Some Notes on Proofs with Alpha Graphs

Frithjof Dau

Technische Universität Dresden, Dresden, Germany
dau@math.tu-dresden.de

Abstract. It is well-known that Peirce's Alpha graphs correspond to propositional logic (PL). Nonetheless, Peirce's calculus for Alpha graphs differs to a large extent to the common calculi for PL. In this paper, some aspects of Peirce's calculus are exploited. First of all, it is shown that the erasure-rule of Peirce's calculus, which is the only rule which does not enjoy the finite choice property, is admissible. Then it is shown that this calculus is faster than the common cut-free calculi for propositional logic by providing formal derivations with polynomial lengths of Statman's formulas. Finally a natural generalization of Peirce's calculus (including the erasure-rule) is provided such that we can find proofs linear in the number of propositional variables used in the formular, depending on the number of propositional variables in the formula.

1 Introduction

At the dawn of modern logic, Peirce invented his system of Existential Graphs (EGs), starting in 1896 and working extensively on it until he passed away in 1914 (see for example [Pei35, Pei92, PS00]). Peirce's EGs are divided into three parts which built upon each other, namely Alpha, Beta, and Gamma. Alpha corresponds to propositional logic (PL), Beta correspond to first order logic, and Gamma, which was never completed, encompasses aspects of higher order logic, modal logic and other features. Although not mathematically formalized, his Alpha and Beta EGs are one of the very early elaborations of mathematical logic.[1] But at the end of the 19th century, symbolic notations had already had taken the vast precedence in the development of formal logic, and EGs did not succeed against symbolic logic.

Several authors investigated Peirce's EGs from different perspectives, some of them aiming to elaborate a (more or less) mathematical theory of them (see for example [Zem64, Rob73, Bur91, Shi02a, Dau06]). Some works focus particularly on Alpha graphs (like [Ham95, Shi02b]) or more particular on finding proofs within Alpha ([Liu05]). But there are only few people who try to implement Peirce's calculus for automated theorem proving (see [HK05, vH03]), and one has to say that in the automated theorem proving community, Peirce's calculus is not acknowledged at all. This paper aims to exploit some aspects of Peirce's calculus which may be helpful for automated theorem proving with this calculus.

Peirce's calculus for Alpha graphs differs to a large extent to the common calculi for propositional logic (PL). In usual calculi for PL, the transformation rules are defined along the inductive construction of formulas. That is, each transformation rule modifies formulas only on their top-level of their construction trees. In other words: We have *shallow* inferences. In contrast to that, Peirce's calculus allows to transform *arbitrary deep subformulas* of given formulas, i.e. to carry out *deep* inferences. To the best of my knowledge, there is only one proof-system which employs deep inferences as well, namely the *calculus of structures* of Gulielmi (see [Brü03]). But deep inference systems, particularly Peirce's rules for EGs, have some

[1] Although Peirce did not provide any mathematical definitions for EGs, a mathematical elaboration of EGs can be obtained from a closer scrutiny of his works. See [Dau06].)

interesting properties which are of interest for automated theorem proving, as it is argued by Gulielmi for his calculus of structures. Some of these properties of Peirce's rules for Alpha graphs are investigated in this paper.

The organization of the paper is as follows: In Sec. 2, the basic notations, including syntax, semantics, and the calculus for Alpha graphs are introduced. Due to space limitations, we will use the linear notation for Alpha graphs. In Sec. 3, some basic theorems for Alpha graphs are provided. In Sec. 4, it is shown how Alpha graphs can be converted to normalforms, and, in order to obtain an analytic calculus, it is proven that the erasure-rule of the calculus can be removed. In Sec. 5 it is proven that the calculus is faster than the common calculi for propositional logic by showing that Statman's formulas can be proven in polynomial time. In Sec. 6, a version of the calculus with generalized rules is introduced, and it is shown that with this calculus, the number of steps of a proof for a formula f depends linearly from the number of propositional variables which occur in f. Finally in Sec. 7, the paper concludes with a discussion of the results.

2 Basic Notations for Alpha Graphs

In this paper, we will use the *linear* notion for Peirce's Alpha graphs. More precisely: Alpha graphs are introduced as formulas of propositional logic, equipped with an equivalence relation which encompasses the syntactical properties of Alpha graphs, mainly the commutativity and associativity of the juxtaposition of graphs, which corresponds on the semantical side to the commutativity and associativity of conjunction.[2]

The formulas of propositional logic, thus Alpha graphs as well, are built over a set $\mathcal{P} := \{P_1, P_2, P_3, \ldots\}$ of propositional variables and a symbol $\top \notin \mathcal{P}$ for truth, and we use the logical junctors $\neg$ and $\wedge$. Now each P_i for $i \in \mathbb{N}$ and $\top$ are formulas, if f is a formula, then $\neg f$ is a formula, and if f_1, f_2 are formulas, then $(f_1 \wedge f_2)$ is a formula. We will omit brackets if it is convenient. As usual, the formulas P_i and $\neg P_i$ with $i \in \mathbb{N}$ are called LITERALS. We will use the letters A, B to denote propositional variables as well, and the letters f, g, h, k, l to denote formulas.

In Peirce's calculus for EGs, the transformation rules allow to modify arbitrary subgraphs in arbitrary contexts. This idea will be carried over to the symbolic notion of propositional logic. First of all, when we speak in this paper about subformulas, we mean subformula *occurrences*. For example, for the formula $P_1 \wedge P_1$, as P_1 appears twice in this formula, we will say that it has *two* subformulas P_1. Square brackets are used to denote contexts. For example, with $f[g]$ we denote a formula f with a subformula g. A subformula g of f is EVENLY ENCLOSED resp. is PLACED IN A POSITIVE CONTEXT if it is a subformula of an even number of subformulas $\neg h$ of f. Otherwise it is said to be ODDLY ENCLOSED resp. to be PLACED IN A NEGATIVE CONTEXT. This will be denoted by $f[g]^+$ resp. $f[g]^-$. This notation can be nested. For example, with $f[P_2 \wedge g[h]]$, it is expressed that g is a formula with a subformula h, and f is a formula with the subformula $P_2 \wedge g$ $(= P_2 \wedge g[h])$.

In Peirce's graphs, conjunction can only be expressed up to commutativity and associativity. Moreover, empty negations are allowed: For this reason, we had to add the symbol $\top$ to our alphabet. In the following, formulas are considered only

[2] A similar approach is common in mathematical logic as well. For example, sequents in a sequent calculus are usually defined as *multisets* of formulas, thus we already have on the syntactical side encompassed commutativity and associativity of conjunction. Similarly, sometimes formulas are considered only modulo an equivalence relation. The equivalence classes are called *structures*. See for example [Brü03]).

up to the following equivalence relation $\sim$:

$$\begin{array}{lrcll}
\textbf{Commutativity:} & (f \wedge g) & \sim & (g \wedge f) & \\
\textbf{Associativity:} & ((f \wedge g) \wedge h) & \sim & (f \wedge (g \wedge h)) & \\
\textbf{Truthelement:} & (f \wedge \top) & \sim & f & \\
\textbf{Congruence:} & f[g] & \sim & f[h] & \text{if } g \sim h
\end{array}$$

Each class of formulas corresponds to a Peircean Alpha graph, thus this definition of propositional logic can be understood as a formalization of Peirce's Alpha system.

