Proceedings of the Twenty-Seventh International Conference on Automated Planning and Scheduling (ICAPS 2017)

# Boosting Search Guidance in Problems with Semantic Attachments

**Sara Bernardini**
Department of Computer Science
Royal Holloway, University of London
Egham, Surrey, UK, TW20 0EX
*sara.bernardini@rhul.ac.uk*

**Maria Fox** and **Derek Long**
Department of Informatics
King's College London
London, UK, WC2R 2LS
*firstname.lastname@kcl.ac.uk*

**Chiara Piacentini**
Department of Mechanical
and Industrial Engineering
University of Toronto
Toronto, Canada, ON M5S
*chiara.piacentini@utoronto.ca*

**Abstract**

Most applications of planning to real problems involve complex and often non-linear equations, including matrix operations. PDDL is ill-suited to express such calculations since it only allows basic operations between numeric fluents. To remedy this restriction, a generic PDDL planner can be connected to a specialised advisor, which equips the planner with the ability to carry out sophisticated mathematical operations. Unlike related techniques based on semantic attachment, our planner is able to exploit an approximation of the numeric information calculated by the advisor to compute informative heuristic estimators. Guided by both causal and numeric information, our planning framework outperforms traditional approaches, especially against problems with numeric goals. We provide evidence of the power of our solution by successfully solving four completely different problems.

## 1 Introduction

Planning for real-world applications usually requires temporal domain models with complex numeric effects that involve algebraic expressions, analytic functions, non-linear equations and matrix calculations (e.g. the Space Telescope Slew Manoeuvre domain (Löhr et al. 2012) and the Machine Tool Calibration domain (Parkinson et al. 2014)). PDDL (Fox and Long 2003) is inadequate to express these calculations as it only allows basic operations between numeric fluents. To cope with this problem, an external advisor is typically attached to a PDDL planner and tasked with handling complex mathematical operations. However, existing solutions provide limited heuristic guidance about such operations and are not general enough to encompass all the temporal and numeric features that can be found in real-world problems. In this paper, we present our approach to combining a specialised advisor with a generic PDDL planner, which improves the state of the art in two ways: (i) we provide the planner with heuristic insights into the effects of the advisor on numeric state variables, which enhances search control; and (ii) we demonstrate capabilities in four completely different domains borrowed from the literature, attesting to the generality of our approach.

More specifically, complex mathematical operations have so far been dealt with *semantic attachments*. Introduced by Weyhrauch (1980) as a way to interpret predicate symbols using external procedures, semantic attachments have been incorporated in PDDL as *modules* (Dornhege, Eyerich, and Keller 2009), extending the PDDL language into PDDL/M. However, the TFD/M planner has no heuristic guidance over the effects of semantic attachments and often judges problems unsolvable because it cannot identify any actions that can help achieve the goals. Action descriptions, in fact, play two roles for a planner: (i) allowing it to predict the states resulting from action application; and (ii) helping it identify actions that are relevant and useful to achieve state conditions. This second role is key in guiding search. Without guidance, the planner is forced to resort to blind search, progressing states and probing their properties. TFD/M also does not support a full range of temporal plan structures (it uses a constrained choice in which new actions are restricted to start only at the start of the plan or immediately after other actions have started or ended).

We propose a principled and general technique to build approximations of the numeric information calculated by an advisor and use them to compute effective heuristic values. Since we integrate the approximations in the domain (as an extra effect of the actions that rely on the advisor to calculate their exact numeric effects), the planner can use them directly in the standard heuristic evaluation, without the need to build a specialised heuristic. We can also dynamically inform this heuristic using the external advisor to rapidly compute first order approximations of effects that can be transparently exploited within the heuristic computation.

Although our method is general, we implement it in the context of a powerful planner, POPF-TIF (POPF with Timed Initial Fluents) (Piacentini et al. 2015). This planner supports required concurrency, metric variables, predictable exogenous events and external advisors. The combination of a powerful planner with good heuristic guidance, both causal and numeric, allows us to successfully solve, within the same framework, complex problems in four very diverse domains: (i) Earth Observations problem (Aldinger and Löhr 2013); (ii) the Hydraulic Blocks World problem (Ivankovic et al. 2014); (iii) the Voltage Control problem (Piacentini et al. 2015); and (iv) large scale Search-and-Tracking missions (Bernardini et al. 2016). When possible, we compare our solution to related techniques, providing evidence of the benefits of our approach.

## 2 Semantic Attachment in PDDL

Semantic attachments (Dornhege, Eyerich, and Keller 2009; Piacentini et al. 2015) evaluate fluents using externally specified functions. Here we consider numeric fluents only.

We implement a simple interface between the planner and the specialised advisor. The interface works by categorising the numeric variables $V$ into three sets:

(i) $V^{ind}$ (*indirect* variables) are calculated by the external advisor based on the context provided by the planner;

(ii) $V^{dir}$ (*direct* variables) are determined by the planner and their values affect the $V^{ind}$ variables;

(iii) $V^{free}$ (*free* variables) are evaluated by the planner and do not result in any external computation.

When the planner queries the advisor, the planner passes the $V^{dir}$ variables and the advisor hands back the $V^{ind}$ variables. Hence, the values of the $V^{ind}$ variables can be seen as *indirect effects* of the choices made by the planner. Formally, we define a *semantic attachment* as a function $\phi$ that depends on the variables $V^{dir}$ and calculates the values of the variables $V^{ind}$, $\phi : \vec{V}^{dir} \rightarrow \vec{V}^{ind}$.

Goals and conditions can be expressed in terms of all of the variables in $V^{dir} \cup V^{free} \cup V^{ind}$. Action effects can affect only the variables in $V^{dir} \cup V^{free}$ , but cannot directly modify the $V^{ind}$ ($V^{ind} \cap V^{dir} = \emptyset$). However, whenever a change occurs in one of the $V^{dir}$ variables by means of the direct effects of an action, the state is updated by also considering the *indirect effects* of such action since the values of the $V^{ind}$ variables are immediately calculated by the advisor according to the new values of the $V^{dir}$ variables.

