Coles 2012) considers over-subscription planning with resource uncertainty. In her approach, the configuration of the world and all goals, including opportunities and their locations, are known in the initial state, which makes it unsuitable for tackling the problem we have characterised.

Burns *et al* (Burns et al. 2012) use an online receding horizon approach to consider anticipatory on-line planning in which plans take into account goals that are likely to arise, in order to be better prepared for achieving them efficiently. This is a relevant idea, but the difficulty in applying it to the problem we present is that, in our model, opportunities are assumed to be rare, making it unlikely that investment of resource in searching for an opportunity, rather than in simply completing the main mission goals, will pay dividends.

All of these approaches rely on some knowledge of the PDF over opportunities. By contrast, *replanning* does not require any knowledge about probability distributions, either over the opportunity space or over the use of resources by actions. In a reactive online method, a replanning strategy responds to an opportunity by throwing away the plan under execution, and building a new, conservative, plan for the union of the hard goal set and the opportunity. It then executes this plan instead, whenever its available resources are sufficient to achieve the new goal set.

Replanning is therefore a plausible approach to our problem. However, we hypothesise that replanning will be unnecessarily expensive, because it will replan parts of the problem for which there is already a detailed, and resource-valid, plan structure in place.

7 The Proposed Approach

We propose an approach to solving the opportunistic planning problem (Definition 5) as described in Section 5. Our approach tackles opportunities (Definition 4) by inserting opportunistic plan fragments (Definition 7) into an existing robust plan, generated using conservative planning.

In our implementation of opportunistic planning, we consider the distributions of the action durations and plan at the 95th percentiles of these distributions. This provides a stable baseline for robust confidence in the completion of the plan. Conservative planning seems an inefficient way of allocating time to tasks, but this apparent inefficiency is offset by the fact that plan utility is likely to be improved upon at execution time. The executive may decide to use any resource gained during execution to carry out extra tasks, such as pursuing opportunities, on top of the basic plan.

We manage execution of an opportunistic plan via the use of an execution stack. When a decision is made to pursue

an opportunity, the tail of the executing plan is pushed onto the stack. The initial state is monotonically extended (Definition 6) and replanning is initiated so that an opportunistic plan fragment is constructed. As long as this successfully completes within the planning time bound, and the resulting plan fragment can be executed within allocated resources, execution of the opportunistic plan fragment begins. When the opportunistic plan fragment has finished executing, the remainder of the main plan is popped off the stack, and its execution is then resumed. With this execution method, it is possible for an opportunistic plan fragment under execution to be suspended and stacked, if a new opportunity is detected during its execution. This is illustrated in figure 2.

When an opportunistic plan fragment is incorporated, some steps from the main plan might become redundant. In this case, some reasoning is needed to return to the latest possible state on the main plan trajectory (obviating as many redundant steps as possible). There is much work on the task of *plan merging*, e.g. (Alami et al. 1998; Alami, Ingrand, and Qutub 1998). In our approach we simply prune redundant steps and insert the plan fragment. In particular, when an opportunity is planned, the main plan suffix is pruned by removing all of the navigation actions at the front of the suffix. Figure 1, part (a), shows an opportunistic plan fragment that has been inserted into the plan, while part (b) shows the structure of a contingent branching plan. It can be seen that, in principle, opportunistic planning explores many fewer states.

