A First Order System with Finite Choice of Premises

Kai Brünnler
kai.bruennler@gmx.net
Technische Universität Dresden, Fakultät Informatik, 01062 Dresden, Germany
and
Institut für Informatik und angewandte Mathematik, Fachgruppe für theoretische Informatik und Logik, Neubrückstr. 10, CH–3012 Bern, Switzerland
Alessio Guglielmi
alessio.guglielmi@inf.tu-dresden.de
Technische Universität Dresden, Fakultät Informatik, 01062 Dresden, Germany

1 Introduction

We call *finitely generating* an inference rule in a sequent system [7] if, given its conclusion, there is only a finite set of premises to choose from. This property is desirable from the viewpoint of proof search, since it implies that the search tree is finitely branching. It is also desirable for showing consistency, since the biggest obstacle to showing consistency is the cut rule, which is not finitely generating.

Much effort has been devoted to eliminating the cut rule in various systems: theorems of cut elimination are at the core of proof theory. In addition to the cut, there is another source of infinite choice in the bottom-up construction of a first order proof, namely the choice in instantiating an existentially quantified variable. Research grounded in Herbrand's theorem [11] deals with this aspect and is at the core of automated deduction and logic programming.

Hendricks et al. (eds.):
First-Order Logic Revisited
Logos Verlag Berlin (2004), 1–16

Implementations of sequent systems for first order predicate logic are usually based on rules that are finitely generating. Such rules can be obtained for example as follows: first prove cut elimination to get rid of the cut rule, then use unification instead of blindly guessing instantiations.

This paper shows how one can eliminate all sources of infinite choice in a system of first order classical logic in a much simpler way, in particular without the use of cut elimination. To do so we use the *calculus of structures* [9], a formalism based on *deep inference*, which is the possibility of applying inference rules deep inside formulae. Systems in the calculus of structures offer the same proof theoretical properties as systems in the sequent calculus, in particular it is possible to prove cut elimination and many other normalisation results [9, 3, 1, 2]. The point here is that it is possible to obtain finitely generating systems without having to use these complex methods.

The main idea we exploit is that there are actually two sources of infinite choice in the cut rule: an infinite choice of atoms and an infinite choice in how these atoms can be combined for making formulae. Deep inference allows us to separate these two kinds of infinite choice. First we reduce the cut rule to atomic form, which is immediate thanks to deep inference. Then we eliminate only those cuts which (seen bottom-up) introduce atoms that do not occur in the conclusion. This is much simpler than a full-blown cut elimination. The instances of cut that we retain introduce atoms that already occur in the conclusion, so they are finitely generating.

Just like the cut rule, the rule for existential quantification also has two sources of infinite choice: an infinite choice of function symbols and an infinite choice in how these function symbols can be combined for making terms. We eliminate them by using the same technique that we used for the cut: first we reduce the instantiation rule to a form which instantiates only with one function symbol at a time. Then we eliminate those instances which introduce function symbols that do not occur in the conclusion. We are left with a finitely generating instantiation rule.

In the sequent calculus, which restricts the application of inference rules to the main connective of a formula, it is impossible to eliminate infinite choice in such a simple manner. First, the cut rule can not immediately be reduced to atomic form: one has to use cut elimination for that. Second, the rule for instantiating existentially quantified variables can not be restricted to instantiating with only one function symbol. The adoption of deep inference instead allows this.

When proving in the system we obtain, the only infinity that remains is

in the unboundedness of the proofs themselves, every other aspect in proof construction is finite: at any given step, there are finitely many inferences possible, and each inference rule can only be applied in a finite number of different ways. Also, the consistency of the system is easily shown.

The point we make in this paper is not so much the existence of the finitely generating system that we show, but the simplicity of the techniques that are used to obtain it, which are purely syntactic and much less complex than cut elimination.

The notion of a finitely generating inference rule is closely related to that of an *analytic* rule, cf. Smullyan [12]. An analytic rule is one that obeys the subformula property. We tend to think of the notion of being finitely generating as a more general, weaker subformula property: there are interesting rules that are finitely generating but do not obey the subformula property, for example in system GS4ip, cf. Dyckhoff [6]. However, not all analytic rules are finitely generating, as witnessed by the existential-right rule. This is due to the fact that analyticity is defined with respect to the notion of Gentzen subformula (where instances of subformulae count as subformulae), rather than the literal notion of subformula.

In previous work, Brünnler and Tiu proved that classical logic can be presented in the calculus of structures in such a way that applying a rule only requires a bounded effort [1, 3]. This paper improves on that result by bounding choice.

