Available at
www.ElsevierComputerScience.com
POWERED BY SCIENCE DIRECT®

Electronic Notes in Theoretical Computer Science

ELSEVIER Electronic Notes in Theoretical Computer Science 91 (2004) 158–170
www.elsevier.com/locate/entcs

CHR: A Constructive Relevant Natural-deduction Logic

Neil Leslie [1],[2]

Centre for Logic, Language and Computation &
School of Mathematical and Computing Sciences
Victoria University of Wellington
Wellington, New Zealand

Edwin D Mares[3]

Centre for Logic, Language and Computation &
School of History, Philosophy, Political Science and International Relations
Victoria University of Wellington
Wellington, New Zealand

Abstract

In this paper we develop a natural-deduction logic which is both constructive and relevant. We use a proof-theoretic argument to justify the rules of the logic. The detailed framework we use to develop our system is modeled on that used to develop Martin-Löf's Type Theory.

Keywords: Relevant logic, Constructive logic, Proof theory, Natural deduction

1 Introduction and outline

In this paper we develop a natural-deduction logic which is both constructive and relevant. We use a proof-theoretic argument, in the style of Dummett [6] and Prawitz [15], to justify the rules of the logic.

[1] We would like to thank the anonymous referees for their helpful comments.
[2] Email: Neil.Leslie@vuw.ac.nz
[3] Email: Edwin.Mares@vuw.ac.nz

doi:10.1016/j.entcs.2003.12.011

The informal motivation behind relevant logic is to have a notion of valid argument which requires that the premisses *really* be used to infer the conclusion [10]. This imposes restrictions on the form of acceptable proofs in a natural deduction system for a relevant logic. The informal motivation behind constructive logic is that arguments should be *effective* [5]. This condition also imposes a restriction on the form of acceptable proofs. In both cases we are concerned not merely with *what* we can prove, but also in *how* we prove it.

The framework we use to develop our system is modeled on that used to develop Martin-Löf's Type Theory [11,12]. In using the Curry-Howard analogy to analyse the implicational fragments of relevant logics we follow the lead of Gabbay and de Queiroz [7], Helman [2], Meyer, Bunder and Powers [13], and Restall [17]. Furthermore, the use of the Curry-Howard analogy makes this work applicable to programming languages. The notion of *relevance* in a proof is related to the notion of *strictness* of a function [2,3,4].

The paper develops as follows.

- In Section 2 we outline the Brouwer-Heyting-Kolmogorov (BHK) [9] interpretation of intuitionistic logic, and the related proof-theoretic justification of the logical laws. We pay particular attention to the Curry-Howard terms.
- In Section 3 we introduce **CHR**. For brevity, we only present the propositional fragment.
- In Section 4 we outline the proper proof reductions required to show that **CHR** proofs are normalisable.
- In Section 5 we outline the proof that **CHR** is a relevant logic.
- In Section 6 we present some derivations in **CHR**.
- Our conclusions are presented in Section 7.

2 Proof-theoretical justifications of the logical laws

We shall only outline the main points of the process that one goes through to give a proof-theoretical justifications of the logical laws. This process is dealt with with in more detail and with more care by, in particular, Michael Dummett and Dag Prawitz (see, for example, [5,6,15]). The process we use differs slightly from that described by Dummett.

- The argument usually take some rules as being "self-evident", typically single-ended introduction rules meeting the complexity condition [8,6].
- These rules alone do *not* define the meanings of the logical constants. We make two observations. First, consider tonk [16]. The constants tonk and $\vee$ share the same introduction rules, but not the same meaning. Second,

the meanings of the constants of **NK** and **NJ** must be different. But, as Gentzen [8] points out, **NK** is **NJ** + DN. DN is certainly not an introduction rule.

- We claim that the solution for constructivists is to seek harmony in computation [5,12,15]. Formally, in Martin-Löf's Type Theory we use the computation rules to further specify the meanings of the connectives.
- The form of the elimination rule for each connective is then defined by the combination of introduction and computation rules.
- Consequently we will present slightly different computation rules from those used for intuitionistic logic, and hence produce a natural constructive relevant logic.

