Planning for Continuous Domains

Fabio Mercorio

Department of Computer Science,University of L'Aquila,
Via Vetoio Loc. Coppito, I-67010 L'Aquila - Italy
fabio.mercorio@univaq.it - http://www.di.univaq.it/fabio.mercorio

Introduction and problem description

My research activity focuses on *planning* and *universal planning* for systems with continuous and (possibly) nonlinear behaviour. Indeed, many realistic planning problems in which planning can be applied involve a mixture of continuous and discrete behaviours that make difficult to deal with such kind of systems. Some examples are: product processing in a plant [1], activity management of an autonomous vehicle [14], voltage regulation planning [2], solar array operations on the International Space Station [15] or oil refinery operations planning [3]. These problems are described by hybrid systems where continuous physical rules are managed by discrete digital equipments. In such a context, it is crucial to reason about continuous change during the planning process.
For these reasons, in the past years the planning community has made a great effort to extend the current planning systems and the standard planning definition language (PDDL) to support such kind of domains. This effort has lead to the definition of PDDL+ [12] that makes it possible to capture continuous processes and events.

However several real world planning problems present complex *nonlinear* behaviours where nonlinearity can arise from the intrinsic dynamics of the system (e.g., the regulation of a steering antenna, which leads to an inverted pendulum problem), or the saturation of actuators (e.g., valves that cannot open more than a certain limit). Thus, planning with continuous nonlinear change is a challenging issue.

Background and overview of the existing literature

Given a system S modeled in some formalism, a planning problem consists of finding a sequence of actions which guarantees to achieve the goal for a given initial condition of the system. A *planner* is a software system able to synthesise a plan by taking as input (1) the domain description and (2) a goal description. A Universal Planner [16] is a software system that synthesises a *set of plans* (or set of policies) able to bring the system to the goal from any feasible state reachable from the initial one.
The Planning Domain Definition Language (PDDL) is a STRIPS based standard modelling language for planning problems that uses a simple Lisp-like syntax. Many versions of PDDL have been proposed (e.g., PDDL2.1, PDDL2.2, PDDL3) and the most expressive one is the PDDL+, that allows one to model continuous change through the initiation and termination of *processes* which continuously

modify the value of the numeric components of the state. [1] Processes and events allow one to model in a more realistic way the behaviour of continuous systems, and this represents a further step in applying planning technology to real world problems.

Although a great number of PDDL planners have been developed, none of them is able to deal with *full* PDDL+ semantics. Recently many techniques have been developed to handle continuous domains: for example the mixing of SAT solver with LP solver in TM-LPSAT planner [17]. TM-LPSAT can deal with processes modelled in PDDL2.1, however it is limited to small linear problems. In Kongming [14] the concept of Flow Tubes was applied in order to compactly represent hybrid plans and encode hybrid flow graphs as a mixed logic linear/nonlinear program, solvable using an off-the-shelf solver. However, it can only address planning problems with constant action duration. COLIN [7] is a powerful tool for planning in domains with linear continuous processes that integrates a guided state space search with linear programming supporting only linear continuous change. Also the well-known *Planning-as-model-checking* technique has been used exploiting states reachability analysis to find plans. The Model Checking Integrated System (MIPS) [11] is a very powerful and complex framework that makes use of a combination of both explicit and symbolic model checking based heuristic search. The MIPS performed very well in different planning competitions, however it is restricted to PDDL2.1.

Finally, regarding to the universal planning problem (introduced in [16]), a *symbolic* model checking approach [6] was used to synthesise optimal universal plans for non-deterministic plants. On the other hand, the DPlan and FPlan tools use an explicit state-based representation instead of OBDDs. However, they both require the explicit definition of an inverse function for each operator used in the domain, and thus their application is hard when dealing with systems whose dynamics is difficult to invert (the typical situation for hybrid and/or nonlinear systems).

Goal of the research

Planning with continuous nonlinear change is still a challenging issue since it is difficult to deal with real world applications using domain independent planners. Indeed, in many cases an ad-hoc planner is required (e.g. the MAPGEN tool [4] has been used for the Deep Space 1 mission).

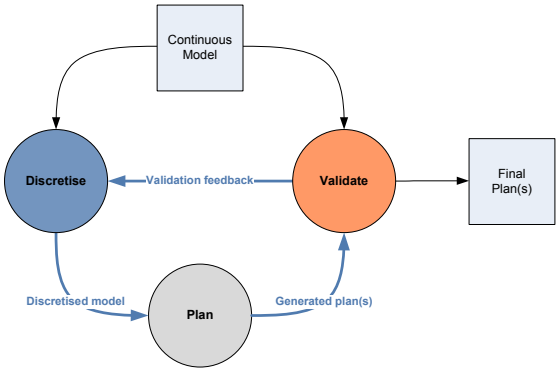
Our research aims to contribute in developing techniques and tools able to handle hybrid planning problems, in order to apply them to real-world systems.

