Structural Normalization for Classical Natural Deduction

William Lovas (wlovas@cs.cmu.edu)
Karl Crary (crary@cs.cmu.edu)

December 22, 2006

Abstract

We present a judgemental formulation of natural deduction for classical logic, similar in spirit to Wadler's dual calculus, but founded on the logical judgements A **true** and A **false**; proof-by-contradiction, which puts these two judgements in opposition, lies at the heart of our system. We then show directly a normalization property for this system by a purely syntactic structural induction.

1 Introduction

The Curry-Howard correspondence allows us to discover new type systems by exploring logics and conversely to discover new logics by examining type systems. Typically these enterprises are carried out in the setting of natural deduction [9], a method of describing logics that defines what it means to be a proof of a proposition. Natural deduction proofs of certain propositions correspond cleanly and elegantly to expressions of certain types in a type system.

Gentzen proved natural deduction sound by appealing to the *sequent calculus*: first, a proof in the natural deduction system can be translated into a deduction in the sequent calculus using a "cut" rule; then, instances of the "cut" rule in a sequent calculus deduction can be systematically propagated toward the leaves, and eventually, removed entirely; finally, such "cut"-free sequent deductions can be reflected back into irreducible natural deduction proofs.

To this day, many consistency proofs for natural deduction systems are carried out by translating to a sequent calculus that admits cut. But if one is only interested in the natural deduction, this detour through the sequent calculus is unwelcome—one would prefer to do one's consistency proof directly in the natural deduction system. Furthermore, it has long been known that cut elimination for the sequent calculus may be proven by a simple nested structural induction [21, 22], and similar structural proofs have been explored for natural deduction systems with proof terms [3, 13].

The present work shows a direct proof of classical logic's consistency as a weak normalization property of the proof terms of classical natural deduction. Not only is this proof direct, but also it is mathematically simple, proceeding by straightforward lexicographic structural induction on types and terms. It relies only on syntactic notions; no semantic models, logical relations, or saturated sets are needed, in contrast to most work on strong normalization [7]. Furthermore, the proof exposes the computational content of consistency as a (non-deterministic) normalization procedure for classical proofs.

In what follows, we introduce a natural deduction system for classical logic and discuss some of its properties (Section 2), presenting its semantics in the judgemental style of Martin-Löf [14] along with several examples of classical theorems; we then characterize the normal forms of our calculus and outline a weak normalization theorem, proving a few interesting cases along the way (Section 3); finally, we demonstrate that our method may be straightforwardly extended to classical *linear* logic (Section 4).

2 Classical natural deduction

2.1 Syntax and judgements

The following system for classical natural deduction is based on Martin-Löf's separation of judgements from propositions [14, 23, 1]. It is similar in spirit to Wadler's dual calculus [24, 25], but founded on the judgements A **true** and A **false** and their interplay rather than on the classical sequent calculus. The presentation shown here is due to Nanevski [18]; in what follows, we will refer to the system as JCL, for Judgemental Classical Logic.

The syntax of JCL is shown in Figure 1. First, we have the propositions A, with all the usual classical connectives: truth, conjunction, falsehood, disjunction, and negation. (The omission of implication is intentional, since it may be defined in classical logic in terms of negation and either disjunction or conjunction [24].) Then we have expressions e, which represent proofs of a proposition's truth, and continuations k, which represent proofs of a proposition's falsehood. Contradictions c are the combination of a continuation with an expression. Computationally, a contradiction represents the state of a computation; the notation $k \triangleleft e$ is meant to suggest a value e being passed to a continuation k, as per the computational interpretation of classical proofs [12, 17, 16, 10].

We will use the word "term" to refer generically to any expression, continuation, or contradiction.

Judgementally, the expressions e will be proof terms for deductions of the judgement A **true** and the continuations k will be proof terms for deductions of the judgement A **false**. In general, these categorical judgements must be extended to hypothetical judgements. In doing so, it is convenient, but inessential, to separate hypotheses about truth (in contexts Γ) from hypotheses about falsehood (in contexts Δ), leading to judgements of the form $\Gamma; \Delta \vdash e : A$ **true** and $\Gamma; \Delta \vdash k : A$ **false**. (Note that we abbreviate true hypothesis $x : A$ **true** as simply $x{:}A$ and false hypotheses $u : A$ **false** as $u{:}\overline{A}$.) Finally, we have a judgement representing a proof of contradiction, $\Gamma; \Delta \vdash c : \#$; this

$$
\begin{array}{lr}
A ::= \mathbf{1} \mid A_1 \times A_2 \mid \mathbf{0} \mid A_1 + A_2 \mid \neg A & \textit{propositions} \\
\\
e ::= x \mid \langle\rangle \mid \langle e_1, e_2 \rangle \mid \mathsf{inl}\, e \mid \mathsf{inr}\, e \mid \mathsf{not}\, k \mid u{:}\overline{A}.\, c & \textit{expressions} \\
k ::= u \mid k \circ \mathsf{fst} \mid k \circ \mathsf{snd} \mid [] \mid [k_1, k_2] \mid \mathsf{not}\, e \mid x{:}A.\, c & \textit{continuations} \\
c ::= k \triangleleft e & \textit{contradictions} \\
\\
\Gamma ::= \cdot \mid \Gamma, x{:}A & \textit{true contexts} \\
\Delta ::= \cdot \mid \Delta, u{:}\overline{A} & \textit{false contexts}
\end{array}
$$

Figure 1: Syntax of JCL's types, proofs, and contexts

states that the true hypotheses found in Γ are in contradiction with the false hypotheses found in Δ, and the term c witnesses this contradiction. Typing rules defining these judgements appear in Figure 2.

When $k \triangleleft e$ is a contradiction, with $k : A$ **false** and $e : A$ **true**, we call A the *mediating type* of the contradiction.

The expressions are essentially familiar introduction forms: a proof of conjunction is a pair of proofs $\langle e_1, e_2 \rangle$, and a proof of disjunction is a proof tagged with inl or inr. A proof of negation is a continuation reified as an expression, an essential concept in defining functions and implication.

The continuations roughly represent elimination forms: refutations of conjunction are continuations that perform the first or second projection and then pass the result to another continuation; refutations of disjunction represent case analysis as a pair of continuations $[k_1, k_2]$. A refutation of negation represents an expression suspended as a continuation.

Both expressions and continuations have an unusual binding syntactic form representing proof-by-contradiction. The expression $u{:}\overline{A}.\, c$ corresponds roughly to a **call**/**cc** construct in a functional language with control effects: it grabs the current continuation, binds it to u, and continues with another contradiction, inside of which a value e may be sent to u by placing the two in contradiction with $u \triangleleft e$ — a kind of **goto** with an argument. The continuation $x{:}A.\, c$ is simply its dual; we might whimsically refer to it as **call**/**ce**.

All three of the hypothetical judgements admit substitution principles that allow proofs or refutations to take the place of truth or falsehood hypotheses.

Theorem 1 (Substitution). Let J be any of $e' : B$ **true**, $k' : B$ **false**, or $c : \#$. Then

1. If $(\Gamma, x{:}A); \Delta \vdash J$ and $\Gamma; \Delta \vdash e : A$ **true**, then $\Gamma; \Delta \vdash \{e/x\}\, J$.
2. If $\Gamma; (\Delta, u{:}\overline{A}) \vdash J$ and $\Gamma; \Delta \vdash k : A$ **false**, then $\Gamma; \Delta \vdash \{k/u\}\, J$.

