Implementation and Evaluation of Contextual Natural Deduction for Minimal Logic

Bruno Woltzenlogel Paleo

Vienna University of Technology
Australian National University
bruno.wp@gmail.com

Abstract. The *contextual* natural deduction calculus ($\mathbf{ND^c}$) extends the usual natural deduction calculus (**ND**) by allowing the implication introduction and elimination rules to operate on formulas that occur inside contexts. It has been shown that, asymptotically in the best case, $\mathbf{ND^c}$-proofs can be quadratically smaller than the smallest **ND**-proofs of the same theorems. In this paper we describe the first implementation of a theorem prover for minimal logic based on $\mathbf{ND^c}$. Furthermore, we empirically compare it to an equally simple **ND** theorem prover on thousands of randomly generated conjectures.

1 Introduction

Natural deduction was introduced by Gentzen in [13] and one of its distinguishing features is that the meaning of a logical connective is determined by elimination and introduction rules, and not by axioms. As a result, formal natural deduction proofs are considered to be similar in structure to their informal counterparts and hence more *natural*. This subjective claim is corroborated by the observation that widely used proof assistants[1] follow a natural deduction style.

However, as exemplified in [21], the inference rules of natural deduction style calculi can be inconvenient, lengthy and ultimately unnatural for formalizing reasoning steps that modify a deeply located subformula of a formula, such as: Tseitin's transformation, skolemization, double negation elimination, quantifier shifting, prenexification... Because these deep reasoning steps are commonly used by automated deduction tools during preprocessing of the theorem to be proved, the resulting proofs may contain deep inferences [10]. Therefore, automatically replaying (i.e. reproving) these proofs in proof assistants (e.g. when an automated deduction tool is integrated within a proof assistant [2]) can be inefficient in terms of proving time and size of the generated shallow proof.

These challenges motivated the invention (in [21]) of the *contextual natural deduction calculus* ($\mathbf{ND^c}$), which is a simple extension of the usual natural deduction calculus (here called **ND**) allowing introduction and elimination rules

[1] e.g. Isabelle (`www.cl.cam.ac.uk/research/hvg/Isabelle/`) and Coq (`http://coq.inria.fr`).

to operate on formulas occurring inside contexts. The goals in [21] were purely theoretical. It was shown that $\mathbf{ND^c}$ is sound and complete, that proofs can be normalized, and that some proofs can be quadratically smaller than the smallest proofs of the same theorem in the usual natural deduction calculus. In contrast, the main goal of the work reported in the present paper is to evaluate $\mathbf{ND^c}$ empirically. This is important, because asymptotic proof-complexity results can be misleading when the assymptotic behaviour they describe for particular worst cases or best cases is not observed in cases that occur most often in practice.

$\mathbf{ND^c}$ can be regarded not only from a theorem proving perspective but also from a *proof compression* point of view: a given $\mathbf{ND}$-proof ψ could be compressed by transforming it to a smaller $\mathbf{ND^c}$-proof. Since every $\mathbf{ND}$-proof is also an $\mathbf{ND^c}$-proof, a straightforward proof compression algorithm could simply try to reprove ψ's theorem using an $\mathbf{ND^c}$ theorem prover.

The implementation of prototypical theorem provers based on $\mathbf{ND}$ and $\mathbf{ND^c}$ within the `Skeptik` framework (`github.com/Paradoxika/Skeptik`) is discussed in Section 3. These provers are restricted to minimal logic (intuitionistic logic having only the implication connective). Although it would be straightforward to extend the contextual techniques to inference rules for other connectives as well, the restriction to minimal logic implies less implementation effort and is sufficient to estimate how promising the idea of contextual natural deduction might be in practice. An experimental infra-structure, including a random formula generator, also had to be implemented, as briefly described in Section 4. The experimental results are shown and analyzed in Section 5.