Current status of the research

We have proposed a *discretise and validate* approach to deal with PDDL+ domains having a complex nonlinear dynamics. In this approach, the continuous problem is relaxed into a discretised one using rounded values and uniform discrete time steps. A schematic view of the approach is shown in Figure 1a. Given a

[1] For instance, if a pump is switched on, then a filling process starts that continuously increases the water level in a tank and terminates when the pump is switched off. Moreover, PDDL+ allows the modelling of predictable *exogenous events* that occur as a consequence of some process. For example, a tank will eventually overflow if the pump is left switched on.

first discretisation, the continuous model is relaxed into a discretised one. Then, a forward reachability analysis is used to perform universal planning on the discretised model. The generated plans are validated against the continuous model using the VAL [13] plan validator. Since VAL computes analytic solutions to the differential equations describing the system dynamics, its use is effective to prove the soundness of the discretised solutions. Moreover, the output of VAL is also used to determine whether a finer discretisation is required. The process loops until the validation returns a satisfying result. This approach has been implemented in *UPMurphi*, a model-checking based tool which performs universal planning on PDDL+ problems, i.e., it generates a *set of policies* for all states reachable from the initial ones. In particular, by exploiting the discretisation, UPMurphi is able to build and analyse the transition graph for a large class of systems, including systems whose dynamics is nonlinear and hard to be inverted. Obviously, in order to have a precise analysis of a hybrid domain, a suitable discretisation of the continuous behaviour is required. On the other hand, the finer is the discretisation used to round the continuous component of the state, the bigger is the resulting state space. To this aim, model checking algorithms offer powerful compression techniques and very effective compact encodings, that make UPMurphi able to deal with huge state spaces.



(a) D&V approach

	Planetary	Chemical
State Space size	10^{24}	10^{17}
Reachable states	31,965,200	29,968,861
Generated plans	5,309,514	7,154,464
Plan gen. time	2,265 sec	6,319 sec
Memory Peak	1800 MB	630 MB
Start States	102	23

(b) Case studies statistics

Moreover, the discretisation of the continuous behaviour of a system involves a quantisation of the timeline, where the time flow is modelled using uniform discrete time steps. This, in turn, requires to encode the PDDL+ description of the domain into such discretised setting. To this aim, we designed and implemented a complete compilation process (the *PDDL+ to UPMurphi compiler)* based on a formal mapping between the grammars of PDDL+ and UPMurphi. It takes in input a domain/problem pair written in PDDL+ and generates the corresponding discretised model to be used by UPMurphi. This completely eliminates the need of learning any planner-specific language and manually re-encoding domains with different formalisms.

Therefore, UPMurphi aims to offer a fully PDDL+ compliant universal planner that is able to cope with problems that are very hard to handle by the current state-of-the-art tools.

Preliminary results accomplished

We have tested UPMurphi in three real world case studies, namely the *planetary lander* [8], the *batch chemical plant* [9], where UPMurphi was successfully applied to perform universal planning, and in the Autonomous Planetary Vehicle problem [10] to control the rover engine. Moreover, thanks to the PDDL+ to UPMurphi compiler we were able to work *directly* on the PDDL+ model of these domains.

The Planetary Lander case study: is inspired by the specifications of "Beagle 2" Mars probe, designed to operate on the Mars surface with tight resource constraints. Basically, the lander must perform two observation actions, called *Observe1* and *Observe2*. However, before making each observation, it must perform the corresponding preparation task, called *prepareObs1* and *prepareObs2*, respectively. Alternatively, the probe may choose to perform a cumulative preparation task for both observations by executing the single long action *fullPrepare*. The shorter actions have higher power requirements than the single preparation action. The power needed to perform these operations comes from the probe solar panels. The energy generated by the panels is influenced by the position of the sun. Power coming from the solar panels is also used to charge a battery, which is then discharged to give power to the lander when the panels do not produce enough energy (e.g., at night). Moreover, the probe must always ensure a minimum battery level to keep its instruments warm. Universal Plan statistics are shown in Table 1b.

