Vol. 39 No. 12 Transactions of Information Processing Society of Japan Dec. 1998

Regular Paper

Calculus of Classical Proofs II☆

KEN-ETSU FUJITA†

We provide a simple natural deduction system of classical propositional logic called λ_{exc}, and demonstrate the proof-theoretical and computational properties of the system from a programming viewpoint. The introduction of λ_{exc} is a consequence of our observations on the existence of a special form of cut-free LK proofs, which we call LJK proofs with invariants. We first show the existence of LJK proofs with invariants for each tautology. On the basis of the proof-theoretical result, we then present (1) a strict fragment of λ_{exc} that is complete with respect to classical provability, (2) a translation from arbitrary proofs to LJK-style proofs, (3) the Church-Rosser and strong normalization properties of λ_{exc}, and (4) an isomorphism between Parigot's $\lambda\mu$-calculus and λ_{exc}, and a comparison with related work.

1. Introduction

The computational meaning of proofs has been investigated in a wide range of fields, including only intuitionistic logic[16),18),23)] and constructive type theory[24)], but also classical logic[3),15),22),26),32)] and modal logic[19)]. Algorithmic contents of proofs can be used to obtain correct programs that satisfy logical specifications. In this paper, our motivation is to study the computational aspects of a simple classical natural deduction system, which arises from our proof-theoretical observations on the existence of a special form of cut-free LK proofs for each tautology. We call the special form of LK proofs LJK proofs with invariants[8)]. In LJK proofs, the succedent of each sequent in the proof is such that every occurrence of the succedent, except for at most one occurrence, is the same as the invariant throughout the proof. Following the proof-theoretical result, we provide a classical natural deduction system λ_{exc}[9)] by using a variant of the *ex*cluded middle. In this paper, on the basis of the Curry-Howard isomorphism[18)], we investigate from a programming viewpoint the computational properties of classical proofs as programs.

In Section 2, we provide a sequent calculus LJK and demonstrate the proof-theoretical properties of the system. In Section 3, on the basis of the existence of LJK proofs, we introduce a simple natural deduction system, λ_{exc}, of classical propositional logic using a variant of the *ex*cluded middle. In λ_{exc}, we study a computational property of classical proofs, and discuss the meaning of the existence of LJK proofs from a programming viewpoint. We also show a direct translation from any proof in λ_{exc} to an LJK-style proof. In Section 4, we prove that λ_{exc} has the Church-Rosser and strong normalization properties. Finally in Section 5, we compare it with related work—Parigot's $\lambda\mu$-calculus, Reholf & Sørensen's λ_{Δ}, and Felleisen's λ_c—to clarify their similarities and differences, from which we obtain an isomorphism between $\lambda\mu$-calculus and λ_{exc}, and the strong normalization property of λ_{exc}.

From a programming viewpoint, this work follows from our previous work[8)]. The terminology of LJK proofs denotes exactly the same style of proofs as μ-head form proofs in Ref. 8). Another follow-on study was devoted to a call-by-value programming language based on classical proofs[9),11)].

2. Sequent Calculus LJK

In sequent calculi, we can distinguish classical systems and intuitionistic systems by the cardinality restriction on the succedent of the sequent[35)]. This restriction is critical in some systems such as L'J[20)], the Beth-tableau system in Ref. 36), and $IL^{>}$[33)]. By introducing LJK proofs, we first show that at most two kinds of formulae on the succedent are enough to prove any tautology.

On the other hand, the role of structural rules in sequent calculi is very important; in fact, by

† Faculty of Computer Science and Systems Engineering, Kyushu Institute of Technology

☆ A preliminary version of this paper was presented at New Aspects in Non-Classical Logics and Their Kripke Semantics, Kyoto University, RIMS, March (1997). The revised version was in part presented at Workshop on Theories of Types and Proofs, Tokyo Institute of Technology, September (1997).

adding and/or deleting structural rules, the systems may drastically change their logical properties as well as decidability properties. The notion of LJK proofs is obtained from observation of the effect of the right contraction rule. Careful consideration naturally leads to separation of the succedent into two parts, namely, a contractable part and a non-contractable part. We will discuss whether the right contraction rule can be applied to certain subformulae among the given formulae. The subformulae to which the right contraction rules are applied are specified in terms of the notion of "invariants" used in LJK proofs.

Simple examples of LJK proofs (to be defined later) of Peirce's law are given below in terms of LK. The following proof 1 below is called an LJK proof of $((A \to B) \to A) \to A$ with the invariant A, and proof 2 an LJK proof with the invariant $((A \to B) \to A) \to A$. In LJK proofs, the succedent of each sequent is such that every occurrence of formulae in the succedent, except for at most one occurrence, is the same as the invariant.

proof 1:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{A \Longrightarrow A}{A \Longrightarrow A, B}}{\Longrightarrow A, A \to B} \quad A \Longrightarrow A}{(A \to B) \to A \Longrightarrow A, A}}{(A \to B) \to A \Longrightarrow A}}{\Longrightarrow ((A \to B) \to A) \to A}$$

proof 2:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{A \Longrightarrow A}{(A \to B) \to A, A \Longrightarrow A}}{A \Longrightarrow ((A \to B) \to A) \to A}}{A \Longrightarrow ((A \to B) \to A) \to A, B}}{\Longrightarrow ((A \to B) \to A) \to A, A \to B} \quad A \Longrightarrow A}{(A \to B) \to A \Longrightarrow ((A \to B) \to A) \to A, A}}{\Longrightarrow ((A \to B) \to A) \to A}$$

To specify LJK proofs, we introduce a sequent calculus called LJK☆. The system uses a sequent of the form $\Gamma \Longrightarrow \Delta; A^*$, where Δ consists of at most one occurrence, and A^* consists of a finite number possibly zero of occurrences of a formula A. Roughly speaking, with the antecedent Γ and the first part Δ of the succedent we simulate intuitionistic inference by forbidding the right contraction rule for Δ. On the other hand, the right contraction rule can be applied to the second part A^*. Thus our idea is to distinguish an intuitionistic part from a non-intuitionistic part in classical proofs.

LJK:

(Axioms)

$$B \Longrightarrow B;$$

(Structural Rules)

$$\dfrac{\Gamma \Longrightarrow \Delta; A^*}{C, \Gamma \Longrightarrow \Delta; A^*}\ (w \Rightarrow)$$

$$\dfrac{\Gamma \Longrightarrow\ ; A^*}{\Gamma \Longrightarrow B; A^*}\ (\Rightarrow w_i) \qquad \dfrac{\Gamma \Longrightarrow \Delta; A^*}{\Gamma \Longrightarrow \Delta; A, A^*}\ (\Rightarrow w_c)$$

$$\dfrac{C, C, \Gamma \Longrightarrow \Delta; A^*}{C, \Gamma \Longrightarrow \Delta; A^*}\ (c \Rightarrow)$$

$$\dfrac{\Gamma \Longrightarrow \Delta; A, A, A^*}{\Gamma \Longrightarrow \Delta; A, A^*}\ (\Rightarrow c)$$

$$\dfrac{\Gamma, C, D, \Pi \Longrightarrow \Delta; A^*}{\Gamma, D, C, \Pi \Longrightarrow \Delta; A^*}\ (e \Rightarrow)$$

$$\dfrac{\Gamma \Longrightarrow A; A^*}{\Gamma \Longrightarrow\ ; A, A^*}\ (\Rightarrow s_c) \qquad \dfrac{\Gamma \Longrightarrow\ ; A, A^*}{\Gamma \Longrightarrow A; A^*}\ (\Rightarrow s_i)$$

$$\dfrac{\Gamma \Longrightarrow B; A^{*(1)} \quad B, \Pi \Longrightarrow \Delta; A^{*(2)}}{\Gamma, \Pi \Longrightarrow \Delta; A^{*(1)}, A^{*(2)}}\ (cut_i)$$

$$\dfrac{\Gamma \Longrightarrow \Delta; A, A^{*(1)} \quad A, \Pi \Longrightarrow\ ; A^{*(2)}}{\Gamma, \Pi \Longrightarrow \Delta; A^{*(1)}, A^{*(2)}}\ (cut_c)$$

(Logical Rules)

$$\dfrac{\Gamma \Longrightarrow B; A^{*(1)} \quad C, \Pi \Longrightarrow \Delta; A^{*(2)}}{B \to C, \Gamma, \Pi \Longrightarrow \Delta; A^{*(1)}, A^{*(2)}}\ (\to\Rightarrow)$$

$$\dfrac{B, \Gamma \Longrightarrow C; A^*}{\Gamma \Longrightarrow B \to C; A^*}\ (\Rightarrow\to)$$

$$\dfrac{\Gamma \Longrightarrow B; A^*}{\neg B, \Gamma \Longrightarrow\ ; A^*}\ (\neg \Rightarrow) \qquad \dfrac{B, \Gamma \Longrightarrow\ ; A^*}{\Gamma \Longrightarrow \neg B; A^*}\ (\Rightarrow \neg)$$

Definition 1
(LJK Proofs with Invariants)
A proof in the system LJK is said to be an LJK proof with the invariant A, if the second part of succedents consists of occurrences of A throughout the proof.

From the above definition, LJK proofs with empty invariants can be identified with LJ proofs. We give examples of Peirce's law in the following.

