A formulae-as-types interpretation of Subtractive Logic

Tristan Crolard

Laboratory of Algorithmics, Complexity and Logic[1]
University of Paris 12
e-mail: crolard@univ-paris12.fr

February 28, 2003

Abstract. We present a formulae-as-types interpretation of Subtractive Logic (*i.e.* bi-intuitionistic logic). This presentation is two-fold: we first define a very natural restriction of the $\lambda\mu$-calculus which is closed under reduction and whose type system is a constructive restriction of the Classical Natural Deduction. Then we extend this deduction system conservatively to Subtractive Logic. From a computational standpoint, the resulting calculus provides a type system for first-class coroutines (a restricted form of first-class continuations).

Introduction

Subtractive logic, also called "bi-intuistionistic logic", is an extension of intuitionistic logic with a new connector, the *subtraction* (called *pseudo-difference* in Rauszer's original work [40, 41, 42]), which is dual to implication. This duality has already been widely investigated from algebraic, relational, axiomatic and sequent perspectives by various authors. In particular, a connection with category theory is presented in the author's previous work [4, 7, 6], while a unified proof-theoretic approach can be found in Goré's recent paper [20].

We present in this paper a Curry-Howard correspondence for this logic. In other words, we define an extension of the λ-calculus together with its type system such that the logical interpretation of this type system (as a natural deduction system) is exactly subtractive logic. This work is mainly motivated by the following two facts:

- subtractive logic is not a constructive logic (disjunction and existence properties do not hold in it). In fact, subtractive logic already combines many flavors of classical logic [7]. However, subtractive logic is conservative over some constructive logic (see below). In other words, disjunction and existence properties hold for formulas which contain no subtraction. Therefore, it is likely that a faithfull computational interpretation of subtractive logic would shed new light onto the boundary between constructive and classical logics.
- In a recent paper [8], Curien and Herbelin demonstrated a striking result: duality in classical logic exchanges call-by-value with call-by-name. Actually, a first attempt at a categorical continuation semantics was Filinski's pioneering work [16]. Then Selinger [46] has investigated thoroughly this duality (but still in a categorical setting). On the other hand, Curien and Herbelin's work is based on Parigot's $\lambda\mu$-calculus and its type system, the Classical Natural Deduction (CND) [33, 34]. In order to complete the duality, they are led to extend CND with the subtraction. However, this new connective is presented in a purely formal way (without any computational meaning). We believe that an accurate computational interpretation of subtraction will arise naturally from a formulas-as-types investigation of subtractive logic.

1. Université Paris XII - Val de Marne. 61, avenue du Général de Gaulle. 94010 Créteil Cedex France.

Subtractive Logic

Propositionnal subtractive logic is conservative intuitionistic logic. However, in the first-order framework, subtractive logic is no longer convervative over intuitionistic logic but over Constant Domain Logic (CDL). This logic as been studied for instance in [19, 21, 32] and [42]. Although CDL is not conservative over intuitionistic logic (in the first-order framework), it is important to note that CDL is still a constructive logic (*i.e.* both disjunction and existence properties hold in it [21]). Moreover, CDL if fully axiomatized as the first order intuitionistic logic extended with the following axiom schema (called DIS in [42]) where x does not occur in B (see [19, 32] for instance):

$$\forall x(A \vee B) \vdash \forall x\, A \vee B$$

The usual way to define a deduction system (sound and faithfull) for subtractive logic is first to add rules for subtraction (which are symmetrical to the rules for implication) and then to restrict sequents to singletons on the left/right sides [20]. Unfortunately, such a calculus is not closed under reduction. We shall thus define a weaker restriction which takes into account "dependencies" between hypothesis and conclusions of a derived sequent. Since these dependencies are defined in a symmetrical manner, our restriction can be dualized to subtraction. We shall call Symmetric Natural Deduction (SND) the resulting system.

Our method for achieving this goal is, on one hand, to derive introduction and elimination rules for subtraction from its definition in classical logic, and on the other hand, first to restrict the Classical Natural Deduction to intuitionistic logic (resp. CDL) and then to extend it again to subtractive logic. This process is summarized in the following diagram:

$$\begin{tikzcd}
 & \begin{array}{c}\mathrm{CND}_{\to\vee\wedge}\\ \text{classical logic}\end{array} \arrow[dl, "(2)"] \arrow[dr, "(1)"'] & \\
\begin{array}{c}\mathrm{SND}_{\to\vee\wedge}\\ \text{intuitionistic logic}\\ \text{(resp. CDL)}\end{array} \arrow[dr, "(3)"'] & & \begin{array}{c}\mathrm{CND}_{\to\vee\wedge-}\\ \text{classical logic}\end{array} \arrow[dl, "(4)"] \\
 & \begin{array}{c}\mathrm{SND}_{\to\vee\wedge-}\\ \text{subtractive logic}\end{array} &
\end{tikzcd}$$

1. Definition of subtraction in classical logic ($A - B \equiv A \wedge \neg B$).
2. Restriction of $\mathrm{CND}_{\to\vee\wedge}$ to intuitionistic logic (resp. CDL).
3. Extension of $\mathrm{SND}_{\to\vee\wedge}$ with restricted rules for subtraction.
4. Dualized restriction of $\mathrm{CND}_{\to\vee\wedge-}$ to subtractive logic.

Formulae-as-types

The restriction above can be rephrased for the pure (*i.e.* untyped) $\lambda\mu$-calculus, in such a way that the former restriction is exactly the latter when we consider typed terms. We shall call "safe $\lambda\mu^{\to+\times-}$-calculus" (resp. "safe $\lambda\mu^{\to+\times}$-calculus") the subset of restricted $\lambda\mu^{\to+\times-}$-terms (resp. $\lambda\mu^{\to+\times}$-terms). This terminology comes from the computational interpretation of this restriction (see below). The main result of this paper is the proof that these subsets are closed under reduction, which means that they actually define a calculus. The diagram above is thus completed accordingly:

$$\begin{tikzcd}
 & \begin{array}{c}\lambda\mu^{\to+\times}\text{-calculus}\\ \mathrm{CND}_{\to\vee\wedge}\end{array} \arrow[dl] \arrow[dr] & \\
\begin{array}{c}\text{safe } \lambda\mu^{\to+\times}\text{-calculus}\\ \mathrm{SND}_{\to\vee\wedge}\end{array} \arrow[dr] & & \begin{array}{c}\lambda\mu^{\to+\times-}\text{-calculus}\\ \mathrm{CND}_{\to\vee\wedge-}\end{array} \arrow[dl] \\
 & \begin{array}{c}\text{safe } \lambda\mu^{\to+\times-}\text{-calculus}\\ \mathrm{SND}_{\to\vee\wedge-}\end{array} &
\end{tikzcd}$$