So, the pattern for each connective we deal with will be:

- introduction rule or rules, taken as self-evident;
- computation rule or rules, taken to embody relevance;
- elimination rule, drawn from the introduction and computation rules.

This pattern is just that used when presenting Martin-Löf's Type Theory [11,14]. Our syntax is presented using aritied expressions, as used by Martin-Löf. The detailed development of a theory of aritied expressions is given in [14], and we will not repeat this here. Similarly our inference rules are presented in a natural deduction format modeled on that of [11]. Our computation rules also follow the format of those in [11].

The following table gives the BHK interpretation of intuitionistic propositional logic. The direct proof tells us what the introduction rule or rules will look like, and the Curry-Howard terms provide us with formal proof objects for valid propositions. Another way to see the Curry-Howard terms is that they are labels which just name the proof. This table is really the starting point for the proof-theoretical justification of the laws of intuitionistic logic, and forms the basis for extending this technique to other logics.

Proposition	Direct proof	Curry-Howard term
$\perp$	-	-
$A \vee B$	A proof of A or B, and an indication of which.	$\mathsf{inl}(a)$ or $\mathsf{inr}(b)$
$A \,\&\, B$	A proof of A and a proof of B.	$\mathsf{pair}(a, b)$
$A \supset B$	A method to construct a proof of B given a proof of A.	$\lambda(b)$

We employ the following definition:

$$\neg A =_{\mathrm{def}} A \supset \bot$$

3 Curry-Howard Relevant logic: CHR

Now we start to develop **CHR**. The obvious starting point is that proofs of relevant implications are *strict* functions. By this we mean that we can only abstract over variables which actually occur inside the term. Note that this is not exactly the notion of strictness used in functional programming where we treat strictness of a function f as the property that $f(\bot)$ is equal to $\bot$, where $\bot$ here is a non-terminating term.

3.1 Relevant implication

For clarity, we use $\xrightarrow{\mathsf{rel}}$ for relevant implication and λ_s for strict function abstraction. Figure 1 is then the introduction rule for relevant implication. Notice the side-condition.

$$\frac{\begin{array}{c}[x:A]\\ \vdots\\ b(x):B\end{array}}{\lambda_s(b):A \xrightarrow{\mathsf{rel}} B}$$

Side condition: $b(x)$ is *strict* in x.

Fig. 1. Relevant implication introduction

We can see that there is no *normal* proof[4] of $A \xrightarrow{\mathsf{rel}} B \xrightarrow{\mathsf{rel}} A$. Figure 2 is a normal-form proof of $A \supset B \supset A$ in intuitionistic logic, where we see that the formal proof object involves vacuous abstraction.

$$\dfrac{\dfrac{[x:A]^1}{\lambda((y)x):B\supset A}\supset \mathrm{I}}{\lambda((x)\lambda((y)x)):A\supset(B\supset A)}\supset I^1$$

Fig. 2. A non-relevant proof

The rule for $\mathsf{funsplit}$ computation is given by Figure 3[5].

[4] Of course this does not preclude their being a proof with a detour, as we have not yet proved that all proofs are normalisable.

[5] We have an η rule at the level of syntax, so b is syntactically equal to $(x)b(x)$ (x not free in b) [14]. In practice, one only ever abstracts over one of the arguments to a non-

$$\frac{f \longrightarrow \lambda_s(b) \qquad d(b) \longrightarrow d'}{\mathsf{funsplit}(d, f) \longrightarrow d'} \ \mathsf{funsplit}\ \text{Comp.}$$

Fig. 3. $\mathsf{funsplit}$ computation

The rule for $\xrightarrow{\mathsf{rel}}$ elimination, following the pattern of Π-elimination from [11], is given by Figure 5.

$$\frac{f : A \xrightarrow{\mathsf{rel}} B \qquad \begin{array}{c}[y(x) : B[x : A]] \\ \vdots \\ a(y) : C\end{array}}{\mathsf{funsplit}(a, f) : C} \ \xrightarrow{\mathsf{rel}} \text{Elim.}$$