Indirect effects are a special case of PDDL+ *events* (Fox and Long 2006). Events are similar to actios, in that they trigger transitions from one state to another, but they necessarily apply as soon as their preconditions are satisfied. Indirect effects can be seen as events that are provoked by changes in the $V^{dir}$ variables.

The $V^{ind}$ variables can also be seen as a generalisation of *derived predicates* (Hoffmann and Edelkamp 2005) involving numeric fluents, where the $V^{dir}$ and $V^{free}$ are the basic variables and the function $\phi$ is the rule that determines the values of the derived variables. In PDDL, however, derived predicates are limited to propositional variables, while in our framework we focus on numeric fluents.

As an alternative to events and derived predicates, relations between variables (either numeric or propositional) can be modelled via *global constraints* (Ivankovic et al. 2014) on the values that they can assume. The variables are divided in two classes, *primary* and *secondary*, which are similar to the direct and indirect variables in our framework. The difference is that the secondary variables that appear in the global constraints do not need to be assigned to specific values as long as they respect all the constraints, while in our case the $V^{ind}$ variables have specific values, which are calculated by the advisor. One limitation of the global constraints is that they only capture the relations between variables at the same time point. In our setting, instead, it is possible to store the values of the $V^{ind}$ variables at a certain time step by using auxiliary $V^{dir}$ variables, allowing the specialised advisor to access their values in calculations at subsequent time points. Finally, although global constraints can be incorporated in delete relaxation heuristics (Ivankovic et al. 2014), this requires explicitly calling the external constraint solver to check the satisfiability of an action, which is a computationally expensive operation.

In previous works, semantic attachments are supported via external modules. In particular, TFD/M (Eyerich, Mattmüller, and Röger 2009) allows three types of modules: condition-checkers, effect-applicators and cost-calculator. The actions in the model must be annotated with the specific external modules that the planner needs to invoke to check the conditions and to calculate the effects. Therefore, PDDL is extended into PDDL/M, which allows such annotations.

Our approach to semantic attachment is related to Planning Modulo Theories (PMT) (Gregory et al. 2012). PMT is an extension of planning to support first order theories as parameters: new types (e.g. sets) can be added in the domain as modules to allow custom operations (e.g. set unions, set intersections). The core planner exploits abstractions of the theories to guide the search. For new types, these abstractions are automatically derived by sampling reachable values from the initial state. Semantic attachment can be seen as a form of PMT where an external module constitutes a new theory. A key difference between our work and PMT is that PMT offers a way to construct heuristic guidance that is specifically designed for structured types, while we propose an approach for non-linear functions of numeric variables.

Other approaches to semantic attachment, such as OPL (Hertle et al. 2012) and PDDLx (Bajada 2016), are compatible with our framework, which is independent of the syntactic form in which the attachments are specified. However, we have no direct experience with them.

## 3 Benchmark Domains

We show now how our benchmark domains (available at *https://github.com/popftif/popf-tif*) can be modelled by exploiting $V^{ind}$, $V^{dir}$ and $V^{free}$ variables.

### 3.1 Earth Observation Domain

The EO problem (Aldinger and Löhr 2013) concerns finding a sequence of poses for a satellite to maximise the number of observations. The satellite needs to perform a series of slew manoeuvres to reach the right conditions for executing an observation. Hence, the angles between *x,y,z*-axes, the angular rate of the satellite and the scan velocity of an observation site must be calculated after every manoeuvre.

This domain was first modelled using PDDL/M and solved with TFD/M. The model consists of three actions: positioning of the satellites to point towards an observation site, making an observation and skipping a site. A plan is found when every observation site has been either scanned or ignored. A `condition-checker` module is introduced to verify that the satellite can transition from one observation site to the next one. An `effect` module calculates the changes on the numeric variables after each scan operation, while `cost` modules determine the duration of each action.

The EO problem finds an easy and compact model using the formalism in Section 2. The $V^{ind}$ variables are:

the action duration, the angular rates and values and auxiliary variables that represent whether the satellite can transition between observation sites. The values of these variables depend on the observation site that has been targeted or scanned, which is part of the $V^{dir}$ variables.

### 3.2 Hydraulic Blocks World (HBW) Model

The HBW domain was introduced to explore the use of global constraints to filter valid states (Ivankovic et al. 2014). This domain is an extension of blocks world (BW) where the towers of blocks sit on pistons that are inside vertical cylinders connected to a reservoir of hydraulic fluid. The columns can be of different areas and the blocks can have different sizes and weights. The goal of the planning problem is to reach a given configuration of blocks, avoiding any piston going above the top or below the bottom of its cylinder. Whenever a block is put down or picked up from a cylinder, the total weight of the column changes and the fluid rearranges itself to balance the forces in the system.

The HBW domain can be modelled in PDDL similarly to the traditional BW domain, with the addition of metric information. In HBW, there are two numeric variables per cylinder, $w_k$, the weight of the tower in cylinder $k$, and $h_k$, the height of the fluid in cylinder $k$, as well as four constant parameters: $V$ (total fluid volume), $a_k$ (cross section of cylinder $k$), $\rho$ (density of the fluid), and $h_k^{max}$ (maximum height of cylinder $k$). The usual BW actions *pickup/putdown* and *stack/unstack* have an additional parameter that represents the cylinder in which the block that is to be moved rests. When one of these actions is performed on block $b$ in cylinder $k$, its effects change the weight $w_k$ of the tower in $k$, which will increase or decrease a quantity corresponding to the weight of $b$, and the height $h_k$ of the fluid.

