DAG Sequent Proofs with a Substitution Rule

Marcelo Finger

ABSTRACT. In this paper we study an extension to classical sequent calculus with a substitution rule, which is normally admissible in classical logic. The structure of proofs is also extended to permit DAG shaped proofs. We analyse several properties of this system, such as the complexity of cut-elimination, and propose an extended tableau proof system, called s-tableau, that corresponds to the DAG-sequent calculus. We show how the pigeon hole principle can be solved linearly solved in s-tableaux.

1 Introduction

In this paper, we investigate a sequent proof method known to have short proofs even for the hardest known propositional formulas. We explore some of the proof-theoretical properties of this inference system and investigate how it can be transformed in a tableau-like decision procedure.

This work is in the spirit of recent work on efficient propositional inference systems, by Dov Gabbay and the author, in which we studied several families of tractable subclassical logics that are less complex than propositional classical logic [Finger and Gabbay, 2005]. Each element of those families has a polynomial time decision procedure. That investigation restricted the use of the cut rule in a non cut-eliminable formulation of propositional classical logic.

In this work, we investigate classical proof theory, especially the role of admissible rules, in another direction. Here add the admissible *substitution rule* (or s-rule) to the set of inference rules in a Gentzen sequent system. Let a *substitution* σ be a formula transformation that maps atoms into formulas, and is extended to all formulas in a homomorphic way. If A is a formula let $A\sigma$ be the application of σ on A. Similarly, if Γ is a set of formulas, then $\Gamma\sigma$ is the result of applying σ to every formula in Γ. In this setting, the s-rule states that if $\Gamma \vdash \Delta$ is a derivable sequent and σ is a substitution, then we can infer $\Gamma\sigma \vdash \Delta\sigma$. Furthermore, this rule can be *defocusing*, that is, the same source sequent $\Gamma \vdash \Delta$ can receive several substitutions, transforming the usual tree structure of a sequent proof into a direct acyclic graph (DAG), as illustrated in Figure 1.

$$
\begin{tikzcd}
 & \Gamma \vdash \Delta \arrow[dl, "\sigma_1"'] \arrow[dr, "\sigma_n"] & \\
\Gamma\sigma_1 \vdash \Delta\sigma_1 & \cdots & \Gamma\sigma_n \vdash \Delta\sigma_n
\end{tikzcd}
$$

Figure 1. The Defocusing Substitution Rule

$$
\begin{tikzcd}
\mathsf{premiss}_1 \arrow[dr] & \cdots & \mathsf{premiss}_k \arrow[dl] \\
 & \mathsf{conclusion} &
\end{tikzcd}
$$

(a) Focusing Rule

$$
\begin{tikzcd}
 & \mathsf{premiss} \arrow[dl] \arrow[dr] & \\
\mathsf{conclusion}_1 & \cdots & \mathsf{conclusion}_k
\end{tikzcd}
$$

(b) Defocusing Rule

Figure 2. Focusing and Defocusing Rules

In a usual tree-like sequent proofs, a rule may contain one premiss or several premisses; in the latter case, the rule is called *focusing* according to the terminology of [Carbone and Semmes, 2000] illustrated in Figure 2. Usual rules (see Figures 3 and 4) are linear or *focusing*, that is, rules are viewed as providing directed edges from the premisses to the conclusion, such that there is only one conclusion but possibly one or more premisses; the former is considered a linear rule, the latter a focusing one. With usual rules, no defocusing is possible, so the proof necessarily has a tree-like structure with a single directed path between any node and the deduced sequent at the root of the tree.

In a DAG proof, due to the presence of defocusing substitution nodes (or s-nodes) there may be more than one directed path from any sequent in the proof to the root sequent. In this way, a DAG proof avoids repetition of isomorphic branches and is thus more compact than a tree proof, so proofs of the same sequent may be shorter with a DAG structure. As the substitution σ may be the identity substitution, there is no need to have any other defocusing rule.

Notation: The propositional connectives we consider here are $\wedge$, $\vee$, $\rightarrow$ and $\neg$. We measure the size of a formula A, $|A|$, as the number of symbols it contains. There are basically two ways of measuring the size of a proof:

(a) The *number of lines*: this is the number of sequents (usually called in the literature the number of lines) occurring in a proof Π, represented by $|\Pi|$.

(b) The *number of symbols*: this is the sum of the sizes of all formulas occurring in a proof Π, represented by $\|\Pi\|$.

We say that a proof system S_1 p-simulates a proof system S_2 if there exists a polynomial $p(x)$ such that for every proof Π_2 of a theorem A in S_2 there is a proof Π_1 of A in S_1 such that $|\Pi_1| \leq p(|\Pi_2|)$. We say that two proof systems are equivalent if each one p-simulates the other.