In Section 2 we introduce first order logic in the calculus of structures and in Section 3 we show how to eliminate infinitely generating rules and we show the consistency argument.

2 First Order Logic in the Calculus of Structures

Variables are denoted by x and y. *Terms*, denoted by τ, are defined as usual in first-order predicate logic. *Atoms*, denoted by a, b, etc., are expressions of the form $p(\tau_1, \ldots, \tau_n)$, where p is a *predicate symbol* of *arity* n and $\tau_1, \ldots, \tau_n$ are terms. The negation of an atom is again an atom.

The *structures* of the language KSq are generated by

$$S ::= \mathsf{f} \mid \mathsf{t} \mid a \mid [\underbrace{S, \ldots, S}_{>0}] \mid (\underbrace{S, \ldots, S}_{>0}) \mid \exists x S \mid \forall x S \mid \bar{S} \quad .$$

where t and f are the units *true* and *false*, $[S_1, \ldots, S_h]$ is a *disjunction*, $(S_1, \ldots, S_h)$ is a *conjunction*, $\exists$ is the *existential quantifier* and $\forall$ is the

Associativity

$$[\vec{R}, [\vec{T}], \vec{U}] = [\vec{R}, \vec{T}, \vec{U}]$$
$$(\vec{R}, (\vec{T}), \vec{U}) = (\vec{R}, \vec{T}, \vec{U})$$

Commutativity

$$[R, T] = [T, R]$$
$$(R, T) = (T, R)$$

Units

$$(\mathsf{f}, \mathsf{f}) = \mathsf{f} \qquad [\mathsf{f}, R] = R$$
$$[\mathsf{t}, \mathsf{t}] = \mathsf{t} \qquad (\mathsf{t}, R) = R$$

Negation

$$\overline{\mathsf{f}} = \mathsf{t}$$
$$\overline{\mathsf{t}} = \mathsf{f}$$
$$\overline{[R, T]} = (\bar{R}, \bar{T})$$
$$\overline{(R, T)} = [\bar{R}, \bar{T}]$$
$$\overline{\exists x R} = \forall x \bar{R}$$
$$\overline{\forall x R} = \exists x \bar{R}$$
$$\bar{\bar{R}} = R$$

Context Closure

$$\text{if} \quad R = T \quad \text{then} \quad \begin{array}{c} S\{R\} = S\{T\} \\ \bar{R} = \bar{T} \end{array}$$

Vacuous Quantifier

if y is not free in R then $\forall y R = \exists y R = R$

Variable Renaming

$$\text{if } y \text{ is not free in } R \quad \text{then} \quad \begin{array}{l} \forall x R = \forall y R[x/y] \\ \exists x R = \exists y R[x/y] \end{array}$$

Figure 1.1: Syntactic equivalence of structures

universal quantifier. $\bar{S}$ is the *negation* of the structure S. The units are not atoms. Structures are denoted by S, R, T, U and V. *Structure contexts*, denoted by $S\{\ \}$, are structures with one occurrence of $\{\ \}$, the *empty context* or *hole*, that does not appear in the scope of a negation. $S\{R\}$ denotes the structure obtained by filling the hole in $S\{\ \}$ with R. We drop the curly braces when they are redundant: for example, $S[R, T]$ stands for $S\{[R, T]\}$. Structures are *equivalent* modulo the smallest equivalence relation induced by the axioms shown in Fig. 1.1, where $\vec{R}$ and $\vec{T}$ are finite, non-empty sequences of structures. In general we do not distinguish between equivalent structures.

An *inference rule* is a scheme of the kind $\rho\,\dfrac{S\{T\}}{S\{R\}}$, where ρ is the *name* of the rule, $S\{T\}$ is its *premise* and $S\{R\}$ is its *conclusion*. A (*formal*) *system* $\mathscr{S}$ is a set of inference rules. The *dual* of a rule is obtained by exchanging premise and conclusion and replacing each connective by its De Morgan dual.