Fig. 4. $\xrightarrow{\mathsf{rel}}$ elimination

We can make the following definition:

$$\mathsf{apply}(f, x) =_{\mathrm{def}} \mathsf{funsplit}((y)(y(x)), f)$$

Figure 5 shows how we evaluate $\mathsf{apply}(f, a)$. Evaluation of $b(a)$ will involve evaluation of a.

$$\frac{\dfrac{f \longrightarrow \lambda_s(b) \qquad \dfrac{b(a) \longrightarrow c}{(y)(y(a))b \longrightarrow c} \equiv, \beta}{\mathsf{funsplit}((y)(y(a)), f) \longrightarrow c} \ \mathsf{funsplit}\ \text{Comp.}}{\mathsf{apply}(f, a) \longrightarrow c} =_{\mathrm{def}}$$

Fig. 5. Evaluating $\mathsf{apply}(f, a)$

We can use apply to give ourselves a derived elimination rule for $\xrightarrow{\mathsf{rel}}$, after the fashion of *Modus Ponens*. We use this rule in Figures 27, 28, 29, 31, and 32, where we label it $\xrightarrow{\mathsf{rel}}$ *E'*.

3.2 Conjunctions

We have both an intensional and an extensional conjunction. Figure 6 is the rule for *intensional* conjunction introduction. There is no side-condition on this rule.

Figure 7 is rule for *extensional* conjunction introduction. This rule has the side condition that the proofs of both conjuncts share the same free variables.

canonical constant, the one which has the type being eliminated. If this is the right-most argument we can make expressions less cluttered. Hence, for all the non-canonical constants we adopt the convention that the argument which has the type being eliminated appears

$$\frac{a : A \qquad b : B}{\langle a, b \rangle : A \bullet B}$$

Fig. 6. Intensional conjunction introduction

$$\frac{a : A \qquad b : B}{\mathsf{pair}(a, b) : A \wedge B}$$
$$\text{Side condition: } \mathsf{FV}(a) = \mathsf{FV}(b).$$

Fig. 7. Extensional conjunction introduction

The difference between $\bullet$ and $\wedge$ is further shown in the elimination rules, and this reflects the difference in the computation rules for the two kinds of splitting we have. The rule for $\mathsf{split}_\mathsf{i}$ computation is given in Figure 8, and the rule for $\mathsf{split}_\mathsf{e}$ computation is given in Figure 9. The difference comes about from whether c is strict or not.

$$\frac{p \longrightarrow \langle a, b \rangle \qquad a \longrightarrow a' \qquad b \longrightarrow b' \qquad c(a', b') \longrightarrow c'}{\mathsf{split}_\mathsf{i}(c, p) \longrightarrow c'}\ \mathsf{split}_\mathsf{i}\ \text{Comp.}$$

Fig. 8. $\mathsf{split}_\mathsf{i}$ computation

$$\frac{p \longrightarrow \mathsf{pair}(a, b) \qquad c(a, b) \longrightarrow c'}{\mathsf{split}_\mathsf{e}(c, p) \longrightarrow c'}\ \mathsf{split}_\mathsf{e}\ \text{Comp.}$$

Fig. 9. $\mathsf{split}_\mathsf{e}$ computation

Figures 10 and 11 are the elimination rules for $\bullet$ and $\wedge$, respectively. In $\bullet$ elimination the side-condition is that c is strict in both its arguments, in $\wedge$ elimination c may be non-strict in both arguments.

$$\frac{p : A \bullet B \qquad \begin{matrix}[x : A, y : B] \\ \vdots \\ c(x, y) : C\end{matrix}}{\mathsf{split}_\mathsf{i}(c, p) : C}\ \bullet\ \text{Elim.}$$
$$\text{Side condition: } c \text{ is strict in both arguments.}$$

Fig. 10. $\bullet$ elimination

Figure 12 is a derivation of $A \wedge B \vdash A \bullet B$.

as the right-most argument.

$$\dfrac{p : A \wedge B \qquad \begin{array}{c}[x : A, y : B]\\ \vdots \\ c(x, y) : C\end{array}}{\mathsf{split_e}(c, p) : C} \wedge \text{Elim.}$$