1.1 Related Works

In their seminal work, Cook and Reckhow [1979] defined a generalisation of the usual propositional proof system called *Frege systems*. A Frege system is based on a finite and complete set of propositional connectives, and has a finite set of schematically defined rules of inference with one or more premisses, and one conclusion, such that the set of rules is sound and complete. A *proof* is a direct acyclic graph, where each node is labelled with a formula or a sequent. The class of Frege systems include inference systems like Hilbert-style axiomatisations, Natural Deduction and Gentzen sequent systems. Furthermore, Cook and Reckhow showed that any Frege system $\mathcal{F}_1$ can p-simulate any other Frege system $\mathcal{F}_2$.

Cook and Reckhow [1979] also defined the notion of an *extended Frege system*, which is a Frege proof system augmented with the introduction of inferences of the form:

$$\vdash p \leftrightarrow A$$

where p is a propositional symbol that does not occur in A, nor in any previous formula in that branch of the proof, nor in the final final formula at the root of the proof. This inference allows the atom p to be an abbreviation of the formula A, which has the potential of reducing the number of symbols in the proof. Extended Frege systems were also shown to p-simulate each other. It remains an open problem if Frege systems can p-simulate extended Frege systems.

In a similar way, the notion of *substitution Frege system* consists of a Frege system augmented with the substitution rule for formulas (or sequents), as in Figure 1. It was shown that substitution Frege systems and extended Frege systems are equivalent, that is, any substitution Frege system $s\mathcal{F}$ can p-simulate any extended Frege proof system $e\mathcal{F}$ and vice-versa [Cook and Reckhow, 1979, Krajíček and Pudlák, 1989]. It was also shown that this result holds even if the substitution is restricted to the mere renaming of propositional symbols [Buss, 1995].

Thus, if we extend a sequent system with a defocusing renaming rule as a weaker version of the substitution rule, we are still guaranteed to have a proof system that can p-simulate any extended Frege system.

One thing that comes to mind when one is discussing extended Frege systems, or any of its equivalent formulations, is if they can ever be used in practice in a real prover. This paper tries to contribute to this question.

1.2 Organisation of this Paper

Most of the literature on substitution and renaming Frege system concentrated basically on the length of proofs. Here we take one particular substitution system and study its intrinsic logic properties.

We start by defining the DAG-sequent proof system and show its soundness and completeness, and show how the substitution rule can be eliminated (Section 2). We then study the complexity of cut elimination and show that, unlike traditional sequent system, in the presence of the substitution rule cut elimination does not provoke an exponential blow up in the size of proofs.

This motivates us to examine how this technique can be brought to semantic tableaux (Section 3). We thus present s-tableaux by extending semantic tableaux with a substitution closure rule, and we prove that it actually corresponds to the DAG-sequent proof system.

As an example of s-tableaux, we apply it to the family of formulas that encode the pigeon hole principle (PHP_n), and show that we have linear s-tableau proofs for PHP_n (Section 4).

2 DAG proofs

We start by formally defining the sequent proof system we will be studying. A sequent is a pair of the form $\Gamma \vdash \Delta$, where Γ and Δ are *multisets*; Γ is the antecedent and Δ is the consequent of the sequent. We then have schematic rules that apply to sequents in a rule, divided in the usual two groups of logical and structural rules. The logical (connective) rules are shown in Figure 3. The structural rules are shown in Figure 4.

Due to the definition of the antecedent and consequent as multisets, the structural rules of *associativity* and *commutativity* are implicit in this formulation. The structural rule of *monotonicity* or *weakening* is taken care of by the *Axiom Rule*. The rules in Figures 3 and 4 are focusing rules and thus generate only tree-like proofs.

The generalisation to DAGs comes when we introduce the defocusing *substitution Rule* (s-rule); see Figure 1. A *substitution* σ is a set of pairs of propositional atoms and formulas, that we represent as $\sigma = [p_1 := A_1, \ldots, p_k := A_k]$. If the pair $\langle p_i, A_i \rangle$ is in σ we write $(p_i := A_i) \in \sigma$. For sets, multisets or sequents, the application of σ means the application of substitution to each of its elements.

$\dfrac{A, \Gamma \vdash \Delta}{A \wedge B, \Gamma \vdash \Delta}\ (\wedge \vdash)$	$\dfrac{\Gamma \vdash \Delta, A \quad \Gamma \vdash \Delta, B}{\Gamma \vdash \Delta, A \wedge B}\ (\vdash \wedge)$
$\dfrac{A, \Gamma \vdash \Delta \quad B, \Gamma \vdash \Delta}{A \vee B, \Gamma \vdash \Delta}\ (\vee \vdash)$	$\dfrac{\Gamma \vdash \Delta, A}{\Gamma \vdash \Delta, A \vee B}\ (\wedge \vdash)$
$\dfrac{\Gamma_1 \vdash \Delta_1, A \quad B, \Gamma_2 \vdash \Delta_2}{A \to B, \Gamma_1, \Gamma_2 \vdash \Delta_1, \Delta_2}\ (\to\vdash)$	$\dfrac{\Gamma, A \vdash \Delta, B}{\Gamma \vdash \Delta, A \to B}\ (\vdash\to)$
$\dfrac{\Gamma \vdash \Delta, A}{\neg A, \Gamma \vdash \Delta}\ (\neg \vdash)$	$\dfrac{A, \Gamma \vdash \Delta}{\Gamma \vdash \Delta, \neg A}\ (\vdash \neg)$