Now we are prepared to introduce the calculus. It consists of the following six rules (where f, g, h, i denote arbitrary formulas).

$$\begin{array}{lrcl}
\textbf{Erasure:} & f[g \wedge h]^+ & \vdash & f[g]^+ \\
\textbf{Insertion:} & f[g]^- & \vdash & f[g \wedge h]^- \\
\textbf{Iteration:} & f[g \wedge h[i]] & \vdash & f[g \wedge h[g \wedge i]] \\
\textbf{Deiteration:} & f[g \wedge h[g \wedge i]] & \vdash & f[g \wedge h[i]] \\
\textbf{Double Cut i):} & f[\neg\neg g] & \vdash & f[g] \\
\textbf{Double Cut ii):} & f[g] & \vdash & f[\neg\neg g]
\end{array}$$

Let f, g be two graphs. Then g CAN BE DERIVED FROM f (which is written $f \vdash g$), if there is a finite sequence $(f_1, f_2, \ldots, f_n)$ with $f = f_1$ and $g = f_n$ such that each f_{i+1} is derived from f_i by applying one of the rules of the calculus. The sequence is called A PROOF or DERIVATION FOR $f \vdash g$ (OF LENGTH $n-1$). Two graphs f, g with $f \vdash g$ and $g \vdash f$ are said to be PROVABLY EQUIVALENT.

If F is a set of graphs, we write $F \vdash f$ if there are $f_1, \ldots, f_i \in F$ with $f_1 \wedge \ldots \wedge f_i \vdash f$. With $f \vdash^n g$ we mean that g can be derived from f in (at most) n steps. For $\top \vdash f$, we write more simply $\vdash f$ resp. $\vdash^n f$. This set of rules is (strongly) sound and complete, as it is shown in [Dau04]. We use the usual abbreviation, i.e., $f \vee g$ is a (mere syntactical) abbreviation for $\neg(\neg f \wedge \neg g)$, $f \rightarrow g$ abbreviates $\neg(f \wedge \neg g)$, and $f \leftrightarrow g$ abbreviates $(f \rightarrow g) \wedge (f \rightarrow g)$, that is $\neg(f \wedge \neg g) \wedge \neg(g \wedge \neg f)$.

The semantics are now defined in the usual way. A VALUATION or MODEL is a mapping $val : \mathcal{P} \cup \{\top\} \mapsto \{\mathtt{ff}, \mathtt{tt}\}$ with $val(\top) = \mathtt{tt}$. Let $val : \mathcal{P} \mapsto \{\mathtt{ff}, \mathtt{tt}\}$ be a valuation. We set $val \models P_i :\Leftrightarrow val(P_i) = \mathtt{tt}$, $val \models (f \wedge g) :\Leftrightarrow val(f) = \mathtt{tt} = val(g)$, and $val \models \neg f :\Leftrightarrow val(f) = \mathtt{ff}$. For $val \models f$, we say that f HOLDS IN val. If we have two formulas f, g such that $val \models g$ for each valuation val with $val \models f$, we write $f \models g$, and we say that f ENTAILS g. Finally, a formula f is called SATISFIABLE, iff there exists a valuation val with $val \models f$, it is called VALID or A TAUTOLOGY, iff $val \models f$ for each valuation val, and it is called CONTRADICTORY, iff $val \not\models f$ for each valuation val.

3 Some Simple Theorems

In [Pei35] Peirce provided 16 useful transformation rules for EGs which he derived from his calculus. These rules are logical metalemmata in the sense that they show some *schemata* for proofs with EGs, i.e., they are derived 'macro'-rules. In this section we provide the formal Alpha graph versions for two of these transformation rules. We start with a (weakened) version of the first transformation rule of Peirce.

Lemma 1 (Reversion Theorem). *Let f and g be two formulas. Then we have:*

$$f \vdash^n g \quad \Rightarrow \quad \neg g \vdash^n \neg f \quad \textit{and} \quad \neg g \vdash^n \neg f \quad \Rightarrow \quad f \vdash^{n+2} g$$

Proof: Let $(h_1, h_2, \ldots, h_n)$ with $h_1 = f$ and $g = h_n$ be a proof for $f \vdash g$. Then, due to the symmetry of the calculus, $(\neg h_n, \neg h_{n-1}, \ldots, \neg h_1)$ is a proof for $\neg g \vdash \neg f$.

Analogously, from $\neg g \vdash^n \neg f$ we conclude $\neg\neg f \vdash^n \neg\neg g$. An additional application of the double cut rule at the beginning and the end of the proof yields $f \vdash^{n+2} g$. □

Let g be a subformula of f. With $f[h/g]$ we denote the graph where g is substituted by h. If g is a subgraph in a positive context, we will more explicitely write $f[h/g]^+$, and analogously $f[h/g]^-$ for negative contexts.

All rules in the calculus which are applied in a context only depend on whether the context is positive or negative. In particular if a proof for $f \vdash g$ is given, this proof can be carried out in arbitrary positive contexts. Together with the previous lemma, this yields the following lemma. It can also be found in [Sow97] (from where we adopted the name of the theorem).

Lemma 2 (Cut-And-Paste-Theorem I). *Let $g \vdash^n h$ for formulas g, h. Then:*

$$f \vdash^n f[h/g]^+ \quad \text{and} \quad f \vdash^{n+2} f[g/h]^-$$

Particularly, tautologies can be inserted into arbitrary contexts of arbitrary formulas.

With $f[\![h/g]\!]$ we denote the formula we obtain from f by substituting *every* subformula (i.e., every occurence of the subformula) g by h.

Lemma 3 (Cut-And-Paste-Theorem II). *Let g be a formula with $\vdash^n g$, let P_i be a propositional variable and f be another formula. Then we have $\vdash^n f[\![g/P_i]\!]$.*

Proof: Let $(h_0, h_2, \ldots, h_n)$ with $h_n = f$ be a proof for f. Then it is easy to see that $(h_0[\![g/P_i]\!], h_2[\![g/P_i]\!], \ldots, h_n[\![g/P_i]\!])$ is a proof for $f[\![g/P_i]\!]$. □

The next two lemmata are two other metalemmata which ease the handling of proofs (they will be needed in Sec. 6). To ease the readability of the proofs, we have sometimes underlined the subformulas which will be used in the next step (for example, by deiterating them).

Lemma 4 (Proof by Distinction). *Let f, g be formulas. Then we have*

$$(g \to f) \wedge (\neg g \to f) \ \vdash^7 f$$

Proof :

$$\begin{aligned}
(g \to f) \wedge (\neg g \to f) &= \underline{\neg(g \wedge \neg f)} \wedge \neg(\neg g \wedge \neg f) \\
&\overset{\text{it.}}{\vdash} \underline{\neg(g \wedge \neg f)} \wedge \neg(\neg(g \wedge \neg(g \wedge \neg f)) \wedge \neg f) \\
&\overset{\text{era.}}{\vdash} \neg(\neg(g \wedge \neg(g \wedge \underline{\neg f})) \wedge \neg f) \\
&\overset{\text{deit.}}{\vdash} \neg(\neg(g \wedge \underline{\neg g}) \wedge \neg f) \\
&\overset{\text{deit.}}{\vdash} \neg(\neg(\underline{g} \wedge \neg\top) \wedge \neg f) \\
&\overset{\text{era.}}{\vdash} \neg(\underline{\neg\neg}\top \wedge \neg f) \\
&\overset{\text{dc.}}{\vdash} \neg(\top \wedge \neg f) \\
&\sim \underline{\neg\neg} f \\
&\overset{\text{dc.}}{\vdash} f
\end{aligned}$$

□

Lemma 5. *Let f, g be formulas. Then we have* $(f \leftrightarrow g) \leftrightarrow g \ \vdash^{14} f$.