The Chemical Plant case study: Its purpose is to produce saline solution with a given concentration recycling the unused part and preparing for another production cycle. The plant (omitted for lack of space) is composed of 7 tanks connected through a complex pipeline, whose flow is regulated by 26 valves and two pumps, one heater, a condenser and two cooling circuits. A set of sensors provide information to the plant controller about the filling level of tanks the pump pressure and the condenser status. When tanks 1 and 2 are appropriately filled, the plant can start the *production phase*. Tank 3 is partially filled with the solution from tank 1, which is then diluted using the water from tank 2 up to the requested concentration. The resulting saline solution can be taken from the output valve of tank 3. If the product is not completely used, the plant recycles it in the next production cycle. To this aim, the solution in moved between tanks and boiled by the heater until it reaches the concentration c_{high}. The steam produced by this process is piped to the condenser that fills tank 6 with the resulting water. Finally, recycled water and solution are pumped to tanks 1 and 2, respectively. The system goal is to synthesise a *production policies* able to bring the system to the goal in any possible configuration obtained after the recycle phase. However, the duration of these activities is not known *a priori*, thus the planner should determine the *optimal* time point at which the tank capacity (or required concentration) is reached. The UP statistics are shown in Table 1b.

Open issues and expected achievements

We plan to extend our technique to deal with planning in nondeterministic

domains where the objective is to find a plan that is guaranteed to achieve the goal regardless of nondeterminism. Indeed, many real world systems are nondeterministic, since the outcomes of the actions cannot be completely predictable. In this context, many efforts have been made (see, e.g. [5]). However, as in the deterministic case, dealing with continuous nonlinear behaviours is an open issue also in the nondeterministic one.

References

1. Aylett, R., Soutter, J.K., Petley, G.J., Chung, P.W.H.: AI planning in a chemical plant domain. In: Proceedings of ECAI. pp. 622–626 (1998)
2. Bell, K.R.W., Coles, A.J., Coles, A.I., Fox, M., Long, D.: The role of AI planning as a decision support tool in power substation management. AI Communications 22(1), 37–57 (2009)
3. Boddy, M.S., Johnson, D.P.: A new method for the global solution of large systems of continuous constraints. In: COCOS. pp. 142–156 (2002)
4. Bresina, J.L., K.Jonsson, A., Morris, P.H., Rajan, K.: Activity planning for the Mars exploration rovers pp. 40–49 (2005)
5. Cimatti, A., Roveri, M., Traverso, P.: Strong planning in non-deterministic domains via model checking. In: Proceedings of AIPS. pp. 36–43 (1998)
6. Cimatti, R., Roveri, M., Traverso, P.: Automatic OBDD-based generation of universal plans in non-deterministic domains. pp. 875–881. AAAI Press (1998)
7. Coles, A.J., Coles, A.I., Fox, M., Long, D.: Temporal planning in domains with linear processes. In: Proceedings of the Twenty-First International Joint Conference on Artificial Intelligence (IJCAI-09) (July 2009)
8. Della Penna, G., Intrigila, B., Magazzeni, D., Mercorio, F.: UPMurphi: a tool for universal planning on PDDL+ problems. In: Proceeding of ICAPS 2009. pp. 106–113. AAAI Press (2009)
9. Della Penna, G., Intrigila, B., Magazzeni, D., Mercorio, F.: A PDDL+ benchmark problem: The batch chemical plant. In: Proceedings of ICAPS 2010. pp. 222–224. AAAI Press (2010)
10. Della Penna, G., Intrigila, B., Magazzeni, D., Mercorio, F.: Resource-optimal planning for an autonomous planetary vehicle. International Journal of Artificial Intelligence & Applications (IJAIA) 1(3), 15–29 (2010)
11. Edelkamp, S.: Mixed propositional and numeric planning in the model checking integrated planning system. In: AIPS Workshop on Planning for Temporal Domains. pp. 47–55 (2002)
12. Fox, M., Long, D.: Modelling mixed discrete-continuous domains for planning. Journal of Artificial Intelligence Research 27, 235–297 (2006)
13. Howey, R., Long, D., Fox, M.: VAL: Automatic plan validation, continuous effects and mixed initiative planning using PDDL. In: Proceedings of ICTAI 2004. IEEE Computer Society, Boca Raton, Florida, USA
14. Léauté, T., Williams, B.C.: Coordinating agile systems through the model-based execution of temporal plans. In: Proceedings of AAAI 2005. pp. 114–120 (2005)
15. Reddy, S.Y., Iatauro, M.J., Kürklü, E., Boyce, M.E., Frank, J.D., Jónsson, A.K.: Planning and monitoring solar array operations on the ISS. In: Proc. Scheduling and Planning Applications Workshop (SPARK), ICAPS (2008)
16. Schoppers, M.: Universal plans of reactive robots in unpredictable environments. In: Proc. IJCAI 1987. (1987)
17. Shin, J.A., Davis, E.: Processes and continuous change in a SAT-based planner. Artif. Intell. 166(1-2), 194–253 (2005)