Let us now rephrase precisely the relation between safe $\lambda\mu^{\to+\times-}$-terms (resp. safe $\lambda\mu^{\to+\times}$-terms) and proofs of $\mathrm{SND}_{\to\vee\wedge-}$ (resp. $\mathrm{SND}_{\to\vee\wedge}$). This relation can be stated as follows: given the derivation of a typing judgement of a $\lambda\mu^{\to+\times-}$-term (resp. $\lambda\mu^{\to+\times}$-term) t by some sequent in $\mathrm{CND}_{\to\vee\wedge-}$ (resp. $\mathrm{CND}_{\to\vee\wedge}$), if t is safe then this derivation belongs to $\mathrm{SND}_{\to\vee\wedge-}$ (resp. $\mathrm{SND}_{\to\vee\wedge}$). As a consequence, the subject reduction for $\mathrm{CND}_{\to\vee\wedge-}$ (resp. $\mathrm{CND}_{\to\vee\wedge}$) together with the closure under reduction of the safe $\lambda\mu^{\to+\times-}$-calculus (resp. $\lambda\mu^{\to+\times}$-calculus) provides the subject reduction for $\mathrm{SND}_{\to\vee\wedge-}$ (resp. $\mathrm{SND}_{\to\vee\wedge}$).

Strong normalization and the subformula property

Since our calculi are derived from de Groote's $\lambda\mu^{\to\wedge\vee\perp}$, they enjoy the properties listed in [12], namely:

a) all connectives are taken as primitive;

b) normal deductions satisfy the subformula property;

c) the reduction relation is defined by means of local reduction steps;

d) the reduction relation is strongly normalizing ;

e) the reduction relation is Church-Rosser;

f) the reduction relation is defined at the untyped level;

g) the reduction relation satisfies the subject reduction property.

In [40], C. Rauszer defined a sequent calculus for subtractive logic which enjoy cut-elimination (and the subformula property which follows). As a by-product of the previous properties, we obtain a new proof of this result (actually, we prove a stronger result, namely the *strong* normalization). As a corollary of the subformula property, we obtain that the normal form of the deduction of a sequent in $\mathrm{CND}_{\to\vee\wedge-}$ (resp. $\mathrm{SND}_{\to\vee\wedge-}$) which does not contain any occurence of subtraction belongs to $\mathrm{CND}_{\to\vee\wedge}$ (resp. $\mathrm{SND}_{\to\vee\wedge}$).

Computational interpretation

During the past two decades (since Griffin's pionnering work [22]) the extension of the well-known formulas-as-types paradigm to classical logic has been widely investigated for instance by Murthy [28], Barbanera and Berardi [1], Rehof and S⊘rensen [43], de Groote [10, 12], Krivine [27]. We shall consider here Parigot's $\lambda\mu$-calculus mainly because it is confluent and strongly normalizing in the second order framework [35] and also because its extension with a primitive disjunction has already been investigated by Pym, Ritter and Wallen [38, 36, 37] and de Groote [12].

The computational interpretation of classical logic is usually given by a λ-calculus extended with some form of control (such as the famous **call/cc** of Scheme or the catch/throw mechanism of Lisp) or similar formulations of first-class continuation constructs. Continuations are used in denotational semantics to describe control commands such as jumps. They can also be used as a programming technique to simulate backtracking and coroutines. For instance, first-class continuations have been sucessfully used to implement Simula-like cooperative coroutines in Scheme [49, 17, 18]. This approach has been extended in the Standard ML of New Jersey (with the typed counterpart of Scheme's **call/cc** [14]) to provide simple and elegant implementations of light-weight processes (or threads), where concurrency is obtained by having individual threads volontarily suspend themselves [45, 39] (providing time-sliced processes using pre-emptive scheduling requires additional run-time system support [3, 44]). The key point in these implementations is that control operators make it possible to switch between coroutine contexts, where the context of a coroutine is encoded as its continuation.

The definition of a catch/throw mechanism is straightforward in the $\lambda\mu$-calculus: just set $\mathbf{catch}\,\alpha\; t \equiv \mu\alpha[\alpha]t$ and $\mathbf{throw}\,\alpha\; t \equiv \mu\delta[\alpha]t$ where δ is a μ-variable which does not occur in t (see [5] for a study of the sublanguage obtained when we restrict the $\lambda\mu$-terms to these operators). Then a μ-variable α may be reified as the first-class continuation $\lambda x.\mathbf{throw}\,\alpha\, x$. However, the type of such a $\lambda\mu$-term is the excluded-middle $\vdash A; \neg A$. Thus, a continuation cannot be a first-class citizen in a constructive logic. To figure out what kind of restricted use of continuations are allowed in $\mathrm{SND}_{\rightarrow\vee\wedge}$, we observe that in the restricted $\lambda\mu^{\rightarrow+\times}$-calculus, even if continuations are no longer first-class objects, the ability of context-switching remains (in fact, this observation is easier to make in the framework of abstract state machine). However, a context is now a pair $\langle$*environment*, *continuation*$\rangle$. Note that such a pair is exactly what we expect as the context of a coroutine, since a coroutine should not access the local environment (the part of the environment which is not shared) of another coroutine. Consequently, we say that a $\lambda\mu^{\rightarrow+\times}$-term t is "safe with respect to coroutine contexts" (or just "safe" for short) if no coroutines of t access the local environment of another coroutine.

The extension of this interpretation to $\mathrm{SND}_{\rightarrow\vee\wedge-}$ is now straightforward. We already know (from its definition in classical logic) that subtraction is a good candidate to type a pair $\langle$*environment*, *continuation*$\rangle$. $\mathrm{SND}_{\rightarrow\vee\wedge-}$ is thus interpreted as a type system for first-class first-class coroutines. The dual restriction from the safe $\lambda\mu^{\rightarrow+\times-}$-calculus just ensures that one cannot obtain full first-class continuations back from first-class coroutines.