Fig. 11. $\wedge$ elimination

$$\dfrac{\dfrac{p : A \wedge B \qquad [x : A]^1}{\mathsf{split_e}((x, y)x, p) : A} \wedge \text{E.}^1 \qquad \dfrac{p : A \wedge B \qquad [y : B]^2}{\mathsf{split_e}((x, y)y, p) : B} \wedge \text{E.}^2}{\langle \mathsf{split_e}((x, y)x, p), \mathsf{split_e}((x, y)y, p)\rangle : A \bullet B} \bullet \text{I.}$$

Fig. 12. $A \wedge B \vdash A \bullet B$

We can show now that there is no *normal* proof of $A \bullet B \vdash A \wedge B$. Such a proof would either be like Figure 13 or Figure 14. Figure 13 fails because the free variables differ, Figure 14 fails because of the strictness condition.

$$\dfrac{p : A \bullet B \qquad \dfrac{[x : A]^1 \qquad [y : B]^1}{\mathsf{pair}(x, y) : A \wedge B} \wedge \text{I.}}{\mathsf{split_i}((x, y)\mathsf{pair}(x, y), p) : A \wedge B} \bullet \text{E.}^1$$

Fig. 13. A non-proof of $A \bullet B \vdash A \wedge B$

$$\dfrac{\dfrac{p : A \bullet B \qquad [x : A]^1}{\mathsf{split_i}((x, y)x, p) : A} \bullet \text{E.}^1 \qquad \dfrac{p : A \bullet B \qquad [y : B]^2}{\mathsf{split_i}((x, y)y, p) : B} \bullet \text{E.}^2}{\mathsf{pair}(\mathsf{split_i}((x, y)x, p), \mathsf{split_i}((x, y)y, p)) : A \wedge B} \wedge \text{I.}$$

Fig. 14. Another non-proof of $A \bullet B \vdash A \wedge B$

3.3 Disjunction

The rules for $\vee$ introduction are given in Figures 15 and 16, and the computation rules for when are given by Figures 17 and 18.

$$\dfrac{a : A}{\mathsf{inl}(a) : A \vee B} \vee \text{I l}$$

Fig. 15. $\vee$ Intro L

Figure 19 is the elimination rule for $\vee$. This rule has the the side condition that d and e are strict. We may also add the side condition that $\mathsf{FV}(d) = \mathsf{FV}(e)$ to enable us to prove that **CHR** is a relevant logic in the sense of Anderson and

$$\frac{b : B}{\mathsf{inr}(b) : A \vee B} \vee \mathrm{I\ r}$$

Fig. 16. ∨ Intro R

$$\frac{f \longrightarrow \mathsf{inl}(l) \qquad d(l) \longrightarrow d'}{\mathsf{when}(d, e, f) \longrightarrow d'} \mathsf{when}\ \mathrm{Comp}$$

Fig. 17. when computation 1

$$\frac{f \longrightarrow \mathsf{inr}(r) \qquad e(r) \longrightarrow e'}{\mathsf{when}(d, e, f) \longrightarrow e'} \mathsf{when}\ \mathrm{Comp}$$

Fig. 18. when computation 2

Belnap [1]. We do not, however, know that any irrelevances are derivable if this side condition is omitted. Thus we assume the side-condition henceforth.

$$\frac{f : A \vee B \qquad \begin{array}{c}[x : A]\\ \vdots\\ d(x) : C\end{array} \qquad \begin{array}{c}[y : B]\\ \vdots\\ e(y) : C\end{array}}{\mathsf{when}(d, e, f) : C} \vee \mathrm{E}$$

Side condition: d and e are strict

Fig. 19. ∨ elimination

3.4 *The absurd*

There is no introduction rule for $\perp$. The computation rule for $\mathsf{case}_{\{\}}$ is given by Figure 20, and the elimination rule for $\perp$ by Figure 21.

$$\frac{e \longrightarrow e'}{\mathsf{case}_{\{\}}(f) \longrightarrow e'} \mathsf{case}_{\{\}}\ \mathrm{Comp}$$

Fig. 20. $\mathsf{case}_{\{\}}$ computation

$$\frac{f : \perp}{\mathsf{case}_{\{\}}(f) : C} \perp \mathrm{E}$$

Fig. 21. $\perp$ elimination

Relevant negation can then be defined in the expected way.