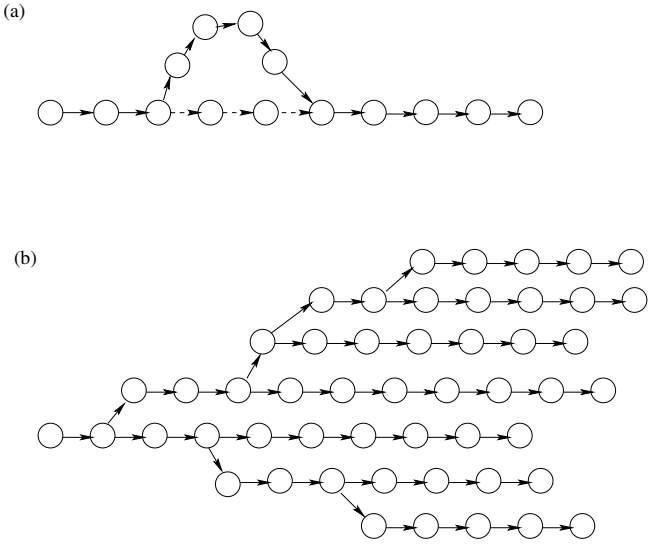

Figure 1: (a) The main plan with an opportunistic plan fragment attached. The fragment rejoins the plan suffix at the first necessary point for completion of the hard goal set. (b) The structure of a contingent plan. Each branch leads to a different goal set, depending on resource availability at the branch nodes.

7.1 The Implementation

In our implementation, the types of opportunities that can be identified are *inspections* and *investigations*. In particular, we restrict our attention to pillar inspections and a particular kind of investigation called chain-following. New objects of types Chain and Pillar are detected during AUV operations. When a new object is spotted, a new opportunity is created, by instantiating the corresponding opportunistic goal, as described in definition 4. The consequent construction processes, by which the extended initial state and the new soft goal are set up, are described in Definition 6. Our implementation of the Opportunistic Planning method behaves as follows, and is detailed in Algorithm 1:

- construct a sequential strategic plan to achieve the goal set (top level missions) within a conservative resource bound;
- start executing that plan under operational control, keeping track of unspent resources;
- branch off the plan to handle an opportunity within the unspent resource bound, storing the plan suffix (this is recursive);
- return to the plan suffix as soon as possible.

The execution of this algorithm, showing the management of the plan stack, is shown in Figure 3.

A limitation of our approach so far is that we treat navigation actions as different from any other actions. They are only needed to move the AUV to places where tasks can be done, and are never in the plan to achieve top-level goals. They can therefore always be safely removed from the suffix as long as we can reach the next interesting waypoint after completion of an opportunistic plan fragment. Integrat-

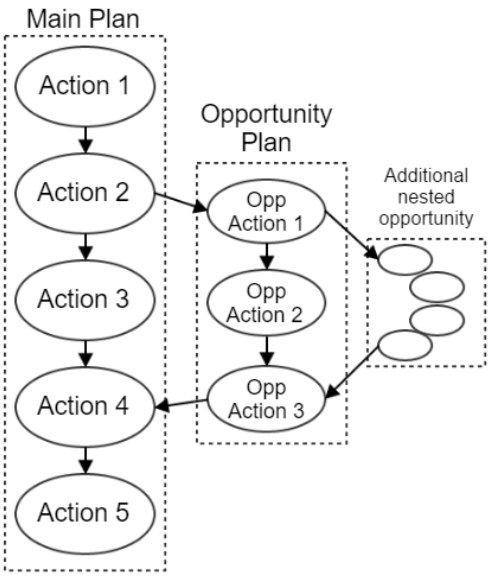

Figure 2: Plan execution with opportunity insertion. Actions 2, 3, and 4 of the main plan are navigation actions (or more generally "support actions") which are subsumed by the opportunistic plan fragment, and may be skipped. The opportunistic plan achieves the "weakest preconditions" of the tail end of the main plan, while adhering to the deadline constraints.

