

20 Years of leanCoP – An Overview

– Talk Abstract –

Jens Otten

Department of Informatics, University of Oslo, Norway

Abstract

The talk gives a comprehensive overview of the automated theorem prover leanCoP and all its variants for classical and non-classical first-order logics. It includes historical details describing how seven lines of Prolog code turned into some of the most popular and efficient connection provers for classical and non-classical logics. The talk also provides an overview of the non-clausal version nanoCoP and on other implementations inspired by leanCoP, in particular those integrating Machine Learning.

Keywords

leanCoP, connection calculus, logic, automated reasoning, non-classical logics

leanCoP – How it Started

20 years ago, the paper “leanCoP: Lean Connection-based Theorem Proving” [1] started the development of a series of compact connection provers for classical and several non-classical logics. The very first version of leanCoP was written a few years earlier, when the author was asked to take on a lecture of Wolfgang Bibel’s course “Inferenzmethoden” at TU Darmstadt, because he was out of town. This prover was intended to show the students a compact Prolog implementation of the (classical) *connection calculus* [2, 3] which was the topic of the lesson at that time. Slightly simplifying that code and adding the *positive start clause technique* resulted in leanCoP 1.0, whose compact version of the Prolog code is shown in the abstract of the 2003 leanCoP article. With a size of 333 bytes, the core code was smaller than that of the first popular *lean* prover leanTAP [4], whose compact Prolog code had a size of 360 bytes.

Even though leanCoP 1.0 already outperformed the famous Otter theorem prover [5] on a small number of problems, it was not until the year 2006 when the development of leanCoP got a huge boost. The good preliminary results on the problems of the *MPTP challenge* led to the integration of further optimizations, e.g. *regularity*, *lemmata*, *strategy scheduling*, *definitional clausal form*, and a more efficient representation of the input clauses. But most importantly, it included *restricted backtracking* [6], a novel and simple, but powerful strategy to significantly reduce the amount of backtracking during the proof search. At CASC in 2007, leanCoP 2.0 [7, 6] had its debut and proved more problems than four other participating systems. It proved four problems that were not solved by the Vampire prover [8] and was awarded *Best Newcomer* [9]. The compact version of the leanCoP 2.0 core prover is still only 555 bytes in size.

AReCCa 2023: Automated Reasoning with Connection Calculi, 18 September 2023, Prague, Czech Republic


✉ jeotten@ifi.uio.no (J. Otten)

🌐 <http://jens-otten.de/> (J. Otten)

🆔 0000-0002-4331-8698 (J. Otten)



© 2023 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

leanCoP 2.1 returns a readable connection proof, uses a shell script with an improved strategy scheduling, and directly supports the TPTP input syntax with equality. At CASC in 2008 and 2009, leanCoP is among the top three (original) provers that output a proof in the core FOF division. At CASC in 2009, leanCoP-SiNE, which integrates the SiNE preprocessor, ends up in third place in the proof class of the SUMO reasoning prize. At CASC in 2010, leanCoP- Ω , which integrates the arithmetic Omega test, wins the first (linear integer) arithmetic TFA division.

ileanCoP and MleanCoP – Dealing with Non-classical Logics

ileanCoP 1.0 [10] was the first version of the connection prover for intuitionistic first-order logic. It is based on leanCoP 1.0 extended by prefixes and a prefix unification procedure [11, 12]. ileanCoP 1.2 [7] integrates all the techniques and strategies of leanCoP 2.0. At CASC in 2007, ileanCoP proved two problems intuitionistically valid that the Vampire prover could not prove classically valid, even though reasoning in intuitionistic logic is significantly harder.

MleanCoP 1.2 [13] and MleanCoP 1.3 [14] are connection provers for the modal first-order logics D, T, S4 and S5 with constant, cumulative and varying domains. They are based on leanCoP 2.1 extended by prefixes and prefix unification procedures that determine the specific modal logic. Up to the release of nanoCoP-M 2.0 these were the fastest provers for modal logics.

nanoCoP – Non-clausal Reasoning

nanoCoP [15, 16] is a series of compact Prolog implementations of the non-clausal connection calculus. The *non-clausal* connection calculus for classical [17, 18] and non-classical logics [19] generalizes the *clausal* connection calculus. It uses the original input formula directly, without translating it into any clausal form. Instead, the structure of the input formula is preserved throughout the proof search. nanoCoP combines the advantages of more natural sequent and tableau provers with the systematic and goal-oriented proof search of connection provers.

The nanoCoP 2.0 provers [20] integrate most of the optimization techniques of leanCoP 2.1 and can provide detailed non-clausal connection proofs. nanoCoP-i and nanoCoP-M are now some of the fastest provers for intuitionistic and modal first-order logic, respectively.