4 Proof reductions

The normalisation theorem states that a proof with a maximum formula can be converted into a proof of the same thing without a maximum formula. We present the proper proof reductions which give us the core of the normalisation procedure. For brevity we do not present the commuting conversions. As usual we define a single-step of reduction, $\rhd_1$, whose reflexive, transitive closure, $\rhd_*$, is confluent. The relation $\rhd_1$ is derived from the computation rules, of course. The single-step reductions on proof fragments are given in Figures 22 to 26. Again, for brevity, we do not show the confluence of $\rhd_*$ here.

$$\dfrac{\begin{array}{c}[x:A]\\ \vdots\\ \lambda_s(b):A\xrightarrow{\mathsf{rel}}B\end{array}\qquad \begin{array}{c}[y(z):B[z:A]]\\ \vdots\\ d(y):C\end{array}}{\mathsf{funsplit}(d,\lambda_s(b)):C} \quad \rhd_1 \quad \begin{array}{c}[b(x):B[x:A]]\\ \vdots\\ d(b):C\end{array}$$

Fig. 22. Removing a $\xrightarrow{\mathsf{rel}}$ maximum

$$\dfrac{\dfrac{\begin{array}{c}\vdots\\ a:A\end{array}\qquad\begin{array}{c}\vdots\\ b:B\end{array}}{\langle a,b\rangle:A\bullet B}\qquad \begin{array}{c}[x:A,y:B]\\ \vdots\\ d(x,y):C\end{array}}{\mathsf{split_i}(d,\langle a,b\rangle):C} \quad \rhd_1 \quad \begin{array}{c}\begin{array}{cc}\vdots & \vdots\\ a:A & b:B\end{array}\\ \vdots\\ d(a,b):C\end{array}$$

Fig. 23. Removing a $\bullet$ maximum

$$\dfrac{\dfrac{\begin{array}{c}\vdots\\ a:A\end{array}\qquad\begin{array}{c}\vdots\\ b:B\end{array}}{\mathsf{pair}(a,b):A\wedge B}\qquad \begin{array}{c}[x:A,y:B]\\ \vdots\\ d(x,y):C\end{array}}{\mathsf{split_e}(d,\mathsf{pair}(a,b)):C} \quad \rhd_1 \quad \begin{array}{c}\begin{array}{cc}\vdots & \vdots\\ a:A & b:B\end{array}\\ \vdots\\ d(a,b):C\end{array}$$

Fig. 24. Removing a $\wedge$ maximum

5 CHR is a relevant logic

In this section we outline a proof that **CHR** is a relevant logic in the sense of Anderson and Belnap [1]. To prove this we simply assign a distinct number to each variable which occurs in any proof object in a **CHR** proof. We then replace each proof object with the set of numbers that occur in it. By a simple

$$\dfrac{\dfrac{\begin{array}{c}\vdots\\ a : A\end{array}}{\mathsf{inl}(a) : A \vee B} \qquad \begin{array}{c}[x : A]\\ \vdots\\ d(x) : C\end{array} \qquad \begin{array}{c}[y : B]\\ \vdots\\ e(y) : C\end{array}}{\mathsf{when}(d, e, \mathsf{inl}(a)) : C} \quad \rhd_1 \quad \begin{array}{c}a : A\\ \vdots\\ d(a) : C\end{array}$$

Fig. 25. Removing a $\vee$ left maximum

$$\dfrac{\dfrac{\begin{array}{c}\vdots\\ b : B\end{array}}{\mathsf{inr}(b) : A \vee B} \qquad \begin{array}{c}[x : A]\\ \vdots\\ d(x) : C\end{array} \qquad \begin{array}{c}[y : B]\\ \vdots\\ e(y) : C\end{array}}{\mathsf{when}(d, e, \mathsf{inr}(b)) : C} \quad \rhd_1 \quad \begin{array}{c}b : B\\ \vdots\\ e(b) : C\end{array}$$

Fig. 26. Removing a $\vee$ right maximum

induction over the complexity of proofs, we can show that the re-annotated proofs are notational variants of proofs in Anderson and Belnap's natural deduction system for their logic **R** of relevant implication [1].

