On timeline-based games and their complexity

Nicola Gigante[1] Angelo Montanari[2] Andrea Orlandini[3]
Marta Cialdea Mayer[4] Mark Reynolds[5]

[1]Free University of Bozen-Bolzano, Italy [2]University of Udine, Italy
[3]CNR-ISTC, Rome, Italy [4]University of Roma Tre, Rome, Italy
[5]University of Western Australia

Timeline-based planning

Timeline-based planning is an approach to planning, born in the space sector, mostly focused on temporal reasoning [1, 2]:

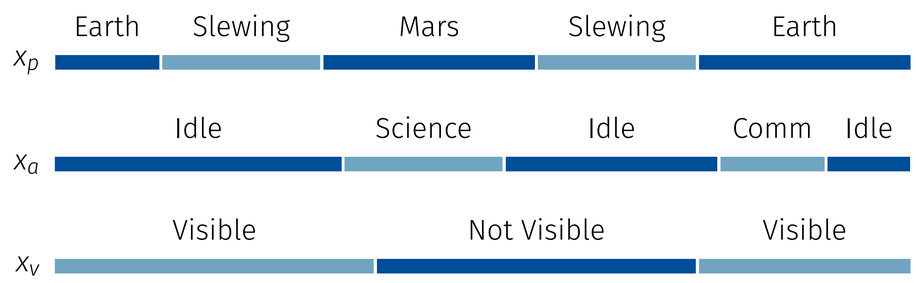
- no clear separation between actions, states, and goals;
- planning problems are modeled as systems made of a number of independent, but interacting, components;
- components are described by state variables;
- the timelines describe their evolution over time;
- the evolution of the system is governed by a set of temporal constraints called synchronization rules.

Synchronization rules have a fixed structure, e.g.:

$$a[x_a = \mathit{Science}] \rightarrow \exists b[x_p = \mathit{Mars}] \,. \quad \mathit{start}(b) \leq \mathit{start}(a) \wedge \mathit{end}(a) \leq \mathit{end}(b)$$

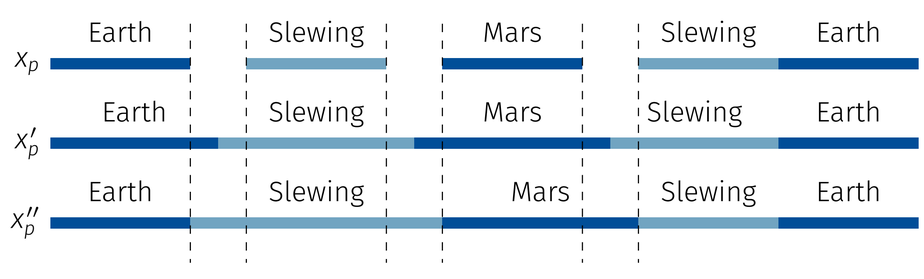
which means:
for each time interval a where $x_a = \mathit{Science}$,
there is another one b where $x_p = \mathit{Mars}$,
such that a is contained in b.

Timelines are the evolution over time of the state variables:



Uncertainty

Temporal uncertainty is handled by means of flexible plans [2], envelopes of possible solutions that differ by the precise timings:



Once a flexible plan has been found, it has to be checked for weak/strong/dynamic controllability, similarly to STNUs.

However, the focus on temporal uncertainty means flexible plans cannot represent strategies involving non-temporal choices.

- flexible plans are inherently sequential;
- the control strategy can only choose the timings of the already fixed sequence of tokens;
- if the expected non-temporal behavior of external variables mismatches during the execution, re-planning is needed [3].

We want to extended the approach to handle general nondeterminism.

Timeline-based games

A timeline-based game is a tuple $G = (SV_C, SV_E, S, D)$.

- Two players, Charlie (the controller) and Eve (the environment);
- players play by starting and ending tokens, building a plan;
- Charlie can start tokens for variables in SV_C, Eve those for variable in SV_E;
- Charlie decides when to stop controllable tokens, while Eve decides when to stop uncontrollable ones;
- Charlie tries to satisfy the set S of system rules, whatever the behavior of Eve;
- both players are assumed to satisfy the set D of domain rules.

Charlie has a winning strategy if, for any behavior of Eve that satisfies the domain rules, he can satisfy the system rules.

Advantages

The approach has advantages with regards to dynamically controllable flexible plans:

- a general form of nondeterminism is handled in this way, not only temporal uncertainty;
- no need for re-planning, as the winning strategy can already handle any behavior of Eve;
- greater modeling flexibility: domain rules allow to describe complex interactions between the agent and the environment;
- provably subsumes the approach based on dynamically controllable flexible plans;
- but how hard is it to find such a strategy?

Theorem 1

Winning strategies for timeline-based games are strictly more general than dynamically controllable flexible plans.

Theorem 2

Deciding whether a given timeline-based game admits a winning strategy for Charlie is 2EXPTIME-complete.

References

[1] Nicola Muscettola. “HSTS: Integrating Planning and Scheduling.” In: *Intelligent Scheduling*. Ed. by Monte Zweben and Mark S. Fox. Morgan Kaufmann, 1994. Chap. 6, pp. 169–212.

[2] Marta Cialdea Mayer, Andrea Orlandini, and Alessandro Umbrico. “Planning and execution with flexible timelines: a formal account.” In: *Acta Informatica* 53.6-8 (2016), pp. 649–680. DOI: 10.1007/s00236-015-0252-z.

[3] Alessandro Umbrico et al. “PLATINUm: A New Framework for Planning and Acting.” In: *Proceedings of the 16th International Conference of the Italian Association for Artificial Intelligence*. Ed. by Floriana Esposito et al. Vol. 10640. Lecture Notes in Computer Science. Springer, 2017, pp. 498–512. DOI: 10.1007/978-3-319-70169-1_37.

Contact: nicola.gigante@unibz.it