Other CoPs – Re-Implementations and Machine Learning

Re-implementations and extensions of leanCoP include lolliCoP (implemented in the linear logic programming language Lolli) [21], fCoP (implemented in OCaml) [22], “C-leanCoP” (implemented in C) [23], RACCOON/leanCoR (for the description logic ALC) [24], fleanCoP/fnanoCoP (implemented in OCaml) [25], SATCoP (integrating a SAT solver) [26], meanCoP (implemented in Rust) [27], Connect++ (in C++) [28], and pyCoP/ipyCoP/mpyCoP (in Python) [29].

At TABLEAUX 2011, the MaLeCoP [30] prover was presented, one of the first implementations that integrated Machine Learning (ML) into a theorem prover. It was the starting point for a whole series of ML provers based on or inspired by leanCoP. Among them are FEMaLeCoP (an optimized MaLeCoP) [31], rlCoP (reinforcement learning using Monte Carlo search) [32], plCoP (adds Monte Carlo tree search to leanCoP) [33], graphCoP (uses graph neural network models) [34], monteCoP (using Monte Carlo search) [25], lazyCoP (implements lazy paramodulation using deep neural networks) [35], and FLoP (geared towards finding longer proofs) [36].

References

- [1] J. Otten, W. Bibel, leanCoP: lean connection-based theorem proving, *Journal of Symbolic Computation* 36 (2003) 139–161.
- [2] W. Bibel, Matings in matrices, *Commun. ACM* 26 (1983) 844–852.
- [3] W. Bibel, *Automated Theorem Proving*, Artificial intelligence, F. Vieweg und Sohn, 1987.
- [4] B. Beckert, J. Posegga, leanTAP: Lean tableau-based deduction, *Journal of Automated Reasoning* 15 (1995) 339–358.
- [5] W. McCune, L. Wos, Otter – the CADE-13 competition incarnations, *Journal of Automated Reasoning* 18 (1997) 211–220.
- [6] J. Otten, Restricting backtracking in connection calculi, *AI Commun.* 23 (2010) 159–182.
- [7] J. Otten, leanCoP 2.0 and ileanCoP 1.2: High performance lean theorem proving in classical and intuitionistic logic, in: A. Armando, P. Baumgartner, G. Dowek (Eds.), *IJCAR 2008*, volume 5195 of *LNAI*, Springer, 2008, pp. 283–291.
- [8] L. Kovacs, A. Voronkov, First-Order Theorem Proving and Vampire, in: N. Sharygina, H. Veith (Eds.), *Proceedings of the 25th International Conference on Computer Aided Verification*, number 8044 in *LNAI*, Springer-Verlag, 2013, pp. 1–35.
- [9] G. Sutcliffe, The CADE-21 automated theorem proving system competition, *AI Commun.* 21 (2008) 71–81.
- [10] J. Otten, Clausal connection-based theorem proving in intuitionistic first-order logic, in: B. Beckert (Ed.), *TABLEAUX 2005*, volume 3702 of *LNAI*, Springer, 2005, pp. 245–261.
- [11] L. A. Wallen, *Automated Deduction in Nonclassical Logics*, MIT Press, Cambridge, 1990.
- [12] J. Otten, W. Bibel, Advances in connection-based automated theorem proving, in: M. Hinchey, J. P. Bowen, E.-R. Olderog (Eds.), *Provably Correct Systems, NASA Monographs in Systems and Software Engineering*, Springer, Cham, 2017, pp. 211–241.
- [13] J. Otten, Implementing connection calculi for first-order modal logics, in: K. Korovin, S. Schulz, E. Ternovska (Eds.), *IWIL 2012*, volume 22 of *EPiC Series in Computing*, EasyChair, 2012, pp. 18–32.
- [14] J. Otten, MleanCoP: A connection prover for first-order modal logic, in: S. Demri, D. Kapur, C. Weidenbach (Eds.), *IJCAR 2014*, volume 8562 of *LNAI*, Springer, 2014, pp. 269–276.
- [15] J. Otten, nanoCoP: A non-clausal connection prover, in: N. Olivetti, A. Tiwari (Eds.), *IJCAR 2016*, volume 9706 of *LNAI*, Springer, Heidelberg, 2016, pp. 302–312.
- [16] J. Otten, nanoCoP: Natural non-clausal theorem proving, in: C. Sierra (Ed.), *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17, Sister Conference Best Paper Track*, IJCAI, 2017, pp. 4924–4928.
- [17] J. Otten, A non-clausal connection calculus, in: K. Brunnler, G. Metcalfe (Eds.), *TABLEAUX 2011*, volume 6793 of *LNAI*, Springer, Heidelberg, 2011, pp. 226–241.
- [18] W. Bibel, J. Otten, From Schütte’s formal systems to modern automated deduction, in: R. Kahle, M. Rathjen (Eds.), *The Legacy of Kurt Schütte*, Springer, Cham, 2020, pp. 217–251.
- [19] J. Otten, Non-clausal connection calculi for non-classical logics, in: R. Schmidt, C. Nalon (Eds.), *TABLEAUX 2017*, volume 10501 of *LNAI*, Springer, Cham, 2017, pp. 209–227.
- [20] J. Otten, The nanoCoP 2.0 connection provers for classical, intuitionistic and modal logics, in: A. Das, S. Negri (Eds.), *TABLEAUX 2021*, volume 12842 of *LNAI*, Springer, 2021, pp. 236–249.