☆ The notion of LJK proofs was introduced by the author independently of Girard's LC [13] and LU [14], but it can be regarded as a fragment of LC. Here, for simplicity, we consider only the fragment of implication and negation. Our discussion can be extended to the full LK: see Ref. 8).

proof 3:

$$
\begin{array}{c}
A \Longrightarrow A; \\ \hline
A \Longrightarrow \ ; A \\ \hline
A \Longrightarrow B; A \\ \hline
\Longrightarrow A \to B; A \quad A \Longrightarrow A; \\ \hline
(A \to B) \to A \Longrightarrow A; A \\ \hline
(A \to B) \to A \Longrightarrow \ ; A, A \\ \hline
(A \to B) \to A \Longrightarrow \ ; A \\ \hline
(A \to B) \to A \Longrightarrow A; \\ \hline
\Longrightarrow ((A \to B) \to A) \to A;
\end{array}
$$

proof 4:

$$
\begin{array}{c}
A \Longrightarrow A; \\ \hline
(A \to B) \to A, A \Longrightarrow A; \\ \hline
A \Longrightarrow ((A \to B) \to A) \to A; \\ \hline
A \Longrightarrow \ ; ((A \to B) \to A) \to A \\ \hline
A \Longrightarrow B; ((A \to B) \to A) \to A \\ \hline
\Longrightarrow A \to B; ((A \to B) \to A) \to A \quad A \Longrightarrow A; \\ \hline
(A \to B) \to A \Longrightarrow A; ((A \to B) \to A) \to A \\ \hline\hline
\Longrightarrow ((A \to B) \to A) \to A;
\end{array}
$$

Lemma 1 (Embedding of LJK Proofs)
If $\Gamma \Longrightarrow \Delta; A^*$ is provable with the invariant A in LJK, then $\Gamma, \neg A \Longrightarrow \Delta$ is provable in LJ.

Proof. By induction on the derivation. □

For example, from proof 3 and proof 4 one can easily obtain LJ proofs of $\neg A \Longrightarrow ((A \to B) \to A) \to A$ and $\neg(((A \to B) \to A) \to A) \Longrightarrow ((A \to B) \to A) \to A$, respectively.

Let Γ/A be the sequence obtained from Γ by deleting all occurrences of the formula A. The following lemma plays an important role in our discussion:

Lemma 2
(From LJ Proofs to LJK Proofs)
If $\Gamma \Longrightarrow B$ is provable in LJ, then $\Gamma/\neg A \Longrightarrow B; A$ is provable with the invariant A in LJK. In particular, cut-free LJK proofs with some invariants are obtained from cut-free LJ proofs.

Proof. By induction on the derivation. □

Corollary 1 (Cut-Free LJK Proofs)
If we have an LJK proof of $\Gamma \Longrightarrow \Delta; A^*$ with the invariant A, then there exists a cut-free LJK proof of $\Gamma \Longrightarrow \Delta; A$ with the invariant A.

Proof. From the above two lemmata and the cut-elimination property of LJ. □

To obtain a candidate for invariants to which the right contraction is applied, we decompose a formula into its components and assumptions.

Definition 2
(Invariants and Corresponding Assumptions)
Given a formula A, the collection of candidates for invariants denoted by $CI(A)$ is defined as a collection of strictly positive subformulae of A with respect to $\to$. In other words, when A is decomposed by the following rule $\rightsquigarrow$, $CI(A)$ is a collection of the second elements of all the pairs appearing in the decomposition process starting from $([\,], A)$:

$$([\Gamma], A_1 \to A_2) \rightsquigarrow ([\Gamma, A_1], A_2),$$

where Γ is a sequence of formulae. For each $A_i \in CI(A)$, we will write $\mathrm{Assume}(A_i, A)$ for the first element $[\Gamma]$ of the pair $([\Gamma], A_i)$ appearing in the decomposition of A.

For instance, in the case of Peirce's law $Peirce \equiv ((A \to B) \to A) \to A$ we have $CI(Peirce) = [Peirce, A]$, where $Peirce$ is called the outermost invariant, and A is the innermost invariant. Then $\mathrm{Assume}\ (Peirce, Peirce) = [\,]$ and $\mathrm{Assume}\ (A, Peirce) = [(A \to B) \to A]$.

Theorem 1 (Existence of LJK Proofs)
If we have $\Gamma \Longrightarrow A$ in LK, then for any B in $CI(A)$, there is a cut-free LJK proof of $\Gamma \Longrightarrow A;$ with the invariant B.

Proof. Assume that one has $\Gamma \Longrightarrow A$ in LK; then one also has $\Gamma, \mathrm{Assume}(B, A) \Longrightarrow B$ in LK. From Glivenko's theorem, we have $\Gamma, \mathrm{Assume}(B, A), \neg B \Longrightarrow B$ in LJ, and $\Gamma, \mathrm{Assume}(B, A) \Longrightarrow B; B$ with invariant B in cut-free LJK from Lemma 2 and Corollary 1. Hence, we have $\Gamma \Longrightarrow A;$ with invariant B in cut-free LJK. □

By contraposition of Lemma 1, we can check which subformulae of the given formula can be invariants. For instance, in the case of Peirce's law there are only two invariants among the subformulae, namely, the cases of proof 3 and proof 4. From Theorem 1, moreover, for each tautology there are invariants among the subformulae, and the strictly positive subformulae can be invariants.

The notion of invariants gives a general form of Glivenko's theorem in the sense that if $\Gamma \Longrightarrow A;$ is provable with the invariant B, then the formula obtained from A by replacing the occurrence B of the invariant with $\neg\neg B$ is provable from Γ in LJ. The obtained formula is denoted by $A^{\neg\neg B}$. For example, from proof 3 one can obtain the LJ proof of $\Longrightarrow ((A \to B) \to A) \to \neg\neg A$. In proof 4, the application of the right contraction rule is delayed to the end, and the proof can be translated into the proof of

$\Longrightarrow \neg\neg(((A \to B) \to A) \to A)$ in LJ, which is a consequence of Glivenko's theorem.

Proposition 1
(Double-Negation Translation)
If $\Gamma \Longrightarrow A$ is provable in LK, then $\Gamma \Longrightarrow A^{\neg\neg B}$ is provable in LJ for any B in $CI(A)$.

This proposition gives another double negation translation depending on the invariants. However, the embedded formulae by distinct invariants become intuitionistically equivalent, since $A \to \neg\neg B \Leftrightarrow \neg\neg(A \to B)$ in LJ.

3. Application to Programming

3.1 Natural Deduction System λ_{exc}

It has become well-known from the work of Griffin[15], Murthy[22], and others, that classical proofs of Π_2^0 statements can be interpreted as programs with control operators. On the basis of the Curry-Howard isomorphism[18], the key notion of LJK proofs also provides a simple method for obtaining exception-handling programs. Following our discussion in the previous section, we present a simple classical natural deduction system λ_{exc} and analyze the computational content of the proofs. It will be observed that an invariant computationally plays the role of a type of exceptional parameter.

As stated in the proof of Theorem 1, we have the following equivalence between LK sequents and LJ sequents:

Proposition 2 Let B be in $CI(A)$. $\Gamma \Longrightarrow A$ in LK iff $\Gamma, \mathrm{Assume}(B, A), \neg B \Longrightarrow B$ in LJ.

This approach would be different from the existing ones with the double-negation elimination rule in the sense that classical proofs are derived from two intuitionistic proofs by application of the classical cut-rules with the invariant B, or equivalently the excluded middle. Following the observation, we define a classical natural deduction system☆, and study the computational meaning of proofs in this system. The types are defined as usual by type variables, a constant $\bot$, and $\to$. The terms are defined by two kinds of variables $x's$ and $y's$, where $y's$ (called exceptional variables or continuation variables) are used only for negation types $\neg A$ defined as $A \to \bot$. $FV(M)$ stands for the set of free variables in M.

☆ At first appearance, our system λ_{exc} seems to be different from the existing ones. However, as a consequence of Section 5.1, we will have an isomorphism between the finite type fragment of Parigot's $\lambda\mu$[26] and λ_{exc}.

λ_{exc}:

Types
$$A ::= \alpha \mid \bot \mid A \to A$$
Contexts
$$\Gamma ::= \langle\,\rangle \mid x{:}A, \Gamma \mid y{:}\neg A, \Gamma$$
Terms
$$M ::= x \mid \lambda x.M \mid MM \mid \mathrm{raise}(M) \mid yM \mid \{y{:}\neg A\}M$$
Type Assignment
$$\Gamma \vdash x : \Gamma(x)$$
$$\frac{\Gamma, x{:}A \vdash M : B}{\Gamma \vdash \lambda x.M : A \to B}\ (\to I)$$
$$\frac{\Gamma \vdash M_1 : A \to B \quad \Gamma \vdash M_2 : A}{\Gamma \vdash M_1 M_2 : B}\ (\to E)$$
$$\frac{\Gamma \vdash M : \bot}{\Gamma \vdash \mathrm{raise}(M) : A}\ (\bot E)$$
$$\frac{\Gamma \vdash M : A}{\Gamma \vdash yM : \bot}\ (\bot I)\ \mathit{if}\ \Gamma(y) \equiv \neg A$$
$$\frac{\Gamma, y{:}\neg A \vdash M : A}{\Gamma \vdash \{y{:}\neg A\}M : A}\ (exc)$$

We denote by $\lambda^{\to\bot}$ the system λ_{exc} with $(\bot I)$ replaced by $(\to E)$ and (exc) deleted.