While the change in weight can be directly modelled in PDDL by using the INCREASE and DECREASE operators in the actions' effects, the same does not hold for the global change in the height of the fluid in every cylinder. This is where semantic attachment comes into play: we handle the change in height by calling an external advisor that encapsulates the fluid dynamics. Thus, in our formulation of the HBW, the weight variables $w_k$ are the $V^{dir}$ and the height variables $h_k$ are the $V^{ind}$. A specialised advisor calculates the $V^{ind}$ based on the $V^{dir}$ by implementing the following system of equations that describe the fluid dynamics:

$$\begin{cases} \sum_{k=1}^{m} a_k h_k = V \quad (m = number\ of\ cylinders) \\ \rho\,(h_k - h_{k+1}) = w_{k+1}/a_{k+1} - w_k/a_k \ \forall k = 1, ..., m-1 \end{cases}$$

### 3.3 Voltage Control Model

The VC problem is a power system problem where the goal is to maintain the voltage of the busbars of an electrical network within safety boundaries. In power systems, the electrical current follows sinusoidal waves, so the relevant quantities are related to each other by non-linear equations. Given a network with $N$ busbars, the voltage of a busbar $k$ depends on the power injected and ejected through a set of non-linear equations, know as *power flow equations*.

In the VC problem, part of the power cannot be controlled as it is given by the customers' behaviours and the level of power generated by the units. For the part that can be controlled, electrical devices are used: capacitor banks, which can inject a given amount of reactive power, and transformers, which modify the reactance of a line. Planning can be used to choose the configuration of the controllable devices that keeps the voltage within given constant limits, while the power consumed and generated are changing over time. In the work by Piacentini et al. (2015), a specific solution to formulate the VC problem in PDDL2.2 with semantic attachment is presented, where the voltages of the busbars are the $V^{ind}$ variables, which are calculated by the advisor via the following equations:

$$P_k = V_k \sum_{n=1}^{N} V_n \left[G_{kn} \cos\left(\delta_k - \delta_n\right) + B_{kn} \sin\left(\delta_k - \delta_n\right)\right]$$

$$Q_k = V_k \sum_{n=1}^{N} V_n \left[G_{kn} \sin\left(\delta_k - \delta_n\right) - B_{kn} \cos\left(\delta_k - \delta_n\right)\right]$$

where $Q$ is the reactive power, the root mean square of the product of current and voltage, $P$ is the real power, $\delta$ is the phase angle of the voltage and $G_{kn}$ and $B_{kn}$ are constants called, respectively, reactance and susceptance of the line between busbars $k$ and $n$. The variables $P_k$ and $Q_k$ constitute the set of $V^{dir}$.

### 3.4 Search-and-Track Model

In a Search-and-Track (SaT) mission (Stone 1975), a searching vehicle, the observer, wishes to locate the position of a moving object, the target, and to track it to a destination upon finding it. Bernardini et al. (2016) propose a solution to SaT based on automated planning in which standard search patterns (e.g. spirals and lawnmowers) are employed by the observer to survey the search region. Starting from an initial pool of promising candidate patterns $\Sigma$, which are scattered throughout the search area, the planner finds a sequence of patterns in $\Sigma$, $S = (\sigma_1, \sigma_2, \ldots, \sigma_{\bar{k}})$ that maximises the probability of finding the target. Based on the target's last known position and a set of possible destinations $\mathcal{D}$, the planner populates the plan $S$ by making predictions about which destination the target is aiming for. The planner updates these predictions over time in view of the target's motion and the results of the previous searches. In a Bayesian fashion, at every iteration $k$, the planner calculates the probability of finding the target within $k$ time steps:

$$P^{(k)}(S) = P^{(k-1)}(S) + P_{S*}^{(k-1)}(1 - P^{(k-1)}(S)) \quad (1)$$

with $P^{(0)}(S) = 0$ and $P_{S*}^{(k-1)}$ being the probability that the target is found during the execution of pattern $\sigma_k$ conditioned to the event that it has not been discovered earlier. This term depends on the probability that the observer finds the target when it is in view, i.e. the detection probability $\gamma_{\sigma_k}$ and on the probability that the target has chosen any of the destinations compatible with $\sigma_k$. The probability that the target is moving towards $x$ at step $k$ after executing $S$ can be

formulated as follows:

$$P_S^{(k)}(x) = \begin{cases} \frac{P_S^{(k-1)}(x)\,(1-\gamma_{\sigma_k})}{1-P_{S*}^{(k-1)}} & \text{if } x \in \mathcal{D}_{\sigma_k} \\ \\ \frac{P_S^{(k-1)}(x)}{1-P_{S*}^{(k-1)}} & \text{if } x \notin \mathcal{D}_{\sigma_k} \end{cases} \quad (2)$$

with $P_S^{(0)}(x) = \frac{1}{d}$ since a uniform prior PD for the destinations is initially assumed and $\mathcal{D}_{\sigma_k}$ indicating the destinations compatible with the pattern $\sigma_k$.

We use an advisor to calculate Equations 1 and 2 at each iteration. The destination and the total probabilities, $P_S^{(k)}(x)$ and $P^{(k)}(S)$, are the $V^{ind}$ variables calculated by the advisor. They depend on their own values in the preceding state and on the last pattern added to the plan ($P_S^{(k-1)}(x)$ and $P^{(k-1)}(S)$). We create auxiliary $V^{dir}$ variables to store these values, which are updated when a new search pattern is executed, triggering a call to the external advisor.

## 4 Combining POPF-TIF with an Advisor

POPF-TIF, built on top of POPF2 (Coles et al. 2010; 2011), is a PDDL planner based on a forward heuristic search engine and exploits partial ordering. The partial ordering is obtained by delaying the commitment to ordering the actions until some constraint forces it. The planner performs Enforced Hill Climbing (EHC) with helpful actions pruning and resorts to Best First Search (BFS) when EHC does not find a solution. The planner uses a temporal extension of the Metric-FF heuristic (Hoffmann 2003), TRPG (Temporal Relaxed Planning Graph) (Coles et al. 2010), to guide the search: facts and action layers are annotated to record the minimum time-step at which they appear.

We create a modular architecture to handle semantic attachments by coupling POPF-TIF with user-defined advisors (Figure 1). Our framework is open source and can be downloaded at *https://github.com/popftif/popf-tif*.

