The Benchmark Domains of the Deterministic Part of IPC-5

Yannis Dimopoulos^{+} **Alfonso Gerevini*** **Patrik Haslum**$^{\circ}$ **Alessandro Saetti***
$^{+}$ Department of Computer Science, University of Cyprus, Nicosia, Cyprus
* Department of Electronics for Automation, University of Brescia, Brescia, Italy
$^{\circ}$ Department of Computer and Information Science, Linköping University, Linköping, Sweden
$^{+}$yannis@cs.ucy.ac.cy *{gerevini,saetti}@ing.unibs.it $^{\circ}$pahas@ida.liu.se

Abstract

We present a set of planning domains and problems that have been used as benchmarks for the fifth International planning competition. Some of them were inspired by different types of logistics applications, others were obtained by encoding known problems from operation research and bioinformatics. For each domain, we developed several variants using different fragments of PDDL3 with increasing expressiveness.

Introduction

The language of the fifth International planning competition (IPC-5), PDDL3.0 (Gerevini & Long 2005), is an extension of the previous versions of PDDL (Fox & Long 2003; Edelkamp & Hoffmann 2004) that aims at a better characterization of plan quality. The new language allows us to express *strong and soft constraints on plan trajectories* (i.e., constraints over intermediate states reached by the plan), as well as *strong and soft problem goals*. Strong trajectory constraints and goals must be satisfied by any valid plan, while soft trajectory constraints and goals (called *preferences*) express desired constraints and goals, which do not necessarily have to be achieved. In PDDL3.0, the plan metric expression can include weighted penalty terms associated with the violation of the soft trajectory constraints and goals in the problem.

This paper gives an informal presentation of the benchmark domains and problems that we developed for IPC-5, and that include most of the new features of PDDL3.0.[1] We designed five new domains, as well as some new variants of two domains that have been used in previous planning competitions. In order to make the language more accessible to the the IPC-5 competitors, we developed for each domain several variants, using different fragments of PDDL3.0. The "propositional" and "metric-time" variants use only the constructs of PDDL2.2 (Edelkamp & Hoffmann 2004); the "simple preferences" variant extends the propositional with preferences over the problem goals; the "qualitative preferences" variant also includes preferences over state trajectory constraints; the "metric-time constraints" variant extends the metric-time variant with strong state trajectory constraints; and, finally, the "complex preferences" variant uses the full power of the language, including soft trajectory constraints and goals. However, not all the different variants of each domain actually use the full fragment "allowed" for that variant.

In the domain variants involving preferences we created for each planning problem a plan metric incorporating terms specifying the penalties for violations of the preference. The metric is a very important part of the problem statements in such domains, since it determines which is the best trade-off between different, perhaps mutually exclusive, preferences, and we tried with much care to ensure that the metrics in the test problems give rise to challenging optimization problems.

The IPC-5 test domains have different motivations. Some of them were inspired by real world applications, (e.g., `storage`, `trucks` and `pathways`); others were aimed at exploring the applicability and effectiveness of automated planning for new applications (`pathways`), or for known problems that have been addressed in other fields of computer science (`TPP` and `openstacks`); finally, two domains were taken from previous competitions, as sample references for the advancement of automated planning with respect to the existing benchmarks (`rovers` and `pipesworld`).

For some domains, the problems we generated have many solutions. In these problems, the most challenging aspect is finding plans of good quality. Other problems are challenging for different reasons: the expressiveness of the planning language used to model the problem including some of the new features of PDDL3.0, the large size of the problem, or the known NP-hardness of the computational problem they model. In most cases, the test problems were automatically (or semi-automatically) generated by using dedicated software tools.

[1]A detailed description of the IPC-5 benchmarks is outside the scope of this short paper; their PDDL formalization is available from the IPC-5 website: `http://ipc5.ing.unibs.it`.

The Travelling Purchaser Domain

This is a relatively recent planning domain that has been investigated in operations research (OR) for several years, e.g., (Riera-Ledesma & Salazar-Gonzalez 2005). The Travelling Purchaser Problem (TPP) is a known generalization of the Travelling Salesman Problem, and is defined as follows. We have a set of products and a set of markets. Each market can provide a limited amount of each product at a known price. The TPP consists in selecting a subset of markets such that a given demand for each product can be purchased, minimizing the combined travel and purchase cost. This problem arises in several applications, mainly in routing and scheduling contexts, and it is NP-hard. In OR, computing optimal or near optimal solutions for the TPP instances is still an active research topic.