Related work: due to the increasing maturity of automated deduction tools, there has been a lot of recent work on proof production [11] and on the development of algorithms for simplifying the generated proofs in a post-processing phase. These methods have focused mostly on propositional resolution proofs output by SAT- and SMT-solvers so far [1, 12, 4, 23, 9, 3], but generalizations to first-order resolution have been proposed as well [14]. There are also algorithms aimed at compressing and structuring sequent calculus proofs by eliminating or introducing cuts [26, 17, 22, 25] or by extracting Herbrand sequents from proofs [18–20]. [21] is probably the first work considering, from a theoretical perspective, the compressibility of natural deduction proofs, and this paper reports the first realization of this idea in practice. Contextual inferences have a lot in common with the related idea of *deep inference*, which has been intensively investigated in the last decade, especially for classical logic (e.g. [5, 8, 7, 16]) but also for intuitionistic logic [24, 6, 15]. Despite the technical differences, deep inference calculi were an inspiration for the development of contextual natural deduction.

2 Contextual Natural Deduction

In this paper a *derivation* is a tree of inferences (instances of the inference rules), operating on sequents of the form $\Gamma \vdash t : T$, where Γ is a (possibly empty) set of named *hypotheses* $h_1 : H_1, \ldots, h_n : H_n$, t is a (contextual) lambda term (whose free variables are among the names in Γ and whose bound variables are assumed

to have unique names) and T is a minimal logic formula (or equivalently, by the Curry-Howard isomorphism, the type of t). A derivation ψ is a *proof* of a theorem T if and only if its leaves are axiom inferences and it ends in $\vdash t : T$, for some term t. Figure 1 shows the rules of a natural deduction calculus for minimal logic (here called **ND**).

$$\frac{}{\Gamma, a : A \vdash a : A}$$

$$\frac{\Gamma, a : A \vdash b : B}{\Gamma \vdash \lambda a^A.b : A \to B} \to_I$$

$$\frac{\Gamma \vdash f : A \to B \qquad \Gamma \vdash a : A}{\Gamma \vdash (f\ a) : B} \to_E$$

Fig. 1: The natural deduction calculus **ND**

Figure 2 shows the inference rules of the *contextual natural deduction calculus* $\mathbf{ND^c}$ along with a corresponding extension of the lambda calculus. $\mathbf{ND^c}$ extends **ND** by allowing the inference rules to operate on subformulas located deeply inside the premises. The notation $\mathcal{C}_\pi[F]$ indicates a formula that has the subformula F in position π. $\mathcal{C}_\pi[\_]$ is called the *context* of F in the formula $\mathcal{C}_\pi[F]$. A *position* π is encoded as a binary string indicating the path from the root of $\mathcal{C}_\pi[F]$ to F in the tree structure of $\mathcal{C}_\pi[F]$; thus, a subformula at position π of a formula P, denoted $\mathrm{At}_\pi(P)$, can be retrieved by traversing the formula according to the following inductive definition:

$$\mathrm{At}_\epsilon(A) = A \qquad \mathrm{At}_{0\pi}(A \to B) = \mathrm{At}_\pi(B) \qquad \mathrm{At}_{1\pi}(A \to B) = \mathrm{At}_\pi(A)$$

A position is said to be *positive* (*negative*) if and only if it contains an even (odd) number of digits 1. In other words, in the tree structure of a formula, a node and its left (right) child always occupy positions with opposite (same) polarities, and the root position is positive. Moreover, a position is *strongly positive* if and only if it does not contain any digit 1. $\mathbf{ND^c}$ has two implication elimination rules (i.e. $\to_E^{\leftarrow}$ and $\to_E^{\rightarrow}$) because the contexts can be combined in two different ways, which are indicated by the superscript left and right harpoons. The natural deduction calculus **ND** can be considered a restriction of $\mathbf{ND^c}$ enforcing empty contexts. Examples of $\mathbf{ND^c}$-proofs are available in [21].

3 Implementation

Implementation was done in `Scala`, leveraging and extending the `Skeptik` proof compression library. Although `Skeptik`'s original focus was on propositional resolution proofs, the addition of data structures for natural deduction was easy

Note: π, π_1 and π_2 must be positive positions.

$$\frac{}{\Gamma, a : A \vdash a : A}$$

$$\frac{\Gamma, a : A \vdash b : \mathcal{C}_\pi[B]}{\Gamma \vdash \lambda_\pi a^A.b : \mathcal{C}_\pi[A \to B]} \to_I (\pi)$$

Contextual Soundness Condition:
a is allowed to occur in b only if π is strongly positive.