The planner receives the PDDL domain, the problem files and the advisor as inputs. The model contains all the variables, while their classification into $V^{ind}$, $V^{dir}$ and $V^{free}$ is stored in the advisor. The role of the advisor is to compute the $V^{ind}$ variables whenever it is invoked by the planner, which happens when the planner applies an action that changes the $V^{dir}$ variables. The planner can access the values of the $V^{ind}$ variables that are calculated by the advisor, but does not require a declarative description of how the action effects change them. Such an abstraction is needed whenever actions have effects that cannot be modelled in an action-based declarative language such as PDDL.

An important advantage of this abstraction is that the user can define a *relaxation* of the effects on the $V^{ind}$ that the planner can use during the heuristic evaluation of a state. Hence, the planner does not apply actions in a way that is blind to the semantic attachment, but instead exploits it to approximate its effects and to evaluate alternative states in the search space. In the state progression phase of the search, the $V^{ind}$ variables are updated by the advisor, while in the TRPG construction an approximation of the effects is used.

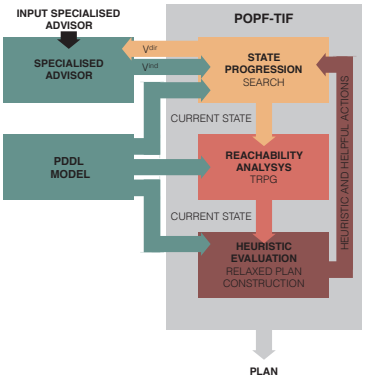


Figure 1: A general framework to incorporate an external advisor into a generic PDDL planner.

A domain designer can achieve a semantic attachment in any domain by adhering to the following procedure: 1) Identify the $V^{ind}$, $V^{dir}$ and $V^{free}$ variables, set up the advisor for calculating the $V^{ind}$ variables and establish a communication line between the planner and the advisor, which uses the $V^{dir}$ variables. This mechanism is domain independent. 2) For each variable $v \in V^{ind}$, identify an approximation of how $v$ changes (see Section 4.2) and add it in the domain as an extra effect of those actions that rely on the advisor to calculate the exact value of $v$.

### 4.1 State Evaluation and External Advisor

When the planner updates a state, the $V^{free}$ and the $V^{dir}$ are calculated first. If there is a change in one of the $V^{dir}$, then the $V^{ind}$ values are calculated by calling the advisor, using as input all the values of the $V^{dir}$ and other optional parameters that can be provided to the advisor.

For example, in the HBW domain, when a block $b$ is unstacked from a cylinder $k$, the planner starts by calculating the effects on the variable $w_k$, expressed in PDDL as `(decrease (weight-c k) (weight-b b))`, where the fluent `(weight-c k)` represents $w_k$ and `(weight-b b)` is the constant weight of the block $b$. Then, the planner invokes the advisor and provides it with the updated values of $w_i\ \forall i = 1, ..., m$. These values, together with the other constant parameters ($\rho$, V, $a_i$), are used by the advisor to calculate the values of all the variables $h_i\ \forall i = 1, ..., m$, which are then communicated back to the planner.

POPF-TIF is implemented in C++ and the external advisor is a dynamically loaded shared library, which is given to the planner as input when invoked. This allows the user to implement different external advisors for different purposes. To successfully load the advisor, the planner and the advisor need to share a common interface. Since the advisor deals only with numeric variables, without any continuous effect, a state in the advisor is given by a map from the names of the fluents to their values. The interface consists of:

(i) a method to initialise the advisor, possibly loading input parameters;

(ii) the lists of the $V^{dir}$ and $V^{ind}$ variables, which are

identified by the names of their corresponding fluents in the PDDL domain;

(iii) a method that calculates and updates the $V^{ind}$ based on the $V^{dir}$.

The planner calls the advisor only when the application of an action changes the $V^{dir}$.

## 4.2 Heuristic Guidance and Approximations

In our framework, we provide the planner with some coarse information about this change to help it guide its choice of actions. In addition, if the goal refers to the $V^{ind}$ variables, we include an approximation of their values in the model to support the planner in recognising what actions are helpful for achieving the goal. This approach offers a general and principled way to harness the heuristic content of the model within the standard planning approach, while still using the external advisor to manage state progression.

The relaxation of numeric state variables used in POPF-TIF for reachability analysis and heuristic evaluation is based on the same principles introduced in Metric-FF (Hoffmann 2003), which can handle only numeric effects expressed in *linear normal form* (LNF). In our framework, however, the values of the $V^{ind}$ for the next states are not accessible to the planner to calculate the heuristic estimators, so we provide the planner with a LNF approximation of them. In principle, more accurate and general approximations can be provided, depending on the capability of the planner. The simplest effect that can be used is a linear effect of first degree: $v \mapsto v + c$ where $c$ is a constant and $v \in V^{ind}$. This effect indicates whether an action increases ($c > 0$), decreases ($c < 0$) or is irrelevant ($c = 0$) to $v$. The value of $c$ should be chosen so that it does not underestimate the real effect on $v$, otherwise possible values might be excluded in the reachability analysis. As proposed by Hoffmann (2003), if the action can be applied multiple times and increases (or decreases) the fluent value, then the upper (or lower) bound of the reachable values becomes $+\infty$ (or $-\infty$), making all the values of the variable $v$ reachable. In this case, the value of $c$ becomes relevant during the relaxed plan extraction, if the action is required to satisfy a numeric goal. This value determines the number of applications of the action, therefore changing the length of the relaxed plan. The approximation of these effects is left to the modeller, who can specify in the model an appropriate linear effect on the $V^{ind}$ variables.

