Efficient Temporal Piecewise-Linear Numeric Planning With Lazy Consistency Checking

Josef Bajada[1] and **Maria Fox**[2] and **Derek Long**[3,4]

[1] *University of Malta*, [2]*British Antarctic Survey*, [3]*Schlumberger*, [4]*King's College London*

ICAPS 2022 Journal Track – IEEE Transactions on Artificial Intelligence – DOI 10.1109/TAI.2022.3146797

Motivation

- We want to scale support for PDDL2.1 Level 4 domains to real-world problem scenarios.
- Support for durative actions with variable durations and continuous effects, with non-constant rate of change (step function).
- We focus on temporal planning algorithms that encode the plan as Linear Programming (LP).



Background

- Each durative action, a, is split into two instant *snap actions*, $a_\vdash$ and $a_\dashv$, representing the start and end discrete endpoints, respectively.
- A Linear Program (LP) finds a valid schedule:
  - LP variables consist of the numeric state variables and the time points of each discrete step in the plan.
  - LP constraints consist of action conditions and effects on the numeric state variables, together with the temporal constraints.
  - The rate of change of a continuous effect can change at discrete time points (happenings) of the plan.
- Each new state that is encountered needs to be checked for temporal consistency.
- But using the LP for every state is computationally expensive.
- An LP with n happenings and m numeric fluents will have $n(2m+1)$ LP variables.

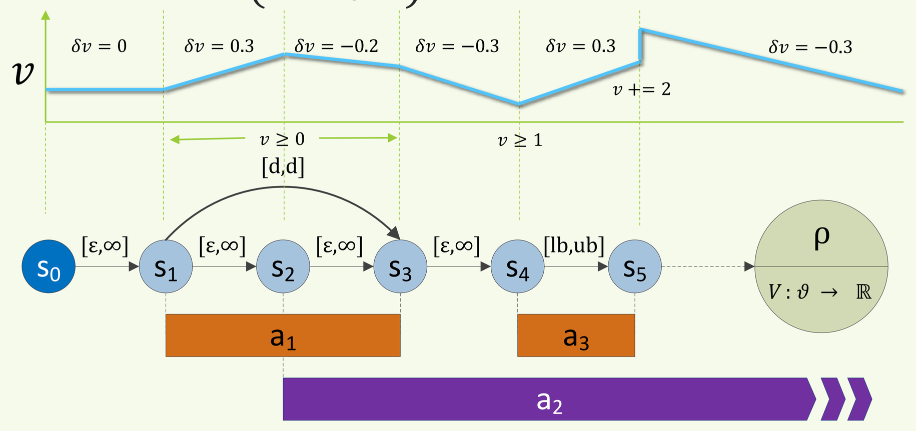

Fig. 1: A temporal plan with continuous and discrete numeric effects, and numeric constraints.

Schedule Dependence

- A numeric fluent, v, is *schedule dependent* at happening j, within a plan, π, if executing the plan up to happening j can lead to v taking a range of different values that the depend on the chosen schedule for the same plan, π.
- A numeric condition, c, over the set of numeric fluents ϑ_c is *schedule dependent* at happening j, within a plan, π, if and only if $\exists v \in \vartheta_c$, where v is schedule dependent at j.
- A discrete numeric effect, $\langle v, \otimes, e \rangle$, with e being an arithmetic expression over the set of numeric fluents ϑ_e is *schedule dependent* at happening j, within a plan, π, if and only if $\exists v_e \in \vartheta_e$, where v_e is schedule dependent at happening j.

Selective LP Execution

- Keep maintaining the STN for each state even when the planner uses the LP.
- When adding a new action, if the conditions (preconditions and invariants) are not schedule dependent, the STN is sufficient to determine consistency.
- When the LP discovers implicit tighter constraints, propagate them into the STN.
- Implicit temporal constraints are encoded back into the STN.

