Dynamic and Temporal Answer Set Programming on Linear Finite Traces*

(Invited Talk)

Pedro Cabalar[1] and Torsten Schaub[2]

[1] University of Corunna, Spain
cabalar@udc.es
[2] University of Potsdam, Germany
torsten@cs.uni-potsdam.de

The logical foundations of Answer Set Programming (ASP; [21]) rest upon the logic of Here-and-There (HT; [18]), or more precisely its equilibrium models [22] that correspond to stable models semantics [15]. For defining extensions to ASP from firm logical principles, it has thus become good practice to first elaborate upon them in the setting of HT in order to afterwards consider the respective language fragments that are well suited in the context of logic programming.

This avenue was also followed in [11], which gave rise to the temporal extension of HT called *Temporal Here-and-There* and its non-monotonic counterpart *Temporal Equilibrium Logic* (for short THT and TEL [1]). More precisely, TEL builds upon an extension of the logic of HT with Linear Temporal Logic (LTL; [23]). This results in an expressive non-monotonic modal logic, which extends traditional temporal logic programming approaches [7] to the general syntax of LTL and possesses a computational complexity beyond LTL [4]. As in LTL, a model in TEL is an *infinite* sequence of states, called a *trace*. However, this rules out computation by ASP technology (and necessitates model checking) and is unnatural for applications like planning, where plans amount to finite prefixes of one or more traces (cf. [2, 12]).

Unlike this, we recently proposed in [10] an alternative combination of the logics of HT and LTL whose semantics rests upon *finite* traces. 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 [12]; see also [2]. 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. We associate this with the following benefits. 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. A similar argument was put forward by [13] in his proposal of "declarative past and imperative future." This format also yields a simpler normal form and lends itself to a systematic modeling methodology which favors

* This is a revised version of the one submitted to *Actions@KR18*.

the definition of states in terms of the past rather than mixing in future operators. For instance, in reasoning about actions, the idea is to derive action effects for the current state and check their preconditions in the previous one, rather than to represent this as a transition from the current to the next state. This methodology aligns state constraints, effect axioms, etc. to capture the present state. 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 [9], 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 [14] using a modular translation into ASP. `telingo` is freely available at `github`[3]. The interested reader might have a good time playing with a few examples given in the `examples` folder at the same site.

Similar to the extension of LTL_f to its (linear) dynamic logic counterpart LDL_f [12], we introduced in [3] a dynamic extension of HT that draws up upon this linear version of dynamic logic. We elaborate upon its restriction to finite traces in [8]. 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, ultimately leading to THT and TEL.

In fact, we show that THT (and its equilibrium counterpart TEL) can be embedded into our new logic DHT (and DEL, respectively) — just as LTL can be put in LDL. Moreover, we prove that 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. In fact, the membership part of this result is obtained by means of an automata-based method for computing DEL models. Finally, we show that the monotonic base logic of DEL, namely DHT, allows us to decide strong equivalence in DEL; this reinforces the adequacy of the relation between both logics.

In the context of the version of DEL for finite traces, DEL_f, we developed a translation of any (converse-free) arbitrary DEL_f theory into a propositional theory (under the semantics of HT) and in turn into a logic program. This translation has turned out to be non-trivial: it is based on unfolding path expressions, something potentially equivalent to the execution of a sequential program. Termination is guaranteed by some preprocessing steps for normalizing path expressions.

These recent results open several interesting topics for future study. As an 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 [6, 5], 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

[3] https://github.com/potassco/telingo

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, like $\mathcal{A}, \mathcal{B}, \mathcal{C}$ [16, 17] or GOLOG [19].