Proof: We provide a formal derivation of $(f \leftrightarrow g) \leftrightarrow g \ \vdash f$. The last step is done with Lem. 4. As we had 7 derivational steps so far, we have a total of 14 steps.

$$
\begin{array}{rcl}
(f \leftrightarrow g) \leftrightarrow g & = & (\neg(f \wedge \neg g) \wedge \neg(g \wedge \neg f)) \leftrightarrow g \\
& = & \neg(\neg(f \wedge \neg g) \wedge \neg(g \wedge \neg f) \wedge \neg g) \wedge \neg(g \wedge \neg(\neg(f \wedge \neg g) \wedge \neg(\underline{g} \wedge \neg f))) \\
& \overset{\text{deit.}}{\vdash} & \neg(\neg(f \wedge \neg g) \wedge \neg(g \wedge \neg f) \wedge \neg g) \wedge \neg(g \wedge \neg(\neg(f \wedge \neg g) \wedge \underline{\neg\neg} f)) \\
& \overset{\text{dc.}}{\vdash} & \neg(\neg(f \wedge \neg g) \wedge \neg(g \wedge \neg f) \wedge \neg g) \wedge \neg(g \wedge \neg(\neg(\underline{f} \wedge \neg g) \wedge f)) \\
& \overset{\text{deit.}}{\vdash} & \neg(\neg(f \wedge \neg g) \wedge \neg(g \wedge \neg f) \wedge \neg g) \wedge \neg(g \wedge \neg(\underline{\neg\neg} g \wedge f)) \\
& \overset{\text{dc.}}{\vdash} & \neg(\neg(f \wedge \neg g) \wedge \neg(g \wedge \neg f) \wedge \neg g) \wedge \neg(g \wedge \neg(\underline{g} \wedge f)) \\
& \overset{\text{deit.}}{\vdash} & \neg(\neg(f \wedge \neg g) \wedge \underline{\neg(g \wedge \neg f)} \wedge \neg g) \wedge \neg(g \wedge \neg f) \\
& \overset{\text{deit.}}{\vdash} & \neg(\neg(f \wedge \underline{\neg g}) \wedge \neg g) \wedge \neg(g \wedge \neg f) \\
& \overset{\text{deit.}}{\vdash} & \neg(\neg f \wedge \neg g) \wedge \neg(g \wedge \neg f) \\
& \sim & \neg(g \wedge \neg f) \wedge \neg(\neg g \wedge \neg f) \\
& = & (g \rightarrow f) \wedge (\neg g \rightarrow f) \\
& \vdash^7 & f
\end{array}
$$

□

4 Normalforms and Admissibility of Erasure

In automatic theorem proving, for tracking back a proof, a desirable feature of the calculus is the so-called *subformula property* which states that all formulas in a derivation are subformulas of the endformula. The essence of the subformula property is the fact that given a conclusion, every inference rule yields a *finite* set of possible premises. Let us call this property *finite choice property* (see for example [Brü03]). It is easy to see that in Peirce's calculus, only the erasure-rule does not satisfy the finite choice property. In this section, it is shown that the erasure-rule is admissible, i.e. the remaining calculus is still complete.

The restricted version of the calculus, where the erasure-rule is removed, is denoted by $\vdash_{-e}$. Due to symmetry reasons, we will consider a calculus $\vdash_{-i}$, that is $\vdash$ without the insertion-rule, as well. In this section, it will be firstly shown how formulas can be converted to normalforms with $\vdash$ and $\vdash_{-e}$, and then how proofs with $\vdash_{-e}$ can be found in an effective way.

Lemma 6 (Reducing Transformation I). *The formulas* $\neg(f \wedge \neg(g \wedge h))$ *and* $\neg(f \wedge \neg g) \wedge \neg(f \wedge \neg h)$ *are provably equivalent in* $\vdash_{-i}$. *More precisely, we have*

$$\neg(f \wedge \neg(g \wedge h)) \quad \vdash^3_{-i} \quad \neg(f \wedge \neg g) \wedge \neg(f \wedge \neg h) \quad \vdash^4_{-i} \quad \neg(f \wedge \neg(g \wedge h)) \tag{1}$$

Proof:

$$
\begin{array}{rcl}
\neg(f \wedge \neg(g \wedge h)) & \overset{\text{it.}}{\vdash_{-i}} & \neg(f \wedge \neg(g \wedge \underline{h})) \wedge \neg(f \wedge \neg(g \wedge h)) \\
& \overset{\text{era.}}{\vdash_{-i}} & \neg(f \wedge \neg g) \wedge \neg(f \wedge \neg(\underline{g} \wedge h)) \\
& \overset{\text{era.}}{\vdash_{-i}} & \neg(f \wedge \neg g) \wedge \underline{\neg(f \wedge \neg h)} \qquad (*) \\
& \overset{\text{it.}}{\vdash_{-i}} & \neg(f \wedge \neg(g \wedge \neg(f \wedge \neg h))) \wedge \underline{\neg(f \wedge \neg h)} \\
& \overset{\text{era.}}{\vdash_{-i}} & \neg(f \wedge \neg(g \wedge \neg(\underline{f} \wedge \neg h))) \\
& \overset{\text{deit.}}{\vdash_{-i}} & \neg(f \wedge \neg(g \wedge \underline{\neg\neg} h)) \\
& \overset{\text{dc.}}{\vdash_{-i}} & \neg(f \wedge \neg(g \wedge h))
\end{array}
$$

The proof until $(*)$ shows the first part of the lemma, the remaining proof shows the second part. □

The proof of this lemma shows even more. It is carried out on the sheet of assertion, thus, due to the Cut-And-Paste-Theorem I (Lem. 2), in can be carried out in positive contexts. Moreover, its inverse direction can be carried out in arbitrary negative contexts, where the rules iteration and deiteration as well as the rules erasure and insertion are mutually exchanged. Thus we immediately obtain the following corollary.

Corollary 1 (Reducing Transformation II).

$$F[\neg(f \wedge \neg(g \wedge h)]^- \vdash^4_{-e} F[\neg(f \wedge \neg g) \wedge \neg(f \wedge \neg h)]^- \vdash^3_{-e} F[\neg(f \wedge \neg(g \wedge h)]^- \quad (2)$$

$$F[\neg(f \wedge \neg(g \wedge h)]^+ \vdash^3_{-i} F[\neg(f \wedge \neg g) \wedge \neg(f \wedge \neg h)]^+ \vdash^4_{-i} F[\neg(f \wedge \neg(g \wedge h)]^+ \quad (3)$$

With these results, it is possible to reduce the depth of a formula and to transform it into its conjunctive normalform. Before we do so, some technical notations have to be introduced. If g is a strict subformula of f (i.e., g is a subformula of f and $g \neq f$), we write $g < f$ resp. $f > g$. A sequence $f = f_0, f_1, f_2, \ldots, f_n$ is called a NEST OF CONTEXTS OF f, if

1. $f_i = \neg f'_i$ for each $i \geq 1$ (i.e., each f_{i+1} begins with a negation sign '$\neg$'),
2. $f_i > f_{i+1}$ for each $i \geq 0$, and
3. For each $0 \leq i \leq n-1$, there is no formula $\neg g$ with $f_i > \neg g > f_{i+1}$.