The use of an approximation of the effects on the $V^{ind}$ variables becomes particularly important when there are *numeric goals* that involve them. In this case, the heuristic must be able to discriminate the actions that are helpful to achieve such goals among all actions. To favour the identification of such actions, we consider different approximations, which reflect different compromises between the accuracy of the approximation and the informativeness of the heuristic. Given a variable $v \in V^{ind}$, a set of actions $\mathcal{A}$ that affect $v$ and the approximated effect $v \mapsto v + c$, we calculate $c$ in different ways, from the least to the most accurate:

(i) $c$ is equal in each $a \in \mathcal{A}$ and its order of magnitude does not necessarily reflect the exact effects calculated by the advisor;

(ii) $c$ is equal in each $a \in \mathcal{A}$ and its order of magnitude reflects the exact effects calculated by the advisor;

(iii) $c$'s absolute value is equal in each $a \in \mathcal{A}$, but its polarity reflects the exact effects calculated by the advisor;

(iv) $c$ is different for every $a \in \mathcal{A}$ and is computed based on the initial state as a first-order approximation of the exact effects calculated by the advisor;

(v) $c$ is different for every $a \in \mathcal{A}$ and is dynamically computed for each state by the advisor along with the exact numeric effects as a first-order approximation of them.

Approximations (iii)-(v) are increasingly more informative and realistic: case (iii) allows the planner to see not only that an effect on the $V^{ind}$ exists, but also its polarity; case (iv) and (v) expose both the magnitude and the polarity of the changes in the $V^{ind}$ to the planner. Cases (i) and (ii) are rather unrealistic. We consider them only to analyse the impact of different approximations on the heuristic guidance.

**Heuristic Guidance for the Earth Observation Domain** In the original domain, there is only one condition on the $V^{ind}$, which involves the auxiliary variable that represent the `condition-checker` module. This condition requires the torque of the satellite not to exceed a conservative upper-bound, ensuring that there is a feasible slew manoeuvre between two observation sites. As a simple approximation, we consider that every action makes this condition true.

**Heuristic Guidance for the Hydraulic Blocks World** We equip the HBW model with information on how moving a block affects the fluid heights in the cylinders. This helps the planner guide search. First, we add the invariant condition $0 \leq h_k \leq h_k^{max}, \forall k = 1, ..., m$ to all the actions. In addition, when the goal involves the $V^{ind}$ variables, we provide the planner with an approximation of their values, which helps the planner identify the actions that are relevant to achieve the goal. For example, suppose that the goal is `(≥ (height-column C1) 6)`, where the heights of the columns are the $V^{ind}$ variables. To guide search, the planner must be able to identify which actions have effects that can change these values. Supplying the model with an approximation of how the column heights change serves this purpose, even if the approximation is not precise. When unstacking a block $b$ from a cylinder $c$, we expect the fluid in the cylinder $c$ to rise and the fluid in the other cylinders to drop. We can express an approximate effect on the $h_k$ in PDDL via a universally quantified effect in the pickup action:

```
(forall (?d - column) (and
    (decrease (height-column ?d)
       (approximation ?d ?c))))
```

where `(approximation ?d ?c)` denotes constant values, instantiated by the modeller in the initial state, positive when `?d` and `?c` are two distinct cylinders and negative otherwise. Any suitable approximation can be used. For example, we can set that the total increase in the height of the fluid in cylinder $c$ is equal to the sum of the fluid displaced from the other two cylinders, each of which are inversely proportional to the areas of the cylinders (assuming that the blocks have constant weight). This approximation corresponds to method (iv) as described in Section 4.2. A simpler and less

informative approximation would be just a constant, representing the fact that the heights in the other columns have decreased by a non-zero value (approximations (i) or (ii)). This approximation is only used by the planner when evaluating the goodness of reachable states.

Since in HBW all the actions have the same preconditions on the $V^{ind}$ variables, these preconditions must hold in every state. Hence, if the goal never mentions the $V^{ind}$ variables, then no approximation is needed and the planner can be blind to how the advisor changes them. The advisor will of course still need to make sure that the values of the $V^{ind}$ variables are properly computed, to ensure state consistency. For example, even if none of the goals refer to the heights of the columns, the advisor must still exclude states that violate the fluid constraints as blocks are moved around.

**Heuristic Guidance for the Voltage Problem** For the voltage problem, the heuristic computation uses an abstraction of the network that is based on a first order approximation of the effects of the actions expressed in the domain, using problem-specific constants provided in the initial state:

```
(forall (?b - bus) (at start
  (increase (voltage ?b) (step-max ?t ?b))))
```

where `(voltage ?b)` are the voltages of the busbars and `(step-max ?t ?b)` are constant values depending on the transformer that was changed and the busbars.

The values of these fluents can be obtained in a preprocessing stage by using different approximations. One way to compute them is to assign a uniform value to the busbars within a given distance from the transformer (approximation (iii) as described in Section 4.2). A more accurate way is obtained by taking, for each busbar, the difference between the voltage at the first time point and the voltage after applying a transformer change (approximation (iv)).

The `(step-max ?t ?b)` values are used in the heuristic evaluation whenever a predicted change of the load or generation profiles rise (lower) the voltage level above (below) the admitted limit. The heuristic computation adds the most appropriate set of actions to the relaxed plan, according to the approximated effects, to re-achieve the numeric goal, changing the heuristic value and the set of helpful actions of the state. The voltage values are then computed by the advisor when the successor states are expanded in the search.

**Heuristic Guidance for Search-and-Track** In the heuristic evaluation of SaT problems, the actions that correspond to the execution of search patterns have an approximate effect on the total probability, which corresponds to a constant value that is set in the initial state and is dependent on the pattern that is executed:

```
(at end (increase (total-probability)
                  (heuristic-approximation ?p)))
```

In this domain, we are interested in obtaining good quality solutions in terms of the specified metric (usually, probability to discover the target). POPF-TIF achieves that by using an any-time search, where the value of the previous solution's metric is the lower limit of an additional metric goal that the following solution must achieve. The approximate effects on the total probability are therefore used in the heuristic computation to select the actions (search patterns) to add in the relaxed plan to achieve this additional numeric goal. To add discriminatory power to the search patterns in the heuristic guidance, we extend our external advisor to calculate the effect `(heuristic-approximation ?p)` after a search pattern action is inserted in the plan. This fluent becomes a $V^{ind}$ variable, which is calculated by the external advisor by applying each search pattern to the current state (approximation (v) as described in Section 4.2). This results in calls to the external advisor that are more expensive, but also more informative for the heuristic evaluation.

| # obs sites | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|
| TFD/M | 0-0.24 | 1 | 1 | 1 | 3 | 7 | 16 | 34 |
| POPF-TIF | 0.17 | 0.52 | 4.33 | 4.99 | 8.78 | 9.40 | 11.82 | 28.70 |

Table 1: Time to find the best quality solution for the EO problems using POPF-TIF and TFD/M.