The classical rule (exc) is a variant of the *exc*luded middle[34]. This rule is introduced independently of $(\bot E)$, in contrast to the double-negation elimination rules, such as $(\bot_C)$, which infers $\Gamma \vdash A$ from $\Gamma, \neg A \vdash \bot$, and $(\mathcal{C})$, which infers $\Gamma \vdash A$ from $\Gamma \vdash \neg\neg A$. We call the rule (exc) a rule of *exc*eption-handling. The type A in (exc) is called the type of an *exc*eptional parameter.

Note the similarity of $(\bot I)$ to $(\to E)$, but note also that $\Gamma \not\vdash y : \neg A$ even if $\Gamma(y) = \neg A$. The negative assumption of the form $y : \neg A$ can be discharged only by (exc) in this system. This style of proof is called a regular proof in Ref. 1). In λ_Δ-calculus[32], not only regular but also non-regular proofs are considered. However, from a non-regular proof we can simply construct a regular proof that has the same assumptions and conclusion.

The reduction rules (e2), (e3), and (e4) below are logically obvious, but they are computationally important. The reduction rule (e5) can be considered as a logical permutative reduction in the sense of Refs. 31) and 1), which is also called the structural reduction in Ref. 26).

Term Reductions:

(e1) $(\lambda x.M)N \ \triangleright\ M[x := N]$;
(e2) $(\mathrm{raise}(M))N \ \triangleright\ \mathrm{raise}(M)$;

(e3-1) $y(\text{raise}(M)) \triangleright M$;
(e3-2) $y(\{y_1 : \neg A\}M) \triangleright yM[y_1 := y]$;
(e4-1) $\{y : \neg A\}M \triangleright M$ if $y \notin FV(M)$;
(e4-2) $\{y : \neg A\}(\text{raise}(yM)) \triangleright \{y : \neg A\}M$;
(e5) $(\{y : \neg(A \to B)\}M)N$
$\triangleright \{y : \neg B\}((M[y \Leftarrow N])N)$,
where $M[y \Leftarrow N]$ is defined as follows:
$x[y \Leftarrow N] = x$;
$(\lambda x.M)[y \Leftarrow N] = \lambda x.M[y \Leftarrow N]$;
$(yM)[y \Leftarrow N] = y(M[y \Leftarrow N]N)$;
$(y'M)[y \Leftarrow N] = y'(M[y \Leftarrow N])$ if $y' \not\equiv y$;
$(M_1 M_2)[y \Leftarrow N] = (M_1[y \Leftarrow N])(M_2[y \Leftarrow N])$;
$(\text{raise}(M))[y \Leftarrow N] = \text{raise}(M[y \Leftarrow N])$;
$(\{y' : \neg A\}M)[y \Leftarrow N] = \{y' : \neg A\}(M[y \Leftarrow N])$.

For technical simplicity, we identify $\{y : \neg A\}\{y_1 : \neg A\}M$ with $\{y : \neg A\}M[y_1 := y]$. The transitive closure of $\triangleright$ is denoted by $\triangleright^+_{exc}$. The reflexive transitive closure of $\triangleright$ is denoted by $\triangleright^*_{exc}$, and the binary relation $=_{exc}$ is defined as the reflexive, symmetric, and transitive closure of $\triangleright$. The relations $\triangleright_\beta$, $\triangleright^*_\beta$, and $=_\beta$ are defined as usual. We sometimes write the term $\{y\}M$ without type information.

Proposition 3 There exists a term M such that $\Gamma \vdash_{\lambda_{exc}} M : A$ iff A as a formula is classically provable from Γ.

Proposition 4 (Subject Reduction) Let $\Gamma \vdash_{\lambda_{exc}} M : A$. If $M \triangleright_{exc} N$, then $\Gamma \vdash_{\lambda_{exc}} N : A$.

Let $C[\,]$ be a context with a single hole $[\,]$ such that $C[\,] ::= [\,] \mid (C[\,])M$. We denote $C[M]$ by the term obtained by replacing $[\,]$ in $C[\,]$ with the term M. We then have $C[\text{raise}(M)] \triangleright^*_{exc} \text{raise}(M)$. Let $\mathcal{P} \equiv \lambda x_1.\{y\}x_1(\lambda x_2.\text{raise}(yx_2))$ of type *Peirce*. Then $\mathcal{P}(\lambda k.C[kM]) \triangleright^*_{exc} M$ if $k \notin C[M]$. Here, the context $C[\,]$ is abandoned, and the term M to be passed on has the same type as that of the exceptional parameter of $\mathcal{P}$. This is why the type A in the definition of (exc) is called the type of an exceptional parameter, and this example can be used to implement a simple exit mechanism. In terms of ML[21], $\{y : \neg A\}M$ may be informally read as `let exception y of A in M handle (yx) => x end`, on the basis of the correspondence of $\bot$ with *exn* (type of exceptions in ML)☆.

We now discuss which subformulae of the given formula can give the type of exceptional parameters. The notion of those invariants to which the right contraction is applied can be taken as the type of exceptional parameters. Here, since $\neg A$ is defined as $A \to \bot$, we redefine the decomposition relation $\rightsquigarrow$ as follows:

$$([\Gamma], A \to B) \rightsquigarrow ([\Gamma, A], B) \quad \text{if } B \not\equiv \bot.$$

Next, we will show that a strict fragment of λ_{exc} in which a single use of (exc) is allowed is complete with respect to classical provability. The terms of the fragment, called LJK-style proofs, are defined as follows:

$M_C ::= \{y\}M_I \mid \lambda x.M_C$;
$M_I ::= x \mid \lambda x.M_I \mid M_I M_I \mid yM_I \mid \text{raise}(M_I)$.

The following proposition shows that the restricted terms M_C that represent some standard form of classical proofs are complete with respect to classical provability, and that the existence of invariants provides an effective way to determine which type has to be assumed in writting classical proofs as programs.

Proposition 5 Let A as a formula be classically provable from Γ, and let $A_i \in CI(A)$. Then there exists a λ_{exc}-proof M_C, called an LJK-style proof with the invariant A_i, such that $\Gamma \vdash M_C : A$ and the type of exceptional parameter is A_i.

This proposition also means that for each tautology A and $A_i \in CI(A)$, we have a classical program M_C of type A with at most one exception and the type of an exceptional parameter A_i. In the next section, we give a concrete method of translation into the LJK-style proofs.

3.2 Translation into LJK-Style Proofs

We give an algorithm for translating arbitrary classical proofs into LJK-style proofs. This analysis gives a new reduction relation to λ_{exc}, which shifts the invariant to the inside. The new reduction can be regarded, in a sense, as an η-like expansion of (e5). To establish this translation, we use an auxiliary type system $\lambda^{\to\bot}$. The translation consists of the following three steps:

(1) Given a proof M of A in λ_{exc}, one obtains an embedding $G(M)$ into $\lambda^{\to\bot}$;
(2) A pullback of $G(M)$, $\{y : \neg A\}\text{raise}(G(M)(\lambda z.yz))$, is an LJK-style proof of A with the invariant A;
(3) To obtain an LJK-style proof of A with the invariant A_i, apply the shift reduction (to be defined later) i times to $\{y : \neg A\}\text{raise}(G(M)(\lambda z.yz))$, where $CI(A)$ is

☆ Although we can write the ML program `fun Peirce(w) = let exception y of '1α in w(fn z => raise(y z)) handle (y x) => x end` for $\mathcal{P}$, whose type can be inferred as `(('1α -> 'β) -> '1α) -> '1α` by the ML system, the correspondence is *informal* in the sense that ML is a call-by-value language and the occurrence of `y` in `exception y` is treated as the name of an exception rather than a variable, as in $\{y\}M$. See also Ref. 9).

$[A_0, \cdots, A_n]$ $(0 \leq i \leq n)$.

Definition 3 The embedding G from the proof terms of λ_{exc} to $\lambda^{\to\perp}$ is defined as follows:

$G(x) = \lambda k.kx$;
$G(\lambda x.M)$
$\quad = \lambda k.k(\lambda x.\mathrm{raise}(G(M)(\lambda m.k(\lambda v.m))))$;
$G(MN) = \lambda k.G(M)(\lambda m.G(N)(\lambda n.k(mn)))$;
$G(\mathrm{raise}(M)) = \lambda k.G(M)(\lambda x.x)$;
$G(yM) = \lambda k.k(G(M)y)$;
$G(\{y\!:\!\neg A\}M) = \lambda y.G(M)y$,

where we assume that two categories of variables are transformed into one category of λ-variables.

Note that the above embedding G, except for the case of $\lambda x.M$, bears a strong resemblance to a call-by-value CPS translation, for instance, see Ref. 9). A classical term $M : A$ is probably interpreted as a function $G(M)$ which takes as an argument a continuation that expects an object of A, where a variable y plays the role of a continuation variable. Let $\rhd'$ consist of (e2), (e3-2), (e4-1), (e4-2), (e5), or (e1) replaced with $(\lambda x.M)V' \rhd' M[x := V']$, where $V' ::= x \mid yM$. In fact, it can be shown that if we have $M \rhd' N$, then $G(M) =_{\beta\eta} G(N)$ in $\lambda^{\to\perp}$ with η-reductions, since $G(M[y \Leftarrow N]) =_\beta G(M)[y := \lambda m.G(N)(\lambda n.y(mn))]$, and $G((\{y\}M)N) =_\beta G(\{y\}M[y \Leftarrow N]N)$, and $G((\lambda x.M)V') \rhd^+_{\beta\eta} G(M[x := V'])$, and so on. However, at the current stage, it is not clear that such a computational property holds, in a natural sense, for the full system with respect to G.