Proof. By simultaneous induction on the hypothetical judgement with a distinguished hypothesis. □

$\boxed{\Gamma;\Delta \vdash e : A\ \mathbf{true}}$

$$\frac{}{(\Gamma, x{:}A);\Delta \vdash x : A\ \mathbf{true}}\ (\text{hypT}) \qquad \frac{\Gamma;(\Delta, u{:}\overline{A}) \vdash c : \#}{\Gamma;\Delta \vdash u{:}\overline{A}.\, c : A\ \mathbf{true}}\ (\#\text{E-T})$$

$$\frac{}{\Gamma;\Delta \vdash \langle\rangle : \mathbf{1}\ \mathbf{true}}\ (\mathbf{1}\text{T}) \qquad \frac{\Gamma;\Delta \vdash e_1 : A_1\ \mathbf{true} \qquad \Gamma;\Delta \vdash e_2 : A_2\ \mathbf{true}}{\Gamma;\Delta \vdash \langle e_1, e_2\rangle : A_1 \times A_2\ \mathbf{true}}\ (\times\text{T})$$

$$\frac{\Gamma;\Delta \vdash e : A_1\ \mathbf{true}}{\Gamma;\Delta \vdash \mathsf{inl}\, e : A_1 + A_2\ \mathbf{true}}\ (+\text{T}_1) \qquad \frac{\Gamma;\Delta \vdash e : A_2\ \mathbf{true}}{\Gamma;\Delta \vdash \mathsf{inr}\, e : A_1 + A_2\ \mathbf{true}}\ (+\text{T}_2)$$

$$\frac{\Gamma;\Delta \vdash k : A\ \mathbf{false}}{\Gamma;\Delta \vdash \mathsf{not}\, k : \neg A\ \mathbf{true}}\ (\neg\text{T})$$

$\boxed{\Gamma;\Delta \vdash k : A\ \mathbf{false}}$

$$\frac{}{\Gamma;(\Delta, u{:}\overline{A}) \vdash u : A\ \mathbf{false}}\ (\text{hypF}) \qquad \frac{(\Gamma, x{:}A);\Delta \vdash c : \#}{\Gamma;\Delta \vdash x{:}A.\, c : A\ \mathbf{false}}\ (\#\text{E-F})$$

$$\frac{\Gamma;\Delta \vdash k : A_1\ \mathbf{false}}{\Gamma;\Delta \vdash k \circ \mathsf{fst} : A_1 \times A_2\ \mathbf{false}}\ (\times\text{F}_1) \qquad \frac{\Gamma;\Delta \vdash k : A_2\ \mathbf{false}}{\Gamma;\Delta \vdash k \circ \mathsf{snd} : A_1 \times A_2\ \mathbf{false}}\ (\times\text{F}_2)$$

$$\frac{}{\Gamma;\Delta \vdash [] : \mathbf{0}\ \mathbf{false}}\ (\mathbf{0}\text{F}) \qquad \frac{\Gamma;\Delta \vdash k_1 : A_1\ \mathbf{false} \qquad \Gamma;\Delta \vdash k_2 : A_2\ \mathbf{false}}{\Gamma;\Delta \vdash [k_1, k_2] : A_1 + A_2\ \mathbf{false}}\ (+\text{F})$$

$$\frac{\Gamma;\Delta \vdash e : A\ \mathbf{true}}{\Gamma;\Delta \vdash \mathsf{not}\, e : \neg A\ \mathbf{false}}\ (\neg\text{F})$$

$\boxed{\Gamma;\Delta \vdash c : \#}$

$$\frac{\Gamma;\Delta \vdash k : A\ \mathbf{false} \qquad \Gamma;\Delta \vdash e : A\ \mathbf{true}}{\Gamma;\Delta \vdash k \mathbin{\triangleleft} e : \#}\ (\#\text{I})$$

Figure 2: Judgemental classical natural deduction

Substitutions $\{e/x\}\,J$ and $\{k/u\}\,J$ are of the standard capture-avoiding variety.

Finally, we have a notion of proof reduction, characterized by three judgements: $e \longrightarrow e'$, $k \longrightarrow k'$, and $c \longrightarrow c'$. Rules defininining these transition relations appear in Figure 3.

As usual, reduction preserves well-typedness of terms.

Theorem 2 (Subject reduction).

1. If $\Gamma; \Delta \vdash e : A$ **true** and $e \longrightarrow e'$, then $\Gamma; \Delta \vdash e' : A$ **true**.
2. If $\Gamma; \Delta \vdash k : A$ **false** and $k \longrightarrow k'$, then $\Gamma; \Delta \vdash k' : A$ **false**.
3. If $\Gamma; \Delta \vdash c : \#$ and $c \longrightarrow c'$, then $\Gamma; \Delta \vdash c' : \#$.

Proof. By simultaneous induction on the transition relations, with appeals to Substitution. □

The interesting reduction rules, corresponding to β-reductions, are all in the $c \longrightarrow c'$ judgement, which specifies how proofs and refutations of the same proposition interact with each other to form smaller contradictions. A projection continuation, when passed a pair, projects the appropriate component and passes it along to the remainder of the continuation; dually, a case analysis, when passed a tagged expression, passes the expression along to the appropriate continuation.

A **call**/**cc** expression can always step when passed to any continuation: the continuation is substituted for the bound variable in the body contradiction. Well-typedness of the result follows from the substitution principle for falsehood assumptions. Similarly, **call**/**ce** continuations can always step when passed any expression.

The remainder of the transition rules are compatibility cases that allow contradiction reduction to occur anywhere inside a term.

2.2 Traditional elimination forms

As a type system, JCL is interesting in that it lacks elimination forms in its expression language. Instead, it has true introductions and false introductions, with all the action happening when they come together. The usual elimination forms are all definable, though, using proof-by-contradiction. To simulate the rules

$$\frac{\Gamma; \Delta \vdash e : A \times B \textbf{ true}}{\Gamma; \Delta \vdash \pi_1\, e : A \textbf{ true}}\ (\times \mathrm{E}_1) \qquad \frac{\Gamma; \Delta \vdash e : A \times B \textbf{ true}}{\Gamma; \Delta \vdash \pi_2\, e : B \textbf{ true}}\ (\times \mathrm{E}_2)$$

$$\frac{\Gamma; \Delta \vdash e : A + B \textbf{ true} \qquad (\Gamma, x{:}A); \Delta \vdash e_1 : C \textbf{ true} \qquad (\Gamma, y{:}B); \Delta \vdash e_2 : C \textbf{ true}}{\Gamma; \Delta \vdash \textbf{case}(e, x{:}A.\, e_1, y{:}B.\, e_2) : C \textbf{ true}}\ (+\mathrm{E}),$$

it suffices to make the following definitions:

$$\pi_1\, e : A \stackrel{\text{def}}{=} u{:}\overline{A}.\, u \circ \mathsf{fst} \triangleleft e$$

$$\pi_2\, e : B \stackrel{\text{def}}{=} u{:}\overline{B}.\, u \circ \mathsf{snd} \triangleleft e$$

$$\textbf{case}(e, x{:}A.\, e_1, y{:}B.\, e_2) : C \stackrel{\text{def}}{=} u{:}\overline{C}.\, [x{:}A.\, u \triangleleft e_1, y{:}B.\, u \triangleleft e_2] \triangleleft e,$$

$\boxed{e \longrightarrow e'}$

$$\frac{c \longrightarrow c'}{u{:}\overline{A}.\, c \longrightarrow u{:}\overline{A}.\, c'} \qquad \frac{e_1 \longrightarrow e_1'}{\langle e_1, e_2\rangle \longrightarrow \langle e_1', e_2\rangle} \qquad \frac{e_2 \longrightarrow e_2'}{\langle e_1, e_2\rangle \longrightarrow \langle e_1, e_2'\rangle}$$

$$\frac{e \longrightarrow e'}{\mathsf{inl}\, e \longrightarrow \mathsf{inl}\, e'} \qquad \frac{e \longrightarrow e'}{\mathsf{inr}\, e \longrightarrow \mathsf{inr}\, e'} \qquad \frac{k \longrightarrow k'}{\mathsf{not}\, k \longrightarrow \mathsf{not}\, k'}$$