A *derivation* Δ is a finite chain of instances of inference rules:

$$\begin{array}{l}\pi'\,\dfrac{T}{V}\\ \pi\,\dfrac{}{\vdots}\\ \rho'\,\dfrac{}{U}\\ \rho\,\dfrac{}{R}\end{array} \quad .$$

A derivation can consist of just one structure. The topmost structure in a derivation is called the *premise* of the derivation, and the structure at the bottom is called its *conclusion*. A derivation Δ whose premise is T, whose conclusion is R, and whose inference rules are in $\mathscr{S}$ will be indicated with $\begin{array}{l}T\\ \Delta\!\parallel_{\mathscr{S}}\\ R\end{array}$. A *proof* Π in the calculus of structures is a derivation whose premise is the unit true. It will be denoted by $\begin{array}{l}\Pi\!\!\top\!\!\!\parallel_{\mathscr{S}}\\ R\end{array}$. A rule ρ is *derivable* for a system $\mathscr{S}$ if for every instance of $\rho\,\dfrac{T}{R}$ there is a derivation $\begin{array}{l}T\\ \parallel_{\mathscr{S}}\\ R\end{array}$. A rule ρ is *admissible* for a system $\mathscr{S}$ if for every proof $\begin{array}{l}\top\!\!\!\parallel_{\mathscr{S}\cup\{\rho\}}\\ S\end{array}$ there is a proof $\begin{array}{l}\top\!\!\!\parallel_{\mathscr{S}}\\ S\end{array}$.

Besides deep inference, the calculus of structures employs a notion of top-down symmetry for derivations. Symmetry makes possible to reduce the cut rule to its atomic form without performing cut elimination: this would be impossible by solely adopting deep inference. The *dual* of a derivation is obtained by flipping it upside-down, negating each structure in it, and replacing each rule by its dual.

System SKSgq, shown in Fig. 1.2, has been introduced and shown to be sound and complete for classical predicate logic in [1]. The first S stands for "symmetric" or "self-dual", meaning that for each rule, its dual (or contrapositive) is also in the system. The K stands for "klassisch" as in Gentzen's LK and the second S says that it is a system in the calculus of structures.

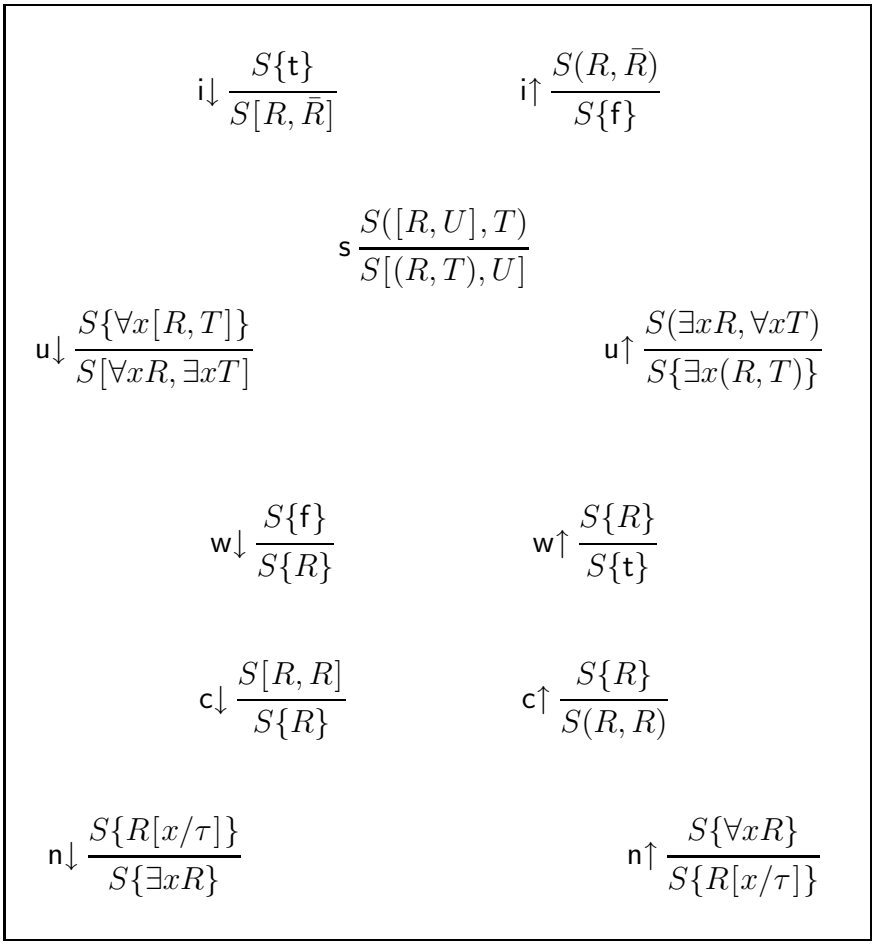

Figure 1.2: System SKSgq

The g is for "general" (as opposed to atomic) contraction. The q denotes (first-order) quantifiers.