Figure 3. Logical Rules for the Sequent Calculus

$\dfrac{}{\Gamma, A \vdash A, \Delta}\ (\text{Axiom})$	$\dfrac{\Gamma_1 \vdash \Delta_1, A \quad A, \Gamma_2 \vdash \Delta_2}{\Gamma_1, \Gamma_2 \vdash \Delta_1, \Delta_2}\ (\text{Cut})$
$\dfrac{\Gamma_1, A, A, \Gamma_2 \vdash \Delta}{\Gamma_1, A, \Gamma_2 \vdash \Delta}\ (\text{Contraction} \vdash)$	$\dfrac{\Gamma \vdash \Delta_1, A, A, \Delta_2}{\Gamma \vdash \Delta_1, A, \Delta_2}\ (\vdash \text{Contraction})$

Figure 4. Usual Structural Rules

There is nothing implicitly "defocusing" in the use of the s-rule, and in principle any node that is used more than once in a proof could be defocusing. We have decided to concentrate the defocusing effect only on the s-rule to obtain a true extension of the usual tree-like sequent proofs.

We can then formally define a *DAG-sequent proof* Π as a direct acyclic graph constructed inductively from the application of only the Axiom, Logical, Structural and Substitution rules. A sequent $\Gamma \vdash \Delta$ is *derivable* or *provable* if there is a DAG sequent proof Π having $\Gamma \vdash \Delta$ as the only node without leaving arrows (a *drain* in the graph). Note that axioms are the only *source nodes* in a proof, that is, the only nodes with no incoming arrows.

A *prefix* Π' of a DAG Π at a node n is the subgraph containing n, such that if all nodes and arcs pointing to to some node in Π' are also in Π'.

LEMMA 1. *Let S be a sequent in a proof Π. Let Π' be the prefix of Π at S. Then Π' is a proof of S.*

Proof. Directly from the definitions of DAG-sequent proofs and of prefix. Just note that S must be a drain in Π' since, by acyclicity of Π, no arc coming out from S may point into Π'. ■

The result above shows that any intermediate sequent generated in a proof is indeed a derivable sequent, as in usual sequent proofs. We say that a sequent proof is *usual* if it is a proof without the use of the substitution rule, and hence it has a tree-like structure.

Consider the classical semantics for propositional formulas based on propositional valuations. We write $\Gamma \models A$ if any valuation that satisfies all formulas in Γ also satisfies A. *Soundness* means that $\Gamma \vdash A$ implies $\Gamma \models A$ and *completeness* means that $\Gamma \models A$ and $\Gamma \vdash A$.

THEOREM 2. *The DAG-sequent calculus is sound and complete.*

Proof. Completeness is trivial, since the usual sequent calculus is contained in the DAG-sequent calculus. For soundness, given the soundness of the usual calculus, all we have to do is show that the substitution rule takes a valid sequent into a valid sequent, which follows directly from the fact that the substitution has the effect of reducing the number of valuations available to the valid sequent in the premiss of a substitution rule. ■

We now consider the admissibility of the substitution and cut rules from DAG proof, by showing their elimination. It must be obvious from the soundness and completeness results that the substitution rule does not add or remove any derivable sequents from classical ones. However, what we are concerned here with the exponential explosion that occurs when we eliminate it from the proof.

LEMMA 3. *If there is a DAG-sequent proof* Π *of sequent* S *then there is a DAG-sequent proof* Π' *of* S *without the use of the substitution rule, such that* $|\Pi'|$ *is bounded by an exponential function on* $|\Pi|$.

Proof. (Sketch) The exponential explosion occurs when we eliminate a convergence point generated by two or more *distinct* applications of the substitution rule, to a single node, a illustrated below.

$$\begin{tikzcd}[column sep=small] & \begin{array}{c}\Pi \\ \Gamma \vdash \Delta\end{array} \arrow[dl, "\sigma_1"'] \arrow[dr, "\sigma_2"] & \\ \Gamma\sigma_1 \vdash \Delta\sigma_1 & & \Gamma\sigma_2 \vdash \Delta\sigma_2 \end{tikzcd}$$

Here we see that a proof Π leads to a sequent $\Gamma \vdash \Delta$ to which the substitution rule is applied twice (or n-times, for $n \geq 2$). When we eliminate this application of the rule, two instances of the proof proof Π are created: $\Pi\sigma_1$, and $\Pi\sigma_2$, where by $\Pi\sigma$ we mean the application of the substitution σ to all formulas in Π, thus generating:

$$\begin{array}{cc} \Pi\sigma_1 & \Pi\sigma_2 \\ \Gamma\sigma_1 \vdash \Delta\sigma_1 & \Gamma\sigma_2 \vdash \Delta\sigma_2 \end{array}$$

The duplication of proof Π leads to the exponential growth both in number of lines and in number of symbols when there is a chain of eliminations on the same path in the proof ■

The fact that the two substitutions σ_1 and σ_2 are distinct is fundamental for the exponential growth, otherwise a simple use of the contraction rule would have done the job without the duplication of Π.