- [21] J. S. Hodas, N. Tamura, lolliCoP - A linear logic implementation of a lean connection-method theorem prover for first-order classical logic, in: R. Goré, A. Leitsch, T. Nipkow (Eds.), IJCAR 2001, volume 2083 of *LNCS*, Springer, 2001, pp. 670–684.
- [22] C. Kaliszyk, J. Urban, J. Vyskočil, Certified Connection Tableaux Proofs for HOL Light and TPTP, in: CPP 2015, CPP '15, ACM, New York, NY, USA, 2015, pp. 59–66.
- [23] C. Kaliszyk, Efficient low-level connection tableaux, in: H. de Nivelle (Ed.), TALBEAUX 2015, volume 9323 of *LNCS*, Springer, 2015, pp. 102–111.
- [24] D. M. Filho, F. Freitas, J. Otten, Raccoon: A connection reasoner for the description logic alc, in: T. Eiter, D. Sands (Eds.), LPAR-21, volume 46 of *EPIc Series in Computing*, EasyChair, 2017, pp. 200–211.
- [25] M. Färber, C. Kaliszyk, J. Urban, Machine learning guidance for connection tableaux, *J. Autom. Reason.* 65 (2021) 287–320.
- [26] M. Rawson, G. Reger, Eliminating models during model elimination, in: A. Das, S. Negri (Eds.), *Automated Reasoning with Analytic Tableaux and Related Methods*, volume 12842 of *LNCS*, Springer, 2021, pp. 250–265.
- [27] M. Färber, Connection Provers in Rust, 2022. URL: <https://github.com/01mf02/cop-rs>.
- [28] S. B. Holden, Connect++: a fast, flexible and modifiable connection prover to support machine learning, in: J. Otten, W. Bibel (Eds.), *Proceedings of the Workshop on Automated Reasoning with Connection Calculi (AReCCa)*, 2023.
- [29] F. Rømming, J. Otten, S. B. Holden, Connections: Markov decision processes for classical, intuitionistic, and modal connection calculi, in: J. Otten, W. Bibel (Eds.), *Proceedings of the Workshop on Automated Reasoning with Connection Calculi (AReCCa)*, 2023.
- [30] J. Urban, J. Vyskočil, P. Štěpánek, MaLeCoP Machine Learning Connection Prover, in: K. Brunnler, G. Metcalfe (Eds.), *Automated Reasoning with Analytic Tableaux and Related Methods*, *Lecture Notes in Computer Science*, Springer, Berlin, Heidelberg, 2011, pp. 263–277.
- [31] C. Kaliszyk, J. Urban, FEMaLeCoP: Fairly efficient machine learning connection prover, in: M. Davis, A. Fehnker, A. McIver, A. Voronkov (Eds.), LPAR-20, volume 9450 of *LNAI*, Springer, Heidelberg, 2015, pp. 88–96.
- [32] C. Kaliszyk, J. Urban, H. Michalewski, M. Olšák, Reinforcement Learning of Theorem Proving, in: *Advances in Neural Information Processing Systems*, volume 31, Curran Associates, Inc., 2018.
- [33] Z. Zombori, J. Urban, C. E. Brown, Prolog technology reinforcement learning prover, in: N. Peltier, V. Sofronie-Stokkermans (Eds.), IJCAR 2020, volume 12167 of *LNAI*, Springer, Cham, 2020, pp. 489–507.
- [34] M. Olšák, C. Kaliszyk, J. Urban, Property invariant embedding for automated reasoning, in: G. D. Giacomo, et al. (Eds.), ECAI 2020, volume 325 of *Frontiers in Artificial Intelligence and Applications*, IOS Press, Amsterdam, 2020, pp. 1395–1402.
- [35] M. Rawson, G. Reger, lazyCoP: Lazy Paramodulation Meets Neurally Guided Search, in: A. Das, S. Negri (Eds.), *Automated Reasoning with Analytic Tableaux and Related Methods*, *Lecture Notes in Computer Science*, Springer, Cham, 2021, pp. 187–199.
- [36] Z. Zombori, A. Csiszárík, H. Michalewski, C. Kaliszyk, J. Urban, Towards Finding Longer Proofs, in: A. Das, S. Negri (Eds.), *Automated Reasoning with Analytic Tableaux and Related Methods*, *Lecture Notes in Computer Science*, Springer, Cham, 2021, pp. 167–186.