For IPC-5, we have formalized several variants of this domain in PDDL. One of them is equivalent to the original TPP, while the others are different formulations or significant (we believe and hope) extensions. In all these domain variants, plan quality is important, although for some instances even finding an arbitrary solution could be quite difficult for a fully-automated planner.

For this domain, we developed both a metric version without time and a metric-time version. We begin the description with the metric version because it is the one equivalent to the original formulation of the TPP.

Metric

This version is equivalent to the original formulation of the TPP in OR. There are only three operators, two of which are used to model the purchasing actions: "buy-all" and "buy-allneeded". The first buys at a certain market (`?m`) the whole amount of a type of goods (`?g`) sold by the market (`?m` and `?g` are operator parameters); while the second one buys at `?m` the amount of `?g` that is needed to complete the purchase of `?g` (as specified in the problem goals). In this version, every market is *directly* connected to every other market and to the depots. Moreover, there is only one depot and only one truck.

Propositional

This version models a variant of the original TPP where: (1) there can be more than one depot and more than one truck; (2) the amount of goods are discrete and represented by qualitative levels; (3) every type of goods has the same price, independent from the market where we buy it; (4) there are two new operators for loading and unloading goods to/from trucks; (5) markets and depots can be indirectly connected.

Simple Preferences

The operators in this domain are the same as in the propositional version. The difference is in the goals, which are all soft goals (preferences). These preferences concern maximizing the level of goods that are stored in the depots, constraints between the levels of different stored goods, and the safety condition that all purchased goods are stored at some market.

Qualitative Preferences

The operators in this version are the same as in the propositional version. All goals are preferences concerning maximizing, for every type of goods, the purchased and stored levels. This version includes preferences over trajectory constraints. These are constraints between the levels of two types of stored goods; constraints about the use of the trucks for loading goods; constraints imposing the use of every truck. Moreover, we have the preference that in the final state all purchased goods are stored at some depot.

Metric-Time

With respect to the simpler metric version, which is equivalent to the original formulation of the TPP, this version has the the following main differences: same as points (1), (4), (5) illustrated in the description of the propositional variants; each action has a duration and the plan quality is a linear combination of total-time (makespan) and the total cost of traveling and purchasing; the operator "`buyall`" has a "`rebate`" rate (if you buy the whole amount of a type of goods that is sold at a market, then you have a discount).

Metric-Time Constraints

The operators in this version are the same as in the metric-time version. In addition, in the domain file, we have some strong constraints imposing that in the final state all purchased goods are stored, every market can be visited by at most one truck at the same time, every truck is used. Moreover, in the problem specification, we have several strong constraints about the relative amounts of different types of goods stored in a depot, the number of times a truck can visit a market, the order in which goods should be stored, the order in which we should store some type of goods and buy another one, and deadlines about delivering goods once they have been loaded in a truck.

Complex Preferences

The operators in this version are the same as in the metric-time version. In addition, it contains many preferences over state trajectory constraints that are similar to those used for the metric-time constraints version.

The Openstacks Domain

The openstacks domain is based on the "minimum maximum simultaneous open stacks" combinatorial optimization problem, which can be stated as follows:

A manufacturer has a number of orders, each for a combination of different products, and can only make one product at a time. The total required quantity of each product is made at the same time (because changing from making one product to making another requires a production stop). From the time that the first product included in an order is made to the time that all products included in the order have been made, the order is said to be “open” and during this time it requires a “stack” (a temporary storage space). The problem is to order the making of the different products so that the maximum number of stacks that are in use simultaneously, or equivalently the number of orders that are in simultaneous production, is minimized (because each stack takes up space in the production area).

This problem, and many related variants, have been studied in operations research (see, e.g., Fink & Voss 1999). It is known to be NP-hard, and equivalent to several other problems (Linhares & Yanasse 2002). This is a pure optimization problem: for any instance of the problem, every ordering of the making of products is a solution, which at worst uses as many simultaneously open stacks as there are orders. Thus, finding a plan is quite trivial (in the sense that there exists a domain-specific linear-time algorithm that solves the problem), but finding a plan of high quality is hard (even for a domain-specific algorithm).

The openstacks problem was recently posed as a challenge problem for the constraint programming community, and, as a result, a large library of problem instances, together with results on those instances for a number of different solution approaches, are available (see Smith & Gent (2005)).

Propositional

This variant is simply an encoding of the original openstacks problem as a planning problem. The encoding is done in such a way that minimizing the length (sequential or parallel) of the plan also minimizes the objective function, i.e., the maximum number of simultaneously open stacks. There are three basic actions to start orders, make products, and ship orders once they are completed, plus an action that “opens” a new stack, but in order to ensure the correspondance between parallel length and the objective function, some of these actions are split in two parts. The domain formulation uses some ADL constructs (quantified disjunctive preconditions), but these can be compiled away with only a linear increase in size.