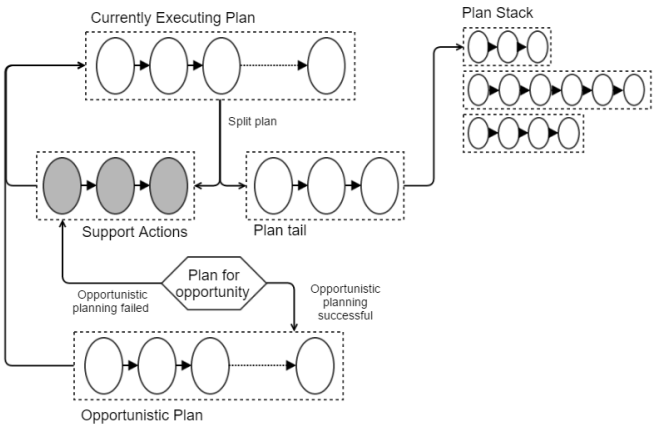

Figure 3: The execution of the algorithm, showing how plan suffixes are stacked. We start at the currently executing plan. When an opportunity is detected, the support actions are pruned and the plan suffix is stacked. Then we plan for the opportunity. If the opportunistic planning is successful, then the opportunistic plan becomes the new currently executing plan, otherwise the support actions are executed.

ing opportunistic plan fragments becomes more complex if other actions are preemptible, and we will consider such extensions in our further work.

In Algorithm 1:

- input: time limit in seconds, missionID identifies the mission goals;
- line 1: now() is the current time. (at AUV MissionEndPoint) is included as a goal;
- line 3: replanRequested can be set true by external processes;
- line 12: We set by hand which actions can dispatch early (all except for turn_valve);
- line 16: AUVs are busy while still executing an action;
- line 18: An action's default timeout can be chosen per operator, either as duration*T for some T, or duration+T for some other T;
- line 22: opportunistic_plan_requested is a communication variable that is declared and initialised externally, and then set true by an external process;
- lines 26-29: These lines find the finishing location for the opportunistic plan, and remove the "goto" actions from the parent plan;
- line 32: If the opportunistic mission was not possible, then the "goto" actions are reinserted at the start.

A final point about the implementation is that we do not currently use the utility component of an opportunity. This is because we have restricted the system to detecting and considering only one opportunity at a time, and an opportunity will always be pursued if time and resources allow. However, in general there might be several opportunities available, in which case a means is required for distinguishing them. Utility provides a way in which opportunities can be ranked for consideration. One approach would be to rank the opportunities by utility, then execute the first one in the ranking that fits into the available time and resources. Again, this is a topic for future work.

8 Experiments

Our hypothesis is that the opportunistic planning approach, just described, is more efficient than replanning the entire hard and soft goal set every time a new soft goal is identified. This might seem to be a "straw man" comparison, because it might seem obvious that replanning is apparently facing a much harder challenge than that oof planning to achieve a local opportunity within a well-defined context. However, this is not always the case. When replanning, the planner throws away all of the constraints of the defunct plan and has complete freedom about the timing of activities, as long as they fit within their respective time windows. This allows the planner to optimise activity around deadlines. By contrast, opportunistic planning has to fit all activity into the local time and resource envelopes of the global plan, which necessitates a *myopic* approach to opportunities (now or never), and might be over-constraining.

We therefore contrast our approach with a replanning method, to identify whether we gain any significant advantage, in terms of overall plan utility and resources spent planning, from the opportunistic approach.

In the case where no opportunity is observed during execution, the replanning strategy and the opportunistic planning strategy will both simply execute the main plan to achieve the hard goals, with no deviation (except in response to plan failure, which we ignore here). Therefore, the differences lie only at the point where an opportunity is discovered. In the replanning case, we construct a new initial state and replan for the entire goal set. In the opportunistic planning case, we plan only for the opportunity, together with a goal to return to the start of the plan suffix. Both approaches begin by constructing the monotonically extended initial and goal states. The opportunistic approach benefits from what is usually a simpler planning problem in exchange for los-