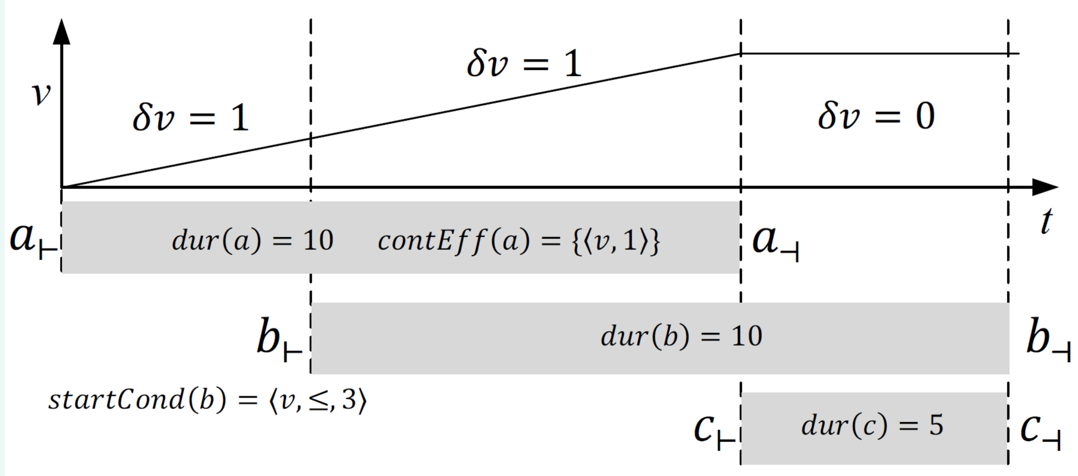

Fig. 2: A plan that is temporally *inconsistent*, detected by the LP but not the STN, unless the implicit constraint that ***b*** has to start at most 3 time units after ***a*** starts is propagated from the LP to the STN.

Schedule-Dependent Numeric Goals

- Schedule-dependent numeric goals need their own specific Linear Program with their constraints added.
- We prime the LP-based goal check when there is an update to a schedule-dependent numeric goal but postpone it to when we find a state where:
  - There are no running actions.
  - All non-schedule-dependent goals are satisfied.
  - There are schedule-dependent goals in the current state.
  - The LP-based goal flag is true.
- When we execute the LP-based goal check, its result will stay valid until another action affects a schedule-dependent numeric goal.

Require: The previous state s, the applied action a, the new state s', and the goal g.
Ensure: Returns *true* if $s' \models g$, false otherwise.

if $\exists v \in \widetilde{\vartheta}_{s'} : (v \in terms(g) \wedge v \in affected(a))$ **then**
  $s'.lpgc \leftarrow true$
else
  $s'.lpgc \leftarrow s.lpgc$
end if
if $s'.Q = \emptyset \wedge \forall c \in nsd(g) : (s' \models c)$ **then**
  if $\widetilde{\vartheta}_{s'} \cap terms(g) = \emptyset$ **then**
    return *true*
  else if $s'.lpgc$ **then**
    $s'.lpgc \leftarrow false$
    return $lpGoalCheck(s')$
  end if
end if
return *false*

Fig. 3: Schedule-dependent numeric goal checking algorithm.

Optimized LP Encoding

- Instead of encoding two LP variables, v_i and v_i' for each numeric fluent at each happening, i, we only generate LP variables for v where v is schedule-dependent.
- Any discrete/non-schedule-dependent updates are propagated forward without the LP.
- A happening, i, where v is schedule-dependent, but there are no updates to v and $\delta v = 0$, will also have its corresponding v_i and v_i' omitted.
- Constraints corresponding to invariant conditions are only checked for happenings where there is a potential constraint-violating change, such as a change in the direction of some threshold.

Empirical Evaluation

- DICE planner is base implementation without these optimisations; DICE2 includes these optimisations.
- **Carpool** domain: Locations, Cars and Trips, with each car having an amount of fuel and each trip having a pickup and drop-off location, together with a number of passengers. (39.22% improvement)
- **Pump Control** domain: Industrial plant where various processes need liquid to be pumped at a certain pressure, some of which concurrently. Pumps need to be adjusted to change the flow rate. (17.48% imp.)
- **Modified Linear Generator** domain: The standard linear generator, but with the refuel action having a variable duration and a goal that the fuel level must be greater than 10 at the end of the plan. (13.48% imp.)
- Evaluation also includes POPF, OPTIC, SMTPlan, UPMurphi, and DiNo, but not all of these planners generated a plan for these problems, in which case there are omitted.