The problems are a selection of the problems used in the constraint modelling challenge, including a few problems that could not be solved (optimally) by any of the CSP approaches, plus a small number of extra small instances.

Time

In this variant of the domain the number of available stacks is fixed, and the objective is instead to minimize makespan. Makespan is dominated by the actions that make products. The number of stacks is for each problem chosen to be somewhere between the optimal and the trivial upper bound (equal to the number of orders).

Metric-Time

In this variant, the objective function is to minimize a (linear) combination of the number of open stacks and the plan makespan. The number of open stacks is modelled using numeric fluents.

Simple Preferences

In this variant, the goal of including all required products in each order is softened, and a “score” (or “reward”) is instead given for each product that is included in an order when it is shipped. The objective is to maximize this score. The maximum number of open stacks is fixed, like in the temporal variant, but at a number slightly less than the optimal number required to satisfy all the requirements of all orders.

This version of the domain uses an ADL construct (a quantified conditional effects) that can only be compiled away at an exponential increase in problem size.

Complex Preferences

This version, like the previous, has soft goals, but also a variable maximum number of open stacks. The objective is to maximize a linear combination of the score (positive) and the number of open stacks (negative). Also like the previous version, the formulation uses a quantified conditional effect.

The Storage Domain

“Storage” is a planning domain involving spatial reasoning. Basically, the domain is about moving a certain number of crates from some containers to some depots by hoists. Inside a depot, each hoist can move according to a specified spatial map connecting different areas of the depot. The test problems for this domain involve different numbers of depots, hoists, crates, containers, and depot areas. While in this domain it is important to generate plans of good quality, for many test problems, even finding any solution can be quite hard for domain-independent planners.

Altogether, the different variants of this domain, involve almost all the new features of PDDL3.0. Note that this domain is basically a propositional domain, where the space for storing crates is represented by PDDL literals. For this domain, instead of a metric-time version, we have a “time-only” version (without numerical fluents).

Propositional

The domain has five different actions: an action for lifting a crate by a hoist, an action for dropping a crate by a hoist, an action for moving a hoist into a depot, an action for moving a hoist from one area of a depot to another one, and finally an action for moving a hoist outside a depot.

Time

This variant is basically the propositional variant where the actions have duration and the plan quality is total-time (plan makespan).

Simple Preference

The operators in this domain are the same as those in the propositional version. The main difference is in the goals. All goals are soft goals (preferences). These preferences concern which depots and depot areas should be used for storing the crates, the desire that only "compatible" crates are stored in the same depot, the desire that the incompatible crates stored in the same depot are located at non-adjacent areas of the depot and, finally, the desire that the hoists are located in depots different from those where we store the crates.

Qualitative Preferences

The operators in this domain are the same as those in the propositional version. The differences are in the preferences over the goals and state trajectory constraints. All goals are soft goals similar to some of the soft goals specified in the simple preferences variant. The preferences over trajectory constraints concern constraints about the use of the available hoists for moving the crates, and about the order in which crates are stored in the depots. Moreover, we have the preference that in any state crossed by the plan, the adjacent areas in a depot can be occupied only by compatible crates.

Time Constraints

The operators in this version are the same as those in the temporal version. The problem goals are specified by an "at-end" constraint imposing that all crates are stored in a depot. The problems have several constraints imposing that a crate can be lifted at most once, ordering constraints about storing certain crates before others, deadlines for storing the crates, and maximum time a hoist can stay outside a depot. There are also constraints imposing a safety condition, that in the final state, all hoists are inside a depot; some constraints imposing that every hoist is used; and some constraints imposing that incompatible crates are not stored at adjacent areas of the depot.

Time Preferences

The operators in this version are the same as those in the temporal version. In addition, this version contains many preferences over state trajectory constraints that are similar to those used for the time constraints version.

The Trucks Domain

Essentially, this is a logistics domain about moving packages between locations by trucks under certain constraints. The loading space of each truck is organized by areas: a package can be (un)loaded onto an area of a truck only if the areas between the area under consideration and the truck door are free. Moreover, some packages must be delivered within a deadline. In this domain, it is important to find good quality plans. However, for many test problems, even finding one plan could be a rather difficult task.