$\boxed{k \longrightarrow k'}$

$$\frac{c \longrightarrow c'}{x{:}A.\, c \longrightarrow x{:}A.\, c'} \qquad \frac{k \longrightarrow k'}{k \circ \mathsf{fst} \longrightarrow k' \circ \mathsf{fst}} \qquad \frac{k \longrightarrow k'}{k \circ \mathsf{snd} \longrightarrow k' \circ \mathsf{snd}}$$

$$\frac{k_1 \longrightarrow k_1'}{[k_1, k_2] \longrightarrow [k_1', k_2]} \qquad \frac{k_2 \longrightarrow k_2'}{[k_1, k_2] \longrightarrow [k_1, k_2']} \qquad \frac{e \longrightarrow e'}{\mathsf{not}\, e \longrightarrow \mathsf{not}\, e'}$$

$\boxed{c \longrightarrow c'}$

$$\frac{k \longrightarrow k'}{k \triangleleft e \longrightarrow k' \triangleleft e} \qquad \frac{e \longrightarrow e'}{k \triangleleft e \longrightarrow k \triangleleft e'}$$

$$\frac{}{k \triangleleft u{:}\overline{A}.\, c \longrightarrow \{k/u\}\, c} \qquad \frac{}{x{:}A.\, c \triangleleft e \longrightarrow \{e/x\}\, c}$$

$$\frac{}{k \circ \mathsf{fst} \triangleleft \langle e_1, e_2\rangle \longrightarrow k \triangleleft e_1} \qquad \frac{}{k \circ \mathsf{snd} \triangleleft \langle e_1, e_2\rangle \longrightarrow k \triangleleft e_2}$$

$$\frac{}{[k_1, k_2] \triangleleft \mathsf{inl}\, e \longrightarrow k_1 \triangleleft e} \qquad \frac{}{[k_1, k_2] \triangleleft \mathsf{inr}\, e \longrightarrow k_2 \triangleleft e}$$

$$\frac{}{\mathsf{not}\, e \triangleleft \mathsf{not}\, k \longrightarrow k \triangleleft e}$$

Figure 3: Proof reduction

where in each case, u is a fresh continuation variable. It is easy to verify that, for any continuations k of the appropriate type:

$$\begin{aligned} k \triangleleft \pi_1 \langle e_1, e_2 \rangle &\longrightarrow^* k \triangleleft e_1 \\ k \triangleleft \pi_2 \langle e_1, e_2 \rangle &\longrightarrow^* k \triangleleft e_2 \\ k \triangleleft \mathbf{case}(\mathsf{inl}\, e, x{:}A.\, e_1, y{:}B.\, e_2) &\longrightarrow^* k \triangleleft \{e/x\}\, e_1 \\ k \triangleleft \mathbf{case}(\mathsf{inr}\, e, x{:}A.\, e_1, y{:}B.\, e_2) &\longrightarrow^* k \triangleleft \{e/y\}\, e_2 \end{aligned}$$

We may also define a nullary **case** statement, the intuitionistic elimination form for falsehood, usually called **abort**:

$$\frac{\Gamma; \Delta \vdash e : \mathbf{0}\ \mathbf{true}}{\Gamma; \Delta \vdash \mathbf{abort}\, e : A\ \mathbf{true}}\ (\mathbf{0}\mathrm{E})$$

$$\mathbf{abort}\, e : A \stackrel{\text{def}}{=} u{:}\overline{A}.\, [] \triangleleft e$$

2.3 Examples

JCL can be used to produce proof terms of classical theorems with computational content in the form of reduction behavior.

The law of the excluded middle, commonly held as the essential difference between classical and intuitionistic logic, may be represented as the following natural deduction proof:

$$u{:}\overline{A + \neg A}.\, u \triangleleft \mathsf{inr}\, (\mathsf{not}\, (x{:}A.\, u \triangleleft \mathsf{inl}\, x))$$

This exhibits the well-known "time-travelling" behavior characteristic of classical proofs [24, 15]: the proof initially asserts $\neg A$ by constructing an A-accepting continuation, but if it's ever called out by being passed a proof of A, it travels back to the original context and asserts A, using the given proof.

It is convenient to define implication, $A_1 \to A_2$: as usual, its (true) "introduction" form is a λ-abstraction $\lambda x{:}A_1.\, e$. Its (false) "elimination" form is an argument expression and a continuation, written "$e; k$":

$$\frac{(\Gamma, x{:}A_1); \Delta \vdash e : A_2\ \mathbf{true}}{\Gamma; \Delta \vdash \lambda x{:}A_1.\, e : A_1 \to A_2\ \mathbf{true}}\ (\to\mathrm{T})$$

$$\frac{\Gamma; \Delta \vdash e : A_1\ \mathbf{true} \qquad \Gamma; \Delta \vdash k : A_2\ \mathbf{false}}{\Gamma; \Delta \vdash e; k : A_1 \to A_2\ \mathbf{false}}\ (\to\mathrm{F})$$

Logically, the rule $\to$F simply says that an implication is false if its antecedent is true and its conclusion is false—ordinary classical truth-table reasoning. Computationally, the continuation $e; k$, when passed a function, applies the function to the argument e and passes the result to the continuation k.

Classical implication can be defined in two different ways that are dual to one another [24]. For example, we might take:

$$A \to B \stackrel{\text{def}}{=} \neg(A \times \neg B)$$

$$(\lambda x{:}A.\, e) : A \to B \stackrel{\text{def}}{=} \mathsf{not}\,(z{:}A \times \neg B.\, (x{:}A.\, (\mathsf{not}\, e) \circ \mathsf{snd} \triangleleft z) \circ \mathsf{fst} \triangleleft z)$$

$$(e; k) : \overline{A \to B} \stackrel{\text{def}}{=} \mathsf{not}\,\langle e, \mathsf{not}\, k\rangle$$

Using implication, we can prove Peirce's law, $((A \to B) \to A) \to A$:

$$\lambda f{:}(A \to B) \to A.\, u{:}\overline{A}.\, (g_u; u) \triangleleft f \qquad \text{where } g_u = \lambda x{:}A.\, v{:}\overline{B}.\, u \triangleleft x$$

Computationally, this is precisely the expression-oriented version of the **call/cc** primitive. Given a function f, it grabs the current continuation u and calls f with a representation of u as a function, g_u. If f returns normally, its return value is passed to u, but if f invokes g_u with some value, that value will be passed to u instead.

If we wanted our system to be particularly austere, we could eliminate disjunction from JCL altogether and let it be a defined notion:

$$A + B \stackrel{\text{def}}{=} \neg(\neg A \times \neg B)$$

$$(\mathsf{inl}\, e) : A + B \stackrel{\text{def}}{=} \mathsf{not}\,(\mathsf{not}\, e \circ \mathsf{fst})$$

$$(\mathsf{inr}\, e) : A + B \stackrel{\text{def}}{=} \mathsf{not}\,(\mathsf{not}\, e \circ \mathsf{snd})$$

$$[k_1, k_2] : A + B \stackrel{\text{def}}{=} \mathsf{not}\,\langle \mathsf{not}\, k_1, \mathsf{not}\, k_2\rangle$$

Of course, by duality, we could do the same in the reverse direction, defining conjunction in terms of disjunction.

3 Weak normalization

We begin by characterizing the normal forms of JCL. Recall that the only place β-reductions happen is in contradictions, so we focus on characterizing irreducible contradictions. Two important facts are immediately evident:

- $k \triangleleft u{:}\overline{A}.\, c$ and $x{:}A.\, c \triangleleft e$ can both take a step, and
- if $k \triangleleft e$ is well-typed and k and e are both introduction forms—as opposed to variables or binders—then $k \triangleleft e$ can take a step.