Related work

Nakano, Kameyama and Sato [30, 29, 31, 24, 25, 26] have proposed various logical frameworks that are intended to provide a type system for a lexical variant of the catch/throw mechanism used in functional languages such as Lisp. Moreover, Nakano has shown that it is possible to restrict the catch/throw mechanism in order to stay in an intuitionistic (propositional) framework. However, in their approach, a disjunction is used to type first-class exceptions. In this paper, we generalize these results in several ways:

- We use Parigot's $\lambda\mu$-calculus and its type system, the Classical Natural deduction (CND), which is confluent and strongly normalizing in the second order framework [35].
- We consider a type system *à la* Curry, which allows us to rephrase the above restriction on pure (*i.e.* untyped) $\lambda\mu$-terms, and not only on typed terms as in [31].
- We propose a computational interpretation of our restriction (as "coroutines which do not access the local environment on another coroutine").
- Our restriction is symmetrical and thus easily extends to duality (and provides a natural way to define "first-class coroutines").

Plan of the paper

The remainder of the paper is organized as follows. In section 1, we present the system $\mathrm{SND}_{\to\vee\wedge}$ which is our restriction of $\mathrm{CND}_{\to\vee\wedge}$ to intuitionistic logic (resp. CDL). In section 2, we present $\mathrm{CND}_{\to\vee\wedge-}$ which is our extension of $\mathrm{CND}_{\to\vee\wedge}$ with the subtraction together with $\mathrm{SND}_{\to\vee\wedge-}$ which is our restriction of $\mathrm{CND}_{\to\vee\wedge-}$ to subtractive logic. In section 3, we recall the $\lambda\mu^{\to+\times}$-calculus and we introduce the $\lambda\mu^{\to+\times-}$-calculus (where we restrict ourselves to the propositional framework). In section 4, we define the safe $\lambda\mu^{\to+\times}$-calculus and we prove that it is closed under reduction. Then we present its extension to the safe $\lambda\mu^{\to+\times-}$-calculus and we also prove that it is closed under reduction.

1 Constructive restrictions of Classical Natural Deduction

It is well-known that if we restrict the classical sequent calculus LK [48] to sequents with at most one conclusion we obtain the intuitionistic sequent calculus LJ [48]. As for natural deduction, it was originally presented for sequents having one conclusion and formalized intuitionistic logic. Parigot's Classical Natural Deduction (CND) [33] may be seen as an extension of the natural deduction to sequents with several conclusions. As expected, this extension leads to classical logic. In order to stay in a constructive framework, several authors [2, 13, 15] have suggested to restrict only the introduction rule of implication of LK to sequents with at most one conclusion. The same restriction can be applied to CND:

Definition 1.1. *We call* CND^1 *the system* CND *where the introduction rule of implication is restricted to sequents with at most one conclusion.*

Surprisingly, in the first order framework, CND^1 is no longer conservative over intuitionnistic logic. Instead, we have:

Proposition 1.2. *The system* CND^1 *is conservative over* CDL.

Proof. By induction on the derivation. Conversely, here is a proof of DIS in CND^1:

$$\dfrac{\dfrac{\dfrac{\dfrac{\forall x(A\vee B)\vdash\forall x(A\vee B)}{\forall x(A\vee B)\vdash A\vee B}}{\forall x(A\vee B)\vdash A,B}}{\forall x(A\vee B)\vdash\forall x\,A,B}}{\forall x(A\vee B)\vdash\forall x\,A\vee B}$$

□

Remark 1.3. In the proof above, we used the usual introduction rule for $\forall$. One can check that a system where this rule is also restricted to sequents with at most one conclusion is conservative over intuitionistic logic.

However, this restriction of the introduction rule for implication is not closed under substitution (see section 4.1). We shall thus consider another restriction of CND which is closed under reduction and is larger than CND^1.

Remark 1.4. Parigot's original CND is a second-order logic, in which $\vee$, $\wedge$, $\exists$, $\exists^2$ are definable from $\to$, $\forall$, $\forall^2$. However, since we are interested in the subformula property, we shall restrict ourselves to the propositional (resp. first-order) framework, where any connective (and quantifier) is taken as primitive and where the proof normalization includes permutative conversions [12].

1.1 A restriction of CND via hypothesis/conclusion dependencies

We use undirected links to make explicit all *dependencies* between occurrences of hypothesis and occurrences of conclusions in any derived sequent. A link between some occurrence of an hypothesis Γ_i and some occurrence of a conclusion Δ_j may be represented as follows:

$$\Gamma_1, \ldots \Gamma_i, \ldots \Gamma_n \vdash \Delta_1, \ldots \Delta_j, \ldots \Delta_m$$

In order to annotate sequents with such links, we name any *occurrence* of hypothesis $(x, y, z, \ldots)$ and any *occurrence* of conclusions (α, β, γ, ...) in a sequent. We assume that the name of an hypothesis (resp. a conclusion) never occurs twice in a sequent. The links annotating some sequent $\Gamma_1^{x_1}, \ldots, \Gamma_n^{x_n} \vdash \Delta_1^{\alpha_1}, \ldots, \Delta_m^{\alpha_m}$ provide a subset of $\{x_1, \ldots, x_n\} \times \{\alpha_1, \ldots, \alpha_m\}$ which can be represented by annotating either each conclusion by a set of hypothesis or each hypothesis by a set of conclusions.

Example 1.5. Consider the sequent $A^x, B^y, C^z \vdash D^\alpha, E^\beta, F^\gamma, G^\delta$ together with the dependencies $\{(x, \beta), (x, \delta), (z, \alpha), (z, \beta), (z, \delta)\}$:

$$A, \ B, \ C \ \vdash \ D, \ E, \ F, \ G$$

This annotated sequent can be represented in both forms:

- each conclusion is annotated by a set of hypothesis

$$A^x, B^y, C^z \vdash \{z\}: D, \{x, z\}: E, \{\}: F, \{x, z\}: G$$

- each hypothesis is annotated by a set of conclusions

$$\{\beta, \delta\}: A, \{\}: B, \{\alpha, \beta, \delta\}: C \vdash D^\alpha, E^\beta, F^\gamma, G^\delta$$

1.1.1 Annotations for $\mathrm{CND}_{\to\vee\wedge}$

We shall use here the former representation (*i.e.* each conclusion is annotated by a set of hypothesis). The annotations of a sequent derived in $\mathrm{CND}_{\to\vee\wedge}$ are defined by induction on the derivation (we omit annotations which are not modified by the rule).