Proposition 6 If we have $\Gamma \vdash M : A$ in λ_{exc}, then $\Gamma \vdash G(M) : \neg\neg A$ in $\lambda^{\to\perp}$.

Proof. By induction on the derivation. □

We define an invariant shift reduction relation $\rhd_s$ for LJK-style proofs, which changes an outer invariant into an inner invariant:

$$\{y\!:\!\neg(A \to B)\}M \;\rhd_s\; \lambda x.\{y\!:\!\neg B\}(Mx[y := \lambda k.y(kx)]).$$

The i applications of $\rhd_s$ are denoted by $\rhd^i_s$ for $i = 0, 1, 2, \cdots$.

Let $[A_0, A_1, \cdots, A_n]$ be $CI(A)$. Then we assume on the ordering that $A_0 \equiv A$ is the outermost invariant and A_n is the innermost invariant, and that $A_i \equiv A'_i \to A_{i+1}$ for some A'_i where $0 \leq i \leq n-1$.

Lemma 3 Let $[A_0, A_1, \cdots, A_n]$ be $CI(A)$. If we have $\Gamma \vdash M : A$ in λ_{exc}, then for any i in $0 \leq i \leq n$, M' such that

$$\{y\!:\!\neg A\}\mathrm{raise}(G(M)(\lambda k.yk)) \;\rhd^i_s\; M'$$

is an LJK-style proof of A with the invariant A_i.

Proof. By case analysis on the number of i.

Case of $i = 0$:
If $\Gamma \vdash M : A$ in λ_{exc}, then $\Gamma \vdash G(M) : \neg\neg A$ in $\lambda^{\to\perp}$. Hence, $\{y : \neg A\}\mathrm{raise}(G(M)(\lambda k.yk))$ is an LJK-style proof of A with the invariant $A \equiv A_0$.

Case of $i = k+1$ where $0 \leq k \leq n-1$:
Assume that $\lambda x_1 \cdots x_k.\{y\!:\!\neg A_k\}N$ is an LJK-style proof of A with the invariant A_k, where $A_k \equiv A'_k \to A_{k+1}$. Then $\lambda x_1 \cdots x_k.\{y\!:\!\neg A_k\}N \rhd_s M'$ gives an LJK-style proof of A with the invariant A_{k+1} by the following replacement of each yP in N with $(\lambda k.y(kx))P$:

$$\dfrac{\dfrac{\dfrac{\begin{array}{c}\dfrac{[y : \neg A_k]^1 \quad P : A_k}{yP : \perp}\\ \vdots \\ N : A_k\end{array}}{\{y\!:\!\neg A_k\}N : A_k}\,1}{}}{\lambda x_1 \cdots x_k.\{y\!:\!\neg A_k\}N : A}$$

$\rhd_s$

$$\dfrac{\dfrac{\dfrac{\dfrac{\begin{array}{c}\dfrac{\dfrac{\dfrac{[y : \neg A_{k+1}]^1 \quad \dfrac{[k : A_k]^2 \; [x : A'_k]^3}{kx : A_{k+1}}}{y(kx) : \perp}\,2}{\lambda k.y(kx) : \neg A_k} \quad P : A_k}{(\lambda k.y(kx))P : \perp}\\ \vdots \\ N[y := \lambda k.y(kx)] : A_k \quad [x : A'_k]^3\end{array}}{N[y := \lambda k.y(kx)]x : A_{k+1}}}{\{y\!:\!\neg A_{k+1}\}(N[y := \lambda k.y(kx)]x) : A_{k+1}}\,1}{\lambda x.\{y\!:\!\neg A_{k+1}\}(Nx[y := \lambda k.y(kx)]) : A_k}\,3}{\lambda x_1 \cdots x_k x.\{y\!:\!\neg A_{k+1}\}(Nx[y\!:=\!\lambda k.y(kx)]) : A}$$

□

The formula (invariant) to which the right contraction rules are applied in terms of sequent calculus is changed to the inside by the reduction rules $\rhd_s$. On the other hand, the shift of the invariant is characterized in terms of Theorem 1 on Page 39 of Ref. 30); that is, the application of $(\perp_C)$ can be restricted to atomic formulae. Moreover, with respect to LJK-style proofs with the innermost invariant, the application of (exc) is taken to be a strictly positive and atomic subformula of the conclusion in the implication fragment (possibly with $\wedge$). In the more general case of adding a primitive $\vee$, it would *not* be possible to postulate (exc) only for an atomic formula.

It is stated that $\triangleright_s$ and (e5) have a strong connection, such that $(\{y:\neg(A \to B)\}M)N \triangleright_s (\lambda x.\{y:\neg B\}(M[y := \lambda z.y(zx)]x))N \triangleright_\beta \{y:\neg B\}(M[y := \lambda z.y(zN)]N)$, which leads to the same result as the one by (e5), since we know that $M[y := \lambda z.y(zN)] \triangleright_\beta^* M[y \Leftarrow N]$. That is, the two terms $\{y:\neg(A \to B)\}M$ and $\lambda x.\{y:\neg B\}(M[y := \lambda z.y(zx)]x)$ are extensionally equivalent.

4. Strong Normalization and Church-Rosser Properties

In this section, we prove that λ_{exc} has the strong normalization property and the Church-Rosser property☆. To verify that λ_{exc} is strongly normalizable, we first show the postponement property of (e4) and then show that the reduction relations consisting of (e1), (e2), (e3) and (e5) have the strong normalization property, from which we show that all the reductions are strongly normalizable. Next the Church-Rosser property of λ_{exc} can be derived from the weak Church-Rosser property by using Newman's Lemma. Let $\triangleright_{ijkl}$ be a reduction relation consisting of (ei), (ej), (ek), and (el), where $1 \le i, j, k, l \le 5$.

Lemma 4 If $M \triangleright_4 N_1 \triangleright_{1235} P$, then $M \triangleright_{1235}^+ N_2 \triangleright_4^* P$ for some N_2.

Proof. By induction on the derivations of $\triangleright_4$ and $\triangleright_{1235}$. We show one of the cases; $M_1 N \triangleright_4 M_2 N$ is derived from $M_1 \triangleright_4 M_2$:

Case 1. $M_1 \equiv \{y\}(\lambda x.M_3) \triangleright_4 M_2 \equiv \lambda x.M_3$ where $y \notin FV(M_3)$:
We have $M_1 N \triangleright_4 M_2 N \triangleright_1 M_3[x := N]$. Now we also have $M_1 N \equiv (\{y\}(\lambda x.M_3))N \triangleright_5 \{y\}((\lambda x.M_3)N) \triangleright_5 \{y\}M_3[x := N] \triangleright_4 M_3[x := N]$.

Case 2. $M_1 \equiv \{y\}\mathrm{raise}(M_3) \triangleright_4 M_2 \equiv \mathrm{raise}(M_3)$ where $y \notin FV(M_3)$:
We have $M_1 N \triangleright_4 M_2 N \triangleright_2 \mathrm{raise}(M_3)$. On the other hand, $M_1 N \equiv (\{y\}(\mathrm{raise}(M_3)))N \triangleright_5 \{y\}(\mathrm{raise}(\mathrm{M}_3))N \triangleright_2 \{y\}\mathrm{raise}(M_3) \triangleright_3 \mathrm{raise}(M_3)$.

Case 3. $M_1 \equiv \{y\}(\mathrm{raise}(yM_3)) \triangleright_4 M_2 \equiv \{y\}M_3$:
We have $M_1 N \triangleright_4 M_2 N \triangleright_5 \{y\}(M_3[y \Leftarrow N]N)$. Then $(\{y\}(\mathrm{raise}(yM_3)))N \triangleright_5$
$\{y\}(\mathrm{raise}(y(M_3[y \Leftarrow N]N)))N \triangleright_2$
$\{y\}\mathrm{raise}(y(M_3[y \Leftarrow N]N)) \triangleright_4$
$\{y\}(M_3[y \Leftarrow N]N)$.

Case 4. $M_2 N \triangleright_{1235} M_3 N$ from $M_2 \triangleright_{1235} M_3$:
We have $M_1 N \triangleright_4 M_2 N \triangleright_{1235} M_3 N$. From the induction hypothesis, $M_1 \triangleright_{1235}^+ N' \triangleright_4^* M_3$ for some N'. Then $M_1 N \triangleright_{1235}^+ N'N \triangleright_4^* M_3 N$.

Case 5. $M_2 N \triangleright_{1235} M_2 N_2$ from $N \triangleright_{1235} N_2$:
We have $M_1 N \triangleright_4 M_2 N \triangleright_{1235} M_2 N_2$, and then $M_1 N \triangleright_{1235} M_1 N_2 \triangleright_4 M_2 N_2$.
The remaining cases can be similarly confirmed. □

Next we show by the reducibility method of Girard[12),17)] that $\triangleright_{1235}$ has the strong normalization property. Let $\mathcal{SN}$ be a set of terms M such that all the reductions consisting of (e1), (e2), (e3), and (e5) starting at M are finite.