Therefore, for a contradiction to be irreducible, both halves must not intro forms, and neither half may be a binding form. This motivates the idea of a *neutral* term: a neutral term is either a variable or an introduction form, and one can appear on either side of a *normal* contradiction, provided the other side is just a variable.

Although the binding forms $u{:}\overline{A}.\, c$ and $x{:}A.\, c$ may not appear on either side of a normal contradiction, it is clear that any such term should be regarded as normal itself provided its sub-contradiction is normal. Further inspection reveals that these binding forms can even appear deep inside a

normal term, as long as they're not put in contradiction with a term of the opposite sort. Thus, subterms of neutral terms may be merely *normal*.

A complete grammar describing normal and neutral terms appears below. Neutral terms are written as bold terms, and normal terms are written as bold, capitalized terms.

$$\begin{array}{lr}
\mathbf{e} ::= x \mid \langle\rangle \mid \langle \mathbf{E_1}, \mathbf{E_2} \rangle \mid \mathsf{inl}\ \mathbf{E} \mid \mathsf{inr}\ \mathbf{E} \mid \mathsf{not}\ \mathbf{K} & \textit{neutral expressions} \\
\mathbf{E} ::= \mathbf{e} \mid u{:}\overline{A}.\ \mathbf{C} & \textit{normal expressions} \\
\mathbf{k} ::= u \mid \mathbf{K} \circ \mathsf{fst} \mid \mathbf{K} \circ \mathsf{snd} \mid [\,] \mid [\mathbf{K_1}, \mathbf{K_2}] \mid \mathsf{not}\ \mathbf{E} & \textit{neutral continuations} \\
\mathbf{K} ::= \mathbf{k} \mid x{:}A.\ \mathbf{C} & \textit{normal continuations} \\
\mathbf{C} ::= u \triangleleft \mathbf{e} \mid \mathbf{k} \triangleleft x & \textit{normal contradictions}
\end{array}$$

With this definition in hand, we can state a normalization theorem for well-typed terms:

Theorem 3 (Normalization).

1. If $\Gamma; \Delta \vdash e : A$ **true**, then $e \longrightarrow^* \mathbf{E}$.
2. If $\Gamma; \Delta \vdash k : A$ **false**, then $k \longrightarrow^* \mathbf{K}$.
3. If $\Gamma; \Delta \vdash c : \#$, then $c \longrightarrow^* \mathbf{C}$.

where $\longrightarrow^*$ is the usual reflexive, transitive closure of $\longrightarrow$.

Proof. Straightforward induction on the typing derivation of the term. The interesting case is the contradiction case.

Case: $\Gamma; \Delta \vdash k \triangleleft e : \#$

$$\begin{array}{rr}
k \triangleleft e \longrightarrow^* \mathbf{K} \triangleleft e & \text{By induction on } k. \\
\longrightarrow^* \mathbf{K} \triangleleft \mathbf{E} & \text{By induction on } e. \\
\longrightarrow^* \mathbf{C} & \text{By Lemma (below).}
\end{array}$$

□

Thus, normalization is an immediate corollary of a lemma regarding the normalization of contradictions where both sides are normal. This main lemma is the heart of the normalization proof. Its proof relies on the related fact that neutral-neutral contradictions also normalize, as well as a series of "cut" lemmas stating that substitutions of normalizing terms into normalizing terms are themselves normalizing. In all, there are 20 substitution cases, since there are four sorts of things we may substitute (**e**, **E**, **k**, and **K**) and five sorts of things we may substitute *into* (**e**, **E**, **k**, **K**, and **C**). In fact, these seemingly auxiliary facts refer back to the main lemma, so in order to prove it, we must strengthen its statement to include them all; the complete lemma is stated in Figure 4.

Lemma 4 (Natural deduction cut elimination).
If $\Gamma; \Delta \vdash \mathbf{K} : A$ **false** and $\Gamma; \Delta \vdash \mathbf{E} : A$ **true**, then $\mathbf{K} \triangleleft \mathbf{E} \longrightarrow^* \mathbf{C}$.

Proof. We prove, simultaneously, the list of clauses given in Figure 4, by lexicographic induction on three things:

1. The active type. In normalization clauses, this is the mediating type of the contradiction; in "cut" clauses, this is the type of the distinguished free variable.

1. **Neutral/neutral normalization.** Neutral continuations against neutral expressions normalize.
   If $\Gamma; \Delta \vdash \mathbf{k} : A$ **false** and $\Gamma; \Delta \vdash \mathbf{e} : A$ **true**, then $\mathbf{k} \triangleleft \mathbf{e} \longrightarrow^* \mathbf{C}$.
2. **Neutral/* cuts.** Substitutions of neutral terms (i) into neutral terms neutralize, (ii) into normal terms normalize.
   If $\Gamma; \Delta \vdash \mathbf{e} : A$ **true**, then
   (a) if $(\Gamma, x{:}A); \Delta \vdash \mathbf{e}' : B$ **true**, then $\{\mathbf{e}/x\}\,\mathbf{e}' \longrightarrow^* \mathbf{e}''$.
   (b) if $(\Gamma, x{:}A); \Delta \vdash \mathbf{k} : B$ **true**, then $\{\mathbf{e}/x\}\,\mathbf{k} \longrightarrow^* \mathbf{k}'$.
   (c) if $(\Gamma, x{:}A); \Delta \vdash \mathbf{E} : B$ **true**, then $\{\mathbf{e}/x\}\,\mathbf{E} \longrightarrow^* \mathbf{E}'$.
   (d) if $(\Gamma, x{:}A); \Delta \vdash \mathbf{K} : B$ **true**, then $\{\mathbf{e}/x\}\,\mathbf{K} \longrightarrow^* \mathbf{K}'$.
   (e) if $(\Gamma, x{:}A); \Delta \vdash \mathbf{C} : B$ **true**, then $\{\mathbf{e}/x\}\,\mathbf{C} \longrightarrow^* \mathbf{C}'$.

   If $\Gamma; \Delta \vdash \mathbf{k} : A$ **false**, then
   (f) if $\Gamma; (\Delta, u{:}\overline{A}) \vdash \mathbf{e} : B$ **true**, then $\{\mathbf{k}/u\}\,\mathbf{e} \longrightarrow^* \mathbf{e}'$.
   (g) if $\Gamma; (\Delta, u{:}\overline{A}) \vdash \mathbf{k}' : B$ **true**, then $\{\mathbf{k}/u\}\,\mathbf{k}' \longrightarrow^* \mathbf{k}''$.
   (h) if $\Gamma; (\Delta, u{:}\overline{A}) \vdash \mathbf{E} : B$ **true**, then $\{\mathbf{k}/u\}\,\mathbf{E} \longrightarrow^* \mathbf{E}'$.
   (i) if $\Gamma; (\Delta, u{:}\overline{A}) \vdash \mathbf{K} : B$ **true**, then $\{\mathbf{k}/u\}\,\mathbf{K} \longrightarrow^* \mathbf{K}'$.
   (j) if $\Gamma; (\Delta, u{:}\overline{A}) \vdash \mathbf{C} : B$ **true**, then $\{\mathbf{k}/u\}\,\mathbf{C} \longrightarrow^* \mathbf{C}'$.
3. **Neutral-normal and normal-neutral normalization.** Normal terms against neutral terms normalize.
   (a) If $\Gamma; \Delta \vdash \mathbf{k} : A$ **false** and $\Gamma; \Delta \vdash \mathbf{E} : A$ **true**, then $\mathbf{k} \triangleleft \mathbf{E} \longrightarrow^* \mathbf{C}$.
   (b) If $\Gamma; \Delta \vdash \mathbf{K} : A$ **false** and $\Gamma; \Delta \vdash \mathbf{e} : A$ **true**, then $\mathbf{K} \triangleleft \mathbf{e} \longrightarrow^* \mathbf{C}$.
4. **Normal/* cuts.** Substitutions of normal terms (i) into neutral terms normalize, if of the same sort, neutralize if of different sorts, (ii) into normal terms normalize.
   If $\Gamma; \Delta \vdash \mathbf{E} : A$ **true**, then
   (a) if $(\Gamma, x{:}A); \Delta \vdash \mathbf{e} : B$ **true**, then $\{\mathbf{E}/x\}\,\mathbf{e} \longrightarrow^* \mathbf{E}'$.
   (b) if $(\Gamma, x{:}A); \Delta \vdash \mathbf{k} : B$ **true**, then $\{\mathbf{E}/x\}\,\mathbf{k} \longrightarrow^* \mathbf{k}'$.
   (c) if $(\Gamma, x{:}A); \Delta \vdash \mathbf{E}' : B$ **true**, then $\{\mathbf{E}/x\}\,\mathbf{E}' \longrightarrow^* \mathbf{E}''$.
   (d) if $(\Gamma, x{:}A); \Delta \vdash \mathbf{K} : B$ **true**, then $\{\mathbf{E}/x\}\,\mathbf{K} \longrightarrow^* \mathbf{K}'$.
   (e) if $(\Gamma, x{:}A); \Delta \vdash \mathbf{C} : B$ **true**, then $\{\mathbf{E}/x\}\,\mathbf{C} \longrightarrow^* \mathbf{C}'$.