Note 1.6. We shall need the following abbreviation:

$$U[V/x] \equiv \begin{cases} U \backslash \{x\} \cup V \text{ if } x \in U \\ U \text{ otherwise} \end{cases}$$

Axioms

$$A^x \vdash \{x\}: A$$

Weakening rules

$$\frac{\Gamma \vdash \Delta}{\Gamma, A^x \vdash \Delta} \qquad \frac{\Gamma \vdash \Delta}{\Gamma \vdash \Delta, \{\}: A}$$

Contraction rules

$$\frac{\Gamma, A^x, A^y \vdash S_1: \Delta_1, \ldots, S_n: \Delta_n}{\Gamma, A^z \vdash S_1[\{z\}/x, \{z\}/y]: \Delta_1, \ldots, S_n[\{z\}/x, \{z\}/y]: \Delta_n} \qquad \frac{\Gamma \vdash \Delta, U: A, V: A}{\Gamma \vdash \Delta, U \cup V: A}$$

Introduction rule for $\to$

$$\frac{\Gamma, A^x \vdash S_1: \Delta_1, \ldots, S_n: \Delta_n, V: B}{\Gamma \vdash S_1 \setminus \{x\}: \Delta_1, \ldots, S_n \setminus \{x\}: \Delta_n, V \setminus \{x\}: (A \to B)}$$

Elimination rule for $\rightarrow$

$$\frac{\Gamma \vdash \Delta, U\colon (A \rightarrow B) \qquad \Gamma' \vdash \Delta', V\colon A}{\Gamma, \Gamma' \vdash \Delta, \Delta', U \cup V\colon B}$$

Remark 1.7. If an hypothesis (resp. a conclusion) occurs with the same name in Γ and Γ' (resp. in Δ and Δ') then these occurrences are implicitly contracted using the left (resp. right) contraction rule. This ensures that the name of an hypothesis (resp. a conclusion) never occurs twice in a sequent. As a special case, when $\Gamma = \Gamma'$ and $\Delta = \Delta'$, the elimination rule for implication becomes:

$$\frac{\Gamma \vdash S_1\colon \Delta_1, \ldots, S_n\colon \Delta_n, U\colon (A \rightarrow B) \qquad \Gamma \vdash S_1'\colon \Delta_1, \ldots, S_n'\colon \Delta_n, V\colon A}{\Gamma, \Gamma' \vdash S_1 \cup S1'\colon \Delta_1, \ldots, S_n \cup S_n'\colon \Delta_n, U \cup V\colon B}$$

Introduction rule for $\wedge$

$$\frac{\Gamma \vdash \Delta, U\colon A \qquad \Gamma \vdash \Delta, V\colon B}{\Gamma \vdash \Delta, U \cup V\colon A \wedge B}$$

Elimination rule for $\wedge$

$$\frac{\Gamma \vdash \Delta, S\colon A \wedge B}{\Gamma \vdash \Delta\colon S\colon A} \qquad \frac{\Gamma \vdash \Delta, S\colon A \wedge B}{\Gamma \vdash \Delta\colon S\colon B}$$

Remark 1.8. It is straightforward to derive the annotated rules for a primitive disjunction by symmetry. For instance, the left-hand side introduction rule for $\vee$ is dual to the introduction rule for $\wedge$ (where this time hypothesis are annotated):

$$\frac{\Gamma, U\colon A \vdash \Delta \qquad \Gamma, V\colon B \vdash \Delta}{\Gamma, U \cup V\colon A \vee B \vdash \Delta}$$

Let us now annotate this rule on the right-hand side:

$$\frac{\Gamma, A^x \vdash S_1\colon \Delta_1, \ldots, S_n\colon \Delta_n, \qquad \Gamma', B^y \vdash S_1'\colon \Delta_1', \ldots, S_p'\colon \Delta_p'}{\Gamma, \Gamma', A \vee B^z \vdash S_1[\{z\}/x]\colon \Delta_1, \ldots, S_n[\{z\}/x]\colon \Delta_n, S_1'[\{z\}/y]\colon \Delta_1', \ldots, S_p'[\{z\}/y]\colon \Delta_p'}$$

Using the cut rule, we eventually obtain the right-hand side annotations for the usual (in natural deduction style) elimination rule for disjunction. The introduction rules for $\vee$ are easier to annotate.

Introduction rule for $\vee$

$$\frac{\Gamma \vdash \Delta; S\colon A}{\Gamma \vdash \Delta; S\colon A \vee B} \qquad \frac{\Gamma \vdash \Delta; S\colon B}{\Gamma \vdash \Delta; S\colon A \vee B}$$

Elimination rule for $\vee$

$$\frac{\Gamma \vdash \Delta, S\colon A \vee B \qquad \Gamma, A^x \vdash S_1\colon \Delta_1, \ldots, S_n\colon \Delta_n \qquad \Gamma', B^y \vdash S_1'\colon \Delta_1', \ldots, S_p'\colon \Delta_p'}{\Gamma, \Gamma', \Gamma'' \vdash \Delta, S_1[S/x]\colon \Delta_1, \ldots, S_n[S/x]\colon \Delta_n, S_1'[S/y]\colon \Delta_1', \ldots, S_p'[S/y]\colon \Delta_p'}$$

Cut rule

$$\frac{\Gamma \vdash S_1\colon \Delta_1, \ldots, S_n\colon \Delta_n, S\colon A \qquad \Gamma', A^x \vdash S_1'\colon \Delta_1', \ldots, S_p'\colon \Delta_p'}{\Gamma, \Gamma' \vdash S_1\colon \Delta_1, \ldots, S_n\colon \Delta_n, S_1'[S/x]\colon \Delta_1', \ldots, S_p'[S/x]\colon \Delta_p'}$$

1.1.2 First-order logic

As usual, we consider a signature which declares functions and relations (each with an arity). Terms are built recursively from individual variables, constants and function symbols and if p is a predicate symbol of arity n and $t_1, \ldots, t_n$ are terms, then $p(t_1, \ldots, t_n)$ is an atomic formula. Formulas are built up from these by closing under the connectives $\wedge$, $\vee$, $\rightarrow$ and universal/existential quantification with respect to any variable.