$$\frac{\Gamma \vdash f : \mathcal{C}^1_{\pi_1}[A \to B] \qquad \Gamma \vdash a : \mathcal{C}^2_{\pi_2}[A]}{\Gamma \vdash (f\ a)^{\rightarrow}_{(\pi_1;\pi_2)} : \mathcal{C}^1_{\pi_1}[\mathcal{C}^2_{\pi_2}[B]]} \to^{\rightarrow}_E (\pi_1; \pi_2)$$

$$\frac{\Gamma \vdash f : \mathcal{C}^1_{\pi_1}[A \to B] \qquad \Gamma \vdash a : \mathcal{C}^2_{\pi_2}[A]}{\Gamma \vdash (f\ a)^{\leftarrow}_{(\pi_1;\pi_2)} : \mathcal{C}^2_{\pi_2}[\mathcal{C}^1_{\pi_1}[B]]} \to^{\leftarrow}_E (\pi_1; \pi_2)$$

Fig. 2: The contextual natural deduction calculus $\mathbf{ND^c}$

and required no refactoring, because `Skeptik` has always taken advantage of `Scala`'s object-orientation features to be agnostic with respect to proof systems.

In `Skeptik`, inference rules are classes. In order to increase the confidence on the correctness of inference rules, each rule class includes correctness checking code and is kept as small as possible. The 3 classes for the **ND** rules are only 13 lines long. The single class `ImpElimC` for the two contextual implication elimination rules has 17 lines and the soundness condition for the `ImpIntroC` rule is a 10-line long trait. To the extent that these few lines of code are trusted, any proof constructed using these inference rules is correct. Any code that is not essential to the rule is written not in the class but in its companion object.

The class `SimpleProver` implements a theorem prover that is generic in the sense that it takes arbitrary (companion objects of) inferences rules and then performs bottom-up proof search using the given inference rules. For each open goal, the prover tries all inference rules in a bottom-up manner in parallel, generating all possible subgoals. Then it recursively tries to prove the subgoals in parallel. When returning from the recursion, the prover chooses the smallest subproof among all alternative subproofs returned by the recursive calls. This exhaustive search strategy is appropriate in the context of proof compression, where the goal is to find small proofs not necessarily as fast as possible. The depth of the recursion is bounded by the maximum proof height specified as a parameter of `SimpleProver`.

The companion objects of the inference rules implement a standard interface, which provides methods that generate subgoals as required by the prover and reconstruct the proof when the subgoals are proved. In the case of **ND**'s implication elimination rule, when generating subgoals for a goal of the form $\Gamma \vdash B$, it is necessary to guess a formula A in order to generate the subgoals $\Gamma_1 \vdash A \to B$

and $\Gamma_2 \vdash A$. Exhaustively guessing all possible formulas would be inefficient. Instead, the rule searches for hypotheses of the form $D_1 \to (\ldots \to (D_k \to B)\ldots)$ in Γ and then generates the subgoals $\Gamma \vdash (D_k \to B)$ and $\Gamma \vdash D_k$. The search Although proof search is still complete under this restriction, the proofs it finds are always normalized. Consequently, a proof found by this procedure is not necessarily the smallest possible proof, because sometimes non-normal proofs can be smaller.

Example 1. Let $t : T$ be a closed simply typed lambda term corresponding to a proof ψ_t of T with ℓ inferences. Then the term $(\lambda x^T.\lambda c^{T\to T\to Z}.((c\ x)\ x))\ t$ corresponds to a proof ψ of $(T \to T \to Z) \to Z$ with $(\ell + 8)$ inferences having ψ_t as a subproof. The proof search procedure described above, however, would never be able to construct ψ. To do so, it would have to start by applying implication elimination. But since there are no hypotheses, no application of implication elimination is possible according to the proof search procedure described above. Instead, the procedure will have to start with an application of implication introduction and will eventually construct the proof ψ' corresponding to the normalized term $\lambda c^{T\to T\to Z}.((ct)t)$. Because ψ' has $(2\ell+4)$ inferences, ψ is smaller than ψ' and hence ψ' is not the smallest possible proof, for large enough ℓ.