# 5 Experimental Results

We encoded all the benchmark domains within our framework and devised heuristic approximations for them by following our principled approach. We use several heuristic approximations in each case and compare them. Whenever possible, we also compare our solution with other approaches and, to this aim, we built modules for TFD/M for two of the four domains. Our experimentation demonstrates that: (i) from a knowledge engineering perspective, our framework is general enough to capture very diverse domains; and (ii) from a performance perspective, our approach is always comparable with related techniques, but outperforms them on the most sophisticated problems.

## 5.1 Earth Observations

For the EO domain, we compare our planner with TFD/M using the benchmark by Aldinger and Löhr (2013). In Table 1, we show the times at which the best quality solution is found. We consider problems with an increasing number of observation sites. Both planners use an anytime search algorithm (i.e. the search algorithm does not stop when finding the first solution, but keeps generating increasingly better plans until the time-out expires). Since in TFD/M the time at which each solution is found is not shown, we provide the approximative range at which it is found. The upper bound is the time when TFD/M exhausts the exploration of the search space or the time of the first report after the solution is found. Table 1 shows that, in this domain, our framework and TFD/M have comparable performance.

## 5.2 Hydraulic Blocks World

**Planners** We evaluate our framework against the HBW domain by comparing it with two alternative planners: PREFPEA*(Ivankovic et al. 2014) and TFD/M (Dornhege, Eyerich, and Keller 2009). PREFPEA* is an optimal planner that has been specifically built to solve problems with global numerical state constraints, such as HBW.

Since the three planners use different input languages, we model HBW in three different ways. In PREFPEA*, the model exploits global constraints to capture the physical laws described in Section 3.2 and the additional constraint:

$0 \leq h_k \leq h_k^{max}$ $(\forall k = 1, ..., m)$. In TFD/M and POPF-TIF, we do not directly model the HBW equations, but the action effects on $h_k$ are calculated by the external advisor as black boxes. The invariant $0 \leq h_k \leq h_k^{max}$ $(\forall k = 1, ..., m)$ is expressed in PDDL by adding it to all the actions as an `over all` condition and to the goal.

The difference between TFD/M and our framework lies in how the external advisor is handled and exploited. In our approach, there is no need to specify in the model when the external advisor is required because the planner invokes the advisor whenever a change in the $V^{dir}$ variables (i.e., $w_k$) occurs. Instead, in TFD/M, the effect-applicator module requires explicit annotation and it is the responsibility of the modeller to call it at the right time, e.g. when at least one $w_k$ is updated. Moreover, when TFD/M evaluates a state, its external modules use the numeric values of the preceeding state. Hence, the update of $w_k$ and $h_k$ (calculated via the module) must happen at two distinct time points (at the start and at the end of the action, respectively). This delay in the update of the $h_k$ is negligible in HBW, but it can create problems in domains with a rich temporal structure.

**Problems and Results** For evaluating the three planners, we consider two sets of problems, which allow us to observe how the approximation of the $V^{ind}$ variables impacts on the heuristic guidance: (i) the *original* problem set (Ivankovic et al. 2014), which encompasses 120 randomly generated problems with between 4 and 7 blocks; and (ii) a *new* set of 80 problems, where we add a numeric goal on one of the columns in the original problem, which corresponds to the height that must be reached on that column.

When using the *original* set of problems, our planner is blind to the numeric effects on the heights on the column: if a state satisfies all the numeric invariants, these are considered always satisfied in the relaxed graph; if the state does not satisfy them, the state has no successors. When considering the *new* set of problems, the addition of numeric goals has an impact on the approximation of the effects on the $V^{ind}$ variables in the heuristic evaluation. If the state does not already satisfy the additional goal condition, the heuristic evaluation must choose an appropriate set of actions.

The experimental evaluation reveals that POPF-TIF and TFD/M have similar performance in terms of execution time, while PREFPEA* is systematically slower than the other two planners. Although PREFPEA* is an optimal planner while the other two are satisfying planners, most of the solutions found by POPF-TIF (60%) and TFD/M (77%) are optimal (where optimality in the HBW is determined by the number of actions in the plan), as show in Figure 2. Moreover, using an anytime search, both POPF-TIF and TFD/M find the optimal solutions on average faster than PREFPEA* in all but six and two cases, respectively.

As for a direct comparison between POPF-TIF and TFD/M, they achieve comparable results against the *original* set of problems because, in this case, the advisor does not contribute to the calculation of the heuristic, which is only driven by the propositional aspect of the problem. However, the situation is different when considering the *new* set of problems. In evaluating them, we run POPF-TIF with the three different constant approximation settings, and specifically cases (i), (ii) and (iv) of the approximations described in Section 4.2. Note that the first and the second approximations contradict the physical effects that are expected. We consider these case studies to test the impact of the approximations on the heuristic evaluation. Among the 80 *new* problems, TFD/M finds a solution only for 11 problems, while it mistakenly marks the other problems as unsolvable. It is worth noting that TFD/M finds solutions only in the accidental case in which the states that satisfy the propositional goals or their next successors satisfy the numeric goals as well. When additional search is required, the planner fails to recognise that other actions can be used to achieve the goal. On the other hand, POPF-TIF finds solutions for 71 problems with approximations (ii) and (iv) and 60 with approximation (i). The approximations contribute to the total number of states that the planner needs to evaluate and the quality of the solution (measured as plan length), as shown in Table 2. Note that the average plan length for TFD/M is shorter because TFD/M only finds solutions for the smallest problems, which require fewer actions. The results highlight that, for the heuristic to be effective, we must provide approximated effects with the right order of magnitude.