Like the Storage domain, this domain has a "time-only" variant instead of a metric-time variant (i.e., there are no numerical fluents). The other variants make extensive use of the new features of PDDL3.0. We start the description from the time constraint version, because it is the one closest to a realistic problem.

Time Constraints

The domain has four different actions: an action for loading a package into a truck, one for unloading a package from a truck, one for moving a truck, and finally one for delivering a package. The durations of loading, unloading and delivering packages are negligible compared to the durations of the driving actions. The problem goals require that certain packages are at their final destinations by certain deadlines. For this variant, we also created an equivalent version, "Time-TIL", in which the trajectory constraints of type "`within`" are compiled into timed initial literals. Each competing team is free to choose one of the two alternative variants.

Time

The operators are the same as those in the time constraints version, but there is no deadline for delivering packages. Finding a valid plan in this version is significantly easier, but finding a plan with short makespan is still challenging.

Complex Preferences

The operators in this version are the same as those in the constraints version. The deadlines are modeled by preferences. Moreover, this version contains preferences over trajectory constraints. These are constraints imposing some ordering about when delivering packages, constraints about the usage of the areas in the trucks, and constraints about loading packages.

Propositional

The operators in this version are similar to those in the constraints version, with the main difference that time is modeled as a discrete resource (with a fixed number of levels). Moreover, the driving actions cannot be executed concurrently.

Simple Preferences

The operators in this domain are the same as those in the propositional version. The difference concerns the problem goals where the delivering deadlines are modeled by preferences.

Qualitative Preferences

The operators in this domain are the same as those in the propositional version. The difference concerns the problems goals including soft delivering deadlines. Moreover, this version includes many preferences over state trajectory constraints that are similar to those used for the complex preferences version.

The Pathways Domain

This domain is inspired by the field of molecular biology, specifically biochemical pathways. "A pathway is a sequence of chemical reactions in a biological organism. Such pathways specify mechanisms that explain how cells carry out their major functions by means of molecules and reactions that produce regular changes. Many diseases can be explained by defects in pathways, and new treatments often involve finding drugs that correct those defects." (Thagard 2003) We can model parts of the functioning of a pathway as a planning problem by simply representing chemical reactions as actions. The goal in these planning problems is to construct a sequence of reactions that produces one or more substances, using a limited number of substances as input. The planner is partly free to choose which input substances to use, i.e., to choose some aspects of the initial state of the problem. This aspect of the problem is modelled by means of additional actions.

The biochemical pathway domain of the competition is based on the pathway of the Mammalian Cell Cycle Control as it described in (Kohn 1999) and modelled in (Chabrier 2003). There are three different kinds of basic actions corresponding to the different kinds of reactions that can appear in a pathway.

Propositional

This is a simple qualitative encoding of the reactions of the pathway. The domain has five different actions: an action for choosing the initial substances, an action for increasing the quantity of a chosen substance (in the propositional version, quantity coincides with presence, and it is modeled through a predicate indicating if a substance is available or not), an action modeling biochemical association reactions, an action modeling biochemical association reactions requiring catalysts, and an action modeling biochemical synthesis reactions. Also, there is an additional set of "dummy" actions used to encode the disjunctive problem goals.

The goals refer to substances that must be synthesized by the pathway, and are disjunctive with two disjuncts each. Furthermore, there is a limit on the number of input substances that can be used by the pathway.

Simple Preferences

This is similar to the propositional version, with the difference that both the products that must be synthesized by the pathway and the number of the input reactants that are used by the network are turned into preferences. The challenge here is finding plans that achieve a good tradeoff between the different kinds of preferences.

Metric-Time

In this version of the domain, reactions have different durations. The reactions can only happen if their input reactants reach some concentration level, and reactions generate their products in specific quantities. The goals in this version are summations of substance concentrations that must be generated by the reactions of the pathway. The plan metric minimizes some linear combination of the number of input substances and the plan duration.

Complex Preferences

This is an extension of the metric-time version with different preferences concerning the concentration of substances of the pathway, or the order in which substances are produced. The metric is a combination of these preferences, the number of substances used and the plan makespan.

The Extended Rovers Domain

The Rovers domain was introduced in the 2002 planning competition (Long & Fox 2003). It models the problem of planning for a group of planetary rovers to explore the planet they are on (taking pictures and samples from interesting locations).

Propositional and Metric-Time

The propositional and metric-time versions of the domain are the same as in IPC 2002, with the addition of some planning problems.

The domain has nine different actions: an action for moving rovers on a planet surface, two actions for sampling soil and rock, an action for dropping rock or soil, an action for calibrating rover instruments, an action for taking image of interesting objective, and finally three actions for transmitting soil data, rock data or image data.