In the case of $\mathbf{ND^c}$ rules, the generation of subgoals is complicated further by the need to take positions into account. For a goal of the form $\Gamma \vdash F$, the contextual implication elimination rule first searches for all positive positions π_1 and π_2 such that $F = \mathcal{C}_{\pi_1}[\mathcal{C}_{\pi_2}[B]]$ for some B. Then it searches for a hypothesis of the form $\mathcal{C}_{\pi_1}[D_1 \to (\ldots \to (D_k \to B)\ldots)]$ (or $\mathcal{C}_{\pi_2}[D_1 \to (\ldots \to (D_k \to B)\ldots)]$) and, if it succeeds, it generates the subgoals $\Gamma \vdash \mathcal{C}_{\pi_1}[(D_k \to B)]$ and $\Gamma \vdash \mathcal{C}_{\pi_2}[D_k]$ (or, respectively, $\Gamma \vdash \mathcal{C}_{\pi_2}[(D_k \to B)]$ and $\Gamma \vdash \mathcal{C}_{\pi_1}[D_k]$).

Example 2. Consider the goal $h : C \to (D \to B) \vdash (D \to (C \to B)$. There are the following possibilities of values for the pair (π_1, π_2): (ϵ, ϵ), $(\epsilon, 0)$, $(\epsilon, 00)$, $(0, \epsilon)$, $(0, 0)$, $(00, \epsilon)$. Consider the case when $\pi_1 = 0$ and $\pi_2 = 0$. In this case, $\mathcal{C}_{\pi_1}[] = D \to []$ and $\mathcal{C}_{\pi_2}[] = C \to []$. h is of the form $\mathcal{C}_{\pi_2}[D \to B]$. Therefore, for this case, the subgoals $h : C \to (D \to B) \vdash \mathcal{C}_{\pi_2}[D \to B]$ (i.e. $h : C \to (D \to B) \vdash C \to (D \to B)$) and $h : C \to (D \to B) \vdash \mathcal{C}_{\pi_1}[D]$ (i.e. $h : C \to (D \to B) \vdash D \to D$) are generated. The proof search procedure then continues trying to prove these subgoals. After the subgoals are proved, a proof of the original goal can be obtained with an application of contextual implication elimination. The other cases for (π_1, π_2) do not result in contexts that match the hypothesis h; therefore, no subgoals are generated for those other cases.

4 Experimental Setup

In order to evaluate the provers, a random formula generator was implemented. It takes a desired size s and a desired number of distinct atomic formulas q as input. Then it generates a list of length s containing q distinct atomic formulas. The

list is grown recursively, and at each iteration, every atomic formula is equally likely to be selected. At this stage, care is taken to avoid generating formulas that are isomorphic modulo variable renaming (e.g. $A \to B$, $B \to A$, $A \to C$, ...; only $B \to A$ can be generated). Subsequently, the generator transforms this list into a minimal logic formula by recursively introducing implications at random positions in the list. The positions are equally likely to be selected (i.e. $A \to (A \to A)$ and $(A \to A) \to A$ are equally probable to be generated).

The experiments varied the value of s from 3 to 15 and the value of q from 1 to $s-1$. For each pair of values (s, q), 1000 formulas were generated, except for small values of s and q, for which there are less than 1000 distinct formulas. In total, 76755 formulas were generated.

For each generated formula f, the **ND** prover with a timeout of 30 seconds and a maximum proof height of 20. The $\mathbf{ND^c}$ prover, on the other hand, had a timeout of 300 seconds and a maximum proof height of $h+1$, where h is the height of proof of f found by the **ND** prover. The larger timeout was chosen because $\mathbf{ND^c}$'s contextual rules clearly result in a larger search space, with more subgoals to try. With the larger timeout, it is possible to measure the impact of the larger search space in the proof search time.

5 Results of the Experiments

30127 formulas were proved by the **ND** prover. 46628 formulas were shown to be countersatisfiable by the **ND** prover, because it terminated before the timeout exhausting the proof search space without finding a proof. There was no case of timeout for the **ND** prover. There were 533 cases of timeout for the $\mathbf{ND^c}$ prover.

Among the 29594 formulas on which both provers were successful, 2557 (8.49%) had shorter proofs in $\mathbf{ND^c}$ than in **ND**. The total length of the $\mathbf{ND^c}$-proofs was 2.97% lesser than the total length of the proofs found by the **ND** prover on all 29594 formulas. The total length of the $\mathbf{ND^c}$-proofs was 27.8% lesser than the total length of the **ND**-proofs on the 2557 proofs that admit shorter $\mathbf{ND^c}$ proofs.