This phenomenon of exponential growth is also known for usual cut elimination, for there are known sequents whose cut-free traditional proofs can only be exponentially larger than some versions with cuts [Statman, 1979, Orevkov, 1982, Boolos, 1984]. Proofs can, of course, be free of substitution and of cut, by first eliminating substitution as above and then eliminating the cuts by some traditional cut elimination process [Girard, 1987b, Takeuti, 1987].

We now show that cut can be eliminated without exponential explosion if the substitution rule is used.

THEOREM 4. *If there is a usual tree-like sequent proof* Π *of sequent* S*, possibly with the use of cuts, then there is a DAG-sequent proof* Π' *of* S *without the use of the cut-rule, such that* $|\Pi'|$ *is linear with respect to* $|\Pi|$ *and* $\|\Pi'\|$ *is linear with respect to* $\|\Pi\|$.

Proof. The work of Carbone [Carbone, 1997, Carbone and Semmes, 2000] has shown that the duplication of a chunk of a branch during cut elimination occurs when that branch contains a contraction of a formula A followed by a cut over A. The exponential explosion then occurs when a branch contains several occurrences of contraction-cut sequences.

In fact, according to the usual cut elimination processes [Girard, 1987b, Takeuti, 1987], when cut is eliminated from an axiom, the size of Π' actually decreases. When we eliminate a cut where the cut-formula was introduced by a connective rule, the cut is pushed up towards the leaves. At each such step, the number of lines may increase only in a constant way and the number of symbols may increase only linearly, guaranteeing the final linear bound on number of lines and quadratic bound in the number of symbols.

So all we have to do is focus in the case where cut is eliminated from a contracted formula

$$\dfrac{\dfrac{\begin{array}{c}\Pi_1\\ \Gamma_1, A, A \vdash \Delta_1\end{array}}{\Gamma_1, A \vdash \Delta_1} \qquad \begin{array}{c}\Pi_2\\ \Gamma_2 \vdash A, \Delta_2\end{array}}{\Gamma_1, \Gamma_2 \vdash \Delta_1, \Delta_2}\,(\text{Cut})$$

generating the following configuration, with the duplication of branch Π_2:

$$\dfrac{\dfrac{\dfrac{\begin{array}{c}\Pi_1\\ \Gamma_1, A, A \vdash \Delta_1\end{array} \qquad \begin{array}{c}\Pi_2\\ \Gamma_2 \vdash A, \Delta_2\end{array}}{\Gamma_1, \Gamma_2, A \vdash \Delta_1, \Delta_2}\,(\text{Cut}) \qquad \begin{array}{c}\Pi_2\\ \Gamma_2 \vdash A, \Delta_2\end{array}}{\Gamma_1, \Gamma_2, \Gamma_2 \vdash \Delta_1, \Delta_2, \Delta_2}\,(\text{Cut})}{\Gamma_1, \Gamma_2 \vdash \Delta_1, \Delta_2}\,(\text{Contractions})$$

However, with the application of a single substitution rule to $\Gamma_2 \vdash A, \Delta_2$,, with two defocusing applications of identity substitution ι, we can avoid the proof duplication that leads to the exponential explosion

$$\begin{array}{c}\Pi_2\\ \Gamma_2 \vdash A, \Delta_2\end{array}$$

$$\dfrac{\dfrac{\dfrac{\begin{array}{c}\Pi_1\\ \Gamma_1, A, A \vdash \Delta_1\end{array} \qquad \Gamma_2 \vdash A, \Delta_2}{\Gamma_1, \Gamma_2, A \vdash \Delta_1, \Delta_2}\,(\text{Cut}) \qquad \Gamma_2 \vdash A, \Delta_2}{\Gamma_1, \Gamma_2, \Gamma_2 \vdash \Delta_1, \Delta_2, \Delta_2}\,(\text{Cut})}{\Gamma_1, \Gamma_2 \vdash \Delta_1, \Delta_2}\,(\text{Contractions})$$

The proof chunk Π_2 is not repeated, the number of added lines is a constant and the number of added formulas is linearly bounded, just as in the connective cases. ■

Note that to replace cuts with substitution rules all we needed was the employment of identity substitutions. It is not known how to eliminate the use of substitution with the use of cuts. If this were possible, then it would follow that Frege systems can p-simulate extended Frege systems.