Remark 1.9. In this paper, we shall not consider the second-order CND. However, rules for second-order universal quantifications are analogous [34] and can be annotated in the same way.

Intro. rule (where x does not occur in Γ, Δ) and elim. rule for $\forall$

$$\frac{\Gamma \vdash \Delta, S\colon A}{\Gamma \vdash \Delta, S\colon \forall x\, A} \qquad\qquad \frac{\Gamma \vdash \Delta, S\colon \forall x\, A}{\Gamma \vdash \Delta, S\colon A[t/x]}$$

Remark 1.10. The rules for $\forall$ leave the annotations untouched. For $\exists$ we thus obtain (by duality) the following left-hand side rules (where x does not occur in Γ, Δ in the left-hand side introduction rule):

$$\frac{\Gamma, S\colon A \vdash \Delta}{\Gamma, S\colon \exists x\, A \vdash \Delta} \qquad\qquad \frac{\Gamma, S\colon \exists x\, A \vdash \Delta}{\Gamma, S\colon A[t/x] \vdash \Delta}$$

Using the cut rule, we obtain the right-hand side annotations for the usual (in natural deduction style) introduction rule for $\exists$:

$$\frac{\Gamma \vdash \Delta, S\colon A[t/x] \qquad \dfrac{\exists x\, A^z \vdash \{z\}\colon \exists x\, A}{A[t/x]^z \vdash \{z\}\colon \exists x\, A}}{\Gamma \vdash \Delta, S\colon \exists x\, A}$$

Similarly, we obtain the right-hand side annotations for the usual elimination rule for $\exists$:

$$\frac{\Gamma \vdash \Delta, S\colon \exists x\, A \qquad \dfrac{\Gamma', A^z \vdash S_1'\colon \Delta_1', \ldots, S_p'\colon \Delta_p', U\colon C}{\Gamma', \exists x\, A^z \vdash S_1'\colon \Delta_1', \ldots, S_p'\colon \Delta_p', U\colon C}}{\Gamma, \Gamma' \vdash \Delta, S_1'[S/z]\colon \Delta_1', \ldots, S_p'[S/z]\colon \Delta_p', U[S/z]\colon C}$$

Intro. rule and elim. rule (where x does not occur in Γ, Δ) for $\exists$

$$\frac{\Gamma \vdash \Delta, S\colon A[t/x]}{\Gamma \vdash \Delta, S\colon \exists x\, A} \qquad\qquad \frac{\Gamma \vdash \Delta, S\colon \exists x\, A \qquad \Gamma', A^z \vdash S_1'\colon \Delta_1', \ldots, S_p'\colon \Delta_p', U\colon C}{\Gamma, \Gamma' \vdash \Delta, S_1'[S/z]\colon \Delta_1', \ldots, S_p'[S/z]\colon \Delta_p', U[S/z]\colon C}$$

Remark 1.11. If we consider the rules for which a left/right symmetry holds (contraction and weakening) we notice that annotations are also symmetrical to one another. For instance, the left contraction rule might have been annotated as follows (where this time hypothesis are annoted):

$$\frac{\Gamma, U\colon A, V\colon A \vdash \Delta}{\Gamma, U \cup V\colon A \vdash \Delta}$$

Remark 1.12. Clearly, any proof of CND can be annotated *a posteriori*.

1.1.3 The system $\mathbf{SND}_{\to\vee\wedge}$

Definition 1.13. *In an annotated proof of* $\mathrm{CND}_{\to\vee\wedge}$*, we say that an instance of the introduction rule for implication is constructive iff x does not occur in any S_i. The rule becomes then:*

$$\frac{\Gamma, A^x \vdash S_1\colon \Delta_1, \ldots, S_n\colon \Delta_n, V\colon B}{\Gamma \vdash S_1\colon \Delta_1, \ldots, S_n\colon \Delta_n, V \setminus \{x\}\colon (A \to B)} \qquad \textit{where } x \notin S_1 \cup \ldots \cup S_n$$

In other words, to stay in a constructive framework, an hypothesis may be discharged over some conclusion if and only if no other conclusion depends on it. This constraint extends to proofs:

Definition 1.14. *A proof of* $\mathrm{CND}_{\to\vee\wedge}$ *is said to be constructive if any occurrence of the introduction rule for implication is constructive. We call* $\mathrm{SND}_{\to\vee\wedge}$ *the restriction of* $\mathrm{CND}_{\to\vee\wedge}$ *to constructive proofs.*

Remark 1.15. Any proof of CND^1 is obviously a proof of SND.

Theorem 1.16. *A sequent is derivable in* $\mathrm{SND}_{\to\vee\wedge}$ *(resp. first-order* $\mathrm{SND}_{\to\vee\wedge}$*) iff it is valid in intuitionistic logic (resp.* CDL*).*

Proof. This theorem is a corollary of theorem 2.4 (corollary 2.5). ☐

Remark 1.17. CDL is a constructive logic. The first (indirect) proof of this result seems to be due to Görnemann [21]. We sketch a direct proof of the existence property using Constant Domain Kripke models in appendix A.

2 Subtractive logic

We first derive the rule for subtraction in the classical framework (we call $\mathrm{CND}_{\to\vee\wedge-}$ the resulting system). Then we show how to restrict this system to obtain subtractive logic. The restricted system is called $\mathrm{SND}_{\to\vee\wedge-}$ (as expected, $\mathrm{SND}_{\to\vee\wedge}$ is the subtraction-free fragment of $\mathrm{SND}_{\to\vee\wedge-}$).

2.1 The system $\mathrm{CND}_{\to\vee\wedge-}$

We first present the introduction/elimination rules for subtraction that we use in the sequel. The introduction rule is dual to the left-hand side introduction rule for implication (in LK), while the elimination rule is reminiscent of the elimination rule for disjunction.