The first and last column show the rules that deal with quantifiers, in the middle there are the rules for the propositional fragment. The propositional rules $\mathsf{i}\downarrow, \mathsf{s}, \mathsf{w}\downarrow$ and $\mathsf{c}\downarrow$ are called respectively *identity*, *switch*, *weakening* and *contraction*. The rule $\mathsf{u}\downarrow$ is called *universal*, because it roughly corresponds to the $\mathsf{R}\forall$ rule in sequent systems, while $\mathsf{n}\downarrow$ is called *instantiation*, because it corresponds to $\mathsf{R}\exists$.

In the sequent calculus, going up, the $\mathsf{R}\forall$ rule removes a universal quan-

tifier from a formula to allow other rules to access this formula. In system SKSgq, inference rules apply deep inside formulae, so there is no need to remove the quantifier. Note that the premise of the $\mathsf{u}{\downarrow}$ rule implies its conclusion, which is not true for the $\mathsf{R}\forall$ rule of the sequent calculus. In all rules of SKSgq the premise implies the conclusion.

As usual, the substitution operation in the rules $\mathsf{n}{\downarrow}$ and $\mathsf{n}{\uparrow}$ requires τ to be free for x in R: quantifiers in R do not capture variables in τ. The term τ is not required to be free for x in $S\{R\}$: quantifiers in S may capture variables in τ.

The dual of rule carries the same name prefixed with a "co-", so e.g. $\mathsf{w}{\uparrow}$ is called *co-weakening*. The rule s is self-dual. The rule $\mathsf{i}{\uparrow}$ is special, it is called *cut*. Rules with a name that contains an arrow pointing downward are called *down-rules* and their duals are called *up-rules*. The system enjoys cut elimination: all up-rules are admissible, as has been shown in [1].

Sequent calculus derivations easily correspond to derivations in system SKSgq. For instance, the cut of sequent systems in Gentzen-Schütte form [16]:

$$\mathsf{Cut}\,\frac{\vdash \Phi, A \quad \vdash \Psi, \bar{A}}{\vdash \Phi, \Psi} \qquad \text{corresponds to} \qquad \mathsf{i}{\uparrow}\,\frac{\mathsf{s}\,\dfrac{\mathsf{s}\,\dfrac{([\Phi, A], [\Psi, \bar{A}])}{[\Phi, (A, [\Psi, \bar{A}])]}}{[\Phi, \Psi, (A, \bar{A})]}}{[\Phi, \Psi]} \quad .$$

3 Eliminating Infinite Choice in Inference Rules

There are several rules with infinite choice in system SKSgq: the co-weakening, the cut, the instantiation and the co-instantiation rule. The equivalence on structures is infinitely generating as well: equivalence classes are infinite. In the following we will see for each of these rules and the equivalence how to to replace them by finitely generating rules without affecting provability.

3.1 The Co-weakening Rule

The rule $\mathsf{w}{\uparrow}$ is clearly infinitely generating, since there is an infinite choice of atoms, but it can immediately be eliminated by using a cut and an instance of $\mathsf{w}{\downarrow}$ as follows:

$$\mathsf{w}\uparrow \frac{S\{a\}}{S\{\mathsf{t}\}} \qquad \rightsquigarrow \qquad \begin{array}{c} \mathsf{s}\dfrac{S\{R\}}{S(R,[\mathsf{t},\mathsf{f}])} \\ \mathsf{s}\dfrac{S(R,[\mathsf{t},\mathsf{f}])}{S[\mathsf{t},(R,\mathsf{f})]} \\ \mathsf{w}\downarrow\dfrac{S[\mathsf{t},(R,\mathsf{f})]}{S[\mathsf{t},(R,\bar{R})]} \\ \mathsf{i}\uparrow\dfrac{S[\mathsf{t},(R,\bar{R})]}{S[\mathsf{t},\mathsf{f}]} \\ =\dfrac{S[\mathsf{t},\mathsf{f}]}{S\{\mathsf{t}\}} \end{array} .$$

3.2 The Cut Rule

The cut is the most prominent infinitely generating rule. The first source of infinite choice we will remove is the arbitrary size of the cut formula. To that end, consider the *atomic cut* rule:

$$\mathsf{ai}\uparrow \frac{S(a,\bar{a})}{S\{\mathsf{f}\}}$$

The following theorem, also proved in [1], allows us to restrict ourselves to atomic cuts.

Theorem 1. *The rule* $\mathsf{i}\uparrow$ *is derivable for* $\{\mathsf{ai}\uparrow,\mathsf{s},\mathsf{u}\uparrow\}$.