By a slightly more involved process we can show that the set of theorems of **CHR** is just the set of theorems of the logic **LR** of Thistlewaite, Meyer and McRobbie [18].

6 Examples

In Figures 27 to 32 we give examples of some **CHR** proofs. The theorems proved are:

- Fig. 27 $(A \bullet B) \xrightarrow{\mathsf{rel}} C \vdash A \xrightarrow{\mathsf{rel}} (B \xrightarrow{\mathsf{rel}} C)$
- Fig. 28 $A \xrightarrow{\mathsf{rel}} (B \xrightarrow{\mathsf{rel}} C) \vdash (A \bullet B) \xrightarrow{\mathsf{rel}} C$
- Fig. 29 $A \xrightarrow{\mathsf{rel}} (B \xrightarrow{\mathsf{rel}} C) \vdash B \xrightarrow{\mathsf{rel}} (A \xrightarrow{\mathsf{rel}} C)$
- Fig. 30 $A \bullet B \vdash B \bullet A$
- Fig. 31 $(B \bullet A) \xrightarrow{\mathsf{rel}} C \vdash (A \bullet B) \xrightarrow{\mathsf{rel}} C$
- Fig. 32 $A \xrightarrow{\mathsf{rel}} (A \xrightarrow{\mathsf{rel}} B) \vdash A \xrightarrow{\mathsf{rel}} B$

7 Conclusions

We have presented **CHR** a constructive, relevant natural-deduction logic. We have justified the rules of the logic an a proof-theoretic fashion, based on the

$$\dfrac{\dfrac{\dfrac{\dfrac{[x:A]^1 \quad [y:B]^2}{\langle x,y\rangle : A\bullet B}\bullet\mathrm{I} \qquad f:(A\bullet B)\xrightarrow{\mathsf{rel}} C}{\mathsf{apply}(f,\langle x,y\rangle):C}\xrightarrow{\mathsf{rel}}\mathrm{E'}}{\lambda_s((y)(\mathsf{apply}(f,\langle x,y\rangle))):B\xrightarrow{\mathsf{rel}}C}\xrightarrow{\mathsf{rel}}\mathrm{I}^2}{\lambda_s((x)\lambda_s((y)(\mathsf{apply}(f,\langle x,y\rangle)))):A\xrightarrow{\mathsf{rel}}(B\xrightarrow{\mathsf{rel}}C)}\xrightarrow{\mathsf{rel}}\mathrm{I}^1$$

Fig. 27. $(A\bullet B)\xrightarrow{\mathsf{rel}} C\vdash A\xrightarrow{\mathsf{rel}}(B\xrightarrow{\mathsf{rel}}C)$

$$\dfrac{\dfrac{[p:A\bullet B]^1 \qquad \dfrac{[y:B]^2 \qquad \dfrac{[x:A]^2 \quad f:A\xrightarrow{\mathsf{rel}}(B\xrightarrow{\mathsf{rel}}C)}{\mathsf{apply}(f,x):B\xrightarrow{\mathsf{rel}}C}\xrightarrow{\mathsf{rel}}\mathrm{E'}}{\mathsf{apply}(\mathsf{apply}(f,x),y):C}\xrightarrow{\mathsf{rel}}\mathrm{E'}}{\mathsf{split_i}((x,y)(\mathsf{apply}(\mathsf{apply}(f,x),y))),p):C}\bullet\mathrm{E}^2}{\lambda_s(\mathsf{split_i}((x,y)(\mathsf{apply}(\mathsf{apply}(f,x),y)))):(A\bullet B)\xrightarrow{\mathsf{rel}}C}\xrightarrow{\mathsf{rel}}\mathrm{I}^1$$