Introduction rule for $-$

$$\frac{\Gamma \vdash \Delta, A \qquad \Gamma', B \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta', A - B}$$

Elimination rule for $-$

$$\frac{\Gamma \vdash \Delta, A - B \qquad \Gamma', A \vdash \Delta', B}{\Gamma, \Gamma' \vdash \Delta, \Delta', C}$$

Remark 2.1. One can obtain rules for subtraction which are dual to rules for implication in CND. As expected, these rules are left-hand side introduction/elimination rules:

$$\frac{\Gamma, A \vdash \Delta, B}{\Gamma, A - B \vdash \Delta} \qquad\qquad \frac{\Gamma, A - B \vdash \Delta \qquad \Gamma', B \vdash \Delta'}{\Gamma, \Gamma', A \vdash \Delta, \Delta'}$$

However these rules are equivalent to the right-hand side introduction/elimination rules for subtraction (the rules of SND). Indeed, they are derivable from the rules of SND:

- Derivation of the left-hand side introduction rule (for C take any Δ_i)

$$\frac{A - B \vdash A - B \qquad \Gamma, A \vdash \Delta, B}{\Gamma, A - B \vdash \Delta}$$

- Derivation of the left-hand side elimination rule

$$\frac{\dfrac{\Gamma', B \vdash \Delta' \qquad A \vdash A}{\Gamma' \vdash \Delta', A - B} \qquad \Gamma, A - B \vdash \Delta}{\Gamma, \Gamma', A \vdash \Delta, \Delta'}$$

Conversely, the rules of SND are derivable from the left-hand side rules. Indeed:

- Derivation of the introduction rule

$$\frac{\Gamma' \vdash \Delta', A \qquad \dfrac{A - B \vdash A - B \qquad \Gamma, B \vdash \Delta}{\Gamma, A \vdash \Delta, A - B}}{\Gamma, \Gamma' \vdash \Delta, \Delta', A - B}$$

- Derivation of the elimination rule

$$\dfrac{\dfrac{\Gamma \vdash \Delta, A-B \qquad \dfrac{\Gamma', A \vdash \Delta', B}{\Gamma', A-B \vdash \Delta'}}{\Gamma, \Gamma' \vdash \Delta, \Delta'}}{\Gamma, \Gamma' \vdash \Delta, \Delta', C}$$

2.1.1 Annotations for $\mathbf{CND}_{\to\vee\wedge-}$

We derive the annotated rules for subtraction by duality (just as we did for disjunction). For instance, the left-hand side elimination rule for $-$ is dual to the introduction rule for $\to$ (where this time hypothesis are annotated):

$$\dfrac{\Gamma, S\colon A-B \vdash \Delta \qquad \Gamma', U\colon B \vdash \Delta'}{\Gamma, \Gamma', S \cup U\colon A \vdash \Delta, \Delta'}$$

Let us now annotate this rule on the right-hand side:

$$\dfrac{\Gamma, A-B^x \vdash S_1\colon \Delta_1, ..., S_n\colon \Delta_n \qquad \Gamma', B^y \vdash S_1'\colon \Delta_1', ..., S_p'\colon \Delta_p'}{\Gamma, \Gamma', A^z \vdash S_1[\{z\}/x]\colon \Delta_1, ..., S_n[\{z\}/x]\colon \Delta_n, S_1'[\{z\}/y]\colon \Delta_1', ..., S_p'[\{z\}/y]\colon \Delta_p'}$$

Using the cut rule, we obtain the right-hand side annotations for the right-hand side introduction rule for subtraction:

$$\dfrac{\Gamma \vdash \Delta, S\colon A \qquad \dfrac{(A-B)^x \vdash \{x\}\colon A-B \qquad \Gamma', B^y \vdash S_1'\colon \Delta_1', ..., S_p'\colon \Delta_p'}{\Gamma', A^z \vdash S_1'[\{z\}/y]\colon \Delta_1', ..., S_p'[\{z\}/y]\colon \Delta_p', \{z\}\colon A-B}}{\Gamma, \Gamma' \vdash \Delta, S_1'[S/y]\colon \Delta_1', ..., S_p'[S/y]\colon \Delta_p', S\colon A-B}$$

Similarly, the left-hand side introduction rule for $-$ is dual to the introduction rule for $\to$ (where this time hypothesis are annotated):

$$\dfrac{S_1\colon \Gamma_1, ..., S_n\colon \Gamma_n, U\colon A \vdash \Delta, B^\beta}{S_1 \setminus \{\beta\}\colon \Gamma_1, ..., S_n \setminus \{\beta\}\colon \Gamma_n, U \setminus \{\beta\}\colon A-B \vdash \Delta}$$

Let us now annotate this rule on the right-hand side:

$$\dfrac{\Gamma, A^x \vdash S_1\colon \Delta_1, ..., S_n\colon \Delta_n, U\colon B}{\Gamma, \Gamma', A-B^z \vdash S_1[\{z\}/x]\colon \Delta_1, ..., S_n[\{z\}/x]\colon \Delta_n}$$

By applying the cut rule and then the weakening rule, we obtain the right-hand side annotations for the right-hand side elimination rule for subtraction:

$$\dfrac{\dfrac{\Gamma' \vdash \Delta', S\colon A-B \qquad \dfrac{\Gamma, A^x \vdash S_1\colon \Delta_1, ..., S_n\colon \Delta_n, U\colon B}{\Gamma, \Gamma', A-B^z \vdash S_1[\{z\}/x]\colon \Delta_1, ..., S_n[\{z\}[/x]\colon \Delta_n}}{\Gamma', \Gamma \vdash \Delta', S_1[S/x]\colon \Delta_1, ..., S_n[S/x]\colon \Delta_n}}{\Gamma', \Gamma \vdash \Delta', S_1[S/x]\colon \Delta_1, ..., S_n[S/x]\colon \Delta_n, \{\}\colon C}$$

Let us recapitulate the annotated rules for subtraction:

Introduction rule for $-$

$$\dfrac{\Gamma \vdash \Delta, S\colon A \qquad \Gamma', B^y \vdash S_1'\colon \Delta_1', ..., S_p'\colon \Delta_p'}{\Gamma, \Gamma' \vdash \Delta, S_1'[S/y]\colon \Delta_1', ..., S_p'[S/y]\colon \Delta_p', S\colon A-B}$$

Elimination rule for $-$

$$\dfrac{\Gamma' \vdash \Delta', S\colon A-B \qquad \Gamma, A^x \vdash S_1\colon \Delta_1, ..., S_n\colon \Delta_n, U\colon B}{\Gamma', \Gamma \vdash \Delta', S_1[S/x]\colon \Delta_1, ..., S_n[S/x]\colon \Delta_n, \{\}\colon C}$$

2.1.2 Restriction of $\mathbf{CND}_{\to\vee\wedge-}$ to subtractive logic

We first define the restriction dual to the one given in definition 1.13:

Definition 2.2. *In an annotated proof of* $\mathrm{CND}_{\to\vee\wedge-}$*, we say that an occurrence of the elimination rule for subtraction constructive iff* $U=\{\}$ *or* $U=\{x\}$*. The rule becomes then:*

$$\frac{\Gamma'\vdash\Delta',S\colon A-B \qquad \Gamma,A^x\vdash S_1\colon\Delta_1,\dots,S_n\colon\Delta_n,U\colon B}{\Gamma',\Gamma\vdash\Delta',S_1[S/x]\colon\Delta_1,\dots,S_n[S/x]\colon\Delta_n,\{\}\colon C}\qquad \textit{where } U=\{\} \textit{ or } U=\{x\}$$

This definition extends to proofs (where we require also the restriction of definition 1.13):

Definition 2.3. *A proof of* $\mathrm{CND}_{\to\vee\wedge-}$ *is said to be constructive iff:*

1. *any occurrence of the introduction rule for implication is constructive*
2. *any occurrence of the elimination rule for subtraction is constructive.*

We call $\mathrm{SND}_{\to\vee\wedge-}$ *the restriction of* $\mathrm{CND}_{\to\vee\wedge-}$ *to constructive proofs.*

The proof of the following theorem (which is rather technical) is given in appendix C.

Theorem 2.4. *A sequent is derivable in* $\mathrm{SND}_{\to\vee\wedge-}$ *(resp. first-order* $\mathrm{SND}_{\to\vee\wedge-}$*) iff it is valid in subtractive logic (resp. first-order subtractive logic).*

Corollary 2.5. *A sequent is derivable in* $\mathrm{SND}_{\to\vee\wedge}$ *(resp. first-order* $\mathrm{SND}_{\to\vee\wedge}$*) iff it is valid in intuitionistic logic (resp.* CDL*).*

3 The typed $\lambda\mu^{\to+\times-}$-calculus

From now on, we shall restrict ourselves to the propositional framework. We briefly present the $\lambda\mu^{\to+\times}$-calculus which is an extension of Parigot's $\lambda\mu$-calculus [33] where disjunction and conjunction are taken as primitives. Then we extend this calculus with a primitive subtraction and thus obtain the $\lambda\mu^{\to+\times-}$-calculus. Note that in this "unsafe" calculus coroutines are exactly as expressive as first-class continuations (since a coroutine's context is just a pair ⟨value, continuation⟩).

3.1 The $\lambda\mu^{\to+\times}$-calculus

Our $\lambda\mu^{\to+\times}$-calculus corresponds to de Groote's $\lambda\mu^{\to\wedge\vee\perp}$-calculus [12] up to minor differences (namely, we augment the syntax with a primitive **let** and we also consider simplification rules as in Parigot's original calculus). We call **var** the set of λ-variables and μ-**var** the set of μ-variables. Raw $\lambda\mu^{\to+\times}$-terms $M,N,\dots$ are constructed by the following grammar:

$$\begin{array}{rcl} M & ::= & x \;\mid\; (M\,N) \;\mid\; \lambda x.M \;\mid\; \mathbf{let}\,x=M\,\mathbf{in}\,N \;\mid\; \mu\alpha M \;\mid\; [\beta]M \\ & \mid & \mathbf{inl}\,M \;\mid\; \mathbf{inr}\,M \;\mid\; \mathbf{case}\,M\,\mathbf{of}\,(\mathbf{inl}\,x)\mapsto P \mid (\mathbf{inr}\,y)\mapsto Q \\ & \mid & \langle M,N\rangle \;\mid\; \mathbf{fst}\,M \;\mid\; \mathbf{snd}\,M \end{array}$$

where x, y range over **var** and α, β range over μ-**var**. Besides, we call *simple* $\lambda\mu^{\to+\times}$*-contexts* the contexts defined by the following grammar:

$$C \;::=\; ([\,]\;t) \;\mid\; \mathbf{fst}\,[\,] \;\mid\; \mathbf{snd}\,[\,] \;\mid\; \mathbf{case}\,[\,]\,\mathbf{of}\,(\mathbf{inl}\,x)\mapsto M \mid (\mathbf{inr}\,y)\mapsto N$$

3.1.1 Reduction rules

Detour-reduction

a) $(\lambda x.u\;t) \rightsquigarrow u\{t/x\}$

b) $\mathbf{fst}\ \langle t, u\rangle \ \leadsto\ t$

c) $\mathbf{snd}\ \langle t, u\rangle \ \leadsto\ u$

d) $\mathbf{case}\ (\mathbf{inl}\ t)\ \mathbf{of}\ (\mathbf{inl}\, x) \mapsto u \mid (\mathbf{inr}\ y) \mapsto v \ \leadsto\ u\{t/x\}$

e) $\mathbf{case}\ (\mathbf{inr}\ t)\ \mathbf{of}\ (\mathbf{inl}\, x) \mapsto u \mid (\mathbf{inr}\ y) \mapsto v \ \leadsto\ v\{t/y\}$

where $u\{t/x\}$ stands for the usual capture-avoiding substitution.

Structural reduction (μ-reduction and permutative rule)

a) $C[\mu\alpha.u] \ \leadsto\ \mu\alpha.u\{[\alpha]C[t]/[\alpha]t\}$

b) $C[\mathbf{case}\ t\ \mathbf{of}\ x \mapsto u \mid y \mapsto v] \ \leadsto\ \mathbf{case}\ t\ \mathbf{of}\ x \mapsto C[u] \mid y \mapsto C[v]$

where $C[\,]$ ranges over simple $\lambda\mu^{\to+\times}$-contexts and $u\{[\alpha]C[t]/[\alpha]t\}$ stands for the structural substitution [12].

Simplification

a) $\mu\alpha[\alpha]u \ \leadsto\ u$ if α does not occur in u

b) $[\beta]\mu\alpha.t \ \leadsto\ t\{\beta/\alpha\}$

Remark 3.1. We shall often refer to the following macro-definitions: **get-context** $\alpha\ t \equiv \mu\alpha[\alpha]t$ and **set-context** $\alpha\ t \equiv \mu\delta[\alpha]t$ where δ is a μ-variable which does not occur in t (see [5] for a study of the sublanguage obtained when we restrict the $\lambda\mu$-terms to these operators, where they are called respectively **catch** and **throw**). Our terminology in this paper comes from [23].

