Temporal Modalities in Answer Set Programming

Pedro Cabalar
University of Corunna, Spain
cabalar@udc.es

Abstract

Based on the answer set (or stable model) semantics for logic programs, Answer Set Programming (ASP) has become one of the most successful paradigms for practical Knowledge Representation and problem solving. Although ASP is naturally equipped for solving static combinatorial problems up to NP complexity (or Σ_2^P in the disjunctive case) its application to temporal scenarios has been frequent since its very beginning, partly due to its early use for reasoning about actions and change. Temporal problems normally suppose an extra challenge for ASP for several reasons. On the one hand, they normally raise the complexity (in the case of classical planning, for instance, it becomes PSPACE-complete), although this is usually accounted for by making repeated calls to an ASP solver. On the other hand, temporal scenarios also pose a representational challenge, since the basic ASP language does not support temporal expressions. To fill this representational gap, a temporal extension of ASP called Temporal Equilibrium Logic (TEL) was proposed in and extensively studied later. This formalism constitutes a modal, linear-time extension of Equilibrium Logic which, in its turn, is a complete logical characterisation of (standard) ASP based on the intermediate logic of Here-and-There (HT). As a result, TEL is an expressive non-monotonic modal logic that shares the syntax of Linear-Time Temporal Logic (LTL) but interprets temporal formulas under a non-monotonic semantics that properly extends stable models.

2012 ACM Subject Classification Theory of computation → Modal and temporal logics; Theory of computation → Constraint and logic programming; Computing methodologies → Nonmonotonic, default reasoning and belief revision; Computing methodologies → Logic programming and answer set programming; Computing methodologies → Temporal reasoning

Keywords and phrases Logic Programming, Temporal Logic, Answer Set Programming, Modal Logic

Digital Object Identifier 10.4230/LIPIcs.TIME.2020.2

Category Invited Talk

Funding *Pedro Cabalar*: research partially supported by MINECO (grant TIN2017-84453-P) and Xunta de Galicia (grant GPC ED431B 2019/03), Spain.

Acknowledgements This document is a summary of a long term project jointly developed by the Knowledge Representation group (inside IRLab) at the University of Corunna, Spain, led by Pedro Cabalar and the Potassco group at the University of Potsdam, Germany, directed by Torsten Schaub. This includes joint work with, among others, Felicidad Aguado, Martín Diéguez, Roland Kaminski, Fançois Laferriere, Philip Morkisch, Gilberto Pérez, Anna Schuhmann and Concepción Vidal. Authors from other universities that have undoubtely contributed to the project are David Pearce, Philip Balbiani, Luis Fariñas and Jorge Fandinno.

EXTENDED ABSTRACT

Based on the answer set (or stable model) semantics [12] for logic programs, *Answer Set Programming* [4] (ASP) has become one of the most successful paradigms for practical Knowledge Representation and problem solving. Although ASP is naturally equipped for solving static combinatorial problems up to NP complexity (or Σ_2^P in the disjunctive case) its application to temporal scenarios has been frequent since its very beginning, partly due to its early use for reasoning about actions and change [13]. Temporal problems normally suppose an extra challenge for ASP for several reasons. On the one hand, they normally raise the

27th International Symposium on Temporal Representation and Reasoning (TIME 2020).
Editors: Emilio Muñoz-Velasco, Ana Ozaki, and Martin Theobald; Article No. 2; pp. 2:1–2:5
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

complexity (in the case of classical planning, for instance, it becomes PSPACE-complete [5]), although this is usually accounted for by making repeated calls to an ASP solver. On the other hand, temporal scenarios also pose a representational challenge, since the basic ASP language does not support temporal expressions.

To fill this representational gap, a temporal extension of ASP called *Temporal Equilibrium Logic* (TEL) was proposed in [7] and extensively studied later on [1]. This formalism constitutes a modal, linear-time extension of *Equilibrium Logic* [15] which, in its turn, is a complete logical characterisation of (standard) ASP based on the intermediate logic of *Here-and-There* (HT) [14]. As a result, TEL is an expressive non-monotonic modal logic that shares the syntax of Linear-Time Temporal Logic (LTL) [16] but interprets temporal formulas under a non-monotonic semantics that properly extends stable models. This semantics is based on the idea of selecting some LTL temporal models of a theory Γ that satisfy some minimality condition, when examined under the weaker logic of temporal HT (THT). Thus, a temporal stable model of Γ is a kind of selected LTL model of Γ, and so, it has the form of an infinite sequence of states, usually called a *trace*. To put an example, the Yale Shooting scenario [] where we must shoot a loaded gun to kill a turkey, can be encoded in TEL as:

$$\square(\mathit{loaded} \wedge \circ \mathit{shoot} \rightarrow \circ \mathit{dead}) \quad (1)$$
$$\square(\mathit{loaded} \wedge \circ \mathit{shoot} \rightarrow \circ \mathit{unloaded}) \quad (2)$$
$$\square(\mathit{load} \rightarrow \mathit{loaded}) \quad (3)$$
$$\square(\mathit{dead} \rightarrow \circ \mathit{dead}) \quad (4)$$
$$\square(\mathit{loaded} \wedge \neg \circ \mathit{unloaded} \rightarrow \circ \mathit{loaded}) \quad (5)$$
$$\square(\mathit{unloaded} \wedge \neg \circ \mathit{loaded} \rightarrow \circ \mathit{unloaded}) \quad (6)$$