   If $\Gamma; \Delta \vdash \mathbf{K} : A$ **false**, then
   (f) if $\Gamma; (\Delta, u{:}\overline{A}) \vdash \mathbf{e} : B$ **true**, then $\{\mathbf{K}/u\}\,\mathbf{e} \longrightarrow^* \mathbf{e}'$.
   (g) if $\Gamma; (\Delta, u{:}\overline{A}) \vdash \mathbf{k} : B$ **true**, then $\{\mathbf{K}/u\}\,\mathbf{k} \longrightarrow^* \mathbf{K}'$.
   (h) if $\Gamma; (\Delta, u{:}\overline{A}) \vdash \mathbf{E} : B$ **true**, then $\{\mathbf{K}/u\}\,\mathbf{E} \longrightarrow^* \mathbf{E}'$.
   (i) if $\Gamma; (\Delta, u{:}\overline{A}) \vdash \mathbf{K}' : B$ **true**, then $\{\mathbf{K}/u\}\,\mathbf{K}' \longrightarrow^* \mathbf{K}''$.
   (j) if $\Gamma; (\Delta, u{:}\overline{A}) \vdash \mathbf{C} : B$ **true**, then $\{\mathbf{K}/u\}\,\mathbf{C} \longrightarrow^* \mathbf{C}'$.
5. **Normal-normal normalization.** Normal continuations against normal expressions normalize.
   If $\Gamma; \Delta \vdash \mathbf{K} : A$ **false** and $\Gamma; \Delta \vdash \mathbf{E} : A$ **true**, then $\mathbf{K} \triangleleft \mathbf{E} \longrightarrow^* \mathbf{C}$.

Figure 4: Complete statement of Lemma 4.

2. The clause number. Any clause may use an earlier clause at the same active type, where the clause order from earliest to latest is
   (a) Neutral-neutral normalization
   (b) Neutral/* cuts
   (c) Normal-neutral normalization and neutral-normal normalization
   (d) Normal/* cuts
   (e) Normal-normal normalization.
3. The expression being substituted into. Any cut clause may refer to cut clauses of the same number at the same active type provided that they only do so at a smaller expression.

Although there are many cases to prove in this simultaneous induction, most of them are completely straightforward. Since the proof relies only on lexicographic structural induction, it is quite amenable to formalization in a logical framework, and in fact the authors have formalized it in machine-checkable form in the Twelf logical framework.

In the remainder of this section, we show a few of the interesting cases of the proof.

Clause 1 (Neutral-neutral normalization).
If $\Gamma; \Delta \vdash \mathbf{k} : A$ **false** and $\Gamma; \Delta \vdash \mathbf{e} : A$ **true**, then $\mathbf{k} \triangleleft \mathbf{e} \longrightarrow^* \mathbf{C}$.

We proceed by case analysis on $\mathbf{k}$, $\mathbf{e}$, and A.

Case: $\mathbf{k} = u$

$u \triangleleft \mathbf{e} \longrightarrow^* u \triangleleft \mathbf{e}$ By reflexivity.

Case: $\mathbf{e} = x$

$\mathbf{k} \triangleleft x \longrightarrow^* \mathbf{k} \triangleleft x$ By reflexivity.

Case: $A = A_1 \times A_2$

$\mathbf{k} = \mathbf{K_1} \circ \mathsf{fst}$ or $\mathbf{k} = \mathbf{K_2} \circ \mathsf{snd}$, and $\mathbf{e} = \langle \mathbf{E_1}, \mathbf{E_2} \rangle$. By inversion.

Subcase: $\mathbf{k} = \mathbf{K_1} \circ \mathsf{fst}$

$\mathbf{K_1} \circ \mathsf{fst} \triangleleft \langle \mathbf{E_1}, \mathbf{E_2} \rangle \longrightarrow \mathbf{K_1} \triangleleft \mathbf{E_1}$ By rule.
$\longrightarrow^* \mathbf{C}$ By induction at A_1.

Subcase: $\mathbf{k} = \mathbf{K_2} \circ \mathsf{snd}$

$\mathbf{K_2} \circ \mathsf{snd} \triangleleft \langle \mathbf{E_1}, \mathbf{E_2} \rangle \longrightarrow \mathbf{K_2} \triangleleft \mathbf{E_2}$ By rule.
$\longrightarrow^* \mathbf{C}$ By induction at A_2.

Other cases: Similar.

Clause 5 (Normal-normal normalization).
If $\Gamma; \Delta \vdash \mathbf{K} : A$ **false** and $\Gamma; \Delta \vdash \mathbf{E} : A$ **true**, then $\mathbf{K} \triangleleft \mathbf{E} \longrightarrow^* \mathbf{C}$.

We proceed by case analysis on $\mathbf{K}$ and $\mathbf{E}$.

Case: $\mathbf{K} = \mathbf{k}$ and $\mathbf{E} = \mathbf{e}$

$\mathbf{k} \triangleleft \mathbf{e} \longrightarrow^* \mathbf{C}$ By induction at same type, Clause 1.

Case: $\mathbf{K} = x{:}A.\,\mathbf{C}$

$x{:}A.\,\mathbf{C} \triangleleft \mathbf{E} \longrightarrow \{\mathbf{E}/x\}\,\mathbf{C}$ By rule.
$\longrightarrow^* \mathbf{C}'$ By induction at same type, Clause 4.

Case: $\mathbf{E} = u{:}\overline{A}.\,\mathbf{C}$

Similar.

The cut lemmas are straightforward, except for the cases of cutting a normal term into a normal contradiction; we prove one such case in its entirety below.

Clause 4 (Subclause (4e): Normal-e/normal-c cut).
If $\Gamma; \Delta \vdash \mathbf{E} : A$ **true** and $(\Gamma, x{:}A); \Delta \vdash \mathbf{C} : \#$, then $\{\mathbf{E}/x\}\,\mathbf{C} \longrightarrow^* \mathbf{C}'$.

We proceed by case analysis on $\mathbf{C}$.

Case: $\mathbf{C} = \mathbf{k} \triangleleft x$

$\{\mathbf{E}/x\}\,\mathbf{C} = \{\mathbf{E}/x\}\,\mathbf{k} \triangleleft \mathbf{E}$ By the definition of substitution.
$\longrightarrow^* \mathbf{k}' \triangleleft \mathbf{E}$ By induction at smaller term.
$\longrightarrow^* \mathbf{C}'$ By induction at same type, Clause 3.