Proof. By an easy structural induction on the structure that is cut. A cut introducing the structure (R,T) together with its dual structure $[\bar{R},\bar{T}]$ is replaced by two cuts on smaller structures:

$$\mathsf{i}\uparrow \frac{S(R,T,[\bar{R},\bar{T}])}{S\{\mathsf{f}\}} \qquad \rightsquigarrow \qquad \begin{array}{c} \mathsf{s}\dfrac{S(R,T,[\bar{R},\bar{T}])}{S(R,[\bar{R},(T,\bar{T})])} \\ \mathsf{s}\dfrac{S(R,[\bar{R},(T,\bar{T})])}{S[(R,\bar{R}),(T,\bar{T})]} \\ \mathsf{i}\uparrow\dfrac{S[(R,\bar{R}),(T,\bar{T})]}{S(R,\bar{R})} \\ \mathsf{i}\uparrow\dfrac{S(R,\bar{R})}{S\{\mathsf{f}\}} \end{array} .$$

A cut introducing the structure $\forall x R$ together with its dual structure $\exists x \bar{R}$ is replaced by a cut inside an existential quantifier followed by an instance of $\mathsf{u}\uparrow$:

$$\mathsf{i}\uparrow \frac{S(\forall x R, \exists x \bar{R})}{S\{\mathsf{f}\}} \quad \rightsquigarrow \quad \begin{array}{c} \mathsf{u}\uparrow \dfrac{S(\forall x R, \exists x \bar{R})}{S\{\exists x (R, \bar{R})\}} \\ \mathsf{i}\uparrow \dfrac{}{= \dfrac{S\{\exists x \mathsf{f}\}}{S\{\mathsf{f}\}}} \end{array} .$$

These reductions can be repeated until all cuts are atomic. □

The rule $\mathsf{ai}\uparrow$ still is infinitely generating, since there is an infinite choice of atoms. Let us take a closer look at the atoms:

$$\mathsf{ai}\uparrow \frac{S(p(\tau_1, \ldots, \tau_n), \overline{p(\tau_1, \ldots, \tau_n)})}{S\{\mathsf{f}\}} .$$

There are both an infinite choice of predicate symbols p and an infinite choice of terms for each argument of p. Let $\vec{\tau}$ denote $\tau_1, \ldots, \tau_n$ and $\vec{x}$ denote $x_1, \ldots, x_n$. Since cuts can be applied inside existential quantifiers, we can delegate the choice of terms to a sequence of $\mathsf{n}\downarrow$ instances:

$$\mathsf{ai}\uparrow \frac{S(p(\vec{\tau}), \overline{p(\vec{\tau})})}{S\{\mathsf{f}\}} \quad \rightsquigarrow \quad \begin{array}{c} \mathsf{n}\downarrow^n \dfrac{S(p(\vec{\tau}), \overline{p(\vec{\tau})})}{S\{\exists \vec{x} (p(\vec{x}), \overline{p(\vec{x})})\}} \\ \mathsf{ai}\uparrow \dfrac{}{= \dfrac{S\{\exists \vec{x} \mathsf{f}\}}{S\{\mathsf{f}\}}} \end{array} .$$

The remaining cuts are restricted in that they do not introduce arbitray terms but just existential variables. Let us call this restricted form $\mathsf{vai}\uparrow$:

$$\mathsf{vai}\uparrow \frac{S(p(\vec{x}), \overline{p(\vec{x})})}{S\{\mathsf{f}\}} .$$

The only infinite choice that remains is the one of the predicate symbol p. To remove it, consider the rule *finitely generating atomic cut*

$$\mathsf{fai}\uparrow \frac{S(p(\vec{x}), \overline{p(\vec{x})})}{S\{\mathsf{f}\}} \quad \text{where } p \text{ appears in the conclusion.}$$

This rule is finitely generating, and we will show that we can easily transform a proof into one where the only cuts that appear are $\mathsf{fai}\uparrow$ instances.