In this way, under TEL semantics, implication $\alpha \rightarrow \beta$ has a similar behaviour to a directional inference rule, normally reversed as $\beta \leftarrow \alpha$ or $\beta :- \alpha$ in logic programming notation. The last two rules, (5)-(6), encode the *inertia law* for fluents *loaded* and *unloaded*, respectively. Note the use of ¬ in these two rules: it actually corresponds to *default negation*, that is, $\neg\alpha$ is read as "there is no evidence about α." For instance, (5) is read as "if the gun was loaded and we cannot prove that it will become unloaded then it stays loaded."

Computation of temporal stable models is a complex task. THT-satisfiability has been classified [8] as PSPACE-complete, that is, the same complexity as LTL-satisfiability, whereas TEL-satisfiability rises to EXPSPACE-completeness, as proved in [3]. In this way, we face a similar situation as in the non-temporal case where HT-satisfiability is NP-complete like SAT, whereas existence of equilibrium model (for arbitrary theories) is Σ_2^P-complete (like disjunctive ASP). There exist a pair of tools, `STeLP` [6] and `ABSTEM` [9], that allow computing (infinite) temporal stable models (represented as Büchi automata). These tools can be used to check verification properties that are usual in LTL, like the typical safety, liveness and fairness conditions, but in the context of temporal ASP. Moreover, they can also be applied for planning problems that involve an indeterminate or even infinite number of steps, such as the non-existence of a plan. The tool `ABSTEM` also accepts pairs of theories to decide different types of equivalence: LTL-equivalence, TEL-equivalence (i.e. coincidence in the set of TS-models) and strong equivalence (i.e., THT-equivalence). Moreover, when strong equivalence fails, `ABSTEM` obtains a context, that is, an additional formula that added to the compared theories makes them behave differently.

The original definition of TEL was thought as a direct non-monotonic extension of standard LTL, so that models had the form of infinite traces. However, this rules out computation by ASP technology and is unnatural for applications like planning, where plans amount to finite prefixes of one or more traces [11]. In a recent line of research [10], TEL was extended to cope with finite traces (which are closer to ASP computation). On the one hand, this amounts to a restriction of THT and TEL to finite traces. On the other hand, this is similar to the restriction of LTL to LTL_f advocated by [11]. Our new approach, dubbed TEL_f, has the following advantages. First, it is readily implementable via ASP technology. Second, it can be reduced to a normal form which is close to logic programs and much simpler than the one obtained for TEL. Finally, its temporal models are finite and offer a one-to-one correspondence to plans. Interestingly, TEL_f also sheds light on concepts and methodology used in incremental ASP solving when understanding incremental parameters as time points.

Another distinctive feature of TEL_f is the inclusion of future as well as past temporal operators. When using the causal reading of program rules, it is generally more natural to draw upon the past in rule bodies and to refer to the future in rule heads. As well, past operators are much easier handled computationally than their future counterparts when it comes to incremental reasoning, since they refer to already computed knowledge.

TEL_f is implemented in the `telingo` system, extending the ASP system `clingo` to compute the temporal stable models of (non-ground) temporal logic programs. To this end, it extends the full-fledged input language of `clingo` with temporal operators and computes temporal models incrementally by multi-shot solving using a modular translation into ASP. `telingo` is freely available at `github`[1]. The interested reader might have a good time playing with the examples given in the `examples` folder at the same site. For instance, under `telingo` syntax, our theory (1)-(6) would be represented[2] as

```
#program dynamic.
dead :- shoot, 'loaded.
unloaded :- shoot, 'unloaded.
loaded :- load.
dead :- 'dead.
loaded :- 'loaded, not unloaded.
unloaded :- 'unloaded, not loaded.
```

The `telingo` input language actually allows the introduction of arbitrary LTL formulas in constraints or past formulas in the rule bodies (conditions).

Similar to the extension of LTL_f to its (linear) dynamic logic counterpart LDL_f [11], we just introduced in [2] a dynamic extension of HT that draws up upon this linear version of dynamic logic. We refer to the resulting logic as *(Linear) Dynamic logic of Here-and-There* (DHT for short). As usual, the equilibrium models of DHT are used to define temporal stable models and induce the non-monotonic counterpart of DHT, referred to as *(Linear) Dynamic Equilibrium Logic* (DEL). In doing so, we actually parallel earlier work extending HT with LTL, ultimatly leading to THT and TEL. To put an example in DEL, the formula $[\neg help^*](\neg help \rightarrow sos)$ behaves as a logic program rule that repeats sending an *sos* while no evidence of *help* has been received along a sequence of states. DEL is general enough to cover LDL, as it shares the same syntax but introduces non-monotonicity with the definition of temporal stable models. It also covers LTL and TEL as particular cases, since LTL temporal operators can be defined as particular cases of DEL expressions: for instance $\Box\alpha$ (i.e. α always holds) can be represented in DEL as $[\top^*]\alpha$. The satisfiability problem in DEL is ExpSpace-complete; it thus coincides with that of TEL but goes beyond that of LDL and LTL, both being PSpace-complete.