The number n is called the DEPTH of the nest. A formula f is said to have depth n if n is the maximal depth of all nests of f. Such a formula is said to be NORMALIZED TO DEPTH n, if moreover for each nest $f = f_0, f_1, f_2, \ldots, f_n$, there exists a propositional variable P_i, $i \in \mathbb{N}$, with $f_n = \neg P_i$. Consider for example the following formulas:

$$f \;:=\; \neg(P_1 \wedge \neg P_2 \wedge \neg P_3) \wedge \neg P_4 \qquad \text{and} \qquad g \;:=\; \neg(P_1 \wedge \neg(P_2 \wedge P_3)) \wedge \neg P_4$$

Both f and g have depth 2, but only f is normalized to depth 2. A formula f which is normalized to depth 2 is a conjunction of formulas $\neg(g_1 \wedge \ldots \wedge g_n)$, where each g_i is a literal. Thus f can be understood to be in CNF (conjunctive normal form), expressed by means of $\neg$ and $\wedge$ only. As $\vdash$ is sound and complete, it is not surprising that each formula can be transformed into its CNF. This is not possible if we restrict ourselves to $\vdash_{-e}$, but even then, it is possible to normalize each formula to depth 3.

Lemma 7 (Normalform).

1. *Using $\vdash_{-e}$, each formula can effectively be transformed into a provably equivalent formula which is normalized to depth* 3.
2. *Using $\vdash$, each formula can effectively be transformed into a provably equivalent formula which is normalized to depth* 2.

Proof: We first prove 1. Let f be an arbitrary formula, assume that f is not normalized to depth 3. Then there exists a nest $f, \neg f_1, \neg f_2, \neg f_3$ where f_3 is not a propositional variable, i.e., f_3 is either of the form $\neg g_3$, or it is the conjunction of at least two nontrivial formulas, i.e., $f_3 = g_3 \wedge g'_3$, with $g_3, g'_3 \neq \top$.

In the first case, we have more explicitely

$$f = g_0 \wedge \neg f_1 = g_0 \wedge \neg(g_1 \wedge \neg f_2) = g_0 \wedge \neg(g_1 \wedge \neg(g_2 \wedge \neg f_3)) = g_0 \wedge \neg(g_1 \wedge \neg(g_2 \wedge \neg\neg g_3))$$

Obviously, we can apply the double cut rule i) and obtain

$$f \vdash g_0 \wedge \neg(g_1 \wedge \neg(g_2 \wedge g_3)) \vdash f$$

In the latter case, we have $f = g_0 \wedge \neg(g_1 \wedge \neg(g_2 \wedge \neg(g_3 \wedge g'_3)))$. Now Eqn. (2) yields

$$f \vdash g_0 \wedge \neg(g_1 \wedge \neg((g_2 \wedge \neg g_3)) \wedge (g_2 \wedge \neg g'_3))) \vdash f$$

These transformations are carried out until we reach a formula which is normalized to depth 3. Thus 1) is proven.

A formula which is normalized to depth 3 cannot be further reduced with Eqn. (2), but Eqn. (3) can still be applied in the outermost context. Thus an analogous argument shows that with the double cut rule or Eqn. (3), each formula can be transformed into a syntactically equivalent formula normalized to depth 2. □

Example:

$$\neg(P_1 \wedge \neg(P_2 \wedge \neg(P_3 \wedge \neg(P_4 \wedge \neg(P_5 \wedge \neg(P_6 \wedge P_7))))))$$
$$\overset{\text{Cor. 1}}{\vdash_{-e}} \neg(P_1 \wedge \neg(P_2 \wedge \neg(P_3 \wedge \neg(P_4 \wedge \neg P_5) \wedge \neg(P_4 \wedge \neg(\neg(P_6 \wedge P_7))))))$$
$$\overset{\text{dc.}}{\vdash_{-e}} \neg(P_1 \wedge \neg(P_2 \wedge \neg(P_3 \wedge \neg(P_4 \wedge \neg P_5) \wedge \neg(P_4 \wedge P_6 \wedge P_7))))$$
$$\overset{\text{2 x Cor. 1}}{\vdash_{-e}} \neg(P_1 \wedge \neg(P_2 \wedge \neg P_3) \wedge \neg(P_2 \wedge \neg(\neg(P_4 \wedge \neg P_5))) \wedge \neg(P_2 \wedge \neg(\neg(P_4 \wedge P_6 \wedge P_7))))$$
$$\overset{\text{2 x dc.}}{\vdash_{-e}} \neg(P_1 \wedge \neg(P_2 \wedge \neg P_3) \wedge \neg(P_2 \wedge P_4 \wedge \neg P_5) \wedge \neg(P_2 \wedge P_4 \wedge P_6 \wedge P_7))$$

In the following, we will show that each tautology can be derived with $\vdash_{-e}$. A well-known method to check the validity of a formula f is to check whether $\neg f$ is contradictory with the method of resolution. The basic idea of resolution is as follows: If k, l are formulas and if A is a propositional variable which does neither occur in k nor in l, then $(A \vee k) \wedge (\neg A \vee l)$ is satisfiable if and only if $k \vee l$ is satisfiable. Now, in order to check whether $\neg f$ is contradictory, subformulas of the form $(A \vee k) \wedge (\neg A \vee l)$ are successively replaced by $k \vee l$ until a formula is reached from which it can be easily decided whether it is satisfiable.

For the $\neg, \wedge$-formalization of propositional logic, this basic transformation can be reformulated as follows: Let k, l formulas, let A be a propositional variable which does neither occur in k nor in l. Then $\neg(A \wedge k) \wedge \neg(\neg A \wedge l)$ is satisfiable if and only if $\neg(k \wedge l)$ is satisfiable. The next lemma shows that the inverse direction of the transformation of resolution can be derived in negative contexts with $\vdash_{-e}$.

Lemma 8 (Inverse Resolution). *Let A be a propositional variable, let k, l be formulas where A does not occur. Then we have:*

$$f[\neg(k \wedge l)]^- \vdash_{-e} f[\neg(A \wedge k) \wedge \neg(\neg A \wedge l)]^-$$

Moreover, $\neg(k \wedge l)$ is satisfiable if and only if $\neg(A \wedge k) \wedge (\neg A \wedge l)$ is satisfiable.

Proof:
$$\begin{aligned} f[\neg(k \wedge l)]^- &\overset{\text{ins.}}{\vdash_{-e}} f[\neg(A \wedge k) \wedge \neg(k \wedge l)]^- \\ &\overset{\text{dc.}}{\vdash_{-e}} f[\neg(A \wedge k) \wedge \neg(\neg\neg k \wedge l)]^- \\ &\overset{\text{ins.}}{\vdash_{-e}} f[\neg(A \wedge k) \wedge \neg(\neg(A \wedge \neg k) \wedge l)]^- \\ &\overset{\text{it.}}{\vdash_{-e}} f[\neg(A \wedge k) \wedge \neg(\neg(A \wedge \neg(A \wedge k)) \wedge l)]^- \\ &\overset{\text{deit.}}{\vdash_{-e}} f[\neg(A \wedge k) \wedge \neg(\neg A \wedge l)]^- \end{aligned}$$
□

Now we are prepared to show that the erasure-rule is admissible.

Theorem 1 (Erasure is Admissible). *If f is a tautology, we have $\vdash_{-e} f$.*

Proof: Due to Lem. 7, we can assume that f is normalized to depth 3, and f cannot be a literal. For $f = g_1 \wedge g_2$, $\vdash_{-e} g_1$ and $\vdash_{-e} g_2$ yield $\vdash_{-e} f$. Thus without loss of generality, we can assume that $f = \neg g$ for a formula g. Obviously, g is normalized to depth 2, and g is contradictory (which is equivalent to f being tautologous).