Definition 4 (Reducibility)
Define a set of reducible terms of type A, $\mathcal{R}(A)$ by induction on the structure of A:
(1) For M of atomic type A, $M \in \mathcal{R}(A)$ iff $M \in \mathcal{SN}$;
(2) For M of type $A \to B$, $M \in \mathcal{R}(A \to B)$ iff $MN \in \mathcal{R}(B)$ for any $N \in \mathcal{R}(A)$.

Lemma 5 Let A be a type. Let $n \ge 0$.
(a) $xN_1 \cdots N_n \in \mathcal{R}(A)$ for $N_i \in \mathcal{SN}$ and N_i of type A_i $(1 \le i \le n)$ and x of type $A_1 \to \cdots \to A_n \to A$.
(b) $\mathcal{R}(A) \subseteq \mathcal{SN}$.

Lemma 6 Let M be a term of type B, and N be a term of type A. Let $N \in \mathcal{R}(A)$ if $x \notin FV(M)$. If $M[x := N] \in \mathcal{R}(B)$, then $(\lambda x.M)N \in \mathcal{R}(B)$.

The above two lemmata can be proved as in Refs. 12) or 17).

Lemma 7 (1) Let M be a term of type A, and y of $\neg A$. If $M \in \mathcal{SN}$, then $yM \in \mathcal{R}(\bot)$. Hence, if $M \in \mathcal{R}(A)$, then $yM \in \mathcal{R}(\bot)$.
(2) Let M be a term of type $\bot$. $M \in \mathcal{R}(\bot)$ iff $\mathrm{raise}(M) \in \mathcal{R}(A)$.
(3) Let $n \ge 0$. Let M be a term of type $A \equiv A_1 \to \cdots \to A_n \to \alpha$, where α is atomic, and let y be a variable of $\neg A$. Let N_i be a term of type A_i where $1 \le i \le n$. $[y \Leftarrow \vec{N}]$ is denoted by $[y \Leftarrow N_1] \cdots [y \Leftarrow N_n]$, where $\vec{N}$ is $N_1 \cdots N_n$. If $M[y \Leftarrow \vec{N}] \in \mathcal{R}(A)$ for any $N_i \in \mathcal{R}(A_i)$, then $\{y\}M \in \mathcal{R}(A)$.

Proof. (1) We prove the first part by double induction on the maximal reduction length from M, denoted by $\nu(M)$, and the term length of M, denoted by $l(M)$. The second part is derived from the first by using Lemma 5. Since $yM \in \mathcal{R}(\bot)$ iff $yM \in \mathcal{SN}$, it is enough to prove

☆ From the isomorphism between $\lambda\mu$ and λ_{exc} obtained in Section 5.1, it would not be essential to establish the fundamental properties of λ_{exc}. However, the "proof" of Church-Rosser for $\lambda\mu$, given in Ref. 26) contains a fatal error. The straightforward use of the Tait & Martin-Löf parallel reduction could not work for proving the Church-Rosser property for λ_{exc} including (e3-2), i.e., renaming rules. See also the footnote to Lemma 8.

that for each N if $yM \triangleright_{1235} N$, then $N \in \mathcal{SN}$. The case analysis on N is as follows:

Case 1. $M \equiv \mathrm{raise}(M_1) \triangleright_{1235} \mathrm{raise}(M_2)$:
We have $yM \triangleright_3 M_1$ with $M_1 \in \mathcal{SN}$ since $M \in \mathcal{SN}$.

Case 2. $M \equiv \{y_1\}M_1 \triangleright_{1235} \{y_1\}M_2$:
We have $yM \triangleright_3 yM_1[y_1 := y]$. From $M \in \mathcal{SN}$, M_1 and $M_1[y_1 := y]$ are in $\mathcal{SN}$. Here, $\nu(M_1) = \nu(M)$ and $l(M_1) < l(M)$. Hence, we have $yM'[y_1 := y] \in \mathcal{SN}$.

Case 3. $M \triangleright_{1235} M_1$:
We have $yM \triangleright_{1235} yM_1$. Since $M_1 \in \mathcal{SN}$ and $\nu(M_1) < \nu(M)$, we have $yM_1 \in \mathcal{SN}$.

(2) and (3) can be verified similarly to Lemma 6. □

We denote by $[\overrightarrow{N}/z]$ one of the following: $[x := N]$ with $N \in \mathcal{R}(A)$, where type of x is A; or $[y \Leftarrow \overrightarrow{N}]$ with $N_i \in \mathcal{R}(A_i)$ $(1 \leq i \leq n)$, where $\overrightarrow{N}$ is $N_1 \cdots N_n$, the type of y is $\neg A$, and $A \equiv A_1 \to \cdots \to A_m$ $(n < m)$.

Proposition 7 Let the type of M be A and $n \geq 1$. Assume that $z_i \not\equiv z_j$ for $i \neq j$ and $z_i \notin \bigcup_{j \neq i} FV(\overrightarrow{N_j})$. Then

$$M[\overrightarrow{N_1}/z_1] \cdots [\overrightarrow{N_n}/z_n] \in \mathcal{R}(A).$$

Proof. By induction on the structure of M. We show the two cases. Let $A \equiv A_1 \to \cdots \to A_m \to \alpha$ where α is atomic $(m \geq 0)$.

Case 1. $M \equiv yM_1$:
From the induction hypothesis, $M_1[\overrightarrow{N_1}/z_1] \cdots [\overrightarrow{N_n}/z_n] \in \mathcal{R}(A)$. If $z_i \equiv y$ for some i, then we also have $M_1[\overrightarrow{N_1}/z_1] \cdots [\overrightarrow{N_i}/z_i] \cdots [\overrightarrow{N_n}/z_n]\overrightarrow{N_i} \in \mathcal{R}(A_{k+1} \to \cdots \to A_m \to \alpha)$ with $N_{ij} \in \mathcal{R}(A_j)$ $(\overrightarrow{N_i} \equiv N_{i1} \cdots N_{ik};\ k \leq m)$ for some A_j. From Lemma 7, $y(M_1[\overrightarrow{N_1}/z_1] \cdots [\overrightarrow{N_i}/z_i] \cdots [\overrightarrow{N_n}/z_n]\overrightarrow{N_i}) \in \mathcal{R}(\bot)$, and hence $(yM_1)[\overrightarrow{N_1}/z_1] \cdots [\overrightarrow{N_n}/z_n] \in \mathcal{R}(\bot)$ since $y \equiv z_i \notin \bigcup_{j \neq i} FV(\overrightarrow{N_j})$. If there is no z_i such that $z_i \equiv y$, then it is straightforward.

Case 2. $M \equiv \{y\}M_1$:
From the induction hypothesis, $M_1[\overrightarrow{N_1}/z_1] \cdots [\overrightarrow{N_n}/z_n][y \Leftarrow \overrightarrow{P}] \in \mathcal{R}(A)$ for $P_i \in \mathcal{R}(A_i)$ where $\overrightarrow{P} \equiv P_1 \cdots P_m$ and $1 \leq i \leq m$. From Lemma 7, we have $\{y\}(M_1[\overrightarrow{N_1}/z_1] \cdots [\overrightarrow{N_n}/z_n]) \in \mathcal{R}(A)$, and hence $(\{y\}M_1)[\overrightarrow{N_1}/z_1] \cdots [\overrightarrow{N_n}/z_n] \in \mathcal{R}(A)$. □

Theorem 2 (Strong Normalization) Well-typed λ_{exc}-terms are strongly normalizable.

Proof. From Proposition 7 and Lemma 5, the reductions consisting of (e1), (e2), (e3), and (e5) are strongly normalizable. Following Lemma 4 and the fact that the length of terms decreases after the reduction of (e4), we can show that λ_{exc} has the strong normalization property. □

Lemma 8 (1) If $M \triangleright_{exc} N$, then $M[x := P] \triangleright_{exc} N[x := P]$.
If $M \triangleright_{exc} N$, then $M[y \Leftarrow P] \triangleright^{+}_{exc} N[y \Leftarrow P]$☆.
(2) If $M \triangleright_{exc} N$, then $P[x := M] \triangleright^{*}_{exc} P[x := N]$.
If $M \triangleright_{exc} N$, then $P[y \Leftarrow M] \triangleright^{*}_{exc} P[y \Leftarrow N]$.

Proof. (1) By induction on the derivation of $\triangleright_{exc}$. (2) By induction on the structure of P. □

Proposition 8 (Weak Church-Rosser Property) If $M \triangleright_{exc} M_1$ and $M \triangleright_{exc} M_2$, then $M_1 \triangleright^{*}_{exc} N$ and $M_2 \triangleright^{*}_{exc} N$ for some N.

Proof. By induction on the derivation of $\triangleright_{exc}$. We show one of the cases; $\{y\}M \triangleright_{exc} \{y\}N$ is derived from $M \triangleright_{exc} N$:

Case 1. $\{y\}M \triangleright_{exc} M$ where $y \notin FV(M)$:
Since $y \notin FV(N)$, we have $\{y\}N \triangleright_{exc} N$. From the assumption, $M \triangleright_{exc} N$.

Case 2. $\{y\}M \equiv \{y\}(\mathrm{raise}(yM_1)) \triangleright_{exc} \{y\}M_1$:
$M \equiv \mathrm{raise}(yM_1) \triangleright_{exc} N \equiv \mathrm{raise}(N_1)$ is derived from $yM_1 \triangleright_{exc} N_1$. For the derivation of $yM_1 \triangleright_{exc} N_1$, there are three cases:

Case 2-1. $yM_1 \equiv y(\mathrm{raise}(M_2)) \triangleright_{exc} N_1 \equiv M_2$:
From the assumption, $\{y\}M_1 \equiv \{y\}\mathrm{raise}(M_2) \equiv \{y\}\mathrm{raise}(N_1) \equiv \{y\}N$.