Take a proof in the system we obtained so far, that is SKSgq without $\mathsf{w}\uparrow$, and with $\mathsf{vai}\uparrow$ instead of $\mathsf{i}\uparrow$. Individuate the bottommost instance of $\mathsf{vai}\downarrow$ that violates the proviso of $\mathsf{fai}\uparrow$:

$$\mathsf{vai}\uparrow \frac{\begin{array}{c}\vdots\\ S(p(\vec{x}), \overline{p(\vec{x})})\end{array}}{S\{\mathsf{f}\}} \quad ,$$

where p does not appear in $S\{\mathsf{f}\}$. We can then replace all instances of $p(\vec{x})$ and $\overline{p(\vec{x})}$ in the proof above the cut with t and f, respectively, to obtain a proof of $S\{\mathsf{f}\}$. It is easy to check that all rule instances stay valid or become trivial; the cut

$$\mathsf{vai}\uparrow \frac{S(\mathsf{t},\mathsf{f})}{S\{\mathsf{f}\}} \quad ,$$

can just be removed, since $(\mathsf{t},\mathsf{f}) = \mathsf{f}$.

Please notice that if p appeared in $S\{\mathsf{f}\}$, then this would not work, because it could destroy the rule instance below $S\{\mathsf{f}\}$.

Proceeding inductively upwards, we remove all infinitely generating atomic cuts.

3.3 The Instantiation Rule

The same techniques also work for instantiation. Consider these two restricted versions of $\mathsf{n}\downarrow$:

$$\mathsf{n}\downarrow_1 \frac{S\{R[x/f(\vec{x})]\}}{S\{\exists x R\}} \qquad \text{and} \qquad \mathsf{n}\downarrow_2 \frac{S\{R[x/y]\}}{S\{\exists x R\}} \quad .$$

An instance of $\mathsf{n}\downarrow$ that is not an instance of $\mathsf{n}\downarrow_2$ can inductively be replaced by instances of $\mathsf{n}\downarrow_1$ (chose variables for $\vec{x}$ that are not free in R):

$$\mathsf{n}\downarrow \frac{S\{R[x/f(\vec{\tau})]\}}{S\{\exists x R\}} \quad \rightsquigarrow \quad \begin{array}{c} \mathsf{n}\downarrow^n \dfrac{S\{R[x/f(\vec{\tau})]\}}{S\{\exists \vec{x} R[x/f(\vec{x})]\}} \\ \mathsf{n}\downarrow_1 \dfrac{}{= \dfrac{S\{\exists \vec{x} \exists x R\}}{S\{\exists x R\}}} \end{array} \quad .$$

This process can be repeated until all instances of $\mathsf{n}\!\downarrow$ are either instances of $\mathsf{n}\!\downarrow_1$ or $\mathsf{n}\!\downarrow_2$.

Now consider the following rules, which are finitely generating

$$\mathsf{fn}\!\downarrow_1 \frac{S\{R[x/f(\vec{x})]\}}{S\{\exists x R\}} \qquad \text{and} \qquad \mathsf{fn}\!\downarrow_2 \frac{S\{R[x/y]\}}{S\{\exists x R\}} \quad .$$

where $\mathsf{fn}\!\downarrow_1$ carries the proviso that the function symbol f either occurs in the conclusion or is a fixed constant c, and $\mathsf{fn}\!\downarrow_2$ carries the proviso that the variable y appears in the conclusion (no matter whether free or bound or in a vacuous quantifier).

Infinitely generating instances of $\mathsf{n}\!\downarrow_1$ and $\mathsf{n}\!\downarrow_2$, i.e. those that are not instances of $\mathsf{fn}\!\downarrow_1$ and $\mathsf{fn}\!\downarrow_2$, respectively, are easily replaced by instances of finitely generating rules similarly to how the infinitely generating cut was eliminated. Take the constant symbol c that is fixed in the proviso of $\mathsf{fn}\!\downarrow_1$, and throughout the proof above an infinitely generating instance of $\mathsf{n}\!\downarrow_1$, replace all terms that are instances of $f(\vec{x})$ by c. For $\mathsf{n}\!\downarrow_2$ we do the same to all occurences of y, turning it into an instance of $\mathsf{fn}\!\downarrow_1$.

3.4 The Equivalence and the Co-instantiation Rule

The equivalence can be broken up into several rules, for associativity, commutativity, and so on. Those rules are clearly finitely generating except for variable renaming and vacuous quantifier, which, technically speaking, have an infinite choice of names for bound variables. The same goes for the co-instantiation rule. Of course these rules can be made finitely generating since the choice of names of bound variables does not matter. There is nothing deep in it: the only reason for us to tediously show this obvious fact is to avoid giving the impression that we hide infinity under the carpet of the equivalence. The need for the argument below just comes from a syntax which has infinitely many different objects for essentially the same thing, e.g. $\forall x p(x), \forall y p(y)$ and $\forall y \forall x p(x)$ .... If you are not concerned about this 'infinite' choice of names of bound variables, then please feel invited to skip ahead to the next section.