Now we can resolve g to a formula h which is not resolvable (i.e., h does not contain any subformula of the form $\neg(A\wedge k)\wedge\neg(\neg A\wedge l)$, that is, the rule of resolution cannot be applied). Then g is satisfiable if and only if h is satisfiable. Next, as g is normalized to depth 2, h is normalized to depth 2, too. Moreover, as the inverse direction of the resolution is derivable in $\vdash_{-e}$ due to Lem. 8, we have $\neg h \vdash_{-e} \neg g$. Thus it is sufficient to show that $\neg h$ is derivable with $\vdash_{-e}$.

As h is not resolvable, no propositional variable appears in different subformulas $\neg h_1$, $\neg h_2$ of h one time in a positive and one time in a negative context. Moreover, due to the iteration-rule, we can assume that each propositional variable $A \in \mathcal{P}$ occurs at most once in each subformula $\neg h'$ of h. Now we can assign the following truth-values to all $P_i \in \mathcal{P}$: We set $val(P_i) := \mathtt{ff}$, if P_i occurs in a negative context of h, and we set $val(P_i) := \mathtt{tt}$ otherwise. It is easy to see that if h is not of the form $\neg\top \wedge k$, then $val \models h$. Thus h has the form $\neg\top \wedge k$. Then $\top \overset{\text{dc.}}{\vdash_{-e}} \neg\neg\top \overset{\text{ins.}}{\vdash_{-e}} \neg(\neg\top \wedge k)$ $(= \neg h)$ is a derivation of $\neg h$ in $\vdash_{-e}$, thus we are done. □

Due to $f \models g \quad \Leftrightarrow \quad \models f \to g$, we can check $f \models g$ with $\vdash_{-e}$ as well. But in general, we do not have $f \models g \quad \Rightarrow \quad f \vdash_{-e} g$, as the simple example $P_1 \wedge P_2 \models P_1$ shows.

5 An exponential speed up

The most prominent rule in sequent-calculi is the cut-rule, a generalized version of the modus ponens: $\dfrac{\Gamma_1 \vdash \Delta_1, A \qquad A, \Gamma_2 \vdash \Delta_2}{\Gamma_1, \Gamma_2 \vdash \Delta_1, \Delta_2}$. Due to the 'erasing of A', this rule does not satisfy the finite choice property. Gentzen's famous cut-elimination-theorem states that the cut-rule is admissible: Every proof using the cut-rule can be converted into another proof without the cut-rule (proofs that do not use the cut-rule are called *analytic*). But by doing so, the size of the proof generally grows exponentially. In particular, there are classes of tautologies such that their proofs in sequent-calculi including the cut-rule grow polynomially with their size, whilst in cut-free sequent-calculi, their proofs grow exponentially. In this section, such a class will be investigated.

In [Sta78], R. Statman studied a class of polynomial-size formulas and investigated their proof-lengths in sequent calculi. First we present the formulas constructed by Statman. Let A_i, B_i with $i \geq 1$ propositional variables. We set:

$$f_i := \textstyle\bigwedge_{k=1}^{i}(A_k \vee B_k)$$

$$g_1 := A_1 \qquad \text{induction start}$$

$$h_1 := B_1 \qquad \text{induction start}$$

$$g_{i+1} := f_i \to A_{i+1} = \textstyle\bigwedge_{k=1}^{i}(A_k \vee B_k) \to A_{i+1} \qquad \text{induction step}$$

$$h_{i+1} := f_i \to B_{i+1} = \textstyle\bigwedge_{k=1}^{i}(A_k \vee B_k) \to B_{i+1} \qquad \text{induction step}$$

$$k_n := ((g_1 \vee h_1) \wedge (g_2 \vee h_2) \wedge \ldots \wedge (g_n \vee h_n)) \to (A_n \vee B_n)$$

For example, we have

$$k_2 = [(A_1 \vee B_1) \wedge (\,((A_1 \vee B_1) \to A_2) \vee ((A_1 \vee B_1) \to B_2)\,)] \to (A_1 \vee B_1)$$

It is straightforward to see that the formulas k_n are tautologies. R. Statman has proven that in cut-free sequent-calculi, the lengths of the proofs for k_n grow exponentially, whereas in sequent-calculi including the the cut-rule, it is possible to find proofs of polynomial length. Gulielmi has proven that k_n can be derived within his cut-free deep inference system, In contrast to usual sequent-calculi, in polynomial time. We provide an analogous result for so-to-speak *analytic* calculus $\vdash_{-e}$. So in this respect, the strong rules of $\vdash_{-e}$, yield an exponentially speed-up in the length of proofs, compared to a analytic sequent-calculus.

Theorem 2 (Statman's formulas can be proven with $\vdash_{-e}$ in polynomial time). *For Statman's formula f_n there exists a formal proof of length $n(n+1)$.*

Proof: We provide a formal derivation of k_n. To ease the readability and to save space, we abbreviate $(A_i \vee B_i)$, i.e., $\neg(\neg A_i \wedge \neg B_i)$, by AB_i.

$$
\begin{array}{lll}
\top & \vdash & \neg\neg\top \\
 & \overset{\text{insertion}}{\vdash} & \neg(AB_1 \wedge AB_2 \wedge \ldots \wedge AB_{n-2} \wedge AB_{n-1} \wedge AB_n \wedge \neg\top) \\
 & \overset{\text{it. } AB_n}{\vdash} & \neg(AB_1 \wedge AB_2 \wedge \ldots \wedge AB_{n-2} \wedge AB_{n-1} \wedge AB_n \wedge \neg AB_n) \\
 & = & \neg(AB_1 \wedge AB_2 \wedge \ldots \wedge AB_{n-2} \wedge AB_{n-1} \wedge \neg(\neg A_n \wedge \neg B_n) \wedge \neg AB_n) \\
 & \overset{2\times\text{it. of } AB_{n-1}}{\vdash} & \neg(AB_1 \wedge AB_2 \wedge \ldots \wedge AB_{n-2} \wedge AB_{n-1} \\
 & & \quad \wedge\neg(AB_{n-1} \wedge \neg A_n \wedge AB_{n-1} \wedge \neg B_n) \wedge \neg AB_n) \\
 & = & \neg(AB_1 \wedge AB_2 \wedge \ldots \wedge AB_{n-2} \wedge \neg(\neg A_{n-1} \wedge \neg B_{n-1}) \\
 & & \quad \wedge\neg(AB_{n-1} \wedge \neg A_n \wedge AB_{n-1} \wedge \neg B_n) \wedge \neg AB_n) \\
 & \overset{4\times\text{it. of } AB_{n-2}}{\vdash} & \neg(AB_1 \wedge AB_2 \wedge \ldots \wedge AB_{n-2} \wedge \neg(AB_{n-2} \wedge \neg A_{n-1} \wedge AB_{n-2} \wedge \neg B_{n-1}) \\
 & & \quad \wedge\neg(AB_{n-2} \wedge AB_{n-1} \wedge \neg A_n \wedge AB_{n-2} \wedge AB_{n-1} \wedge \neg B_n) \wedge \neg AB_n) \\
 & \vdots & \\
 & \overset{2(n-1)\times\text{it. of } AB_1}{\vdash} & \neg(AB_1 \\
 & & \quad \wedge\neg(AB_1 \wedge \neg A_2 \wedge AB_1 \wedge \neg B_2) \\
 & & \quad \wedge\neg(AB_1 \wedge AB_2 \wedge \neg A_2 \wedge AB_1 \wedge AB_2 \wedge \neg B_2) \\
 & & \quad \qquad \vdots \\
 & & \quad \wedge\neg(AB_1 \wedge \ldots \wedge AB_{n-1} \wedge \neg A_n \wedge AB_1 \wedge \ldots \wedge AB_{n-1} \wedge \neg B_n) \\
 & & \quad \wedge\neg AB_n) \\
 & \overset{2(n-1)\times\text{dc.}}{\vdash} & \neg(AB_1 \\
 & & \quad \wedge\neg(\neg\neg(AB_1 \wedge \neg A_2) \wedge \neg\neg(AB_1 \wedge \neg B_2)) \\
 & & \quad \wedge\neg(\neg\neg(AB_1 \wedge AB_2 \wedge \neg A_2) \wedge \neg\neg(AB_1 \wedge AB_2 \wedge \neg B_2)) \\
 & & \quad \qquad \vdots \\
 & & \quad \wedge\neg(\neg\neg(AB_1 \wedge \ldots \wedge AB_{n-1} \wedge \neg A_n) \wedge \neg\neg(AB_1 \wedge \ldots \wedge AB_{n-1} \wedge \neg B_n)) \\
 & & \quad \wedge\neg AB_n) \\
 & = & \neg(AB_1 \\
 & & \quad \wedge((AB_1 \to A_2) \vee (AB_1 \to B_2)) \\
 & & \quad \wedge((AB_1 \wedge AB_2 \to A_2) \vee (AB_1 \wedge AB_2 \to B_2)) \\
 & & \quad \qquad \vdots \\
 & & \quad \wedge((AB_1 \wedge \ldots \wedge AB_{n-1} \to A_n) \vee (AB_1 \wedge \ldots \wedge AB_{n-1} \to B_n)) \\
 & & \quad \wedge\neg AB_n) \\
 & = & k_n
\end{array}
$$