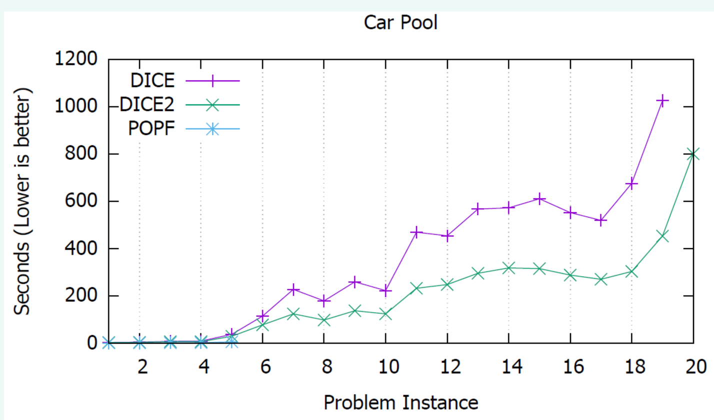

Fig. 4: Carpool

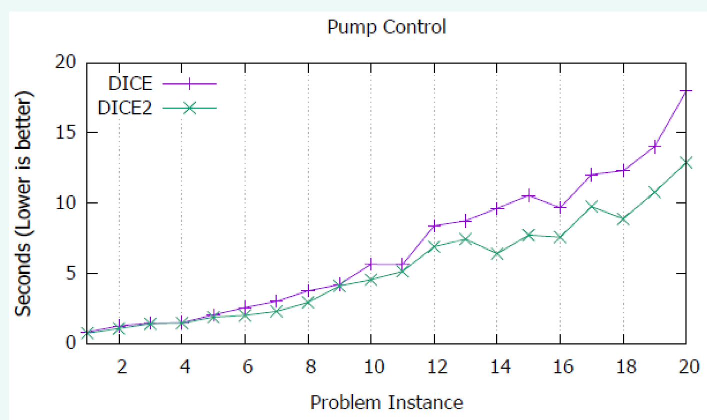

Fig. 5: Pump Control

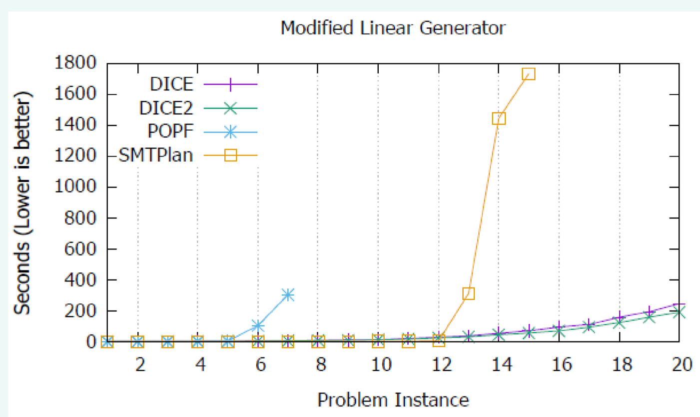

Fig. 6: Modified Linear Generator

References

Bajada, J.; Fox, M.; and Long, D. 2015. *Temporal Planning with Semantic Attachment of Non-Linear Monotonic Continuous Behaviours.* In Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI-15), pp. 1523–1529.

Bajada, J; Fox, M.; and Long, D. 2016, *Temporal planning with constants in context.* In Proceedings of the 22nd European Conference on Artificial Intelligence (ECAI 2016), pp. 1712–1713. IOS Press.

Coles, A.; Coles, A.; Fox, M.; and Long, D. 2012. *COLIN: Planning with Continuous Linear Numeric Change.* Journal of Artificial Intelligence Research, 44:1-96.

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

Denenberg, E., Coles, A., Long, D. 2019. *Evaluating the Cost of Employing LPs and STPs in Planning: Lessons Learned From Large Real-Life Domains.* In Proceedings of the 12th International Scheduling and Planning Applications Workshop (SPARK'19).

Fox, M.; and Long, D. 2003. *PDDL2.1: An extension to PDDL for expressing temporal planning domains.* Journal of Artificial Intelligence Research, 20:61-124. AI Access Foundation.

Acknowledgments

This research was partially funded by the UK Engineering and Physical Sciences Research Council (EPSRC) as part of project: *The Autonomic Power System* (Grant Ref: EP/I031650/1).

Department of Artificial Intelligence,
University of Malta,
Msida, MSD 2080,
Malta
josef.bajada@um.edu.mt