Figure 8 (which takes into account all 76755 formulas) shows that the **ND** prover rarely took longer than 10 milliseconds on a formula. The $\mathbf{ND^c}$ prover was, as expected, significantly slower than the **ND** prover, because the search space for $\mathbf{ND^c}$ proofs is much larger. Nevertheless, the $\mathbf{ND^c}$ prover is surprinsingly faster in a few cases, probably due to the stricter upper-bound on proof height.

The 3D-charts in Figure 7 shed further light on what influences the proportion of formulas that admit shorter $\mathbf{ND^c}$-proofs and the total compression ratios (including all 29594 formulas). The proportion of formulas admitting shorter $\mathbf{ND^c}$-proofs (and their total compression ratios) is higher the lower the number of distinct atoms they contain and the larger they are. The depths[2] of the for-

[2] $\mathrm{depth}(A) = 1$, for an atomic A; $\mathrm{depth}(B \to C) = \max(\mathrm{depth}(B), \mathrm{depth}(C)) + 1$.

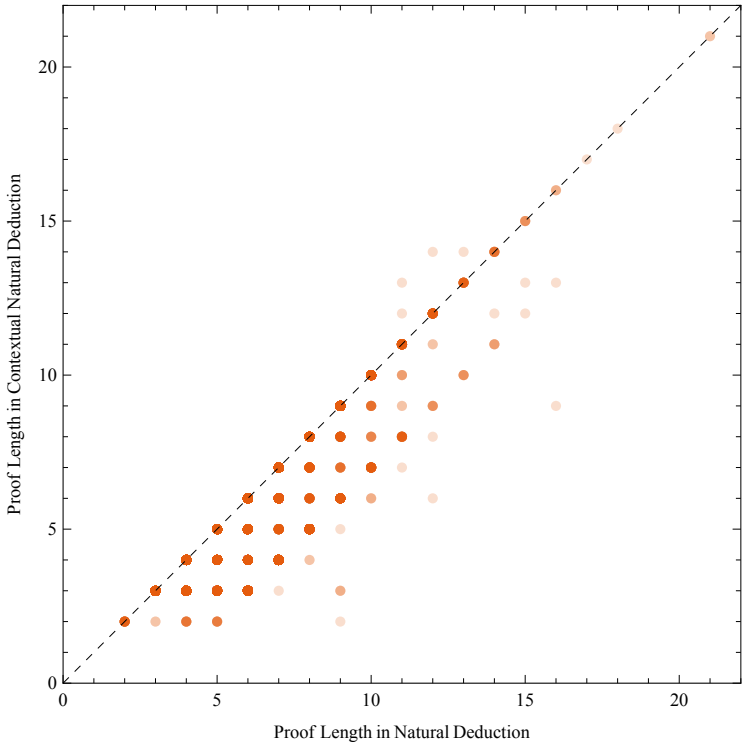

Fig. 3: Scatter Plot of Proof Lengths

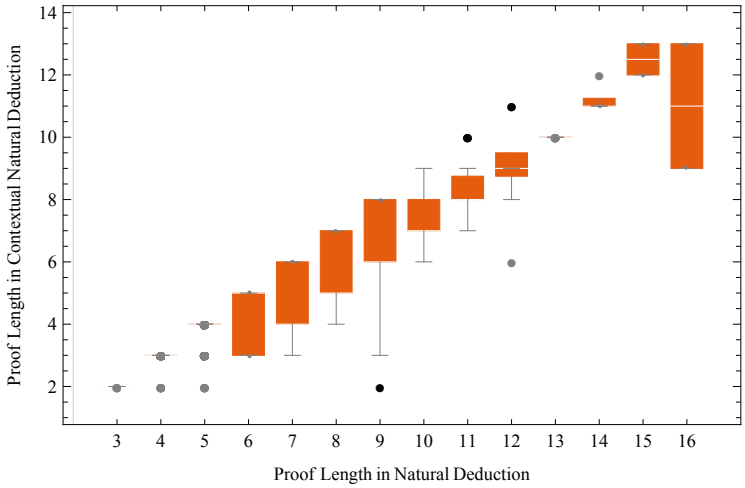

Fig. 4: Box-Whiskers Plot of Proof Lengths

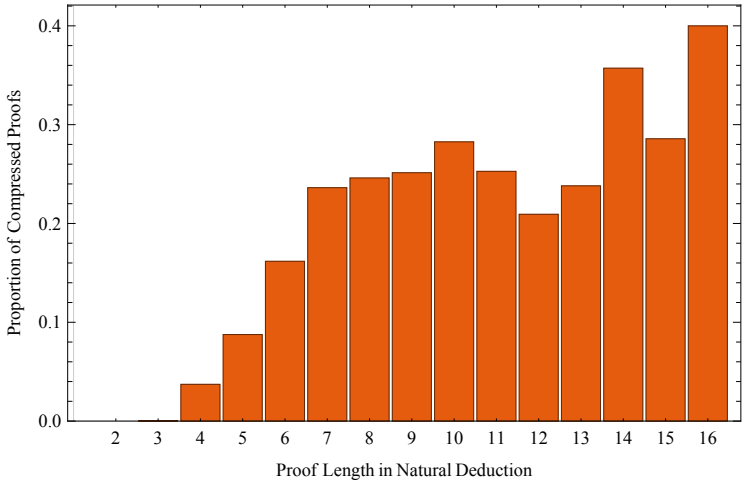

Fig. 5: Proportion of Compressed Proofs by Length

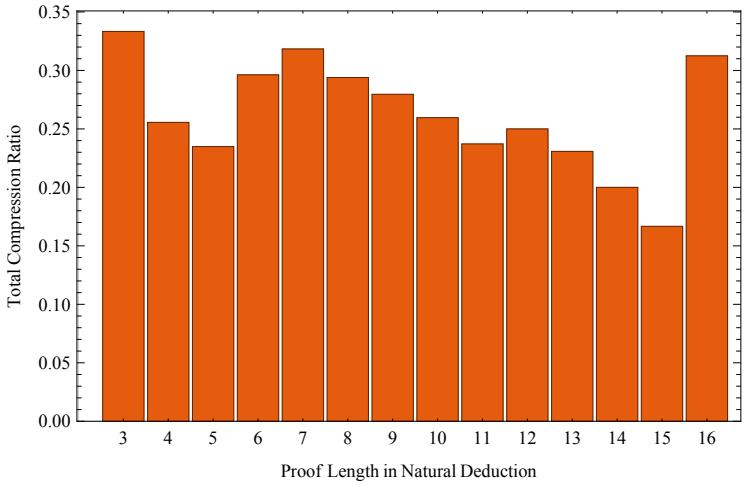

Fig. 6: Total Compression by Length

mulas also seem to play a role, with greater compression proportions and total compression ratios for intermediary depth values.

Figure 8 (which takes into account all 76755 formulas) shows that the **ND** prover rarely took longer than 10 milliseconds on a formula. Surprisingly, the $\mathbf{ND^c}$ prover was faster than the **ND** prover in the majority of the cases (46733 formulas; 60.88%), probably due to the stricter upper-bound on proof height. However, when the $\mathbf{ND^c}$ prover was slower, it tended to be significantly slower, as shown in the Figure.