So we need $1+1+2(1+2+\ldots+(n-1))+2(n-1) = 2(1+\ldots+n) = n(n+1)$ steps to derive f_n. □

6 Proofs of linear length

In [BZ93], Baaz and Zach show that adding the scheme of equivalence (Eq), i.e.,

$$(f \leftrightarrow g) \rightarrow (h[f] \leftrightarrow h[g]) \qquad \text{(Eq)}$$

to an arbitrary hilbert-style calculus H for propositional logic allows to find proofs of linear length, depending on the number of propositional variables in the formula. More precisely, if T_n is the set of all tautologies in up to n propositional variables, they show that there exists a linear function ϕ such that for all n and all $A \in T_n$ it satisfies H+EQ $\vdash^{\phi(n)} A$.

In this section, by adapting the proof of [BZ93] for our system, it will be shown that we can find a similar approximation for prooflengths. In contrast to [BZ93], it is not needed to add new rules or axioms to our calculus. Instead, it suffices to generalize the rules iteration, deiteration and double cut in a natural manner.

Recall the definition of the iteration rule: $f[g \wedge h[i]] \vdash f[g \wedge h[g \wedge i]]$. If f is a formula with a subformula g, then each subformula i such that f has the form $f[g \wedge h[i]]$ is said to be RECEIVABLE FOR THE ITERATION OF g. We now generalize the rules of the calculus. This calculus will be denoted by $\Vdash$.

gen. Iteration: If $f[g]$ is a formula, then it is allowed to add to each context i which is receivable for the iteration of g an arbitrary number of copies of g.
gen. Deiteration: Inverse direction of deiteration.
gen. Double Cut i): An arbitrary number of double negations may be removed from a formula.
gen. Double Cut ii): An arbitrary number of double negations may be added to a formula.

Some simple examples shall illustrate the rules. Consider the following proof, where in each step, the outermost subformula $A \wedge \neg B$ is iterated (one time into the outermost context, two times into the context of $D \wedge F$). In this derivation, the iterated copies of the subformula are underlined.

$$\begin{aligned} A \wedge \neg B \wedge C \wedge \neg(D \wedge F) &\overset{\text{it.}}{\vdash} A \wedge \neg B \wedge \underline{A \wedge \neg B} \wedge C \wedge \neg(D \wedge F) \\ &\overset{\text{it.}}{\vdash} A \wedge \neg B \wedge A \wedge \neg B \wedge C \wedge \neg(\underline{A \wedge \neg B} \wedge D \wedge F) \\ &\overset{\text{it.}}{\vdash} A \wedge \neg B \wedge A \wedge \neg B \wedge C \wedge \neg(A \wedge \neg B \wedge \underline{A \wedge \neg B} \wedge D \wedge F) \end{aligned}$$

This derivation is now consolidated to *one* application of the generalized iteration rule. But a 'nested' application of the iteration-rule is not considered as generalized iteration rule, i.e., although we have

$$\begin{aligned} A \wedge \neg B \wedge C \wedge \neg(D \wedge F) &\overset{\text{it.}}{\vdash} A \wedge \neg B \wedge \underline{A \wedge \neg B} \wedge C \wedge \neg(D \wedge F) \\ &\overset{\text{it.}}{\vdash} A \wedge \neg B \wedge A \wedge \neg(\underline{A \wedge \neg B} \wedge B) \wedge C \wedge \neg(D \wedge F) \end{aligned}$$

the last formula is *not* obtained from the first formula with a single application of the application of the generalized iteration rule, as in the second step, the subformula $A \wedge \neg B$ is iterated into a context which was not created until the first step, i.e., into a context which does not exist in the starting formula.

The generalized double cut rule is easier to understand.

$$A \wedge \neg B \wedge C \wedge \neg(D \wedge F) \overset{\text{gen. dc.}}{\Vdash} A \wedge \neg B \wedge \underline{\neg\neg}(C \wedge \underline{\neg\neg}\neg(\underline{\neg\neg}D \wedge F))$$

We can now prove that with $\Vdash$ we can find derivations of tautologies whose length depend linearly from the number of the propositional variables in the tautology.

Theorem 3 (Proofs of linear length in the generalized calculus). *If f is a tautology with n different propositional variables, we have $\vdash^{24+14n} f$.*

Proof: The proof is done by induction over n.

So, for the induction start, let f be a tautology without propositional variables. For $f \not\sim \top$ and $f \not\sim \neg\top$, f contains $\neg\neg\top$ or $\neg\top \wedge \neg\top$ as subformula. We can successively replace subformulas $\neg\neg\top$ by $\top$ (with the double cut rule) and subformulas $\neg\top \wedge \neg\top$ by $\neg\top$ (by deiterating one occurrence of $\neg\top$). As both rules are equivalence rules, it is easy to see that f is a tautology if and only if this procedure eventually yields $\top$.

This idea is captured by the ongoing proof, which is based on Yukami's trick ([Yuk84]). In the formal derivation of f we have to construct, the manifold replacements of $\neg\neg\top$ by $\top$ of the double cut rule will be performed in one step by an application of the generalized double cut rule. But the manifold replacements of $\neg\top \wedge \neg\top$ by $\neg\top$ cannot be analogously be captured by one application of the generalized deiteration rule, as in the different applications of the deiteration rule take place in different contexts (i.e., different occurrences of $\neg\top$ are used for deiterating other occurrences of $\neg\top$). To overcome with this problem, instead of replacing $\neg\top \wedge \neg\top$ directly by $\neg\top$, we first replace each occurrence $\neg\top \wedge \neg\top$ by $\neg\neg(\neg\top \wedge \neg\top)$ with the generalized double cut rule. Then all occurrences of $\neg(\neg\top \wedge \neg\top)$ are replaced by $\top$ with the generalized deiteration rule, using a subformula $\neg(\neg\top \wedge \neg\top)$ in the uppermost context.