3 Tableaux with Substitution

Theorem 4 motivates us to explore a possible adaptation of the substitution rule to decision algorithms such as analytic tableaux. We follow Smullyan's presentation of tableaux dealing with *signed formulas*, in which formulas are prefixed with an F or T sign [Smullyan, 1968]. The signed formulas $F\ A$ and $T\ A$ are called *conjugates*. Any propositional valuation f is simply extended to signed formulas by making $f(T\ A) = 1$ iff $f(A) = 1$ and $f(F\ A) = 1$ iff $f(A) = 0$.

A tableau for a sequent $A_1, \dots A_n \vdash B_1, \dots, B_m$ is an attempt to refute it by asserting the antecedent with T-signed formulas $T\ A_1, \dots T\ A_n$ while denying the consequent with F-signed formulas $F\ B_1, \dots, F\ B_m$. The tableau's *expansion rules* will then expand the tableau into a tree of signed formulas. If every branch of that tree *closes*, than the initial sequent has been shown; otherwise, a *falsifying valuation* is obtained, which validates the sequent's antecedent and falsifies its consequent.

Signed formulas are classified into α, *neg* and β formulas, as indicated in Figure 5; it must be noted that Smullyan splits *neg*-formulas arbitrarily into α and β. The *expansion of a branch* consists in choosing a signed formula in that branch and then proceeding as follows. If it is an α formula, then add both conclusions α_1 and α_2 to the end of the branch. If it is a *neg* formula, add the *pos* formula to the end of the branch. If it is a β formula, split the branch by adding β_1 to one branch and β_2 to the other. The expansion rules are illustrated in Figure 6.

α	α_1	α_2
$T\ A \wedge B$	$T\ A$	$T\ B$
$F\ A \vee B$	$F\ A$	$F\ B$
$F\ A \to B$	$T\ A$	$F\ B$

neg	*pos*
$T\ \neg A$	$F\ A$
$F\ \neg A$	$T\ A$

β	β_1	β_2
$F\ A \wedge B$	$F\ A$	$F\ B$
$T\ A \vee B$	$T\ A$	$T\ B$
$T\ A \to B$	$F\ A$	$T\ B$

Figure 5. Smullyan's Notation for Signed Formulas

$$\frac{\alpha}{\begin{array}{c}\alpha_1\\ \alpha_2\end{array}} \qquad \frac{neg}{pos} \qquad \begin{array}{ccc} & \beta & \\ / & & \backslash \\ \beta_1 & & \beta_2 \end{array}$$

Figure 6. Tableau Expansion Rules

We consider a branch Θ as a set of formulas. A branch is *partially expanded* if some of its signed formulas, but not necessarily all, have been expanded. branch. In usual analytic tableaux, a branch *closes* if it contains a pair of conjugate formulas, meaning that a contradiction was reached on that branch. If all formulas in Θ have been expanded and the branch is not closed, than it is an *open*. The tableau closes if all its branches are closed; if a single branch is open the tableau is open.

Traditional analytic tableaux are always tree-like. By extending tableaux with substitution rules, one could expect us to transform its tree-like structure into a DAG. However, we do not do it. Instead, we add a new branch closure rule.

> **Substitution Closure Rule (s-closure):** If a tableau has partially expanded branches Θ_1 and Θ_2 such that there exists a substitution σ satisfying
>
> $$\Theta_1 \subseteq \Theta_2\sigma$$
>
> where by $\Theta_2\sigma$ we mean the set obtained by the application of σ to every signed formula in Θ_2. Then Θ_2 is closed.

The usual closure of a branch with $T\ A$ and $F\ A$ is a *simple closure*. We call a tableau extended with the substitution closure rule a *substitution tableau* (or an s-tableau).

In order to facilitate the application of the condition $\Theta_1 \subseteq \Theta_2\sigma$, it would be nice if we could increase the number of equivalent formulas that are also identical in their representation. For example, the formula $p_1 \wedge (p_2 \wedge p_3)$ is equivalent to $(p_1 \wedge p_2) \wedge p_3$ but not syntactically identical. The idea is thus to introduce a connective $\bigwedge$ that operate over a set of formulas, such that both formulas can be represented as $\bigwedge\{p_1, p_2, p_3\}$; similarly, we introduce $\bigvee$. By convention, $\bigwedge\{A\} = \bigvee\{A\} = A$, $\bigwedge\varnothing = \top$, $\bigvee\varnothing = \bot$. The transformation of maximal conjunctions and disjunctions into, respectively, $\bigwedge$-conjunctions and $\bigvee$-disjunctions is immediate. And the tableau rules for $F\bigwedge$ and $T\bigvee$ are, obviously, n-ary branching rules, and the tableau rules for $T\bigwedge$ and $F\bigvee$ are linear with multiple consequences. In the following we will assume that formulas are transformed into this set notation for large conjunctions and disjunctions without mentioning it.