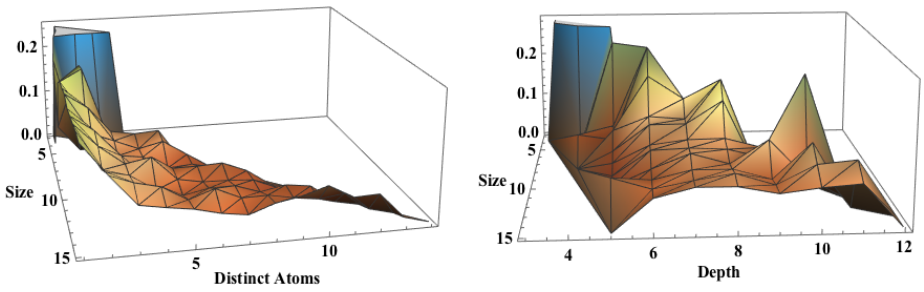

Fig. 7: Proportion of Compressed Proofs

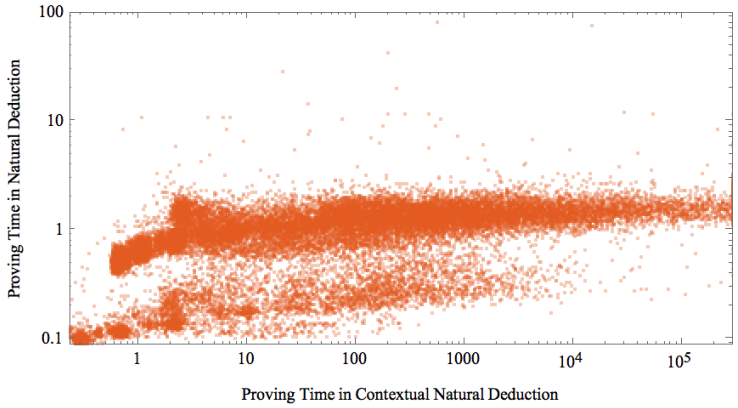

Fig. 8: Scatter Plot of Proving Time

6 Conclusions

The empirical investigation reported in this paper confirms the expectation that $\mathbf{ND^c}$ can provide shorter proofs in a significant number of cases. Because the benchmarks were randomly generated, it is still an open question whether contextual natural deduction will perform well on real-world benchmarks. Nevertheless, the good performance on randomly generated benchmarks complements and the previous theoretical result of asymptotic best-case quadratic compression shown in [21]. Together, the experimental evaluation and the theoretical improved proof complexity constitute strong evidence that contextualization is worth pursuing for the sake of obtaining shorter proofs in natural deduction style.

The price to pay for shorter proofs is currently a much longer proving time. In this paper, proving time was restricted by bounding the proof height during proof search. Other more sophisticated techniques to restrict proof search should be investigated in the future, in order to improve the efficiency of theorem proving in $\mathbf{ND^c}$.

To evaluate the benefit of contextual natural deduction on real-world benchmarks, it is firstly necessary to extend $\mathbf{ND^c}$ to the more expressive higher-order logics used by interactive proof assistants. Their libraries of formalized mathematics contain proofs of theorems for which shorter contextual natural deduction proofs could be possible.

Acknowledgements: This work was supported by an Stipendium of the Österreichische Akademie der Wissenschaften (APART).

References

1. Omer Bar-Ilan, Oded Fuhrmann, Shlomo Hoory, Ohad Shacham, and Ofer Strichman. Linear-time reductions of resolution proofs. In *Hardware and Software: Verification and Testing, 4th International Haifa Verification Conference, HVC 2008, Haifa, Israel, October 27-30, 2008. Proceedings*, pages 114–128, 2008.
2. Sascha Böhme and Tobias Nipkow. Sledgehammer: Judgement day. In Jürgen Giesl and Reiner Hähnle, editors, *IJCAR*, volume 6173 of *Lecture Notes in Computer Science*, pages 107–121. Springer, 2010.
3. Joseph Boudou, Andreas Fellner, and Bruno Woltzenlogel Paleo. Skeptik: A proof compression system. In *Automated Reasoning - 7th International Joint Conference, IJCAR 2014, Held as Part of the Vienna Summer of Logic, VSL 2014, Vienna, Austria, July 19-22, 2014. Proceedings*, pages 374–380, 2014.
4. Joseph Boudou and Bruno Woltzenlogel Paleo. Compression of propositional resolution proofs by lowering subproofs. In *Automated Reasoning with Analytic Tableaux and Related Methods - 22th International Conference, TABLEAUX 2013, Nancy, France, September 16-19, 2013. Proceedings*, pages 59–73, 2013.
5. Kai Brünnler. Atomic cut elimination for classical logic. In Matthias Baaz and Johann A. Makowsky, editors, *CSL*, volume 2803 of *Lecture Notes in Computer Science*, pages 86–97. Springer, 2003.
6. Kai Brünnler and Richard McKinley. An algorithmic interpretation of a deep inference system. In *15th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning*, volume 5330 of *LNCS*, pages 482–496. Springer, 2008.
7. Paola Bruscoli and Alessio Guglielmi. On the proof complexity of deep inference. *ACM Transactions on Computational Logic*, 10:1–34, 2009.
8. Paola Bruscoli, Alessio Guglielmi, Tom Gundersen, and Michel Parigot. A quasipolynomial cut-elimination procedure in deep inference via atomic flows and threshold formulae. In Edmund M. Clarke and Andrei Voronkov, editors, *LPAR (Dakar)*, volume 6355 of *Lecture Notes in Computer Science*, pages 136–153. Springer, 2010.
9. Scott Cotton. Two techniques for minimizing resolution proofs. In *Theory and Applications of Satisfiability Testing - SAT 2010, 13th International Conference, SAT 2010, Edinburgh, UK, July 11-14, 2010. Proceedings*, pages 306–312, 2010.
10. David Deharbe, Pascal Fontaine, and Bruno Woltzenlogel Paleo. Quantifier inference rules in the proof format of verit. In *1st International Workshop on Proof Exchange for Theorem Proving*, 2011.
11. David Delahaye and Bruno Woltzenlogel Paleo, editors. *All about Proofs, Proofs for All*, volume 55 of *Mathematical Logic and Foundations*. College Publications, London, UK, January 2015.