In order to construct the formal derivation, we first define a mapping $\Delta(f)$, which formalizes the three different modifications of formulas as follows:

1. If f contains a double negation $\neg\neg\top$ as subformula, then $\Delta(f)$ is obtained from f by removing the double negation, i.e.: For $f[\neg\neg\top]$ we set
$$\Delta(f[\neg\neg\top]) := f[\top] \quad .$$
2. If f contains $(\neg\top \wedge \neg\top)$ as subformula, then $\Delta(f)$ is obtained from f by replacing this subformula by $\neg\neg(\neg\top \wedge \neg\top)$, i.e.: For $f[\neg\top \wedge \neg\top]$ we set
$$\Delta(f[\neg\top \wedge \neg\top]) := f[\neg\neg(\neg\top \wedge \neg\top)] \quad .$$
3. If f contains $\neg(\neg\top \wedge \neg\top)$ as subformula, then $\Delta(f)$ is obtained from f by removing this subformula, i.e.: For $f[\neg(\neg\top \wedge \neg\top)]$ we set
$$\Delta(f[\neg(\neg\top \wedge \neg\top)]) := f[\top] \quad .$$

Due to the discussion at the beginning of this proof, we know that f is a tautology if and only if there is an n such that $\Delta^n(f) = \top$.

Now let f be a tautology and $n \in \mathbb{N}$ with $\Delta^n(f) = \top$. Let

$$f_d^{-1} := \Delta f \leftrightarrow (\Delta^2 f \leftrightarrow (\Delta^3 f \leftrightarrow \ldots (\Delta^{n-1} f \leftrightarrow \top)\ldots) \quad ,$$

$$\begin{aligned} f_d &:= f \leftrightarrow (\Delta f \leftrightarrow (\Delta^2 f \leftrightarrow \ldots (\Delta^{n-1} f \leftrightarrow \top)\ldots) \\ &= f \leftrightarrow (f_d^{-1}) \qquad \text{and} \end{aligned}$$

$$\begin{aligned} f_d^{\Delta} &:= \Delta f \leftrightarrow (\Delta^2 f \leftrightarrow (\Delta^3 f \leftrightarrow \ldots (\Delta^n f \leftrightarrow \top)\ldots) \\ &= \Delta f \leftrightarrow (\Delta^2 f \leftrightarrow (\Delta^3 f \leftrightarrow \ldots (\top \leftrightarrow \top)\ldots) \end{aligned}$$

Now we can derive f from $\top$. We start with the construction of $\neg(\neg\top \wedge \neg\top)$, and we derive $f_d \leftrightarrow f_d$ as well.

$$
\begin{array}{rcl}
\top & \overset{\text{gen. dc}}{\vdash} & \neg\neg\top \wedge \neg\neg\top \\
& \overset{\text{it.}}{\vdash} & \neg(\neg\top \wedge \neg\top) \wedge \neg\neg\top \\
& \overset{\text{ins.}}{\vdash} & \neg(\neg\top \wedge \neg\top) \wedge \neg(f_d \wedge \neg\top) \\
& \overset{\text{it.}}{\vdash} & \neg(\neg\top \wedge \neg\top) \wedge \neg(f_d \wedge \neg f_d) \\
& \overset{\text{it.}}{\vdash} & \neg(\neg\top \wedge \neg\top) \wedge \neg(f_d \wedge \neg f_d) \wedge \neg(f_d \wedge \neg f_d) \\
& = & \neg(\neg\top \wedge \neg\top) \wedge (f_d \leftrightarrow f_d) \\
& = & \neg(\neg\top \wedge \neg\top) \wedge ((f \leftrightarrow (f_d^{-1})) \leftrightarrow f_d) \\
& \vdash^3 & \neg(\neg\top \wedge \neg\top) \wedge ((f \leftrightarrow (f_d^{-1})) \leftrightarrow f_d^{\Delta})
\end{array}
$$

The last step reflects the discussion at the beginning of the proof. It is carried out each with one application of:

1. the generalized double cut insertion rule
2. the generalized double cut erasure rule
3. the generalized deiteration rule

The formulas f_d^{-1} and f_d^{Δ} differ only in the innermost formula, which is $\top \leftrightarrow \top$ for f_d^{Δ} and $\top$ for f_d^{-1}. We have

$$
\top \leftrightarrow \top \;=\; \neg(\top \wedge \neg\top) \wedge \neg(\top \wedge \neg\top) \;\sim\; \neg\neg\top \wedge \neg\neg\top
$$

Thus the most inner formula $\top \leftrightarrow \top$ of f_d^{Δ} can be replaced with the generalized double cut rule by $\top$. That is, we get:

$$
\begin{array}{rcl}
\neg(\neg\top \wedge \neg\top) \wedge ((f \leftrightarrow (f_d^{-1})) \leftrightarrow f_d^{\Delta}) & \overset{\text{gen. dc.}}{\vdash} & \neg(\neg\top \wedge \neg\top) \wedge ((f \leftrightarrow f_d^{-1}) \leftrightarrow f_d^{-1}) \\
& \overset{\text{era}}{\vdash} & (f \leftrightarrow f_d^{-1}) \leftrightarrow f_d^{-1}
\end{array}
$$

According to Lem. 5, we can derive f from this formula within 14 steps. As we needed 10 steps so far, we see that f can be derived with a total number of 24 steps from $\top$. This finishes the induction start.

Assume now we have shown that the Lemma holds for formulas with at most n propositional variables. Now let f be a tautology with $n+1$ propositional variables, let A be one of these variables. As we have

$$
\models f \quad \Leftrightarrow \quad \models f[\![\top/A]\!] \wedge f[\![\neg\top/A]\!] \quad ,
$$

there exists a formal derivation of $f[\![\top/A]\!] \wedge f[\![\neg\top/A]\!]$ with length $24 + 14n$. From this formula, we proceed as follows:

$$
\begin{array}{rl}
& f[\![\top/A]\!] \wedge f[\![\neg\top/A]\!] \\
\overset{\text{dc.}}{\vdash} & \neg\neg\top \wedge f[\![\top/A]\!] \wedge f[\![\neg\top/A]\!] \\
\overset{\text{ins.}}{\vdash} & \neg(\underline{A} \wedge \neg\top) \wedge f[\![\top/A]\!] \wedge f[\![\neg\top/A]\!] \\
\overset{\text{it. of } A}{\vdash} & \neg(\neg A \wedge A) \wedge f[\![\top/A]\!] \wedge f[\![\neg\top/A]\!] \\
\overset{\text{dc.}}{\vdash} & \neg(\neg A \wedge \neg\neg A) \wedge \underline{f[\![\top/A]\!]} \wedge f[\![\neg\top/A]\!] \\
\overset{\text{it. of } f[\![\top/A]\!]}{\vdash} & \neg(\neg(A \wedge f[\![\top/A]\!]) \wedge \neg\neg A) \wedge f[\![\top/A]\!] \wedge \underline{f[\![\neg\top/A]\!]}
\end{array}
$$