[1] `https://github.com/potassco/telingo`

[2] The left upper commas are read as *previously* and correspond to the past operator dual of *next* "$\circ$". The $\Box$ operator is implicit in all dynamic rules.

These recent results open several interesting topics for future study. First, the version of DEL for finite traces, DEL_f, seems a natural step to follow, similar to the relation of LDL and LDL_f. We plan to propose and analyse this variation in an immediate future. As a second open topic, it would be interesting to adapt existing model checking techniques (based on automata construction) for temporal logics to solve the problem of existence of temporal stable models. This was done for infinite traces in [8, 6], but no similar method has been implemented for finite traces on TEL_f or DEL_f yet. The importance of having an efficient implementation of such a method is that it would allow deciding non-existence of a plan in a given planning problem, something not possible by current incremental solving techniques. Another interesting topic is the optimization of grounding in temporal ASP specifications as those handled by `telingo`. The current grounding of `telingo` is inherited from incremental solving in `clingo` and does not exploit the semantics of temporal expressions that are available now in the input language. Finally, we envisage to extend the `telingo` system with features of DEL in order to obtain a powerful system for representing and reasoning about dynamic domains, not only providing an effective implementation of TEL and DEL but, furthermore, a platform for action and control languages.

References

1 F. Aguado, P. Cabalar, M. Diéguez, G. Pérez, and C. Vidal. Temporal equilibrium logic: a survey. *Journal of Applied Non-Classical Logics*, 23(1-2):2–24, 2013.

2 A. Bosser, P. Cabalar, M. Diéguez, and T. Schaub. Introducing temporal stable models for linear dynamic logic. In M. Thielscher, F. Toni, and F. Wolter, editors, *Proceedings of the Sixteenth International Conference on Principles of Knowledge Representation and Reasoning (KR'18)*, pages 12–21. AAAI Press, 2018.

3 Laura Bozzelli and David Pearce. On the complexity of temporal equilibrium logic. In *Proceedings of the 30th Annual ACM/IEEE Symposium of Logic in Computer Science (LICS'15)*, Kyoto, Japan, 2015. (to appear).

4 G. Brewka, T. Eiter, and M. Truszczyński. Answer set programming at a glance. *Communications of the ACM*, 54(12):92–103, 2011.

5 Tom Bylander. The computational complexity of propositional strips planning. *Artificial Intelligence*, 69(1):165–204, 1994. `doi:10.1016/0004-3702(94)90081-7`.

6 P. Cabalar and M. Diéguez. STELP - a tool for temporal answer set programming. In *LPNMR'11*, volume 6645 of *Lecture Notes in Computer Science*, pages 370–375, 2011.

7 P. Cabalar and G. Perez. Temporal Equilibrium Logic: A First Approach. In *Proceedings of the 11^{th} International Conference on Computer Aided Systems Theory (EUROCAST'07)*, page 241–248, 2007.

8 Pedro Cabalar and Stéphane Demri. Automata-based computation of temporal equilibrium models. In *21st International Symposium on Logic-Based Program Synthesis and Transformation (LOPSTR'11)*, 2011.

9 Pedro Cabalar and Martín Diéguez. Strong equivalence of non-monotonic temporal theories. In *Proceedings of the 14th International Conference on Principles of Knowledge Representation and Reasoning (KR'14)*, Vienna, Austria, 2014.

10 Pedro Cabalar, Roland Kaminski, Torsten Schaub, and Anna Schuhmann. Temporal answer set programming on finite traces. *Theory and Practice of Logic Programming*, 18(3-4):406–420, 2018.

11 G. De Giacomo and M. Vardi. Linear temporal logic and linear dynamic logic on finite traces. In F. Rossi, editor, *Proceedings of the Twenty-third International Joint Conference on Artificial Intelligence (IJCAI'13)*, pages 854–860. IJCAI/AAAI Press, 2013.

12 M. Gelfond and V. Lifschitz. The Stable Model Semantics For Logic Programming. In *Proc. of the 5th International Conference on Logic Programming (ICLP'88)*, page 1070–1080, Seattle, Washington, 1988.

13 Michael Gelfond and Vladimir Lifschitz. Representing action and change by logic programs. *Journal of Logic Programming*, 17(2/3&4):301–321, 1993.

14 A. Heyting. Die formalen Regeln der intuitionistischen Logik. *Sitzungsberichte der Preussischen Akademie der Wissenschaften. Physikalisch-mathematische Klasse*, 1930.

15 D. Pearce. A New Logical Characterisation of Stable Models and Answer Sets. In *Proc. of Non-Monotonic Extensions of Logic Programming (NMELP'96)*, pages 57–70, Bad Honnef, Germany, 1996.

16 A. Pnueli. The temporal logic of programs. In *18th Annual Symposium on Foundations of Computer Science*, pages 46–57. IEEE Computer Society Press, 1977.