Qualitative Preferences

This is the IPC 2002 propositional version with soft trajectory constraints added (constraint types always, sometime and at-most-once are used). The objective is simply to maximize the number of preferences satisfied. The preferences are "artificial", in the sense that they do not encode any "real" preferences on the plan, but are constructed in a way as to make the problem of maximizing the satisfaction of preferences challenging.

Metric Simple Preferences

This version is a special case of the complex preferences version, which has preferences only on the goals of the problem.

This version of the domain poses a so-called "net benefit" problem: goals (atoms, and in some cases conjunction of atoms) have values and actions have cost, and the objective is to maximize the sum values of achieved goals minus the sum of costs of actions in the plan. Only the actions that move the rovers have non-zero cost. The domain uses simple (goal state) preferences to encode goal values and fluents to encode action costs. There are three different sets of problems, with somewhat different properties. In the first, goals are *interfering*, meaning that the cost of achieving any two goals is greater than the sum of achieving them individually. The second has instead *synergy* between the goals, i.e., the cost of achieving several goals is less than the sum of achieving each of them separately, while the third contains goals with relationships of both kinds.

The Extended Pipesworld Domain

The Pipesworld domain was introduced in the previous planning competition (Hoffmann & Edelkamp 2005). It models the transportation of batches of petroleum products in a network of pipelines.

Propositional and Time

The propositional and temporal versions of the domain are the "tankage" variant of the domain used in IPC 2004 The domain has six actions: two actions for moving a batch from a tankage to a pipeline segment (one for the start and one for the end of the activity), two actions for moving a batch from a tankage to a pipeline segment, and two actions for moving a batch from a tankage (or pipeline segment) to a pipeline segment (or tankage) in case the pipes consist of only one segment.

Time Constraints

The time constraints variant is based on the temporal no-tankage variant from IPC 2004, but adds hard deadlines on when each of the goals must be reached. Deadlines are specified using the PDDL3 `within` constraint. The problems also have a number of "triggered" deadline constraints, specified with PDDL3 `always-within` constraint.

Complex Preferences

This variant is similar to the previous, but has soft deadlines instead, encoded with preferences on the constraints. Each goal can have several (increasing) deadline, with different (increasing) penalties for missing them.

Conclusions

We have given an informal description of the benchmark domains that we developed for the deterministic part of the 2006 International Planning Competition. The general aim was to create a new set of problems for the planning community involving new and interesting – and hopefully also useful – issues, in particular planning with (possibly contradicting) preferences over problem goals and state trajectory constraints.

Several competing teams have declared their that their planners are capable of handling parts of the extended PDDL3 language. At the time of writing, benchmark tests are still being run. In addition to their use for the competition, we hope that the new benchmarks will provide a challenging extension to the existing set of planning benchmarks, both those involving PDDL3 constructs and those that can be specified through the previous versions of PDDL.

References

Chabrier, N. 2003. `http://contraintes.inria.fr/BIOCHAM/EXAMPLES/~cell_cycle/cell_cycle.bc`.

Edelkamp, S., and Hoffmann, J. 2004. PDDL2.2: The language for the classic part of the 4th international planning competition. Technical Report 195, Institut für Informatik, Freiburg, Germany.

Fink, A., and Voss, S. 1999. Applications of modern heuristic search methods to pattern sequencing problems. *Computers & Operations Research* 26:17 – 34.

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

Gerevini, A., and Long, D. 2005. Plan constraints and preferences in PDDL3. Technical report rt-2005-08-47, Universitá di Brescia, Dipartimento di Elettronica per l'Automazione.

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

Kohn, K. 1999. Molecular interaction map of the mammalian cell cycle control and dna repair systems. *Mol Biol Cell* 10(8).

Linhares, A., and Yanasse, H. 2002. Connection between cutting-pattern sequencing, VLSI design and flexible machines. *Computers & Operations Research* 29:1759 – 1772.

Long, D., and Fox, M. 2003. The 3rd international planning competition: Results and analysis. *Journal of Artificial Intelligence Research* 20:1 – 59.

Riera-Ledesma, J., and Salazar-Gonzalez, J., J. 2005. A heuristic approach for the travelling purchaser problem. *European Journal of Operational Research* 160(3):599–613.

Smith, B., and Gent, I. 2005. Constraint modelling challenge 2005. `http://www.dcs.st-and.ac.uk/~ipg/challenge/`.

Thagard, P. 2003. Pathways to biomedical discovery. *Philosophy of Science* 70.