Algorithm 1: opportunisticPlanningMethod

```
input  : timelimit : Int, missionID : Int, missionEndPoint :
         Waypoint
output : boolean
problem ← generateProblemFile(now(), missionID,
  missionEndPoint);
plan ← makePlan(problem);
replanRequested ← false;
freeTime ← 0;
if plan.length() > timelimit then
    return false;
end
else
    while plan.length() > 0 do
        currentAction ← plan.pop();
        dispatchTime ← currentAction.dispatchTime;
        if !canDispatchEarly(currentAction) and
          now() < dispatchTime and !replanRequested then
            wait();
        end
        currentAUV ← currentAction.AUV;
        while AUV.isBusy() and !replanRequested do
            wait();
            if now() > currentAction.timeout then
                dispatch(cancelAction);
                replanRequested ← true;
            end
            if opportunistic_plan_requested then
                opportunistic_plan_requested ← false;
                currentEndPoint ←
                  currentLocation();
                prunedActions ← {};
                while plan.first() == "goto" do
                    currentEndPoint ←
                      plan.first().destination;
                    prunedActions.push_back(plan.pop());
                end
                plans.push_back(plan);
                if
                  !opportunisticPlanningMethod(freeTime,
                  opportunisticMissionID, currentEndPoint)
                  then
                    plans.insert(prunedActions, 0);
                end
                plans.pop(plan);
            end
        end
        if !replanRequested then
            dispatch(currentAction);
            freeTime ← now() - dispatchTime;
        end
        else
            replanRequested ← false;
            problem ←
              generateProblemFile(now());
            plan ← makePlan(problem)
        end
    end
end
return true
```

ing the possibility of finding a better plan by exploiting the remaining resources to achieve the opportunity and original goals together. Our experiments consider the situation at a point at which an opportunity has been discovered.

We perform the comparison by setting up a *main mission*, with hard goals, and an opportunity. The main mission is taken to be a valve-turning mission, possibly involving many valves, and the opportunity mission is an inspection (we do not consider investigations in this experiment). In the valve-turning mission, the AUV is required to approach and set two valves within a deadline. The effect of setting a deadline is to bound the resource available for exploiting opportunities. The inspection mission is not time-limited. When it arises as an opportunity, a plan to exploit it must fit within the available resource envelope. Inspection missions are of several sizes, ranging between 2 and 32 inspection points.

In our simulation, the main mission elements and the opportunities are located within an area 50m by 50m and set at least 5m apart. They are positioned successively, with uniform probability over the available area. The deadlines for valves are set to different values, making the planning problems harder as the deadlines are tightened. The opportunistic planning strategy requires the opportunity to be exploited within the free resource window, *before* the completion of other mission components. The replanning strategy does not require this, but both strategies require the overall plan to be completed by the mission deadline.

In Table 1 we report our results for a collection of randomly generated problem instances. The opportunistic planner is given 10 seconds to solve the problem. In general, the window of opportunity is short, partly because it is most often the case that we will discover an opportunity while navigating, in which case we do not want to stop the vehicle unless we decide to pursue the opportunity, and partly because the energy and computational resources on board the AUV is limited. It is also important that the time taken evaluating an opportunity should not be significant compared with execution time of actions, otherwise we endanger the main mission itself by wasting resources on multiple opportunity evaluations. This latter problem arises if the signal processing that leads to recognition of an opportunity is unable to determine that multiple sightings of the same object are actually not distinct opportunities.

The replanning strategy was allowed 30 minutes of CPU time to generate a best possible plan. We report the best plan found in that time, with the time it took to find that plan (POPF2 uses an anytime strategy of plan improvement, reporting plans as they are found).

The bolded results are the cases in which the combined mission is solvable with a higher quality solution within the 30 minute bound. In four of these cases, the replanning strategy would outperform the opportunistic strategy, but in the bold and italicised case, the plan takes so long to find that the combined planning and execution time exceeds the time available for the complete plan. Indeed, in almost all cases, the complete plan is so much longer than the opportunistic plan that it would not be possible to complete within the duration of the intended mission time for the whole problem.

Part of the difficulty for the replanning strategy arises

from the forward search paradigm of POPF2. The existence of deadlines leads to the planner pushing activity later along the time line than is appropriate and it fails to search the parts of the search space in which the short plans exist. In future work we will explore alternative temporal planning strategies in order to better understand the impact of this planning artefact on the qualit of the plans.

