**I) What is our research problem?** The **satisfiability problem** is one of the most fundamental problems studied in computer science, with multiple applications in knowledge representation [17], formal verification [20], and planning [11]. It has also been a central undertaking of mathematical logic since the publication of *Hilbert's Grundzüge der theoretischen Logik*. This problem is parametrised by a formal language $\mathcal{L}$ (a.k.a. *logic*, *e.g.* first-order logic), and asks if an input formula $\varphi$ (written in the syntax of $\mathcal{L}$) is satisfied in *some* structure, *i.e.* it has a *model*. Formally:

| | **(Finite) Satisfiability Problem** | **(Finite) Entailment Problem** |
|---|---|---|
| *Parameters:* | A logic $\mathcal{L}$. | Logics $\mathcal{L}_1$ and $\mathcal{L}_2$. |
| *Input:* | An $\mathcal{L}$-formula $\varphi$. | An $\mathcal{L}_1$-formula $\varphi_1$ and an $\mathcal{L}_2$-formula $\varphi_2$. |
| *Question:* | Does $\varphi$ have a (finite) model? | Do all (finite) models of $\varphi_1$ satisfy $\varphi_2$? |

The satisfiability problem generalises to the above-defined entailment problem.[1] In the restricted case of *finite* satisfiability and entailment problems, we are interested in models of finite sizes only. These problems have served as a yardstick for measuring the "quality" of logics, driving the discovery of decidable first-order fragments [5, 24] in the last decades. Satisfiability underlies symbolic program verification [1], while (query) entailment is the main reasoning problem over incomplete databases in Ontology-Based Data Access (OBDA) [23]. In formal verification, the input formula expresses a system specification, and satisfiability determines its feasibility. In OBDA, we seek certain (true in all extensions) query answers over incomplete data. There, $\varphi_1$ encodes a database $\mathcal{D}$ together with some integrity constraints, while $\varphi_2$ represents the query. Entailment checks whether all "valid" completions of $\mathcal{D}$ (w.r.t. $\varphi_1$) satisfy the query $\varphi_2$.

**II) On what logics do we focus?** FINGO focuses on families of *dynamic logics* and query languages supporting navigational features, and studies their finite satisfiability and entailment problems. In the case of decidability, we also aim to establish tight complexity bounds for these problems.

**We study logics supporting regular path expressions, focusing on (extensions of) well-known program verification logics like Propositional Dynamic Logic (PDL) and the $\mu$-calculus.**

Regular path expressions have become the de facto standard for querying graph-structured data, with languages like Cypher, Gremlin, and PGQL widely adopted in industry [2]. They also arise naturally in bioinformatics and cheminformatics [10, 21], where domain experts frequently seek associations among entities in protein, cellular, drug, and disease networks (stored as graphs). In this setting, identifying gene–disease–drug associations (paths in the data) can aid in the discovery of new treatment strategies. Consult the full proposal for more motivation behind the choice of PDL [9] and the $\mu$-calculus [19] as the main object of study of FINGO.

**III) What are the existing approaches?** Dynamic logics have the *tree-like (counter)model property*, *i.e.* whenever $\varphi_1$ does not entail $\varphi_2$, there exists an infinite "tree-representable" model of $\varphi_1$ violating $\varphi_2$. Hence, one can benefit from the theory of formal languages and solve the unrestricted satisfiability and entailment problems via **automata-based methods**. One relies on Alternating Parity Tree Automata (APTA) [25], finite-state machines that read infinite node-labelled trees top-down (accepting or rejecting a tree). An algorithm solving the entailment problem via APTA is given on the next page. Despite its simplicity, its adaptation to new settings presents two key challenges. First, defining the right notion of automata and solving its non-emptiness problem.

[1] Note that $\varphi$ entails $\psi$ if and only if $\varphi \wedge \neg\psi$ is unsatisfiable.

Second, designing a suitable encoding of models as tree-like structures.

**Algorithm 1:** Deciding the entailment problem for the input formulæ $\varphi_1$ and $\varphi_2$.

1 Construct APTA $\mathcal{A}_1$ and $\mathcal{A}_2$ accepting all tree-like models of $\varphi_1$ and $\varphi_2$.
2 Construct an APTA $\mathcal{A}_2^{\neg}$ accepting all tree-like structures violating $\varphi_2$. `// The APTA` $\mathcal{A}_2^{\neg}$ `is usually constructed through projection and complementation of` $\mathcal{A}_2$.
3 Return `YES` if the *intersection* of the languages $\mathcal{A}_1$ and $\mathcal{A}_2^{\neg}$ is empty, and `NO` otherwise.