An example of s-tableaux for a family of "hard" propositional formulas is given in Section 4. We now analyse some properties of s-tableaux. First, we note that completeness is trivial for s-tableaux are an extension of usual semantic tableaux and inherit its completeness. Soundness deserves more care.

LEMMA 5. *S-tableaux are sound and complete.*

Proof. Completeness follows immediately from the completeness of usual tableaux, as all usual tableau rules and closure conditions are present in s-tableaux.

For soundness, the tableaux rules maintain their usual property, namely if there is a valuation that satisfies α it also satisfies α_1 and α_2, if there is a valuation that satisfies *neg* it also satisfies *pos* and if there is a valuation that satisfies β it satisfies β_1 of β_2. In this case, if there is an open saturated branch, a valuation can be constructed that falsifies the original sequent; if there is a closed branch in the usual way containing $T\ A$ and $F\ A$ for some A, then no valuation can satisfy all of the branches signed formulas, meaning that the expansion taken on that branch do not lead to a counter-valuation. It remains to be proved that closing a tableau with the substitution closure rule keeps the soundness of the process, that is, we are not closing a branch that has the possibility of becoming saturated open.

In fact, suppose there are partially expanded branches Θ_1 and Θ_2 such that there exists a substitution σ satisfying $\Theta_1 \subseteq \Theta_2\sigma$, so that Θ_2 is closed. It is easy to see that, inductively, every expansion of a signed formula $X\varphi$ in Θ_1 can be mimicked by an expansion of $X\varphi\sigma$ in Θ_2. So if Θ_1 closes due to $T\ A, F\ A\Theta_1$ for some A, Θ_2 will also close due to $T\ A\sigma, F\ A\sigma \in \Theta_2$. On the other hand, if Θ_1 becomes saturated open, the sequent is not provable anyway. Thus, in both cases, the closure of Θ_2 preserves the soundness of the inference. ■

We also have the direct correspondence between closed s-tableaux and DAG sequent proofs with substitution. For a tableau $\mathcal{T}$, we define $|\mathcal{T}|$ as the total number of signed formulas (lines) occurring in it, and $\|\mathcal{T}\|$ as the total number symbols occurring in it.

LEMMA 6. *For every closed s-tableau $\mathcal{T}$ there corresponds a DAG s-sequent proof Π such that $|\Pi|$ is linear with $|\mathcal{T}|$ and $\|\Pi\|$ is quadratic with $\|\mathcal{T}\|$.*

Proof. (Sketch) We apply the usual transformation of tableau proofs into sequent proofs, that is, we transform analytic tableaux into *block tableau*;

see [Smullyan, 1968]. In case a branch Θ_2 is closed due to the substitution closing rule, there is a branch Θ_1' whose prefix Θ_1 is a partially expanded branch verifying $\Theta_1 \subseteq \Theta_2\sigma$. Consider three sub branches: the partially expanded "joint" branch $\Theta_1 \cap \Theta_2$, the "left" expansion $\Theta_1 \setminus \Theta_2$ and the "right" expansion that closes $\Theta_2 \setminus \Theta_1$. The corresponding last elements of the left and right branches correspond to sequents S and $S\sigma$, which become the receiving ends of an application of an s-rule:

$$\begin{tikzcd}
& S \arrow[dl, "\iota"'] \arrow[dr, "\sigma"] & \\
S \arrow[dr, dotted] & & S\sigma \arrow[dl, dotted] \\
& S_\beta &
\end{tikzcd}$$

The formula over which a β-expansion was applied to generate Θ_1 and Θ_2 becomes the sequent S_β, which is the point where the two branches rejoin. The rest of the details can be easily filled in, so it is omitted. ■

4 Example: The Pigeon Hole Problem

The Pigeon Hole Problem (PHP) is a notoriously famous hard case for theorem provers. An initial polynomial-size proof for extended Frege systems for PHP_n was given by [Cook and Reckhow, 1979], and the existence of a polynomial size Frege proof for PHP_n was shown by [Buss, 1987].

The Pigeon Hole Principle of size n (PHP_n) states that if there are $n+1$ pigeons to be placed at n holes, at least one hole will get more than one pigeon. Pigeons are numbered from 1 to $n+1$, holes are numbered from 1 to n, and the fact that pigeon i is placed in hole j is coded by the atomic symbol p_{ij}. These are the only atomic symbols employed, hence there are $n(n+1)$ atomic symbols.