Case 2-2. $yM_1 \equiv y\{y_1\}M_2 \triangleright_{exc} N_1 \equiv yM_2[y_1 := y]$:
We have $\{y\}N \equiv \{y\}\mathrm{raise}(N_1) \equiv \{y\}\mathrm{raise}(yM_2[y_1 := y]) \triangleright_{exc} \{y\}M_2[y_1 := y]$, and also $\{y\}M_1 \equiv \{y\}\{y_1\}M_2 \equiv \{y\}M_2[y_1 := y]$.

Case 2-3. $yM_1 \triangleright_{exc} N_1 \equiv yN_2$ from $M_1 \triangleright_{exc} N_2$:
We have $\{y\}N \equiv \{y\}\mathrm{raise}(N_1) \equiv \{y\}\mathrm{raise}(yN_2) \triangleright_{exc} \{y\}N_2$, and we also have $\{y\}M_1 \triangleright_{exc} \{y\}N_2$ from $M_1 \triangleright_{exc} N_2$.

Case 3. $\{y\}M \triangleright_{exc} \{y\}N_1$ from $M \triangleright_{exc} N_1$:
From the induction hypothesis, $N \triangleright^{*}_{exc} P$ and $N_1 \triangleright^{*}_{exc} P$ for some P. We have $\{y\}N \triangleright^{*}_{exc} \{y\}P$ and $\{y\}N_1 \triangleright^{*}_{exc} \{y\}P$.

The rest of the cases can be similarly confirmed. □

Theorem 3 (Church-Rosser Theorem) If $M \triangleright^{*}_{exc} M_1$ and $M \triangleright^{*}_{exc} M_2$, then $M_1 \triangleright^{*}_{exc} N$ and $M_2 \triangleright^{*}_{exc} N$ for some N.

Proof. From strong normalization, weak Church-Rosser, and Newman's Lemma: see

☆ Even though one defines parallel reduction $\gg$ as usual for λ_{exc} including (e3-2), we cannot establish that if $M_i \gg N_i$ $(i = 1, 2)$, then $M_1[y \Leftarrow M_2] \gg N_1[y \Leftarrow N_2]$, which is a counterpart of fact (iv) in the proof of Theorem 1[26]. This is why we do not use the method of parallel reductions that was tried in Ref. 26), but instead use Newman's lemma to verify Church-Rosser. See also the observation in the proof of Church-Rosser in Ref. 9).

Ref. 2). □

5. Comparison with Related Work

In the following subsection, we briefly compare λ_{exc} with some of the existing systems of call-by-name style, namely, $\lambda\mu$-calculus [26], λ_Δ [32], and a variant of λ_c [6]. As regards the relation between $\lambda\mu$ and λ_{exc}, we can obtain an isomorphism between them and the strong normalization of λ_{exc}. Our observation on the relation between λ_Δ and λ_{exc} suggests a generalization of some reduction rule of λ_Δ, which can lead to an isomorphism between them. We discuss what kind of reduction rule needs to be added to λ_c to make it isomorphic with λ_{exc}.

5.1 Relation to $\lambda\mu$-Calculus

To study computational interpretations of classical proofs, Parigot [26] introduced $\lambda\mu$-calculus of second order classical natural deduction with multiple conclusions. The $\lambda\mu$-calculus has elegant properties: from a proof-theoretical point of view, in contrast to the well-known NK, $\lambda\mu$ has no operational rules such as double-negation elimination or the absurdity rule, but it has multiple conclusions and structural rules. The positive fragment of $\lambda\mu$ is complete with respect to the positive fragment of classical logic; that is, to prove, for example, Peirce's law, we do not have to use a $\bot$ that is not a subformula of the theorem. On the other hand, in the proof of NK, λ_Δ [32], λ_{exc}, and a variant of $\lambda\mu$ à la Ong [25], we have to use $\bot$, which is not contained in the conclusion. Moreover, since in $\lambda\mu$ the name $[\alpha]$ always appears as the form $[\alpha]M$ for some term M, the notion of regularity in Ref. 1) is involved in the system.

From a computational viewpoint, in $\lambda\mu$ [26]~[28] some proof terms of theorems may contain a free name δ of $\bot$; for instance, the term $\lambda x_1.\mu\alpha.[\delta](x_1(\lambda x_2.\mu\delta.[\alpha]x_2))$ of type $\neg\neg A \to A$ has a free name δ. To keep our usual intention of closed terms, we adopt a variant of $\lambda\mu$-calculus à la Ong [25] and study the relation between $\lambda\mu$ à la Ong and λ_{exc}. At first appearance the $\lambda\mu$-calculus has a single conclusion; however, the remaining conclusions are placed on the left side after the semicolon.

The system of $\lambda\mu$ is defined in the following. The types are defined in the normal manner from atomic types that include $\bot$, using $\to$. The context Γ and terms are defined in the usual way. The set of types with names is denoted by Δ.

$\lambda\mu$:

$\Gamma ::= \langle\,\rangle \mid x\!:\!A, \Gamma;\quad \Delta ::= \langle\,\rangle \mid A^\alpha, \Delta;$

$M ::= x \mid MM \mid \lambda x.M \mid [\alpha]M \mid \mu\alpha.M;$

$$\Gamma;\Delta \vdash x : \Gamma(x)$$

$$\frac{\Gamma, x\!:\!A; \Delta \vdash M : B}{\Gamma;\Delta \vdash \lambda x.M : A \to B}$$

$$\frac{\Gamma;\Delta \vdash M : A \to B \quad \Gamma;\Delta \vdash N : A}{\Gamma;\Delta \vdash MN : B}$$

$$\frac{\Gamma;\Delta \vdash M : A}{\Gamma;\Delta, A^\alpha \vdash [\alpha]M : \bot}$$

$$\frac{\Gamma;\Delta, A^\alpha \vdash M : \bot}{\Gamma;\Delta \vdash \mu\alpha.M : A}$$

The reduction relation $\rhd_\mu$ of β-reductions, structural reductions, $(S1)$, and $(S2)$ in Ref. 27) is considered, namely,

$(\lambda x.M)N \rhd_\mu M[x := N];$

$(\mu\alpha.M)N \rhd_\mu \mu\alpha.M[\alpha \Leftarrow N];$

$(S1)$: $[\alpha](\mu\beta.M) \rhd_\mu M[\beta := \alpha];$

$(S2)$: $\mu\alpha.[\alpha]M \rhd_\mu M$ if $\alpha \notin FreeName(M)$.

The binary relations $\rhd_\mu^*$ and $=_\mu$ are defined in the usual way.

Definition 5
(Translation from λ_{exc} to $\lambda\mu$)
$\underline{x} = x$; $\underline{\lambda x.M} = \lambda x.\underline{M}$; $\underline{yM} = [y]\underline{M}$; $\underline{MN} = \underline{M}\underline{N}$; $\underline{\mathrm{raise}(M)} = \mu\alpha.\underline{M}$, where α is a fresh name; $\underline{\{y\}M} = \mu y.[y]\underline{M}$.

For this translation, we separate a context in λ_{exc} into two parts as follows:

$\Gamma ::= \Gamma_1 \mid \Gamma_2;$

$\Gamma_1 ::= \langle\,\rangle \mid x\!:\!A, \Gamma_1;\quad \Gamma_2 ::= \langle\,\rangle \mid y\!:\!\neg A, \Gamma_2.$

$\underline{y\!:\!\neg A, \Gamma_2} = A^y, \underline{\Gamma_2}.$

Proposition 9 If we have $\Gamma_1, \Gamma_2 \vdash_{\lambda_{exc}} M : A$, then $\Gamma_1; \underline{\Gamma_2} \vdash_{\lambda\mu} \underline{M} : A$.

Lemma 9 If $M \rhd_{exc} N$, then $\underline{M} \rhd_\mu \underline{N}$.

Proof. By induction on the derivation $M \rhd_{exc} N$. □

From Lemma 9, Proposition 9, and the strong normalization of $\lambda\mu$ [27),28)], we can also show that well-typed λ_{exc}-terms are strongly normalizable.

Corollary 2 Well-typed λ_{exc}-terms are strongly normalizable.

Definition 6
(Translation from $\lambda\mu$ to λ_{exc})
$\langle x\rangle = x$; $\langle\lambda x.M\rangle = \lambda x.\langle M\rangle$; $\langle MN\rangle = \langle M\rangle\langle N\rangle$; $\langle[\alpha]M\rangle = \alpha\langle M\rangle$; $\langle\mu\alpha.M\rangle = \{\alpha\}\mathrm{raise}(\langle M\rangle)$. $\langle A^\alpha, \Delta\rangle = \alpha\!:\!\neg A, \langle\Delta\rangle$.

Proposition 10 If $\Gamma;\Delta \vdash_{\lambda\mu} M : A$, then $\Gamma, \langle\Delta\rangle \vdash_{\lambda_{exc}} \langle M\rangle : A$.

Lemma 10 If $M \triangleright_\mu N$, then $\langle M\rangle \triangleright^+_{exc} \langle N\rangle$.