References

1. Aguado, F., Cabalar, P., Diéguez, M., Pérez, G., Vidal, C.: Temporal equilibrium logic: a survey. Journal of Applied Non-Classical Logics **23**(1-2), 2–24 (2013)
2. Baier, J., McIlraith, S.: Planning with first-order temporally extended goals using heuristic search. In: Gil, Y., Mooney, R. (eds.) Proceedings of the Twenty-first National Conference on Artificial Intelligence (AAAI'06). pp. 788–795. AAAI Press (2006)
3. Bosser, A., Cabalar, P., Diéguez, M., Schaub, T.: Introducing temporal stable models for linear dynamic logic. In: Thielscher, M., Toni, F., Wolter, F. (eds.) Proceedings of the Sixteenth International Conference on Principles of Knowledge Representation and Reasoning (KR'18). pp. 12–21. AAAI Press (2018)
4. Bozzelli, L., Pearce, D.: On the complexity of temporal equilibrium logic. In: Proceedings of the Thirtieth Annual Symposium on Logic in Computer Science (LICS'15). pp. 645–656. IEEE Computer Society Press (2015)
5. Cabalar, P., Demri, S.: Automata-based computation of temporal equilibrium models. In: Vidal, G. (ed.) Proceedings of the Twenty-first International Symposium on Logic-Based Program Synthesis and Transformation (LOPSTR'11). Lecture Notes in Computer Science, vol. 7225, pp. 57–72. Springer-Verlag (2011)
6. Cabalar, P., Diéguez, M.: STELP — a tool for temporal answer set programming. In: Delgrande, J., Faber, W. (eds.) Proceedings of the Eleventh International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'11). Lecture Notes in Artificial Intelligence, vol. 6645, pp. 370–375. Springer-Verlag (2011)
7. Cabalar, P., Diéguez, M., Vidal, C.: An infinitary encoding of temporal equilibrium logic. Theory and Practice of Logic Programming **15**(4-5), 666–680 (2015)
8. Cabalar, P., Diéguez, M., Schaub, T.: Towards dynamic answer set programming over finite traces. In: Lierler and Woltran [20], to appear.
9. Cabalar, P., Kaminski, R., Morkisch, P., Schaub, T.: telingo = ASP + time. In: Lierler and Woltran [20], to appear.
10. Cabalar, P., Kaminski, R., Schaub, T., Schuhmann, A.: Temporal answer set programming on finite traces. Theory and Practice of Logic Programming **18**(3-4), 406–420 (2018)
11. Cabalar, P., Vega, G.P.: Temporal equilibrium logic: A first approach. In: Moreno-Díaz, R., Pichler, F., Quesada-Arencibia, A. (eds.) Proceedings of the Eleventh International Conference on Computer Aided Systems Theory (EUROCAST'17). Lecture Notes in Computer Science, vol. 4739, pp. 241–248. Springer-Verlag (2007)
12. De Giacomo, G., Vardi, M.: Linear temporal logic and linear dynamic logic on finite traces. In: Rossi, F. (ed.) Proceedings of the Twenty-third International Joint Conference on Artificial Intelligence (IJCAI'13). pp. 854–860. IJCAI/AAAI Press (2013)
13. Gabbay, D.: The declarative past and imperative future: Executable temporal logic for interactive systems. In: Banieqbal, B., Barringer, H., Pnueli, A. (eds.) Proceedings of the Conference on Temporal Logic in Specification. Lecture Notes in Computer Science, vol. 398, pp. 409–448. Springer-Verlag (1987)

14. Gebser, M., Kaminski, R., Kaufmann, B., Schaub, T.: Multi-shot ASP solving with clingo. Theory and Practice of Logic Programming **19**(1), 27–82 (2019), http://arxiv.org/abs/1705.09811
15. Gelfond, M., Lifschitz, V.: The stable model semantics for logic programming. In: Kowalski, R., Bowen, K. (eds.) Proceedings of the Fifth International Conference and Symposium of Logic Programming (ICLP'88). pp. 1070–1080. MIT Press (1988)
16. Gelfond, M., Lifschitz, V.: Action languages. Electronic Transactions on Artificial Intelligence **3**(6), 193–210 (1998)
17. Giunchiglia, E., Lee, J., Lifschitz, V., McCain, N., Turner, H.: Nonmonotonic causal theories. Artificial Intelligence **153**(1-2), 49–104 (2004)
18. Heyting, A.: Die formalen Regeln der intuitionistischen Logik. In: Sitzungsberichte der Preussischen Akademie der Wissenschaften, p. 42–56. Deutsche Akademie der Wissenschaften zu Berlin (1930), reprint in Logik-Texte: Kommentierte Auswahl zur Geschichte der Modernen Logik, Akademie-Verlag, 1986.
19. Levesque, H., Reiter, R., Lespérance, Y., Lin, F., Scherl, R.: GOLOG: A logic programming language for dynamic domains. Journal of Logic Programming **31**(1-3), 59–83 (1997)
20. Lierler, Y., Woltran, S. (eds.): Proceedings of the Fifteenth International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'19), vol. 11481 (2019)
21. Lifschitz, V.: Answer set planning. In: de Schreye, D. (ed.) Proceedings of the International Conference on Logic Programming (ICLP'99). pp. 23–37. MIT Press (1999)
22. Pearce, D.: A new logical characterisation of stable models and answer sets. In: Dix, J., Pereira, L., Przymusinski, T. (eds.) Proceedings of the Sixth International Workshop on Non-Monotonic Extensions of Logic Programming (NMELP'96). Lecture Notes in Computer Science, vol. 1216, pp. 57–70. Springer-Verlag (1997)
23. Pnueli, A.: The temporal logic of programs. In: Proceedings of the Eight-teenth Symposium on Foundations of Computer Science (FOCS'77). pp. 46–57. IEEE Computer Society Press (1977)