Fig. 28. $A\xrightarrow{\mathsf{rel}}(B\xrightarrow{\mathsf{rel}}C)\vdash(A\bullet B)\xrightarrow{\mathsf{rel}}C$

$$\dfrac{\dfrac{\dfrac{[y:B]^1 \qquad \dfrac{[x:A]^2 \quad f:A\xrightarrow{\mathsf{rel}}(B\xrightarrow{\mathsf{rel}}C)}{\mathsf{apply}(f,x):B\xrightarrow{\mathsf{rel}}C}\xrightarrow{\mathsf{rel}}\mathrm{E'}}{\mathsf{apply}(\mathsf{apply}(f,x),y):C}\xrightarrow{\mathsf{rel}}\mathrm{E'}}{\lambda_s((x)(\mathsf{apply}(\mathsf{apply}(f,x),y))):A\xrightarrow{\mathsf{rel}}C}\xrightarrow{\mathsf{rel}}\mathrm{I}^2}{\lambda_s((y)(\lambda_s((x)(\mathsf{apply}(\mathsf{apply}(f,x),y))))):B\xrightarrow{\mathsf{rel}}(A\xrightarrow{\mathsf{rel}}C)}\xrightarrow{\mathsf{rel}}\mathrm{I}^1$$

Fig. 29. $A\xrightarrow{\mathsf{rel}}(B\xrightarrow{\mathsf{rel}}C)\vdash B\xrightarrow{\mathsf{rel}}(A\xrightarrow{\mathsf{rel}}C)$

$$\dfrac{p:A\bullet B \qquad \dfrac{[y:B]^1 \quad [x:A]^1}{\langle y,x\rangle:B\bullet A}\bullet\mathrm{I}}{\mathsf{split_i}((x,y)\langle y,x\rangle,p):B\bullet A}\bullet\mathrm{E}^1$$

Fig. 30. $A\bullet B\vdash B\bullet A$

notion that relevant proof is captured by strict computation. The system we have produced is clear and simple. The proof rules are natural to use, as we have shown in examples. We have outlined how this logic relates to other relevant logics. Because we have developed this logic as a type theory, there is a natural computer science interpretation of propositions in this logic as

$$\dfrac{\dfrac{\dfrac{[p : A \bullet B]^1 \qquad \dfrac{[y : B]^2 \qquad [x : A]^2}{\langle y, x\rangle : B \bullet A}\bullet\,\mathrm{I}}{\mathsf{split_i}((x,y)\langle y,x\rangle, p) : B \bullet A}\bullet\,\mathrm{E}^2 \qquad f : (B \bullet A) \xrightarrow{\mathsf{rel}} C}{\mathsf{apply}(f, \mathsf{split_i}((x,y)\langle y,x\rangle, p)) : C}\xrightarrow{\mathsf{rel}}\mathrm{E'}}{\lambda_s((p)\mathsf{apply}(f, \mathsf{split_i}((x,y)\langle y,x\rangle, p))) : (A \bullet B) \xrightarrow{\mathsf{rel}} C}\xrightarrow{\mathsf{rel}}\mathrm{I}\ ^1$$

Fig. 31. $(B \bullet A) \xrightarrow{\mathsf{rel}} C \vdash (A \bullet B) \xrightarrow{\mathsf{rel}} C$

$$\dfrac{\dfrac{[x : A]^1 \qquad \dfrac{[x : A]^1 \qquad f : A \xrightarrow{\mathsf{rel}} (A \xrightarrow{\mathsf{rel}} B)}{\mathsf{apply}(f, x) : A \xrightarrow{\mathsf{rel}} B}\xrightarrow{\mathsf{rel}}\mathrm{E'}}{\mathsf{apply}(\mathsf{apply}(f, x), x) : B}\xrightarrow{\mathsf{rel}}\mathrm{E'}}{\lambda_s((x)(\mathsf{apply}(\mathsf{apply}(f, x), x))) : A \xrightarrow{\mathsf{rel}} B}\xrightarrow{\mathsf{rel}}\mathrm{I}\ ^1$$

Fig. 32. $A \xrightarrow{\mathsf{rel}} (A \xrightarrow{\mathsf{rel}} B) \vdash A \xrightarrow{\mathsf{rel}} B$

specifying strict functions.