This situation is encoded with a sequent $\Gamma_n \vdash \Delta_n$, where Γ_n expresses that every pigeon goes to a hole, and Δ_n expresses that there is a hole with at least two pigeons. In this way, Γ_n encodes that, for each of the $n+1$ pigeons, it is placed in one of the n holes, that is:

$$\Gamma_n = \left\{ \bigvee_{j=1}^{n} p_{ij} | 1 \leq i \leq n+1 \right\}$$

and Δ_n encodes that for some of the n holes, there are two distinct pigeons placed at it, that is:

$$\Delta_n = \{ p_{kj} \wedge p_{ij} | 1 \leq j \leq n, 1 \leq k < i \leq n+1 \}.$$

An initial tableau for PHP_n is constructed by T-signing all formulas in Γ_n and F-signing all formulas in Δ_n. The size of Γ_n is $O(n^2)$ and the size of Δ_n is $O(n^3)$, so the size of a PHP_n sequent is $O(n^3)$.

The big symmetries fond in the PHP problems have been pointed as the cause of its high complexity, for all pigeons and all holes “look the same”. It is this very symmetry that is exploited to generate a small s-tableau proof. Note that formulas in Γ_n look like a $(n+1) \times n$ matrix, where each line correspond to a pigeon i and each column correspond to a hole j:

$$\begin{array}{rccccccc}
\Gamma_n = & p_{11} & \vee & p_{12} & \vee & \ldots & \vee & p_{1n}, \\
& \vdots & & & & & & \vdots \\
& p_{n+1,1} & \vee & p_{n+1,2} & \vee & \ldots & \vee & p_{n+1,n}
\end{array}$$

which evidences that if we swap lines i' and i'' (that is, if we apply the substitution $\sigma = [i' := i'', i'' := i']$) Γ_n *remains the same*, and similarly, if we swap columns j' and j'' Γ_n also remains the same. It is perhaps harder to see, but no less true, that the same holds for the formulas of Δ_n, that is, if we swap i' with i'', or j' with j'', in all formulas of Δ_n, Δ_n remains the same. Thus, the symmetry of PHP can be expressed by the following.

LEMMA 7. *Let* $\sigma_i = [i' := i'', i'' := i']$ *and* $\sigma_j = [j' := j'', j'' := j']$. *Then* $\Gamma_n\sigma_i = \Gamma_n$, $\Gamma_n\sigma_j = \Gamma_n$, $\Delta_n\sigma_i = \Delta_n$ *and* $\Delta_n\sigma_j = \Delta_n$.

We can then start expanding the tableau with an n-branch over the $(n+1)$st (ie, the last) line of Γ_n, as illustrated below.

$$\begin{array}{cccc}
 & T\ \Gamma_n & & \\
 & F\ \Delta_n & & \\
T\ p_{n+1,1} & T\ p_{n+1,2} & \cdots & T\ p_{n+1,n} \\
\times_\sigma & \times_\sigma & \cdots &
\end{array}$$

By Lemma 7 there is a substitution that, for any pair of partially expanded branches, transforms one branch into the other. We choose to map every branch into the last one, so that all the first $n-1$ branches are s-closed, which is indicated above by the symbol $\times_\sigma$. Let the branch containing $T\ p_{n+1,n}$ be the *main* branch.

We then concentrate on all the n formulas in Δ_n of the form $F\ p_{i,n} \wedge p_{n+1,n}$, $1 \leq i \leq n$. We branch each of these formulas, so that the branch containing $F\ p_{n+1,n}$ will simply close due to the presence of $T\ p_{n+1,n}$, and $F\ p_{i,n}$ is added to the main branch, for $1 \leq i \leq n$. We next consider the first n lines of Γ_n with a branch that generates $T\ p_{i1} \vee p_{i2} \vee \ldots \vee p_{i,n-1}$ and

$T\ p_{in}$. Clearly, the branch containing $T\ p_{i,n}$ will simply close due to the presence of $F\ p_{i,n}$ in the main branch, so that we have added to the main branch n formulas of the form, $T\ p_{i1} \vee p_{i2} \vee \ldots \vee p_{i,n-1}$, $1 \leq i \leq n$, which consists of Γ_{n-1}. By noting that $\Delta_{n-1} \subset \Delta_n$, we have shown how to reduce $\Gamma_n \vdash \Delta_n$ into $\Gamma_{n-1} \vdash \Delta_{n-1}$. We illustrated below the main branch.

$$\begin{array}{c}
T\ \Gamma_n \\
F\ \Delta_n \\
T\ p_{n+1,n} \\
F\ p_{1,n} \\
\vdots \\
F\ p_{n,n} \\
T\ \Gamma_{n-1} \\
F\ \Delta_{n-1} \text{ by copying from above}
\end{array}$$

Note that in this process of reducing PHP_n to PHP_{n-1}, we have used $O(n)$ formulas (lines) and $O(n^2)$ symbols. If we repeat this process n times we end up with $\Gamma_1 \vdash \Delta_1$, which clearly closes. We have thus shown the following.