Consider the following rules for variable renaming and vacuous quantifier, they all carry the proviso that x is not free in R:

$$\alpha\downarrow \frac{S\{\forall xR[y/x]\}}{S\{\forall yR\}} \qquad \alpha\uparrow \frac{S\{\exists xR[y/x]\}}{S\{\exists yR\}}$$

$$\mathsf{v}\downarrow \frac{S\{\exists xR\}}{S\{R\}} \qquad \mathsf{v}\uparrow \frac{S\{R\}}{S\{\forall xR\}}$$

Let us now consider proofs in the system that is obtained from SKSq by restricting the equivalence rule to not include vacuous quantifier and variable renaming and by adding the above rules. This system is strongly equivalent to SKSq as can easily be checked.

The rule $\mathsf{v}\uparrow$ is clearly finitely generating. Let us see how to replace the rule $\mathsf{v}\downarrow$ by finitely generating rules, the same technique also works for the rules $\alpha\uparrow$ and $\alpha\downarrow$. Consider the finitely generating rule $\mathsf{fv}\downarrow_1$, which is $\mathsf{v}\downarrow$ with the added proviso that x occurs in the conclusion (no matter whether bound or free or in a vacuous quantifier) and the infinitely generating rule $\mathsf{v}\downarrow'$ which is $\mathsf{v}\downarrow$ with the proviso that x does not occur in the conclusion.

Fix a total order on variables. Let $\mathsf{fv}\downarrow_2$ be $\mathsf{v}\downarrow$ with the proviso that x is the lowest variable in the order that does not occur in the conclusion. This rule is clearly finitely generating: there is no choice.

Each instance of $\mathsf{v}\downarrow$ is either an instance of $\mathsf{fv}\downarrow_1$ or of $\mathsf{v}\downarrow'$. In a given proof, all instances of $\mathsf{v}\downarrow'$ can be replaced by instances of $\mathsf{fv}\downarrow_2$ as follows, as we we will see now. Starting from the conclusion, going up in the proof, identify the first infinitely generating vacuous quantifier rule:

$$\begin{array}{c} \vdots \\ \mathsf{v}\downarrow' \dfrac{S\{\exists xR\}}{S\{R\}} \quad x \text{ does not occur in } S\{R\} \\ \vdots \\ T \end{array} ,$$

where x is not the lowest in our fixed order that does not occur in the conclusion. Let y be the lowest variable that does not occur in the conclusion. Now, throughout the proof above, do the following:

1. Choose a variable z that does not occur in the proof. Replace y by z.
2. Replace x by y.

By definition neither x nor y occur in the conclusion, so the conclusion is not broken. All the replacements respect that variable occurrences with different names stay different and variable occurrences with the same names stay the same. So the proof above stays intact. Replace the $\mathsf{v}{\downarrow'}$ instance by a $\mathsf{fv}{\downarrow_2}$ instance and proceed inductively upwards.

4 A Finitely Generating System for Predicate Logic

We now define the finitely generating system FKSgq to be

$$(\mathsf{SKSgq} \setminus \{\mathsf{i}{\uparrow}, \mathsf{w}{\uparrow}, \mathsf{n}{\downarrow}\}) \cup \{\mathsf{fai}{\uparrow}, \mathsf{fn}{\downarrow_1}, \mathsf{fn}{\downarrow_2}\} \quad ,$$

and, for what we showed in the previous section, state

Lemma 1. *Each structure is provable in system* SKSgq *if and only it is provable in system* FKSgq*.*

To put the finitely generating system at work, we use it to show consistency of system SKSgq. Of course, for this purpose it suffices to have finitely generating cut. Having infinite choice in instantiation would not affect the following argument.

Lemma 2. *The unit* f *is not provable in system* FKSgq*.*

Proof. No atoms, but only f, t and vacuous quantifiers can appear in such a proof. It is easy to show that f is not equivalent to t. Then we show that no rule can have a premise equivalent to t and a conclusion equivalent to f. This is simply done by inspection of all the rules in FKSgq. □

From the two lemmas above we immediately get consistency:

Theorem 2. *The unit* f *is not provable in system* SKSgq*.*

Now we can make use of symmetry by flipping derivations:

Theorem 3. *For all structures R, if there is a proof of R then there is no proof of $\bar{R}$.*

Proof. We assume that we have both proofs:

$$\begin{array}{c} \top \\ \| \\ R \end{array} \quad \text{and} \quad \begin{array}{c} \top \\ \| \\ \bar{R} \end{array} \quad .$$

Dualise the proof of R to get

$$\begin{array}{c}\bar{R}\\ \|\\ \mathsf{f}\end{array} \quad ,$$

and compose this derivation with the proof of $\bar{R}$ to get a proof of f, which is in contradiction to Theorem 2:

$$\begin{array}{c}\top\\ \|\\ \bar{R}\\ \|\\ \mathsf{f}\end{array} \quad .$$

□

5 Conclusion

In this paper we showed simple proof theoretical techniques for making a system of first order classical logic finitely generating. We believe that these considerations help make clear that finite choice and cut elimination, or other normalisation techniques, are conceptually independent.