In one of our test cases the opportunistic planner failed to find a plan within 10 seconds, so the plan reverts to the main mission plan. In this case, the replanning strategy takes 3 minutes to find a plan that is far too long to be used in place of the main mission plan, so this represents a waste of the time spent in this attempt.

These results show very clearly that the cost of a complete replan is much higher than the cost of planning for an opportunity alone. Even though planning for the combined mission should offer, in principle, a chance to find a better quality solution than the one found by simply linking the opportunistic plan fragment to the front of the existing plan, the reality is that it is very hard to achieve this. A more capable planning strategy might be more successful in finding better plans, but the time taken to do so would certainly be far greater than the time required to find the opportunistic plan. Each such plan construction attempt spends the very resource that is required to exploit the opportunity itself, so it is an impractical approach to repeatedly evaluate opportunities by using a full replanning approach.

9 Conclusions and Future Work

In this paper we have defined the concept of *opportunistic planning*, a method for robust planning and plan execution under limited uncertainty. We have presented a fully implemented method for opportunistic planning of missions and the interleaving of mission execution with utility-increasing opportunities. The results of our experiments show that opportunistic planning is a good compromise between scalability and robustness, allowing the practical management of uncertainty. We have demonstrated that, in terms of time to plan and resulting plan utility, opportunistic planning significantly outperforms a replanning method.

In this paper we focus on long-term maintenance and inspection of underwater installations, using an Autonomous Underwater Vehicle (AUV). The locality of goals and the presence of low probability/high-reward opportunities make this domain an ideal target for the opportunistic planning technique. These aspects are also present in many other robotics domains (e.g. ground robots for disaster recovery). One avenue of future work is to investigate other types of scenario, to discover how the opportunistic planning approach could be generalised to other planning domains.

Our current approach to opportunistic planning demonstrates improvements over a replanning strategy, but has some limitations. In particular: we do not evaluate the expected gain, in terms of accumulated resource, of reducing our confidence in achievement of the hard goal set. For example if, at some point p, into the execution of a plan, we are willing to reduce our confidence in successful execution of the plan suffix to the 94th percentile, how much resource could we save for spending on an opportunity spotted at p?

As an alternative to allowing the expected accumulation of resources following the execution of a sequence of actions it would be possible to adjust the sum of the nominal durations to account for the length of the sequence. So, for k actions each with identical mean and standard deviation, the nominal durations can be reduced to $m + \frac{1.65s}{\sqrt{k}}$.

More generally, where several actions are sequenced to achieve a goal it is possible to discount the sum of the nominal durations to allow for the expected accumulated benefits of using the 95th percentile as the nominal durations of the individual components.

In our future work we intend to experiment with trading off confidence against utility, by doing this reasoning *online* at the point at which we have evaluated an opportunity. The actions in the plan suffix are not changed, but the confidence in completing it successfully is traded for the benefits of the opportunity. For a very high value opportunity it might even be worth, in order to free up more resource, requesting the sacrifice of a component mission from the command level planner.

References

Alami, R.; Chatila, R.; Fleury, S.; Ghallab, M.; and Ingrand, F. 1998. An architecture for autonomy. *The International Journal of Robotics Research* 17(4):315–337.

Alami, R.; Ingrand, F. F.; and Qutub, S. 1998. A scheme for coordinating multi-robots planning activities and plans execution. In *ECAI*, 617–621.

Burns, E.; Benton, J.; Ruml, W.; Yoon, S.; and Do, M. B. 2012. Anticipatory Online Planning. In *Proceedings of 22nd International Conference on Automated Planning and Scheduling (ICAPS'12)*.

Coles, A.; Coles, A.; Fox, M.; and Long, D. 2010. Forward-chaining partial-order planning. In *Proceedings of the 20rd International Conference on Automated Planning and Scheduling (ICAPS'10)*, 42–49.