For the satisfiability problem, we build $\mathcal{A}_1$ and test for its non-emptiness. Algorithm 1 scales well with generalisations of APTA, and was applied by M. Ortiz (the PI's mentor) to solve the entailment problem for Conjunctive Regular Path Queries (CRPQs, an expressive navigational query language) and expressive extensions of PDL [6, 22]. We stress that the correctness of this method heavily relies on the existence of infinite tree-like (counter)models encodable via APTA.

**IV) What about finite-model reasoning?** Unfortunately, the presented technique **fails** when moving to the finite-model reasoning case, as finite models for formulæ written in (extensions of) PDL and the $\mu$-calculus are unlikely to be tree-like. As an example, abstractions of computer programs as well as real-world systems in biochemistry are inherently finite and frequently involve complex motifs like cycles and cliques, absent in tree-like structures. Despite the broad applicability of dynamic logics, only tiny progress has been made in the past on their finite satisfiability and entailment problems. We believe this is largely due to the absence of available automata-based methods,[2] crucial for dealing with regular expressions or fixed-points. We stress:

**Due to the lack of automata-based tools for finite-model reasoning, little progress has been made on finite satisfiability and entailment problems for dynamic logics over the past 30 years.**

The above claim refers to logics for which finite and unrestricted satisfiability or entailment do not coincide and recent work by Figueira [8]). The *only known exceptions* are the entailment of CRPQs over $\mathcal{ALC}$ (PDL without regular expressions), a recent breakthrough by Gutowski et al. [15, 16] established after a series of mathematically laborious works [12, 13, 14], and the finite satisfiability problem for the $\mu$-calculus with converse by M. Bojańczyk [3]. The second paper is based on a simple yet ingenious idea: Bojańczyk replaced APTA by a dedicated finite-graph operating automaton and proved decidability of the non-emptiness problem. His approach effectively solves the finite satisfiability for the logic, but unfortunately does not extend to richer logics, like the graded $\mu$-calculus with converse [4] (for which the finite satisfiability problem has been open for over two decades), or to the entailment problem (*e.g.* of CRPQs). This is where FINGO aims to make a unique and significant contribution.

**V) What are the main goals of FINGO?** The main goal of FINGO can be summarised as follows.

The mission of FINGO is to **alter the landscape of available techniques** for finite-model reasoning and finite entailment problems, and apply them to **solve long-standing open problems** in the area of dynamic logics, with an emphasis on (extensions of) Propositional Dynamic Logic and the $\mu$-calculus. FINGO will achieve this goal by **designing and providing complexity results for novel automata models operating on finite graphs**, extending APTA.

[2] Other possible tools are tableaux methods and cyclic proof systems, but they can be reformulated as automata.

The main research directions of FINGO are structured into work packages (WPs) outlined below. Each work package develops algorithms for novel finite-graph automata, motivated by their applications in OBDA and decision procedures for dynamic logic.

**(WP1) Adapt Algorithm 1 to the setting of finite-graph automata and OBDA.**

**(WP2) Solve the non-emptiness problem for two-way finite-graph automata with counting.**

**(WP3) Understand the limits of automata-based approaches to the finite satisfiability problem by analysing extensions of finite-graph automata with pebbles, counters, or storage.**

(WP1) will provide a uniform algorithm for navigational query answering in OBDA under the finite model semantics. (WP2) will establish a novel automata-based technique for finite-model reasoning for logics with inverses, counting, and regular expressions, and will settle the long-standing open problem of finite satisfiability for the two-way graded $\mu$-calculus. Finally, (WP3) will provide partial progress on techniques for generalisations of PDL like LoopPDL [7] or Non-Regular PDL [18], whose finite satisfiability problems have remained open for over 40 years.


[1] E. Ábrahám and G. Kremer. "Satisfiability Checking: Theory and Applications". In: *SEFM*. 2016. DOI: 10.1007/978-3-319-41591-8_2.

[2] R. Angles, M. Arenas, P. Barceló, A. Hogan, J. Reutter, and D. Vrgoč. "Foundations of Modern Query Languages for Graph Databases". In: *ACM Computing Surveys* (2017). DOI: 10.1145/3104031.

[3] M. Bojańczyk. "Two-Way Alternating Automata and Finite Models". In: *ICALP*. 2002. DOI: 10.1007/3-540-45465-9_71.

[4] P. Bonatti, C. Lutz, A. Murano, and M. Vardi. "The Complexity of Enriched Mu-Calculi". In: *Log. Methods Comput. Sci.* (2008). DOI: 10.2168/LMCS-4(3:11)2008.