Note: Crucially, the active type remains the same in the last inductive appeal: since the variable we're substituting for, x, appears on one side of the contradiction, the contradiction's mediating type is the type of x. Furthermore, it is essential that $\mathbf{k}'$ be neutral and not merely normal—we may *not* appeal to normal-normal normalization, Clause 5, at the same active type.

Case: $\mathbf{C} = \mathbf{k} \triangleleft y$, where $y \neq x$

$\{\mathbf{E}/x\}\,\mathbf{C} = \{\mathbf{E}/x\}\,\mathbf{k} \triangleleft y$ By the definition of substitution.
$\longrightarrow^* \mathbf{k}' \triangleleft y$ By induction at smaller term.
$= \mathbf{C}'$

Note: Here, the active type may change: we know nothing about the type of y. That doesn't matter, though, since $\mathbf{k} \triangleleft y$ is already normal—we need not appeal to any inductive hypothesis.

Case: $\mathbf{C} = u \triangleleft \mathbf{e}$

$\{\mathbf{E}/x\}\,\mathbf{C} = u \triangleleft \{\mathbf{E}/x\}\,\mathbf{e}$ By the definition of substitution.
$\longrightarrow^* u \triangleleft \mathbf{E}'$ By induction at smaller term.

Note: As above, the active type may change: we know nothing about the type of u. Therefore, we may *not* appeal inductively to Clause 3, neutral-normal normalization, even though its clause number is earlier. Instead, we proceed by cases on $\mathbf{E}'$.

$$A ::= \mathbf{1} \mid A_1 \otimes A_2 \mid \mathbf{0} \mid A_1 \oplus A_2 \mid \top \mid A_1 \mathbin{\&} A_2 \mid \bot \mid A_1 \parr A_2 \mid \neg A$$

$$\begin{aligned}
e &::= x \mid \langle\!\langle\rangle\!\rangle \mid \langle\!\langle e_1, e_2 \rangle\!\rangle \mid \mathsf{inl}\, e \mid \mathsf{inr}\, e \mid \langle\rangle \mid \langle e_1, e_2 \rangle \mid [\![]\!].\, c \mid [\![u_1{:}\overline{A_1}, u_2{:}\overline{A_2}]\!].\, c \\
&\quad \mid \mathsf{not}\, k \mid u{:}\overline{A}.\, c \\
k &::= u \mid \langle\!\langle\rangle\!\rangle.\, c \mid \langle\!\langle x_1{:}A_1, x_2{:}A_2 \rangle\!\rangle.\, c \mid [] \mid [k_1, k_2] \mid k \circ \mathsf{fst} \mid k \circ \mathsf{snd} \mid [\![]\!] \mid [\![k_1, k_2]\!] \\
&\quad \mid \mathsf{not}\, e \mid x{:}A.\, c \\
c &::= k \triangleleft e
\end{aligned}$$

$$\Gamma ::= \cdot \mid \Gamma, x{:}A$$

$$\Delta ::= \cdot \mid \Delta, u{:}\overline{A}$$

Figure 5: Syntax of JCLL'c types, proofs, and contexts

Subcase: $\mathbf{E}' = \mathbf{e}'$
$u \triangleleft \mathbf{E}' = u \triangleleft \mathbf{e}' = \mathbf{C}'$.

Subcase: $\mathbf{E}' = v{:}\overline{B}.\, \mathbf{C}''$
$u \triangleleft \mathbf{E}' = u \triangleleft v{:}\overline{B}.\, \mathbf{C}'' \longrightarrow \{u/v\}\, \mathbf{C}'' = \mathbf{C}'$

In the second subcase above, it is evident that substituting a variable for a variable inside a normal contradiction yields another normal contradiction. □

Corollary 5 (Consistency). There is no c such that $\cdot;\cdot \vdash c : \#$.

Proof. By contradiction. Suppose there were such a c. Then it would have a normal form $c \longrightarrow^* \mathbf{C}$. By Subject Reduction, $\cdot;\cdot \vdash \mathbf{C} : \#$. But since a normal contradiction must have a variable on one side, no normal contradiction can be well-typed without hypotheses. □

4 Extension to classical linear logic

Here, we briefly sketch how our result may be extended to a dual-style natural deduction for classical linear logic we call Judgemental Classical Linear Logic, or JCLL.

4.1 Classical linear proof terms

JCLL has similar judgemental foundations to JILL [1], but with the symmetric classical duality of JCL. To JCL, it adds types and proof terms for multiplicative conjunction and disjunction. Figure 5 outlines its syntax.

Hypotheses are now interpreted as linear resources and may only appear linearly; the typing rules guarantee this.

$$\frac{}{x{:}A; \cdot \vdash x : A \ \mathbf{true}} \qquad \frac{}{\cdot; u{:}\overline{A} \vdash u : A \ \mathbf{false}}$$

Hypotheses come equipped with linear substitution principles.

Theorem 6 (Linear substitution). Let J be any of $e' : B$ **true**, $k' : B$ **false**, or $c : \#$. Then

1. If $(\Gamma', x{:}A); \Delta' \vdash J$ and $\Gamma; \Delta \vdash e : A$ **true**, then $(\Gamma, \Gamma'); (\Delta, \Delta') \vdash \{e/x\}\, J$.
2. If $\Gamma'; (\Delta', u{:}\overline{A}) \vdash J$ and $\Gamma; \Delta \vdash k : A$ **false**, then $(\Gamma, \Gamma'); (\Delta, \Delta') \vdash \{k/u\}\, J$.

The usual additive conjunction and disjunction are now $A \mathbin{\&} B$ and $A \oplus B$, with units $\top$ and $\mathbf{0}$, respectively. Their proof terms, typing rules, and operational behavior are the same as those for $A \times B$, $A + B$, $\mathbf{1}$, and $\mathbf{0}$ in the unrestricted calculus.

A proof of multiplicative conjunction $A \otimes B$ is represented by a pair of proofs $\langle\!\langle e_1, e_2 \rangle\!\rangle$ that do not share resources. A refutation of $A \otimes B$ is represented by a contradiction under truth assumptions for its conjuncts.

$$\frac{\Gamma_1; \Delta_1 \vdash e_1 : A \ \mathbf{true} \qquad \Gamma_2; \Delta_2 \vdash e_2 : B \ \mathbf{true}}{(\Gamma_1, \Gamma_2); (\Delta_1, \Delta_2) \vdash \langle\!\langle e_1, e_2 \rangle\!\rangle : A \otimes B \ \mathbf{true}}\ (\otimes\mathrm{T})$$

$$\frac{(\Gamma, x{:}A, y{:}B); \Delta \vdash c : \#}{\Gamma; \Delta \vdash \langle\!\langle x{:}A, y{:}B \rangle\!\rangle.\, c : A \otimes B \ \mathbf{false}}\ (\otimes\mathrm{F})$$

The operational interpretation of the interaction of a proof and a refutation of $A \otimes B$ substitutes the proofs for the hypotheses to yield a new contradiction.

$$\frac{}{\langle\!\langle x{:}A, y{:}B \rangle\!\rangle.\, c \mathrel{\triangleleft} \langle\!\langle e_1, e_2 \rangle\!\rangle \longrightarrow \{e_1, e_2/x, y\}\, c}$$

This corresponds closely to the behavior of the usual intuitionistic "**let**" elimination form.

Proofs and refutations of multiplicative disjunction $A \parr B$ are precisely symmetric: a refutation is a pair of refutations $[\![k_1, k_2]\!]$ and a proof is a contradiction under assumptions of falsehood $[\![u_1{:}\overline{A}, u_2{:}\overline{B}]\!].\, c$.