Proof. By induction on the derivation $M \triangleright_\mu N$. □

Lemma 11 (1) For any λ_{exc}-term M, $\langle \underline{M}\rangle \triangleright^*_{exc} M$.
(2) For any $\lambda\mu$-term N, $\underline{\langle N\rangle} \triangleright^*_\mu N$.

Proof. From the definitions of the translations. □

From Lemmata 9, 10, and 11, with respect to conversions there is an isomorphism between λ_{exc} and $\lambda\mu$.

Proposition 11 ($\lambda_{exc} \simeq \lambda\mu$) λ_{exc} and $\lambda\mu$ are isomorphic in the sense that $M =_\mu N$ iff $\langle M\rangle =_{exc} \langle N\rangle$, and that $M =_{exc} N$ iff $\underline{M} =_\mu \underline{N}$.

In terms of the right structural rules of sequent calculus, the operator μ in $\lambda\mu$ works for both the right contraction and the right weakening. In λ_{exc}, on the other hand, the two roles are separated: the right contraction can be simulated by (exc), and the right weakening by $(\bot I)$ and raise. The logical aspect of the operator μ can be split into two primitive aspects of λ_{exc}, which is also computationally justified under the isomorphism, and is used to define proof terms of classical substructural logics in Ref. 10).

5.2 Relation to λ_Δ-Calculus

For the purpose of establishing the Curry-Howard isomorphism in classical logic, Rehof and Sørensen[32] introduced λ_Δ-calculus by restriction of Felleisen's control operator $\mathcal{C}$ to avoid a breakdown of neat properties like the Church-Rosser property. The λ_Δ-calculus is natural and has good properties not only in terms of proof theory but also in terms of typed calculus. In relation to λ_{exc}, λ_Δ-calculus treats both regular proofs and non-regular proofs; in other words, there is no distinction of variables that are bound by λ-abstraction or Δ-abstraction. Of course, any non-regular proof can be translated into a regular proof without changing the assumptions or conclusion, such that each variable y that is abstracted by Δ is replaced with $\lambda x.yx$. To study the relation between λ_Δ and λ_{exc}, we consider the λ_Δ-proofs under this modification.

The definition of λ_Δ[32] is briefly given below. The syntax of λ_Δ-terms is defined as follows:

$M ::= x \mid \lambda x.M \mid MM \mid \Delta x.M$

The reduction rules are defined as (d1), (d2), and (d3) together with β-reductions.

(d1): $(\Delta x.M)N \triangleright \Delta x.M[x := \lambda z.x(zN)]$;

(d2): $\Delta x.xM \triangleright M$ if $x \notin FV(M)$;

(d3): $\Delta x.x(\Delta d.xM) \triangleright M$ if $x, d \notin FV(M)$.

The type inference rules are $(\to I)$, $(\to E)$, and the following $(\bot_c)$.

$$\frac{\Gamma, x{:}A \to \bot \vdash M : \bot}{\Gamma \vdash \Delta x.M : A}\ (\bot_c)$$

Definition 7
(Translation from λ_Δ to λ_{exc})
$x^\circ = x$; $(\lambda x.M)^\circ = \lambda x.M^\circ$; $(MN)^\circ = M^\circ N^\circ$; $(\Delta x.M)^\circ = \{x\}\mathrm{raise}(M^\circ)$.

Proposition 12 (1) If we have $\Gamma \vdash_{\lambda_\Delta} M : A$, then $\Gamma \vdash_{\lambda_{exc}} M^\circ : A$.
(2) If we have $M \triangleright N$ in λ_Δ, then $M^\circ =_{exc} N^\circ$ in λ_{exc}.

The above proposition can be verified by induction. In particular, (2) can be confirmed by using $(M[y := \lambda z.y(zN)])^\circ \triangleright^*_\beta M^\circ[y \Leftarrow N^\circ]$, where to prove (2), in contrast to Lemma 10, the case of (d1) introduces conversions instead of reductions. From (2), equivalent λ_Δ-terms are translated into equivalent λ_{exc}-terms with respect to conversions (i.e., the correctness of the translation).

Definition 8
(Translation from λ_{exc} to λ_Δ)
$x^+ = x$; $(\lambda x.M)^+ = \lambda x.M^+$; $(yM)^+ = yM^+$; $(MN)^+ = M^+N^+$; $(\mathrm{raise}(M))^+ = \Delta d.M^+$ provided $d \notin FV(M)$; $(\{y\}M)^+ = \Delta y.yM^+$.

Proposition 13 If $\Gamma \vdash_{\lambda_\Delta} M : A$, then $\Gamma \vdash_{\lambda_{exc}} M^+ : A$.

As regards the statement that if we have $M \triangleright_{exc} N$, then $M^+ =_\Delta N^+$ in λ_Δ, where $=_\Delta$ is the reflexive, symmetric, and transitive closure of $\triangleright$ in λ_Δ, our reduction rule of (e4-2) fails even if we drop (e3-1) and (e3-2). Our observation suggests adding a new reduction to λ_Δ, instead of (d3), such that $\Delta x.x(\Delta d.M) \triangleright \Delta x.M$, where $d \notin FV(M)$: (d4). Here, the new rule (d4) is a general form of (d3). The dropped (d3) rule can be covered by (d2) and (d4); and moreover, the simulation of Felleisen's λ_c[6] by λ_Δ (call-by-value variant), which is observed in Ref. 32), is not lost. We can then show that $M^+ \triangleright^* N^+$ in λ_Δ if $M \triangleright_{exc} N$ without (e3-1) and (e3-2). Moreover, we know that $(M^+)^\circ \triangleright^*_{exc} M$ and that $(M^\circ)^+ \triangleright^* M$ in λ_Δ with (d4) instead of (d3). Hence, as in Proposition 11, there is an isomorphism between λ_{exc} without (e3-1),(e3-2) and λ_Δ with (d4) instead of (d3).

With respect to the remaining rules (e3-1) and (e3-2), they can be simulated in λ_Δ by using the following rule:
$y(\Delta x.M) \triangleright M[x := y]$, where the type of the

variable y is of the form $A \to \bot$.

All the above modifications of λ_Δ can lead to an isomorphism ($\lambda_\Delta \simeq \lambda_{exc}$).

5.3 Relation to λ_c-Calculus

For reasoning about a call-by-value language, Felleisen, et al.[6),7)] introduced λ_c-calculus extending the type-free λ_v-calculus of Plotkin[29)] with the control operator $\mathcal{C}$ and the abort operator $\mathcal{A}$. Griffin[15)] used the λ_c-calculus to extend the Curry-Howard isomorphism to classical logic from a computational interest. It is noteworthy that λ_c has the usual reduction rules and the computation rules used only at the top-level, which bring the computation of the top-level continuation to a stop. Since de Groote[4)] proved that there is an isomorphism between $\lambda\mu$ and a call-by-name variant of λ_c, the relation may be obvious. However, we observe that the computation rules in λ_c are necessary for simulating some of the compatible rules in λ_{exc}, and that λ_{exc} would be simulated in λ_c with some reduction rule. Following the observations in Refs. 4) and 32), we consider the following call-by-name variant of λ_c. The terms are defined as usual.

$M ::= x \mid \lambda x.M \mid MM \mid \mathcal{F}M$

The reduction rules are β-reduction, (F_L), and (F_{top}):

(F_L): $(\mathcal{F}M)N \rhd \mathcal{F}(\lambda k.M(\lambda f.k(fN)))$;

(F_{top}): $\mathcal{F}M \rhd \mathcal{F}(\lambda k.M(\lambda f.kf))$.

The operator $\mathcal{F}$ has the type $\neg\neg A \to A$, which is a variant of and can be defined by Felleisen's $\mathcal{C}$, see Ref. 32). In addition, the computation rule is (F_T): $\mathcal{F}M \rhd_T M(\lambda x.x)$, which is applied only at the top-level.

Definition 9

(Translation from λ_c to λ_{exc})

$\langle x\rangle = x$; $\langle \lambda x.M\rangle = \lambda x.\langle M\rangle$; $\langle MN\rangle = \langle M\rangle\langle N\rangle$; $\langle \mathcal{F}M\rangle = \{y\}\mathrm{raise}(\langle M\rangle(\lambda x.yx))$.

Proposition 14 (1) If we have $\Gamma \vdash_{\lambda_c} M : A$, then $\Gamma \vdash_{\lambda_{exc}} \langle M\rangle : A$.

(2) If we have $M \rhd N$ in λ_c, then $\langle M\rangle =_{exc} \langle N\rangle$.

The above proposition can be proved by straightforward induction.

Definition 10

(Translation from λ_{exc} to λ_c)

$\overline{x} = x$; $\overline{\lambda x.M} = \lambda x.\overline{M}$; $\overline{yM} = y\overline{M}$; $\overline{MN} = \overline{M}\,\overline{N}$; $\overline{\mathrm{raise}(M)} = \mathcal{F}(\lambda v.\overline{M})$ where v is fresh; $\overline{\{y\}M} = \mathcal{F}(\lambda y.y\overline{M})$.

Proposition 15 If we have $\Gamma \vdash_{\lambda_{exc}} M : A$, then $\Gamma \vdash_{\lambda_c} \overline{M}$.

With respect to the correctness of the translation, the reduction rules (e2) and (e5) can be simulated by (F_L). We also know that $\langle \overline{M}\rangle \rhd^*_{exc} M$. In contrast, the compatible rules (e4-1) and (e4-2) can be simulated by the use of the non-compatible (F_T). Moreover, for (e3-1) and (e3-2), they can be simulated in λ_c by using the following reduction rule F''_R:

$y(\mathcal{F}M) \rhd M(\lambda x.yx)$, where the type of y is of the form $\neg A$.