THEOREM 8. *There is an s-tableau $\mathcal{T}$ for PHP_n such that $|\mathcal{T}| = O(n^2)$ and $\|\mathcal{T}\| = O(n^3)$.*

The number of atoms in PHP_n is $O(n^2)$ and the number of symbols in PHP_n is $O(n^3)$, so we have a proof for PHP_n whose size in number of formulas is linear in the number of atoms of the input sequent, and whose size in number of symbols is linear in the number of symbols of the input sequent.[1]

There are two interesting points from the proof above we would like to highlight. First, that all the substitutions used in the s-closure of the branches are actually variable renamings. Second, that all the application of s-closure starts with the identification of a set of substitutions *in the original problem* that make the problem invariant, that is, that map the problem into itself. It is the presence of this substitution invariance that allows one to look for substitutions for the application of the s-closure rule. This seems to indicate a way of applying s-closure in practice, that is, the identification of the invariant substitution and the search for adequate substitutions after branching. The problem is that identifying the existence of an initial substitution seems to be as hard as theorem proving itself.

[1]Without the use of the s-closure rule, the corresponding semantic tableau $\mathcal{T}'$ would be such that $|\mathcal{T}'| = O(n!)$ and $\|\mathcal{T}'\| = O(n!)$.

5 Conclusion

We have defined in this paper an extension of classical sequent proofs with a substitution rule and a DAG proof structure, and we have shown how this technique can be transposed to semantic tableaux. This technique can be generalised in several directions.

First, with respect to tableaux, there is nothing particular to semantic tableaux that we have used, and other forms of tableaux can be extended with an s-closure rule, such as KE tableaux.

Second, with respect to sequent proofs, the techniques explored in this paper are not restricted to classical propositional logic and can be directly applied to extensions of propositional logic such as: modal logics, temporal logics, description logics and first-order logic. It may be even possible to apply those techniques to non-classical logics that possess the uniform substitution property, such as most substructural logics [Restall, 2000, Bull and Segerberg, 1984, Dalen, 1984, Girard, 1987a].

It remains an open problem whether Frege proof systems can p-simulate extended Frege proof systems. In the current setting, this problem can be formulated as the search for a systematic way in which the use of the substitution rule can be simulated in ordinary sequent calculus by means of the cut rule.

BIBLIOGRAPHY

[Boolos, 1984] George Boolos. Don't eliminate cut. *Journal of Philosophical Logic*, 13:373–378, 1984.

[Bull and Segerberg, 1984] R. Bull and K. Segerberg. Basic Modal Logic. In D. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic*, volume II, pages 1–88. D. Reidel Publishing Company, 1984.

[Buss, 1987] Samuel Buss. Polynomial size proofs of the propositional pigeonhole principle. *Journal of Symbolic Logic*, 52:916–927, 1987.

[Buss, 1995] Samuel R. Buss. Some remarks on lengths of propositional proofs. *Archive for Mathematic Logic*, 34:377–394, 1995.

[Carbone and Semmes, 2000] Alessandra Carbone and Stephen Semmes. *A Graphic Apology for Symmetry and Implicitness.* Oxford Mathematical Monographs. Oxford University Press, 2000.

[Carbone, 1997] A. Carbone. Interpolants, cut elimination and flow graphs for the propositional calculus. *Annals of Pure and Applied Logic*, 83:249–299, 1997.

[Cook and Reckhow, 1979] S. A. Cook and R. A. Reckhow. The relative efficiency of propositional proof systems. *Journal of Symbolic Logic*, 44:36–50, 1979.

[Dalen, 1984] Dirk Van Dalen. Intuitionistic logic. In D. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic*, volume III, 1984.

[Finger and Gabbay, 2005] Marcelo Finger and Dov Gabbay. Cut and pay. Manuscript, 2005.

[Girard, 1987a] J.-Y. Girard. Linear logic. *Theoretical Computer Science*, 50:1–102, 1987.

[Girard, 1987b] J.-Y. Girard. *Proof Theory and Logical Complexity*, volume volume 1 of *Studies in Proof Theory. Monographs.* Bibliopolis, 1987.

[Krajíček and Pudlák, 1989] Jan Krajíček and Pavel Pudlák. Propositional proof systems, the consistency of first order theories and the complexity of computations. *Journal of Symbolic Logic*, 54(3):1063–1079, 1989.

[Orevkov, 1982] V. Orevkov. Lower bounds for increasing complexity of derivations after cut elimination. *Journal of Soviet Mathematics*, 20(4):2337–2350, 1982.

[Restall, 2000] G. Restall. *An Introduction to Substructural Logics*. Routledge, 2000.

[Smullyan, 1968] Raymond M. Smullyan. *First-Order Logic*. Springer-Verlag, 1968.

[Statman, 1979] R. Statman. Lower bounds on herbrand's theorem. *Proceedings of the American Mathematical Society*, 75(1):104–107, June 1979.

[Takeuti, 1987] G. Takeuti. *Proof Theory*. North Holland, second edition edition, 1987.