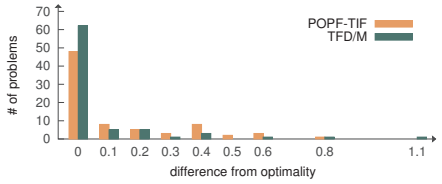


Figure 2: Relative distance from the optimality of solutions obtained by POPF-TIF and TFD/M , with respect to the optimal value found by PREFPEA* for the HBW problems.

| | POPF-TIF | | | TFD/M |
|---|---|---|---|---|
| | $h_{(iv)}$ | $h_{(ii)}$ | $h_{(i)}$ | |
| # problem solved | 71 | 71 | 60 | 11 |
| average #state evaluated | 829 | 904 | 698 | 150 |
| average execution time | 0.18 | 0.18 | 0.24 | 0.2 |
| average plan length | 20 | 20 | 50 | 8 |

Table 2: Average plan length and number of states evaluated for different heuristic approximations with POPF-TIF and TFD/M for the HBW.

## 5.3 The Voltage Control Problem

We compare the solutions of the VC problem that we get within our framework with those that can be obtained without semantic attachment. We cannot offer a direct comparison with TFD/M in this domain because it involves required concurrency, which TFD/M does not fully support.

For this domain, we consider two approximations of the $V^{ind}$ variables, corresponding to case (iii) and (iv) (respectively, AC $h_{(iii)}$ and AC $h_{(iv)}$) as described in Section 4.2.

In modelling the problem without relying on external advisors, we need to use a linearisation of the power flow equations. We model predictable numerical exogenous events, which happen when a change in the load and the genera-

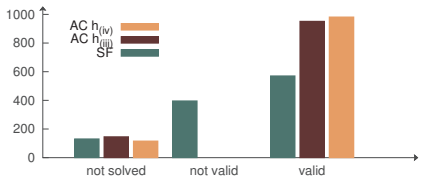


Figure 3: Number of problems solved for the VC problem without external advisor (SF) and with external advisor and heuristic approximations (iii) (AC $h_{(iii)}$) and (iv) (AC $h_{(iv)}$).

tion profile occurs, with TIFs. We divide the voltage into two components: the first does not depend on the actions, but only on the TIFs and can be calculated exactly by applying the power flow equations in a preprocessing step. The second component of the voltage takes into account the variation of the voltage that capacitors and transformers induce. This is the effect that is linearised providing a set of linear sensitivity factors (SF) (Wood, Wollenberg, and Shebleeacute 2013). The total voltage is the sum of the two components. We call this second model SF.

In our experiments, we consider the VC problem for the 33 kV distribution network AuRA-NMS (Davidson et al. 2010) and take the load profiles from the UK power grid (National Grid PLC 2012). We generate 1098 problem instances and solve the same instances with the two models. We validate the plans that we produce against the full power flow equations and determine if there are voltage violations. In the histogram in Figure 3, we report the total number of problems for which each approach: (i) cannot find any solution; (ii) finds an invalid solution; and (iii) finds valid solutions. As expected, using the AC model, POPF-TIF only produces valid solutions, while about 41% of the plans generated by the SF model are invalid. The total success rate of the AC model is 89% with the accurate heuristic (AC $h_{(iv)}$) and 87% with the less accurate one (AC $h_{(iii)}$), compared with 52% of the SF model. The difference between the two heuristics is in the quality of the solutions: with an accurate heuristic, the average plan length is 11, with the other is 21.

### 5.4 Search-and-Tracking

We conducted a series of experiments to assess the gain that we obtain by using our framework in the solution of SaT problems. The advisor allows the dynamic updating of the probabilities over time, which cannot be achieved via a purely deterministic planner. We cannot compare our approach with TFD/M in this domain because TFD/M does not support temporal initial literals, which are needed in modelling SaT as a planning problem.

We consider 14 routes that lead from Stirling (the target starting point) to the 14 most populated cities of Scotland and execute a SaT simulation on each route 1000 times (the simulation has a non-deterministic spotting model and target behaviour) for each of the 2 strategies. The observer is artificially blinded for 15 minutes after it loses the target to make the SaT problems more difficult.

Figure 4 shows how the total probability changes while executing the plans found by POPF-TIF for one of the problems in the dataset and the average among all the problems. In each plot, we show the results found in three cases: (i) without using the external advisor (no Adv) (Bernardini, Fox, and Long 2015); (ii) using the external advisor and a static approximation of the action effects on the total probabilities for heuristic guidance, i.e. approximation (iv) as described in Section 4.2 (Adv $h_{(iv)}$) (Bernardini et al. 2016); and (iii) using the external advisor and exploiting it to dynamically calculate the action approximated effects on the total probabilities at each step for heuristic guidance, i.e. approximation (v) as described in Section 4.2 (Adv $h_{(v)}$). On average, the total objective functions found in the three cases are 0.73, 0.82 and 0.86, respectively. From the plots, we observe that the use of an external advisor is instrumental in improving the quality of the plans found. In addition, we see that exploiting the external advisor to obtain more accurate numeric effects to use in the heuristic evaluation further contributes to obtain better quality plans.

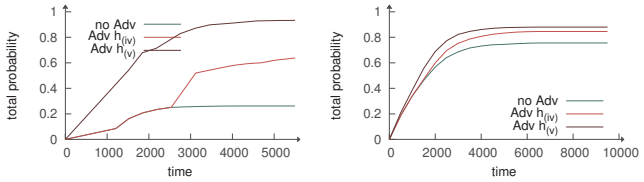


Figure 4: Total probability as a function of the execution time of the plan for the SaT problem for one problem (left) and for all the problems (right).

## 6 Conclusions

In this paper, we present a set of general principles to embed a specialised advisor into a generic PDDL planner. We focus on the method that we use to integrate approximations of the values calculated by the advisor in the domain, which can then be directly used in the standard heuristic evaluation with no need to build a specialised heuristic. The resulting heuristic guidance is informative and allows the planner to maintain its performance against complex problems involving sophisticated mathematical calculations.