[5] E. Börger, E. Grädel, and Y. Gurevich. *The Classical Decision Problem*. 1997. ISBN: 3540423249.

[6] D. Calvanese, T. Eiter, and M. Ortiz. "Answering Regular Path Queries in Expressive Description Logics: An Automata-Theoretic Approach". In: *AAAI*. 2007. ISBN: 9781577353232.

[7] R. Danecki. "Propositional Dynamic Logic with Strong Loop Predicate". In: *MFCS*. 1984. DOI: 10.1007/BFB0030342.

[8] D. Figueira, S. Figueira, and E. P. Baque. "Finite Controllability for Ontology-Mediated Query Answering of CRPQ". In: *KR*. 2020. DOI: 10.24963/kr.2020/39.

[9] M. J. Fischer and R. E. Ladner. "Propositional Dynamic Logic of Regular Programs". In: *J. Comput. Syst. Sci.* (1979). DOI: 10.1016/0022-0000(79)90046-1.

[10] J. Galgonek, T. Hurt, V. Michlíková, P. Onderka, J. Schwarz, and J. Vondrásek. "Advanced SPARQL Querying in Small Molecule Databases". In: *J. Cheminformatics* (2016). DOI: 10.1186/S13321-016-0144-4.

[11] M. Ghallab, D. Nau, and P. Traverso. *Automated Planning - Theory and Practice*. 2004. ISBN: 978-1-55860-856-6.

[12] T. Gogacz, V. Gutiérrez-Basulto, A. Gutowski, Y. Ibáñez-García, and F. Murlak. "On Finite Entailment of Non-Local Queries in Description Logics". In: *KR*. 2020. DOI: 10.24963/KR.2020/43.

[13] T. Gogacz, V. Gutiérrez-Basulto, Y. Ibáñez-García, J. C. Jung, and F. Murlak. "On Finite and Unrestricted Query Entailment beyond SQ with Number Restrictions on Transitive Roles". In: *IJCAI*. 2019. DOI: 10.24963/ijcai.2019/238.

[14] T. Gogacz, Y. A. Ibáñez-García, and F. Murlak. "Finite Query Answering in Expressive Description Logics with Transitive Roles". In: *KR*. 2018. ISBN: 978-1-57735-803-9.

[15] V. Gutiérrez-Basulto, A. Gutowski, Y. Ibáñez-García, and F. Murlak. "Finite Entailment of UCRPQs over ALC Ontologies". In: *KR*. 2022. ISBN: 978-1-956792-01-0.

[16] V. Gutiérrez-Basulto, A. Gutowski, Y. A. Ibáñez-García, and F. Murlak. "Containment of Graph Queries Modulo Schema". In: *PODS* (2024). DOI: 10.1145/3651140.

[17] P. Hitzler, M. Krötzsch, and S. Rudolph. *Foundations of Semantic Web Technologies*. 2010. ISBN: 9781420090505.

[18] T. Koren and A. Pnueli. "There Exit Decidable Context Free Propositional Dynamic Logics". In: *Logics of Programs*. 1983. DOI: 10.1007/3-540-12896-4_369.

[19] D. Kozen. "Results on the Propositional mu-Calculus". In: *Theoretical Computer Science* (1983). DOI: 10.1016/0304-3975(82)90125-6.

[20] D. Kroening and O. Strichman. *Decision Procedures - An Algorithmic Point of View, Second Edition*. 2016. DOI: 10.1007/978-3-662-50497-0.

[21] A. Lysenko, I. Roznovat, M. Saqi, A. Mazein, C. Rawlings, and C. Auffray. "Representing and Querying Disease Networks using Graph Databases". In: *BioData Min.* (2016). DOI: 10.1186/S13040-016-0102-8.

[22] M. Ortiz. "Query Answering in Expressive Description Logics: Techniques and Complexity Results". PhD thesis. TU Wien, AT, 2010.

[23] A. Poggi, D. Lembo, D. Calvanese, G. De Giacomo, M. Lenzerini, and R. Rosati. "Linking Data to Ontologies". In: *Journal on Data Semantics* (2008). DOI: 10.1007/978-3-540-77688-8_5.

[24] I. Pratt-Hartmann. *Fragments of First-Order Logic*. Oxford University Press, 2023. ISBN: 9780192867964.

[25] T. Wilke. "Alternating Tree Automata, Parity Games, and Modal Mu-Calculus". In: (2001). DOI: 10.36045/bbms/1102714178.