# Formal Foundations of Incremental Dynamic Answer Set Programming

Pedro Cabalar[1], Martín Diéguez[2], François Laferrière[3], and Torsten Schaub[3]

[1] University of Corunna, Spain
[2] University of Angers, France
[3] University of Potsdam, Germany

**Abstract.** Extensions of Answer Set Programming (ASP) with language constructs from dynamic and temporal logics provide an expressive computational framework for modeling dynamic applications. In this work, we explore the logical foundations of temporal and dynamic ASP. We focus on the identification and study of fragments of temporal and dynamic logics, with the goal of developing a theory for incremental solving of dynamic ASP.

## 1 Introduction

Representing and reasoning about dynamic domains is a key problem in the field of Knowledge Representation and Reasoning. Temporal and dynamic logics have the ability to reason about a timeline. Therefore, they are commonly used to address those types of problems where there is a need for modeling action and change. With this kind of approach, a solution to a dynamic problem is a sequence of states, called a trace, with each state corresponding to a different time point in the timeline. While most of the work on this topic is based on monotonic logics and focus on infinite traces, the recent rise of Answer Set Programming (ASP; [8]) has increased the interest for approaches using non-monotonic logic and finite traces. The main reasons being that it's better fitted for most real world problems, and is computationally more viable.

Equilibrium Logic (EL; [10]) provides a logical foundation for ASP. To enrich the modeling power of ASP, EL has been extended with temporal logic into Temporal Equilibrium Logic [4] and with dynamic logic into Dynamic Equilibrium Logic [1]. These two logics provide a stepping stone for representing and reasoning about dynamic domains with ASP, but this research area has barely been explored and a lot of ground work remains to be done.

Our work focus on the identification and study of fragments of temporal and dynamic logics.

## 2 Identification of fragments of logic

The first focus of our work is the identification and the study of syntactic fragments of temporal and dynamic logics. The goal is to find interesting fragments from a knowledge representation perspective that can also facilitate reasoning.

Our primary focus is the study of past-future temporal programs [4]. This fragment is quite expressive, in fact, such programs are sufficient to capture common action languages, and hence are enough to model planning problems. The characteristic of those programs is that the past is declarative and the future is imperative. The notions of declarative past and imperative future were first put forward by Gabbay et al.[6], the main idea being that, 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.

Another important reason for studying this fragment is related to incremental reasoning [9]. Roughly speaking, incremental solving is a step-oriented approach that avoids redundancies by gradually processing the extensions to a problem rather than repeatedly re-processing the entire extended problem. When understanding incremental parameters as time points, the incremental approach correspond to looking for solutions, first in traces of length 1, then in traces of length 2, etc., until a (optimal) solution is found. This approach is particularly interesting in real world problems where the minimal number of time points necessary to find a solution is rarely known. Having a modeling language in which the past can depend on the future makes it difficult to apply incremental reasoning [3]. In past-future programs, past is only declarative, therefore it cannot depend on the future. This makes past-future fragments an ideal choice for incremental reasoning.

## 3 Optimization of Tseitin translations

Implementation of temporal and dynamic ASP heavily depends on translation between different syntactic fragments of logic. For instance, in TELINGO [3, 2] which is the extension with temporal and dynamic logics of the ASP system CLINGO [7], dynamic formulas are first translated into a normal form of temporal logic, and temporal formulas are afterward translated into regular logic programs. These translations rely on the introduction of auxiliary atoms in a Tseitin-style [11] for, first, guaranteeing that its result is of polynomial size wrt the input formula, and, second, surmounting the fact that translations between temporal, dynamic and regular formulas are usually impossible without extending the language. Even if those translations are polynomial in size, it doesn't imply easy solving, as the formulas can still end up big.

The second focus of my research is the simplification of the formulas obtained from Tseitin-style translation. One particularly useful simplification can be done using the temporal extension of Ferraris's *completion lemma* [5] which allows the substitution between implication and equivalence in certain context.

## References

1. Cabalar, P., Diéguez, M., Schaub, T.: Towards dynamic answer set programming over finite traces. In: LPNMR. Lecture Notes in Computer Science, vol. 11481, pp. 148–162. Springer (2019)

2. Cabalar, P., Diéguez, M., Schaub, T., Laferrière, F.: Implementing dynamic answer set programming. CoRR **abs/2002.06916** (2020)
3. Cabalar, P., Kaminski, R., Morkisch, P., Schaub, T.: telingo = ASP + time. In: LPNMR. Lecture Notes in Computer Science, vol. 11481, pp. 256–269. Springer (2019)
4. Cabalar, P., Kaminski, R., Schaub, T., Schuhmann, A.: Temporal answer set programming on finite traces. TPLP **18**(3-4), 406–420 (2018)
5. Ferraris, P.: Answer sets for propositional theories. In: International Conference on Logic Programming and Nonmonotonic Reasoning. pp. 119–131. Springer (2005)
6. Gabbay, D.: The declarative past and imperative future. In: Temporal logic in specification. pp. 409–448. Springer (1989)
7. Gebser, M., Kaminski, R., Kaufmann, B., Ostrowski, M., Schaub, T., Wanko, P.: Theory solving made easy with clingo 5. In: ICLP (Technical Communications). OASICS, vol. 52, pp. 2:1–2:15. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik (2016)
8. Gebser, M., Kaminski, R., Kaufmann, B., Schaub, T.: Answer Set Solving in Practice. Synthesis Lectures on Artificial Intelligence and Machine Learning, Morgan & Claypool Publishers (2012)
9. Gebser, M., Kaminski, R., Kaufmann, B., Schaub, T.: Multi-shot ASP solving with clingo. TPLP **19**(1), 27–82 (2019)
10. Pearce, D.: Equilibrium logic. Annals of Mathematics and Artificial Intelligence **47**(1), 3–41 (2006)
11. Tseitin, G.S.: On the complexity of derivation in propositional calculus. In: Automation of reasoning, pp. 466–483. Springer (1983)