Our architecture is general and we demonstrate its power by extending the planner POPF-TIF with plug-in advisors and tackling problems in four completely different domains. Our experiments provide evidence that our approach is effective regardless of the specific domain of application and outperforms related techniques when complex tasks are considered. Our analysis across the four domains using the different approximations shows a trade-off between heuristic calculation time/accuracy and resulting plan quality. The actual magnitude of the trade-off is domain specific.

Furthermore, we show that the enhancement of a high performing planner capable of handling problems with a rich temporal structure, such as POPF-TIF, with an advisor that allows complex metric calculations and a mechanism that supports good heuristic guidance opens the door to solving sophisticated real-world problems.

## Acknowledgments

This research was supported by the UK Engineering and Physical Sciences Research Council (EPSRC) as part of the projects entitled *The Autonomic Power System* (Grant Ref: EP/I031650/1) and *Automated Plan-Based Policy-Learning for Surveillance Problems* (Grant Ref: EP/J012157/1).

We thank Johannes Aldinger and Johannes Löhr for making the EO domain available to us.

## References

Aldinger, J., and Löhr, J. 2013. Planning for Agile Earth Observation Satellites. In *ICAPS Workshop on Planning in Continuous Domains*, 9–17.

Bajada, J. 2016. *Temporal Planning for Rich Numeric Contexts*. Ph.D. Dissertation, Department of Informatics School of Natural and Mathematical Sciences, King's College.

Bernardini, S.; Fox, M.; Long, D.; and Piacentini, C. 2016. Leveraging Probabilistic Reasoning in Deterministic Planning for Large-Scale Autonomous Search-and-Tracking. In *Proceedings of the 26th International Conference on Automated Planning and Scheduling (ICAPS-16)*.

Bernardini, S.; Fox, M.; and Long, D. 2015. Combining Temporal Planning with Probabilistic Reasoning for Autonomous Surveillance Missions. *Autonomous Robots* 1–23.

Coles, A.; Coles, A.; Fox, M.; and Long, D. 2010. Forward-Chaining Partial-Order Planning. In *Proceedings of the 20th International Conference on Automated Planning and Scheduling (ICAPS-10)*, 42–49.

Coles, A.; Coles, A.; Fox, M.; and Long, D. 2011. POPF2: a Forward-Chaining Partial Order Planner. In Angela Garcìa-Olaya, D. J., and Lòpez, C. L., eds., *The 7th International Planning Competition: Description of Participant Planners of the Deterministic Track*. Planning and Learning Group (PLG) Universidad Carlos III de Madrid. 65–70.

Davidson, E. M.; Dolan, M. J.; Ault, G. W.; and McArthur, S. D. J. 2010. AuRA-NMS: An Autonomous Regional Active Network Management System for EDF Energy and SP Energy Networks. In *IEEE Power and Energy Society General Meeting*, 1–6.

Dornhege, C.; Eyerich, P.; and Keller, T. 2009. Semantic Attachments for Domain-Independent Planning Systems. In *Proceedings of the 19th International Conference on Automated Planning and Scheduling (ICAPS-09)*, 114–121.

Eyerich, P.; Mattmüller, R.; and Röger, G. 2009. Using the context-enhanced additive heuristic for temporal and numeric planning. In *Proceedings of the 19th International Conference on Automated Planning and Scheduling (ICAPS-09)*, 130–137.

Fox, M., and Long, D. 2003. PDDL2.1: An Extension to PDDL for Expressing Temporal Planning Domains. *Journal of Artificial Intelligence Research* 20.

Fox, M., and Long, D. 2006. Modelling Mixed Discrete-Continuous Domains for Planning. *Journal of Artificial Intelligence Research* 27:235–297.

Gregory, P.; Long, D.; Fox, M.; and Beck, J. C. 2012. Planning modulo theories: Extending the planning paradigm. In *Proceedings of the 22nd International Conference on Automated Planning and Scheduling (ICAPS-12)*.

Hertle, A.; Dornhege, C.; Keller, T.; and Nebel, B. 2012. Planning with Semantic Attachments: An Object-oriented View. *Frontiers in Artificial Intelligence and Applications* 242:402–407.

Hoffmann, J., and Edelkamp, S. 2005. The deterministic part of IPC-4: An overview. *Journal of Artificial Intelligence Research* 24:519–579.

Hoffmann, J. 2003. The Metric-FF Planning System: Translating "Ignoring Delete Lists" to Numeric State Variables. *Journal of Artificial Intelligence Research* 20:291–341.

Ivankovic, F.; Haslum, P.; Shivashankar, V.; and Nau, D. S. 2014. Optimal Planning with Global Numerical State Constraints. In *Proceedings of the 24th International Conference on Automated Planning and Scheduling (ICAPS-14)*, 145–153.

Löhr, J.; Eyerich, P.; Keller, T.; and Nebel, B. 2012. A Planning Based Framework for Controlling Hybrid Systems. In *Proceedings of the 22nd International Conference on Automated Planning and Scheduling (ICAPS-12)*, 164–171.

National Grid PLC. 2012. National grid uk. http://www.nationalgrid.com/uk/Electricity/Data/Demand+Data/.

Parkinson, S.; Gregory, P.; Longstaff, A.; and Crampton, A. 2014. Automated Planning for Multi-Objective Machine Tool Calibration: Optimising Makespan and Measurement Uncertainty. In *Proceedings of the 24th International Conference on Automated Planning and Scheduling (ICAPS-14)*, 421–429.

Piacentini, C.; Alimisis, V.; Fox, M.; and Long, D. 2015. An Extension of Metric Temporal Planning with Application to AC Voltage Control. *Artificial intelligence* 229:210–245.

Stone, L. D. 1975. *The Theory of Optimal Search*. Operations Research Society of America.

Weyhrauch, R. W. 1980. Prolegomena to a Theory of Mechanized Formal Reasoning. *Artificial intelligence* 13(1A):133–170.

Wood, A.; Wollenberg, B.; and Shebleeacute, G. 2013. *Power Generation, Operation and Control*. New Jersey: Wiley, third edition.