This reduction rule is a special form of $\mathcal{C}''_R$ in Barbanera and Berardi[3)], which is also used in Ref. 4) to simulate $(S1)$ of $\lambda\mu$. With the help of (F_{top}) and (F''_R), we can show that $\overline{\langle \mathcal{F}M\rangle} = \mathcal{F}(\lambda y.y\mathcal{F}(\lambda v.\overline{\langle M\rangle}(\lambda x.yx)))$
$\rhd \mathcal{F}(\lambda y.(\lambda v.\overline{\langle M\rangle}(\lambda x.yx))(\lambda k.yk))$
$\rhd \mathcal{F}(\lambda y.\overline{\langle M\rangle}(\lambda x.yx))$, and then we have that $\overline{\langle \mathcal{F}M\rangle} = \mathcal{F}M$ in λ_c, which can lead to $\overline{\langle M\rangle} = M$ in λ_c. Hence, there is an isomorphism between λ_c and λ_{exc} without (e4-1) and (e4-2), denoted by $\lambda_c \simeq \lambda_{exc}$, which is consistent with $\lambda_{exc} \simeq \lambda\mu$ (Proposition 11) and $\lambda\mu \simeq \lambda_c$[4)]. However, compared with the proof of $\lambda\mu \simeq \lambda_c$, the proof of $\overline{\langle M\rangle} = M$ in λ_c needs one more reduction rule, namely, F''_R, which would reveal another aspect of the relation between λ_{exc} and $\lambda\mu$.

6. Concluding Remarks

We have provided a simple natural deduction system λ_{exc} of classical propositional logic based on our observations of LJK proofs in sequent calculus, and have demonstrated its proof-theoretical and computational properties. The Church-Rosser and strong normalization properties hold in the calculus, and there is an isomorphism between λ_{exc} and $\lambda\mu$ with respect to conversions. We have shown from the existence of LJK proofs that there is a strict fragment of λ_{exc}, which is complete with respect to classical provability and would serve as a standard form of classical proofs. Here, we observed that the invariant to be applied by the right contraction rules, in term of sequent calculus, computationally corresponds to the type of exceptional parameter, and the type can be specified as a strictly positive subformula with respect to $\to$.

The relation between λ_{exc} and λ_c is not exactly clear. To study this, it would be important to investigate the computational properties of a call-by-value version of λ_{exc}. Such an approach would also be worthwhile from a practical programming viewpoint. The computational properties of the call-by-value version λ^v_{exc} have been studied extensively from a programming viewpoint, and classical proofs as programs, including exceptions, are also demonstrated in Ref. 9). De Groote[5] proposed a simple calculus $\lambda^{\rightarrow}_{exn}$ of a call-by-value system to explain the exception-handling mechanism. The relation of λ^v_{exc} to his calculus is also discussed in Ref. 9).

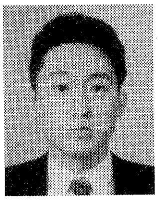
Acknowledgments I would like to thank M. Horai-Takahashi and Y. Andou for helpful comments and discussions. I would also like to thank all of the referees for their most valuable comments.

References

1) Andou, Y.: A Normalization-Procedure for the First Order Classical Natural Deduction with Full Logical Symbols, *Tsukuba Journal of Mathematics*, Vol.19, No.1, pp.153–162 (1995).
2) Barendregt, H.P.: *The Lambda Calculus, Its Syntax and Semantics* (revised edition), North-Holland (1984).
3) Barbanera, F. and Berardi, S.: *Extracting Constructive Context from Classical Logic via Control-like Reductions*, LNCS, Vol.664, pp. 45–59 (1993).
4) de Groote, P.: *On the Relation between the $\lambda\mu$-Calculus and the Syntactic Theory of Sequential Control*, LNAI, Vol.822, pp.31–43 (1994).
5) de Groote, P.: *A Simple Calculus of Exception Handling*, LNCS, Vol.902, pp.201–215 (1995).
6) Felleisen, M., Friedman, D.P., Kohlbecker, E., and Duba, B.: Reasoning with Continuations, *Proc. Annual IEEE Symposium on Logic in Computer Science*, pp.131–141 (1986).
7) Felleisen, M. and Hieb, R.: The Revised Report on the Syntactic Theories of Sequential Control and State, *Theor. Comput. Sci.*, Vol.103, pp.131–141 (1992).
8) Fujita, K.: μ-Head Form Proofs with at Most Two Formulas in the Succedent, *Trans. IPS. Japan*, Vol.38, No.6, pp.1073–1082 (1997).
9) Fujita, K.: *Calculus of Classical Proofs I*, LNCS, Vol.1345, pp.321–335 (1997).
10) Fujita, K.: On Proof Terms and Embeddings of Classical Substructural Logics, *Studia Logica* Vol.61, No.2, pp.199–221 (1998).
11) Fujita, K.: *Polymorphic Call-by-Value Calculus Based on Classical Proofs*, LNAI, Vol.1476, pp.170–182 (1998).
12) Girard, J.-Y., Lafont, Y., Taylor, P.: *Proofs and Types*, Cambridge University Press (1989).
13) Girard, J.-Y.: A New Constructive Logic: Classical Logic, *Math. Struct. Comp. Science*, Vol.1, pp.255–296 (1991).
14) Girard, J.-Y.: On the Unity of Logic, *Annals of Pure and Applied Logic*, Vol.59, pp.201–217 (1993).
15) Griffin, T.G.: A Formulae-as-Types Notion of Control, *Proc. 17th Annual ACM Symposium on Principles of Programming Languages*, pp.47–58 (1990).
16) Hayashi, S. and Nakano, H.: *PX: A Computational Logic*, MIT Press (1988).
17) Hindley, J.R. and Seldin, J.P.: *Introduction to Combinators and λ-Calculus*, Cambridge University Press (1986).
18) Howard, W.: *The Formulae-as-Types Notion of Constructions*, *To H.B. Curry: Essays on combinatory logic, lambda-calculus, and formalism*, pp.479–490, Academic Press (1980).
19) Kobayashi, S.: Monad as Modality, *Theor. Comput. Sci.*, Vol.175, pp.29–74 (1997).
20) Maehara, S.: Eine Darstellung Der Intuitionistischen Logik In Der Klassischen, *Nagoya Mathematical Journal*, pp.45–64 (1954).
21) Millner, R., Tofte, M. and Harper, R.: *The Definition of Standard ML*, MIT Press (1990).
22) Murthy, C.R.: An Evaluation Semantics for Classical Proofs, *Proc. 6th Annual IEEE Symposium on Logic in Computer Science*, pp.96–107 (1991).
23) Nakano, H.: Logical Structures of the Catch and Throw Mechanism, PhD Thesis, University of Tokyo (1995).
24) Nordström, B., Petersson, K., and Smith, J. M.: *Programming in Martin-Löf's Type Theory, An Introduction*, Clarendon Press (1990).
25) Ong, C.-H.L.: A Semantic View of Classical Proofs: Type-Theoretic, Categorical, and Denotational Characterizations, *Linear Logic '96 Tokyo Meeting* (1996).
26) Parigot, M.: *$\lambda\mu$-Calculus: An Algorithmic Interpretation of Classical Natural Deduction*, LNCS, Vol.624, pp.190–201 (1992).
27) Parigot, M.: *Classical Proofs as Programs*, LNCS, Vol.713, pp.263–276 (1993).
28) Parigot, M.: Strong Normalization for Second Order Classical Natural Deduction, *Proc. 8th Annual IEEE Symposium on Logic in Computer Science* (1993).
29) Plotkin, G.: Call-by-Name, Call-by-Value and the λ-Calculus, *Theor. Comput. Sci.*, Vol.1, pp.125–159 (1975).
30) Prawitz, D.: *Natural Deduction: A Proof-Theoretical Study*, Almqvist & Wiksell (1965).
31) Prawitz, D.: Ideas and Results in Proof Theory, *Proc. 2nd Scandinavian Logic Symposium*, Fenstad, N.E. (Ed.), pp.235–307, North-Holland (1971).
32) Rehof, N.J. and Sørensen, M.H.: *The λ_Δ-Calculus*, LNCS, Vol.789, pp.516–542 (1994).
33) Schellinx, H.: Some Syntactical Observations

on Linear Logic, *J. Logic Computat.*, Vol.1, No.4, pp.537–559 (1991).
34) Seldin, J.P.: Normalization and Excluded Middle, I, *Studia Logica XLVIII*, 2, pp.193–217 (1989).
35) Szabo, M.E.: *The Collected Papers of Gerhard Gentzen*, North-Holland (1969).
36) Troelstra, A.S. and van Dalen, D.: *Constructivism in Mathematics, An Introduction*, North-Holland (1988).

(Received October 24, 1997)
(Accepted September 7, 1998)



Ken-etsu Fujita was born in 1959. He received his M.E. and Ph.D. degrees from Tohoku Univ. in 1987 and 1990, respectively. He has been in Kyushu Institute of Technology as a research associate from 1990 to 1994, and as a lecturer since 1994. His current research interests are relationship between type systems and logics, and its application to programming. He is a member of IPSJ, JSSST, JSAI, EATCS, and MSJ.