Coles, A. 2012. Opportunistic Branched Plans to Maximise Utility in the Presence of Resource Uncertainty. In *Proceedings of European Conference on AI (ECAI'2012)*.

Dechter, R.; Meiri, I.; and Pearl, J. 1991. Temporal constraint networks. *Artif. Intell.* 49(1-3):61–95.

Drummond, M.; Bresina, J.; and Swanson, K. 1994. Just-in-Case Scheduling. In *Proceedings of 12th National Conference on Artificial Intelligence (AAAI)*, 1098–1104.

Edelkamp, S., and Hoffmann, J. 2004. PDDL2.2: The language for the classical part of the 4th international planning competition. Technical report, Technical Report 195. Albert Ludwigs Universitat, Institut fur Informatik, Freiburg. Germany.

Fox, M., and Long, D. 2002. Single-trajectory opportunistic planning under uncertainty. In *Proceedings of the UK Planning Special Interest Group (PLANSIG'02)*.

Fox, M., and Long, D. 2003. PDDL2.1: An extension to pddl for expressing temporal planning domains. *Journal of Artificial Intelligence Res. (JAIR)* 20:61–124.

Fox, M., and Long, D. 2006. Modelling mixed discrete-continuous domains for planning. *J. Artif. Intell. Res. (JAIR)* 27:235–297.

Gerevini, A.; Haslum, P.; Long, D.; Saetti, A.; and Dimopoulos, Y. 2009. Deterministic planning in the fifth international planning competition: PDDL3 and experimental evaluation of the planners. *Artif. Intell.* 173(5-6):619–668.

Gough, J.; Fox, M.; and Long, D. 2004. Plan execution under resource consumption uncertainty. In *Proceedings of the ICAPS Workshop on Connecting Planning and Execution*, 24–29.

Kaelbling, L.; Littman, M.; and Cassandra, A. 1995a. Planning and acting in partially observable stochastic domains. *Articial Intelligence* 101:99–134.

Kaelbling, L.; Littman, M.; and Cassandra, A. 1995b. Planning and acting in partially observable stochastic domains. *Articial Intelligence* 101:99–134.

Lee, B.; Preston, F.; and Green, G. 2011. *Preparing for High-impact, Low-probability Events Lessons from Eyjafjallajkull*. Chatam House.

McDermott, D.; Ghallab, M.; Howe, A.; Knoblock, C.; Ram, A.; Veloso, M.; Weld, D.; and Wilkins, D. 1998. PDDL – the planning domain definition language. Technical report, Yale Center for Computational Vision and Control.

Ong, S.; Png, S.; Hsu, D.; and Lee, W. 2009. POMDPs for robotic tasks with mixed observability. *Robotics: Science and Systems*.

Paulos, J.; Eckenstein, N.; Tosun, T.; Seo, J.; Davey, J.; Greco, J.; Kumar, V.; and Yim, M. 2015. Automated self-assembly of large maritime structures by a team of robotic boats. *IEEE T. Automation Science and Engineering* 12(3):958–968.

Pineau, J.; Gordon, G.; and Thrun, S. 2006a. Anytime point-based approximations for large POMDPs. *Journal of Artificial Intelligence Research* 27:335–380.

Pineau, J.; Gordon, G.; and Thrun, S. 2006b. Anytime point-based approximations for large POMDPs. *Journal of Artificial Intelligence Research* 27:335–380.

Pryor, L., and Collins, G. 1996. Planning for contingencies: A decision-based approach. *Journal of Artificial Intelligence Research* 4:287–339.

Ross, S.; Pineau, J.; Paquet, S.; and Chaib-draa, B. 2008. Online planning algorithms for pomdps. *Journal of Artificial Intelligence Research* 32:663–704.

Sanner, S., and Boutilier, C. 2009. Practical Solution Techniques for First-Order MDPs. *Artificial Intelligence* 173(5-6):748–788.