12. Pascal Fontaine, Stephan Merz, and Bruno Woltzenlogel Paleo. Compression of propositional resolution proofs via partial regularization. In *Automated Deduction - CADE-23 - 23rd International Conference on Automated Deduction, Wroclaw, Poland, July 31 - August 5, 2011. Proceedings*, pages 237–251, 2011.
13. G. Gentzen. Untersuchungen über das logische Schließen. *Mathematische Zeitschrift*, 39:176–210,405–431, 1934–1935.
14. Jan Gorzny and Bruno Woltzenlogel Paleo. Towards the compression of first-order resolution proofs by lowering unit clauses. In *Automated Deduction - CADE-25 - 25th International Conference on Automated Deduction, Berlin, Germany, August 1 - August 7, 2015. Proceedings*, 2015.
15. Nicolas Guenot. Nested proof search as reduction in the lambda-calculus. In Peter Schneider-Kamp and Michael Hanus, editors, *PPDP*, pages 183–194. ACM, 2011.
16. Alessio Guglielmi. A system of interaction and structure. *CoRR*, cs.LO/9910023, 1999.
17. Stefan Hetzl, Alexander Leitsch, and Daniel Weller. Towards algorithmic cut-introduction. In *LPAR*, 2012.
18. Stefan Hetzl, Alexander Leitsch, Daniel Weller, and Bruno Woltzenlogel Paleo. Herbrand sequent extraction. In *Intelligent Computer Mathematics, 9th International Conference, MKM 2008, Birmingham, UK, July 28 - August 1, 2008. Proceedings*, pages 462–477, 2008.
19. Bruno Woltzenlogel Paleo. *Herbrand Sequent Extraction.* M.sc. thesis, Technische Universität Dresden; Technische Universität Wien, Dresden, Germany; Wien, Austria, 07 2007.
20. Bruno Woltzenlogel Paleo. *Herbrand Sequent Extraction.* VDM-Verlag, Saarbrücken, Germany, 2008.
21. Bruno Woltzenlogel Paleo. Contextual natural deduction. In *Logical Foundations of Computer Science, International Symposium, LFCS 2013, San Diego, CA, USA, January 6-8, 2013. Proceedings*, pages 372–386, 2013.
22. Bruno Woltzenlogel Paleo. Reducing redundancy in cut-elimination by resolution. *Journal of Logic and Computation*, 2014.
23. Simone Fulvio Rollini, Roberto Bruttomesso, Natasha Sharygina, and Aliaksei Tsitovich. Resolution proof transformation for compression and interpolation. *Formal Methods in System Design*, 45(1):1–41, 2014.
24. Alwen Tiu. A local system for intuitionistic logic. In Miki Hermann and Andrei Voronkov, editors, *LPAR*, volume 4246 of *Lecture Notes in Computer Science*, pages 242–256. Springer, 2006.
25. Bruno Woltzenlogel Paleo. *A General Analysis of Cut-Elimination by CERes.* Ph.d. dissertation, Vienna University of Technology, 11 2009.
26. Bruno Woltzenlogel Paleo. Atomic cut introduction by resolution: Proof structuring and compression. In Edmund Clarke and Andrei Voronkov, editors, *16th International Conference on Logic for Programming Artificial Intelligence and Reasoning (LPAR)*, number 6355 in LNAI, pages 463 – 480. Springer, 2010.