References

[1] Anderson, Alan Ross and Nuel D Belnap, Jr, *Entailment: The Logic of Relevance and Necessity Volume 1*. Princeton University Press, Princeton, New Jersey, USA, 1975.

[2] Anderson, Alan Ross, Nuel D Belnap, Jr and J. Michael Dunn, *Entailment: The Logic of Relevance and Necessity, Volume 2*. Princeton University Press, Princeton, New Jersey, USA, 1992.

[3] Baker-Finch, Clement A., Relevant logic and strictness analysis. In *Workshop on Static Analysis, LaBRI, Bordeaux*, pages 221–228. Bigre 81–82, 1992. Available from http://cs.anu.edu.au/~Clem.Baker-Finch/Research/rlsa/, August 2003.

[4] Baker-Finch, Clement A., Relevance and contraction: A logical basis for strictness and sharing analysis. Technical Report ISE RR 34/94, University of Canberra, 1993. Available from http://cs.anu.edu.au/~Clem.Baker-Finch/Research/rcss/, August 2003.

[5] Dummett, Michael, The philosophical basis of intuitionistic logic. In *Truth and Other Enigmas*, pages 215–247. Duckworth, London, England, 1978. First published in H. E. Rose and J. C. Shepherdson, editors, *Logic Colloquium '73*, North-Holland, Amsterdam, The Netherlands,1975.

[6] Dummett, Michael, *The Logical Basis of Metaphysics*. Duckworth, London, England, 1991.

[7] Gabbay, Dov M. and Ruy J G B de Quieroz, Extending the Curry-Howard interpretation to linear, relevant and other resource logics. *Journal of Symbolic Logic*, 57(4):1319–1365, 1992.

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

[9] Kolmogorov, Andrei, On the interpretation of intuitionistic logic. In Paulo Mancosu, editor, *From Brouwer to Hilbert The Debate on the Foundations of Mathematics in the 1920s*, pages 324–334. Oxford University Press, Oxford, England, 1998. Originally published in German as Zur Deutung der intuitionistichen Logik, *Mathematische Zeitschrift* 35:58–65, 1932.

[10] Mares, Edwin, D., Relevance logic. In Dale Jacquette, editor, *A Companion to Philosophical Logic*, chapter 38, pages 609–627. Blackwell, Oxford, England, 2002.

[11] Martin-Löf, Per, *Intuitionistic Type Theory*, volume 1 of *Studies in Proof Theory Lecture Notes*. Bibliopolis, Napoli, Italy, 1984. Notes taken by Giovanni Sambin from a series of lectures given in Padua, June 1980.

[12] Martin-Löf, Per, On the meanings of the logical constants and the justifications of the logical laws. *Nordic Journal of Philosophical Logic*, 1(1):11–60, 1996. Originally presented at the meeting *Teoria della Dimostrazione e Filosofia della Logica*, Siena 6–9 April, 1983.

[13] Meyer, Robert K., Martin W Bunder and Larry Powers, Implementing the 'Fool's model' of combinatory logic. *Journal of Automated Reasoning*, 7:597–630, 1991.

[14] Nordström, Bengt, Kent Petersson and Jan M Smith, *Programming in Martin-Löf's Type Theory An Introduction*. Clarendon Press, Oxford, England, 1990.

[15] Prawitz, Dag, Meaning and proofs: on the conflict between classical and intuitionistic logic. *Theoria*, 77(1):1–40, 1977.

[16] Prior, Arthur, N., The runabout inference-ticket. *Analysis*, 21:38–39, 1960.

[17] Restall, Greg, *An Introduction to Substructural Logics*. Routledge, London, England, 2000.

[18] Thistlewaite, Paul B., Michael A McRobbie and Robert K Meyer, *Automated theorem-proving in non-classical logics*. Research notes in theoretical computer science. Pitman, London, 1988.