Multiplicative units $\mathbf{1}$ and $\bot$ are nullary versions of $A \otimes B$ and $A \parr B$.

$$\frac{}{\cdot; \cdot \vdash \langle\!\langle\rangle\!\rangle : \mathbf{1} \ \mathbf{true}}\ (\mathbf{1}\mathrm{T}) \qquad \frac{\Gamma; \Delta \vdash c : \#}{\Gamma; \Delta \vdash \langle\!\langle\rangle\!\rangle.\, c : \mathbf{1} \ \mathbf{false}}\ (\mathbf{1}\mathrm{F}) \qquad \frac{}{\langle\!\langle\rangle\!\rangle.\, c \mathrel{\triangleleft} \langle\!\langle\rangle\!\rangle \longrightarrow c}$$

$$\frac{\Gamma; \Delta \vdash c : \#}{\Gamma; \Delta \vdash [\![]\!].\, c : \mathbf{0} \ \mathbf{true}}\ (\bot\mathrm{T}) \qquad \frac{}{\cdot; \cdot \vdash [\![]\!] : \mathbf{0} \ \mathbf{false}}\ (\bot\mathrm{F}) \qquad \frac{}{[\![]\!] \mathrel{\triangleleft} [\![]\!].\, c \longrightarrow c}$$

As expected, in the linear calculus, a contradiction's two halves may not share resources.

$$\frac{\Gamma_1; \Delta_1 \vdash k : A \ \mathbf{false} \qquad \Gamma_2; \Delta_2 \vdash e : A \ \mathbf{true}}{(\Gamma_1, \Gamma_2); (\Delta_1, \Delta_2) \vdash k \mathrel{\triangleleft} e : \#}\ (\#\mathrm{I})$$

The remainder of the typing and reduction rules are similar, and we omit them.

The linear calculus enjoys subject reduction.

Theorem 7 (Linear subject reduction).

1. If $\Gamma; \Delta \vdash e : A$ **true** and $e \longrightarrow e'$, then $\Gamma; \Delta \vdash e' : A$ **true**.
2. If $\Gamma; \Delta \vdash k : A$ **false** and $k \longrightarrow k'$, then $\Gamma; \Delta \vdash k' : A$ **false**.
3. If $\Gamma; \Delta \vdash c : \#$ and $c \longrightarrow c'$, then $\Gamma; \Delta \vdash c' : \#$.

4.2 Classical linear normalization

It is straightforward to define normal and neutral forms for this calculus as in the unrestricted calculus, and the "cut elimination" proof is precisely analogous. Linearity does not interfere with most cases; when it does, it does so by ruling the case out (for example, the linear proof has no cases for $\{\mathbf{e}/x\}\, u$ or $\{\mathbf{e}/x\}\, \langle\!\langle\rangle\!\rangle$).

$$
\begin{aligned}
\mathbf{e} &::= x \mid \langle\!\langle\rangle\!\rangle \mid \langle\!\langle \mathbf{E_1}, \mathbf{E_2} \rangle\!\rangle \mid \mathsf{inl}\, \mathbf{E} \mid \mathsf{inr}\, \mathbf{E} \\
&\quad \mid \langle\rangle \mid \langle \mathbf{E_1}, \mathbf{E_2} \rangle \mid [\![]\!].\, c \mid [\![u_1{:}\overline{A_1}, u_2{:}\overline{A_2}]\!].\, \mathbf{C} \mid \mathsf{not}\, \mathbf{K} \\
\mathbf{E} &::= \mathbf{e} \mid u{:}\overline{A}.\, \mathbf{C} \\
\mathbf{k} &::= u \mid \langle\!\langle\rangle\!\rangle.\, \mathbf{C} \mid \langle\!\langle x_1{:}A_1, x_2{:}A_2 \rangle\!\rangle.\, \mathbf{C} \mid [\,] \mid [\mathbf{K_1}, \mathbf{K_2}] \\
&\quad \mid \mathbf{K} \circ \mathsf{fst} \mid \mathbf{K} \circ \mathsf{snd} \mid [\![]\!] \mid [\![\mathbf{K_1}, \mathbf{K_2}]\!] \mid \mathsf{not}\, \mathbf{E} \\
\mathbf{K} &::= \mathbf{k} \mid x{:}A.\, \mathbf{C} \\
\mathbf{C} &::= u \triangleleft \mathbf{e} \mid \mathbf{k} \triangleleft x
\end{aligned}
$$

Theorem 8 (Linear weak normalization).

1. If $\Gamma; \Delta \vdash e : A$ **true**, then $e \longrightarrow^* \mathbf{E}$.
2. If $\Gamma; \Delta \vdash k : A$ **false**, then $k \longrightarrow^* \mathbf{K}$.
3. If $\Gamma; \Delta \vdash c : \#$, then $c \longrightarrow^* \mathbf{C}$.

5 A note on commuting conversions

Structural normalization proofs for intuitionistic calculi with closed-scope eliminations like **case** require commuting conversions[1] to restore the subformula property [13]. Arbitrary instances of **case** analysis can violate the invariant that eliminations reduce the size of the active type. By reducing the type at which a **case** occurs, commuting conversions permit an induction on the active type to succeed.

Neither of our classical normalization proofs rely on a reduction relation with commuting conversions: ordinary β reductions suffice. One can explain the lack of commuting conversions in two ways. First, although our logic includes sum types $A + B$, our equivalent of a **case** analysis, copairing $[k_1, k_2]$, does not introduce a closed scope. Since the need for commuting

[1] Also called "permutative conversions" or "permutation conversions".

conversions arises from the "parasitic formula" [11] introduced by closed-scope eliminations, and we have no equivalent of closed-scope eliminations, our logic does not need commuting conversions.

Second, one can view proofs in JCL and JCLL as CPS-converted proofs of intuitionistic proofs—control flow is made explicit and all complex subterms are named. As observed by de Groote [6], CPS conversion identifies terms that are commuting convertible. In JCL and JCLL, the translations of commuting convertible terms turn out to be β-convertible. Intuitionistic linear logic provides a convenient example which has the pleasing property that the conversion does not affect the size of the terms:

$$\begin{gathered}\textbf{let}\ \langle\!\langle a{:}A, b{:}B\rangle\!\rangle = (\textbf{let}\ \langle\!\langle c{:}C, d{:}D\rangle\!\rangle = M\ \textbf{in}\ N)\ \textbf{in}\ P : E \\ \longrightarrow_{\text{cc}} \textbf{let}\ \langle\!\langle c{:}C, d{:}D\rangle\!\rangle = M\ \textbf{in let}\ \langle\!\langle a{:}A, b{:}B\rangle\!\rangle = N\ \textbf{in}\ P : E \\ (\text{where } c, d \notin \mathrm{FV}(P))\end{gathered}$$

These translate to the classical terms

$$\begin{gathered}u{:}\overline{E}.\,(\langle\!\langle a{:}A, b{:}B\rangle\!\rangle.\,u \triangleleft P) \triangleleft (v{:}\overline{A \otimes B}.\,(\langle\!\langle c{:}C, d{:}D\rangle\!\rangle.\,v \triangleleft N) \triangleleft M) \\ \longrightarrow_{\text{cc}} u{:}\overline{E}.\,(\langle\!\langle c{:}C, d{:}D\rangle\!\rangle.\,u \triangleleft (v{:}\overline{A \otimes B}.\,(\langle\!\langle a{:}A, b{:}B\rangle\!\rangle.\,v \triangleleft P) \triangleleft N) \triangleleft M \\ (\text{where } c, d \notin \mathrm{FV}(P))\end{gathered}$$

which both β-reduce to the term

$$u{:}\overline{E}.\,(\langle\!\langle c{:}C, d{:}D\rangle\!\rangle.\,(\langle\!\langle a{:}A, b{:}B\rangle\!\rangle.\,u \triangleleft P) \triangleleft N) \triangleleft M.$$

6 Related work

The computational interpretation of classical logic in terms of a λ-calculus enriched with control operators has been discovered, re-discovered, and explored by many over the years. Filinski's symmetric λ-calculus [8] was the first judgemental presentation of classical natural deduction, in the context of categorical duality. Griffin [12], Girard [10], and Murthy [17, 16] all did seminal work exploring the connections between classical logic and control effects in programming languages. Parigot [20] later presented a natural deduction for classical logic, relying not on the judgements A **true** and A **false** as we do, but rather on a notion of multiple conclusions similar to the classical sequent calculus.