Woods, M.; Shaw, A.; Barnes, D.; Price, D.; Long, D.; and Pullan, D. 2009. Autonomous science for an ExoMars Rover-like mission. *J. Field Robotics* 26(4):358–390.

Mission		Opp plan	Full replan	Plan duration		
Main	Opp	time	time	Opp Mission	Complete Opp Plan	Replanned plan
V2_400	I_16	0.36	38.18	851.384	1265.032	2437.496
V2_500	I_16	5.54	7.46	1541.168	2076.155	2596.156
V2_600	I_16	5.34	7.28	1541.168	2117.136	2269.701
V2_700	I_16	5.32	9.56	1541.168	2117.136	2283.134
V2_800	I_16	5.38	6.24	1541.168	2117.136	**2048.833**
V2_900	I_16	5.4	9.16	1541.168	2117.136	**1900.069**
V2_1000	I_16	0.38	21.42	851.384	1265.032	2615.245
V2_1100	I_16	0.34	7.28	888.554	1302.202	2048.833
V2_1200	I_16	2.4	11.9	1440.568	1854.216	2511.960
V2_1300	I_16	0.36	6.34	851.384	1265.032	2772.985
V2_1400	I_16	0.42	6.28	851.384	1265.032	2772.985
V2_1500	I_16	0.34	7.82	851.384	1265.032	2946.391
V2_1600	I_16	0.38	14.54	851.384	1265.032	2175.901
V2_1700	I_16	0.4	15.6	851.384	1265.032	2897.665
V2_1800	I_16	0.42	6.24	851.384	1265.032	2772.985
V2_1900	I_16	0.38	6.44	851.384	1265.032	2772.985
V2_2000	I_16	0.36	2.62	851.384	1265.032	2490.490
V2_400	I_32	5.08	148.17	2233.961	2564.254	3531.784
V2_500	I_32	2.2	165.62	1768.98	2129.213	5332.514
V2_600	I_32	3.7	78.19	1777.177	2137.41	3623.974
V2_700	I_32	4.08	272.84	1815.849	2176.082	4877.45
V2_1000	I_32	4.66	104.04	2686.638	3093.992	4263.605
V2_2000	I_32	4.32	100.16	2457.922	2865.276	3778.601
V2_2500	I_32	4.67	68.78	2457.922	2865.276	4212.37
V2_3000	I_32	4.36	132.32	2469.124	2876.478	3948.493
V2_3500	I_32	4.21	119.14	1861.244	2191.537	4925.07
V2_5000	I_32	5.16	81.04	1997.34	2327.633	5460.531
V2_1000	I_2	0.02	3.36	141.138	678.167	**580.58**
V2_1000	I_6	0.06	182.02	374.415	911.444	***774.337***
V2_1000	I_8	0.06	4.82	504.346	863.481	977.001
V2_1000	I_10	0.1	135.62	582.795	1017.846	1383.791
V2_2000	I_10	0.1	7.44	700.198	1294.007	1585.303
V2_2000	I_12	0.14	3.38	772.926	1545.852	**1414.551**
V2_2000	I_14	0.14	3.70	675.458	1141.406	2040.448
V2_2000	I_16	0.14	2.60	676.458	1142.406	2490.490
V2_2000	I_18	0.18	44.68	878.395	1470.264	3116.591
V2_2000	I_20	0.44	15.18	1231.22	1767.021	3358.226
V2_2000	I_22	1.08	17.42	1752.10	2152.423	4114.774
V2_2000	I_24	0.98	34.48	1643.92	2017.172	2914.554
V2_2000	I_26	2.6	289.40	2141.87	2485.068	6029.480
V2_2000	I_28	3.38	265.72	3088.31	3667.984	5987.822
V2_2000	I_30	-	179.76	-	398.45	4187.185
V2_2000	I_32	4.14	218.74	2689.84	3057.283	4475.005
V2_2000	I_34	5.24	89.32	3125.49	3621.695	4615.062

Table 1: Table of experimental results. Planning time and plan durations are measured in seconds.