$$
\begin{array}{rl}
\text{it. of } f[\![\neg\top/A]\!] & \\
\vdash & \neg(\neg(A \wedge f[\![\top/A]\!]) \wedge \neg(\neg A \wedge f[\![\neg\top/A]\!])) \wedge \underline{f[\![\top/A]\!] \wedge f[\![\neg\top/A]\!]} \\
\text{era.} & \\
\vdash & \neg(\neg(A \wedge f[\![\top/A]\!]) \wedge \neg(\neg A \wedge f[\![\neg\top/A]\!])) \\
\text{gen. it. of } A & \\
\vdash & \neg(\neg(A \wedge f[\![A/A]\!]) \wedge \neg(\neg A \wedge f[\![\neg\top/A]\!])) \\
\text{gen. it. of } \neg A & \\
\vdash & \neg(\neg(A \wedge f[\![A/A]\!]) \wedge \neg(\neg A \wedge f[\![\neg\neg A/A]\!])) \\
\text{gen. dc.} & \\
\vdash & \neg(\neg(A \wedge f[\![A/A]\!]) \wedge \neg(\neg A \wedge f[\![A/A]\!])) \\
= & \neg(\neg(A \wedge f) \wedge \neg(\neg A \wedge f)) \\
\text{era.} & \\
\vdash & \neg(\neg f \wedge \neg(\neg A \wedge f)) \\
\text{era.} & \\
\vdash & \neg(\neg f \wedge \neg f) \\
\text{deit.} & \\
\vdash & \neg\neg f \\
\text{dc.} & \\
\vdash & f
\end{array}
$$

As we needed 14 further steps, we obtain $\top \vdash^{24+14(n+1)} f$, thus we are done. □

7 Further Research

This paper is a first step to the proof-theoretic foundations of Peirce's calculus for Alpha graphs. The calculus has powerful rules, and it has to be investigated whether the results of this paper can be improved. Firstly, it is natural to ask whether the deiteration rule is admissible as well. Kocura uses in [HK05] a system consisting of the rules insertion, iteration, and double cut, but a proof whether this system is complete is still missing. Secondly, one might ask whether the results of the last section hold for the non-generalized calculus as well. I strongly suspect that this is not the case. Consider the formula $f := \neg\neg\top \wedge \ldots \wedge \neg\neg\top$ consisting of 2^n subformulas $\neg\neg\top$. Then f can can be derived with $\vdash$ within $n+1$ steps as follows: First insert a double cut, then in each step, iterate the whole formula derived so far. It is likely that this is the optimal derivation of f, but so far, I did not succeed in proving that.

Besides these two questions, the results of the paper show that Peirce's calculus may be of interest for automated theorem proving, thus it should be investigated further from a proof-theoretic point of view.

References

[Brü03] Kai Brünnler. *Deep Inference and Symmetry in Classical Proofs*. PhD thesis, Technische Universität Dresden, 2003.

[Bur91] Robert W. Burch. *A Peircean Reduction Thesis: The Foundation of Topological Logic*. Texas Tech. University Press, Texas, Lubbock, 1991.

[BZ93] Matthias Baaz and Richard Zach. Short proofs of tautologies using the schema of equivalence. In Egon Börger, Yuri Gurevich, and Karl Meinke, editors, *CSL*, volume 832 of *Lecture Notes in Computer Science*, pages 33–35. Springer, Berlin – Heidelberg – New York, 1993.

[Dau02] Frithjof Dau. An embedding of existential graphs into concept graphs with negations. In Uta Priss, Dan Corbett, and Galia Angelova, editors, *ICCS*, volume 2393 of *LNAI*, pages 326–340, Borovets, Bulgaria, July, 15–19, 2002. Springer, Berlin – Heidelberg – New York.

[Dau04] Frithjof Dau. Types and tokens for logic with diagrams: A mathematical approach. In Karl Erich Wolff, Heather D. Pfeiffer, and Harry S. Delugach, editors, *Conceptual Structures at Work: 12th International Conference on Conceptual Structures*, volume 3127 of *Lecture Notes in Computer Science*, pages 62–93. Springer, Berlin – Heidelberg – New York, 2004.

[Dau06] Frithjof Dau. Mathematical logic with diagrams, based on the existential graphs of peirce. Habilitation thesis. To be published. Available at: `http://www.dr-dau.net`, 2006.

[DMS05] Frithjof Dau, Marie-Laure Mugnier, and Gerd Stumme, editors. *Common Semantics for Sharing Knowledge: Contributions to ICCS 2005*, Kassel, Germany, July, 2005. Kassel University Press.

[Ham95] Eric M. Hammer. *Logic and Visual Information*. CSLI Publications, Stanford, California, 1995.

[HB35] Weiss Hartshorne and Burks, editors. *Collected Papers of Charles Sanders Peirce*, Cambridge, Massachusetts, 1931–1935. Harvard University Press.

[HK05] David P. Hodgins and Pavel Kocura. Propositional theorem prover for peircelogik. In Dau et al. [DMS05], pages 203–204.

[Liu05] Xin-Wen Liu. An axiomatic system for peirce's alpha graphs. In Dau et al. [DMS05], pages 122–131.

[Pap83] Helmut Pape. *Charles S. Peirce: Phänomen und Logik der Zeichen*. Suhrkamp Verlag Wissenschaft, Frankfurt am Main, Germany, 1983. German translation of Peirce's Syllabus of Certain Topics of Logic.

[Pei35] Charles Sanders Peirce. *MS 478: Existential Graphs.* Harvard University Press, 1931–1935. Partly published in of [HB35] (4.394-417). Complete german translation in [Pap83].

[Pei92] Charles Sanders Peirce. Reasoning and the logic of things. In K. L. Kremer and H. Putnam, editors, *The Cambridge Conferences Lectures of 1898*. Harvard Univ. Press, Cambridge, 1992.

[PS00] Charles Sanders Peirce and John F. Sowa. Existential Graphs: MS 514 by Charles Sanders Peirce with commentary by John Sowa, 1908, 2000. Available at: `http://www.jfsowa.com/peirce/ms514.htm`.

[Rob73] Don D. Roberts. *The Existential Graphs of Charles S. Peirce*. Mouton, The Hague, Paris, 1973.

[Rob92] Don D. Roberts. The existential graphs. *Computers Math. Appl.*., 23(6–9):639–63, 1992.

[Sch60] Kurt Schütte. *Beweistheorie*. Springer, Berlin – Heidelberg – New York, 1960.

[Shi02a] Sun-Joo Shin. *The Iconic Logic of Peirce's Graphs*. Bradford Book, Massachusetts, 2002.

[Shi02b] Sun-Joo Shin. Multiple readings in peirce's alpha graphs. In Michael Anderson, Bernd Meyer, and Patrick Olivier, editors, *Diagrammatic Representation and Reasoning*. Springer, Berlin – Heidelberg – New York, 2002.

[Sow84] John F. Sowa. *Conceptual structures: information processing in mind and machine*. Addison-Wesley, Reading, Mass., 1984.

[Sow97] John F. Sowa. Logic: Graphical and algebraic. manuscript, Croton-on-Hudson, 1997.

[Sta78] Richard Statman. Bounds for proof-search and speed-up in predicate calculus. *Annals of Mathematical Logic*, 15:225–287, 1978.

[vH03] Bram van Heuveln. Existential graphs. Presentations and Applications at: `http://www.rpi.edu/ heuveb/research/EG/eg.html`, 2003.

[Yuk84] Tsuyoshi Yukami. Some results on speed-up. *Ann. Japan Assoc. Philos. Sci.*, 6:195–205, 1984.

[Zem64] Jay J Zeman. *The Graphical Logic of C. S. Peirce*. PhD thesis, University of Chicago, 1964. Available at: `http://www.clas.ufl.edu/users/jzeman/`.