Wadler's dual calculus [24] is most closely related to ours, with its symmetric character inspired by Curien and Herbelin [2]; in its most recent presentation [25], the structural rules of weakening and contraction are admissible, so its proof terms are essentially a notational variant of our own. The present work focuses only on β-reductions, though, without treating Wadler's η, ς, and ν expansions. The dualized judgemental interpretation in terms of truth and falsehood, due to Nanevski [18], is novel, as far as we are aware.

Our proof technique is similar to Pfenning's structural cut-elimination proofs [21, 22], adapted to the natural deduction setting. Its discovery parallels that of hereditary substitutions [26, 19], a technique now used

primarily in specifying logical frameworks: both can be seen as taking normalization proof that proceeds by translation through the sequent calculus and turning it into a proof that proceeds entirely in the natural deduction calculus.

There is no shortage of recent work on normalization for classical calculi. For example, de Groote [5] and David [4] exhibit strong normalization proofs for $\lambda\mu$-like calculi. Wadler [24] conjectured without proof that his call-by-value and call-by-name calculi were strongly normalizing in the absence of expansions. Recently, Dougherty, *et al.* [7] published a proof of strong normalization for the non-confluent (neither call-by-value nor call-by-name) version of Wadler's calculus using a logical relations argument. Our proof differs from theirs first by demonstrating only *weak* normalization and second by relying solely on the syntactic method of structural induction.

7 Conclusions

We have presented a judgemental natural deduction system for classical logic, characterized its normal forms, and proven by structural induction that every valid proof term has a normal form. Furthermore, we have formalized our proof in the Twelf logical framework so that it may be verified mechanically. We have also presented a judgemental natural deduction for classical *linear* logic and demonstrated how our normalization proof adapts to the linear case.

More generally speaking, in demonstrating our result, we have also shown that a consistency proof for a logic given in natural deduction style need not step outside the system, either by clunkily translating to another system or by giving a complicated semantic model for the system. A simple syntactic lexicographic induction suffices to show normalization.

References

[1] Bor-Yuh Evan Chang, Kaustuv Chaudhuri, and Frank Pfenning. A judgmental analysis of linear logic. Technical Report CMU-CS-03-131, Carnegie Mellon University, 2003.

[2] Pierre-Louis Curien and Hugo Herbelin. The duality of computation. In *Proceedings of the fifth ACM SIGPLAN International Conference on Functional Programming (ICFP'00)*, pages 233–243, New York, NY, USA, 2000. ACM Press.

[3] René David. Normalization without reducibility. *Annals of Pure and Applied Logic*, 107:121–130, 2001.

[4] René David and Karim Nour. A short proof of the strong normalization of classical natural deduction with disjunction. *Journal of Symbolic Logic*, 68(4):1277–1288, December 2003.

[5] Philippe de Groote. Strong normalization of classical natural deduction with disjunction. In Samson Abramsky, editor, *Fifth International*

Conference on Typed Lambda Calculi and Applications (TLCA'01), volume 2044 of *Lecture Notes in Computer Science*, pages 182–196. Springer, 2001.

[6] Philippe de Groote. On the strong normalisation of intuitionistic natural deduction with permutation-conversions. *Information and Computation*, 178(2):441–464, 2002.

[7] Daniel J. Dougherty, Silvia Ghilezan, Silvia Likavec, and Pierre Lescanne. Strong normalization of the dual classical sequent calculus. In *12th International Conference on Logic for Programming, Artificial Intelligence and Reasoning (LPAR'05)*, December 2005.

[8] Andrzej Filinski. Declarative continuations and categorical duality. Master's thesis, University of Copenhagen, Copenhagen, Denmark, August 1989. (DIKU Report 89/11.).

[9] Gerhard Gentzen. Investigations into logical deduction. In M. E. Szabo, editor, *The Collected Papers of Gerhard Gentzen*. North-Holland, 1969.

[10] Jean-Yves Girard. A new constructive logic: classical logic. *Mathematical Structures in Computer Science*, 1(3):255–296, 1991.

[11] Jean-Yves Girard, Paul Taylor, and Yves Lafont. *Proofs and Types*. Cambridge Tracts in Computer Science. Cambridge University Press, Cambridge, 1989.

[12] Timothy G. Griffin. A formulae-as-types notion of control. In *17th Annual ACM Symposium on Principles of Programming Languages (POPL'90)*, San Francisco, January 1990. ACM Press.

[13] Felix Joachimski and Ralph Matthes. Short proofs of normalization. *Archive of Mathematical Logic*, 42(1):59–87, 2003.

[14] Per Martin-Löf. On the meanings of the logical constants and the justifications of the logical laws. *Nordic Journal of Philosophical Logic*, 1(1):11–60, 1996.

[15] Tom Murphy, VII, Karl Crary, and Robert Harper. Distributed control flow with classical modal logic. In Luke Ong, editor, *Computer Science Logic, 19th International Workshop (CSL 2005)*, Lecture Notes in Computer Science. Springer, August 2005.

[16] Chetan R. Murthy. Classical proofs as programs: How, what, and why. In J. Paul Myers Jr. and Michael J. O'Donnell, editors, *Constructivity in Computer Science*, volume 613 of *Lecture Notes in Computer Science*, pages 71–88. Springer, 1991.

[17] Chetan R. Murthy. An evaluation semantics for classical proofs. In *6th IEEE Symposium on Logic in Computer Science (LICS'91)*, pages 96–109, 1991.

[18] Aleksandar Nanevski. Judgemental reconstruction of classical logic. Personal communication, 2004.

[19] Aleksandar Nanevski, Frank Pfenning, and Brigitte Pientka. Contextual modal type theory. *Under consideration for publication in the ACM Transactions on Computational Logic*, September 2005.

[20] Michel Parigot. $\lambda\mu$-calculus: An algorithmic interpretation of classical natural deduction. In *Proceedings of the International Conference on Logic Programming and Automated Reasoning (LPAR'92)*, pages 190–201, London, UK, 1992. Springer-Verlag.

[21] Frank Pfenning. Structural cut elimination in linear logic. Technical Report CMU-CS-94-222, Carnegie Mellon University, 1994.

[22] Frank Pfenning. Structural cut elimination: I. intuitionistic and classical logic. *Information and Computation*, 157(1-2):84–141, 2000.

[23] Frank Pfenning and Rowan Davies. A judgmental reconstruction of modal logic. *Mathematical Structures in Computer Science*, 11(4):511–540, 2001.

[24] Philip Wadler. Call-by-value is dual to call-by-name. In *ICFP '03: Proceedings of the Eighth ACM SIGPLAN International Conference on Functional Programming*, volume 38, pages 189–201, New York, NY, USA, September 2003. ACM Press.

[25] Philip Wadler. Call-by-value is dual to call-by-name - reloaded. In Jürgen Giesl, editor, *RTA*, volume 3467 of *Lecture Notes in Computer Science*, pages 185–203. Springer, 2005.

[26] Kevin Watkins, Iliano Cervesato, Frank Pfenning, and David Walker. A concurrent logical framework I: Judgements and properties. Technical Report CMU-CS-02-101, Carnegie Mellon University, March 2002. Revised May 2003.