Some of the techniques we used, for example the replacement of an atom and its dual by t and f, are folklore. However, in order to produce a finitely generating system they have to be combined with the reduction of the cut rule to its atomic form. This crucial ingredient is provided by deep inference and top-down symmetry, which are not available in the sequent calculus.

In the calculus of structures, there are presentations of various modal logics [13], linear logic [14, 15] and various extensions of it [9, 10, 4] and noncommutative logics [5]. All these systems are similar to system SKSgq in that they include rules which follow a scheme [8], which ensures atomicity of cut and identity. The techniques shown here thus also work for these logics.

6 Bibliography

[1] Kai Brünnler. Locality for classical logic. Technical Report WV-02-15, Dresden University of Technology, 2002. Available at http://www.wv.inf.tu-dresden.de/~kai/LocalityClassical.pdf.

[2] Kai Brünnler. Atomic cut elimination for classical logic. In M. Baaz and J. A. Makowsky, editors, *CSL 2003*, volume 2803 of *Lecture Notes in Computer Science*, pages 86–97. Springer-Verlag, 2003.

[3] Kai Brünnler and Alwen Fernanto Tiu. A local system for classical logic. In R. Nieuwenhuis and A. Voronkov, editors, *LPAR 2001*, volume 2250 of *Lecture Notes in Artificial Intelligence*, pages 347–361. Springer-Verlag, 2001.

[4] Paola Bruscoli. A purely logical account of sequentiality in proof search. In Peter J. Stuckey, editor, *Logic Programming, 18th International Conference*, volume 2401 of *Lecture Notes in Artificial Intelligence*, pages 302–316. Springer-Verlag, 2002.

[5] Pietro Di Gianantonio. Structures in cyclic linear logic. Technical report, Università di Udine, 2003.

[6] Roy Dyckhoff. Contraction-free sequent calculi for intuitionistic logic. *The Journal of Symbolic Logic*, (57):795–807, 1992.

[7] Gerhard Gentzen. Investigations into logical deduction. In M. E. Szabo, editor, *The Collected Papers of Gerhard Gentzen*, pages 68–131. North-Holland Publishing Co., Amsterdam, 1969.

[8] Alessio Guglielmi. Recipe. Manuscript. http://www.wv.inf.tu-dresden.de/~guglielm/Research/Notes/AG2.pdf, 2002.

[9] Alessio Guglielmi. A system of interaction and structure. Technical Report WV-02-10, Technische Universität Dresden, 2002. Available at http://www.wv.inf.tu-dresden.de/~guglielm/Research/Gug/Gug.pdf.

[10] Alessio Guglielmi and Lutz Straßburger. A non-commutative extension of MELL. In Matthias Baaz and Andrei Voronkov, editors, *Logic for Programming, Artificial Intelligence, and Reasoning, LPAR 2002*, volume 2514 of *LNAI*, pages 231–246. Springer-Verlag, 2002.

[11] J. Herbrand. *Recherches sur la théorie de la démonstration*. PhD thesis, Université de Paris, 1930.

[12] Raymond M. Smullyan. Analytic cut. *The Journal of Symbolic Logic*, 33:560–564, 1968.

[13] Charles Stewart and Phiniki Stouppa. A systematic proof theory for several modal logics. Technical Report WV-03-08, Technische Universität Dresden, 2003. Submitted.

[14] Lutz Straßburger. A local system for linear logic. In Matthias Baaz and Andrei Voronkov, editors, *Logic for Programming, Artificial Intelligence, and Reasoning, LPAR 2002*, volume 2514 of *LNAI*, pages 388–402. Springer-Verlag, 2002.

[15] Lutz Straßburger. MELL in the Calculus of Structures. *Theoretical Computer Science*, 309(1–3):213–285, 2003.

[16] Anne Sjerp Troelstra and Helmut Schwichtenberg. *Basic Proof Theory*. Cambridge University Press, 1996.