3.2 Typing rules for the $\lambda\mu^{\to+\times}$-calculus

Axiom

$$x{:}\, A^x \vdash A$$

Rules for $\to$

$$\frac{t{:}\, \Gamma, A^x \vdash \Delta; B}{\lambda x.t{:}\, \Gamma \vdash \Delta; A \to B} \qquad \frac{u{:}\, \Gamma \vdash \Delta; A \to B \qquad v{:}\, \Gamma \vdash \Delta; A}{(u\, v){:}\, \Gamma \vdash \Delta; B}$$

Naming rules

$$\frac{t{:}\, \Gamma \vdash \Delta; A}{[\alpha]t{:}\, \Gamma \vdash \Delta, A^\alpha;} \qquad \frac{t{:}\, \Gamma \vdash \Delta, A^\alpha;}{\mu\alpha.t{:}\, \Gamma \vdash \Delta; A}$$

Rules for $\vee$

$$\frac{t{:}\, \Gamma \vdash \Delta; A}{\mathbf{inl}\, t{:}\, \Gamma \vdash \Delta; A \vee B} \qquad \frac{t{:}\, \Gamma \vdash \Delta; B}{\mathbf{inr}\, t{:}\, \Gamma \vdash \Delta; A \vee B}$$

$$\frac{t{:}\, \Gamma \vdash \Delta; A \vee B \qquad u{:}\, \Gamma, A^x \vdash \Delta; C \qquad v{:}\, \Gamma, B^y \vdash \Delta; C}{\mathbf{case}\, t\, \mathbf{of}\, (\mathbf{inl}\, x) \mapsto u \mid (\mathbf{inr}\ y) \mapsto v{:}\, \Gamma \vdash \Delta; C}$$

Rules for $\wedge$

$$\frac{t{:}\, \Gamma \vdash \Delta; A \wedge B}{\mathbf{fst}\ t{:}\, \Gamma \vdash \Delta; A} \qquad \frac{t{:}\, \Gamma \vdash \Delta; A \wedge B}{\mathbf{snd}\ t{:}\, \Gamma \vdash \Delta; B}$$

$$\frac{t{:}\, \Gamma \vdash \Delta; A \qquad u{:}\, \Gamma \vdash \Delta; B}{\langle t, u\rangle{:}\, \Gamma \vdash \Delta; A \wedge B}$$

Cut rule

$$\frac{u\colon \Gamma \vdash \Delta; A \qquad t\colon \Gamma, A^x \vdash \Delta; B}{\mathbf{let}\ x = u\ \mathbf{in}\ t\colon \Gamma \vdash \Delta; B}$$

Remark 3.2. The typing rule for **get-context** and **set-context** are easily derivable. We obtain:

$$\frac{t\colon \Gamma \vdash \Delta; A}{\mathbf{set\text{-}context}\ \alpha\ t\colon \Gamma \vdash \Delta, A^\alpha; B} \qquad \frac{\Gamma \vdash \Delta, A^\alpha; A}{\mathbf{get\text{-}context}\ \alpha\ t\colon \Gamma \vdash \Delta; A}$$

3.3 The $\lambda\mu^{\to+\times-}$-calculus

We present an extension of the $\lambda\mu^{\to+\times}$-calculus with first-class coroutines. This new calculus is called the $\lambda\mu^{\to+\times-}$-calculus (and its type system is $\mathrm{CND}_{\to\vee\wedge-}$, see section 3.4).

3.3.1 Syntax

We add two term constructors to the syntax of the raw $\lambda\mu^{\to+\times}$-calculus:

$$M \ ::= \ \ldots \mid \mathbf{make\text{-}coroutine}\, M\ N \mid \mathbf{resume}\ M\ \mathbf{with}\ x \mapsto N$$

We define the simple $\lambda\mu^{\to+\times-}$-contexts by extending the grammar of simple $\lambda\mu^{\to+\times}$-contexts:

$$C \ ::= \ \ldots \mid \mathbf{resume}\ []\ \mathbf{with}\ x \mapsto N$$

3.3.2 Continuation contexts

Definition 3.3. *Continuation contexts are built inductively from simple $\lambda\mu^{\to+\times-}$contexts.*

Remark 3.4. We shall abbreviate as C_α a pair $(C[\,], \alpha)$ where $C[\,]$ is a continuation context.

3.3.3 Reduction rules

We extend the reduction rules of the $\lambda\mu^{\to+\times}$-calculus with this new detour-reduction rule:

f) $\mathbf{resume}\ (\mathbf{make\text{-}coroutine}\, t\, C_\alpha)\ \mathbf{with}\ x \mapsto u \rightsquigarrow \mathbf{set\text{-}context}\ \alpha\ C[u\{t/x\}]$

and the structural reduction is now defined by the following rules:

a) $C[\mu\alpha.u] \ \rightsquigarrow \ \mu\alpha.u\{[\alpha]C[t]/[\alpha]t\}$

b) $C[\mathbf{case}\ t\ \mathbf{of}\ x \mapsto u \mid y \mapsto v] \ \rightsquigarrow \ \mathbf{case}\ t\ \mathbf{of}\ x \mapsto C[u] \mid y \mapsto C[v]$

c) $C[\mathbf{resume}\ t\ \mathbf{with}\ x \mapsto u] \ \rightsquigarrow \ \mathbf{resume}\ t\ \mathbf{with}\ x \mapsto u$

where $C[\,]$ ranges over simple $\lambda\mu^{\to+\times-}$-contexts.

3.4 Typing rules for coroutines

The typing rules for **make-coroutine** and **resume** are respectively introduction and elimination rules for subtraction:

$$\frac{t\colon \Gamma \vdash \Delta; A \qquad C[x]\colon \Gamma, D^x \vdash \Delta; B}{\mathbf{make\text{-}coroutine}\, t\, C_\beta\colon \Gamma \vdash \Delta, B^\beta; A - D} \qquad \frac{t\colon \Gamma \vdash \Delta; A - B \qquad u\colon \Gamma, A^x \vdash \Delta; B}{\mathbf{resume}\ t\ \mathbf{with}\ x \mapsto u\colon \Gamma \vdash \Delta; C}$$

Remark 3.5. As a typing rule for **make-coroutine**, one would expect the simpler following rule (which corresponds to the case $C[\,] = [\,]$):

$$\frac{t\colon \Gamma \vdash \Delta; A}{\mathbf{make\text{-}coroutine}\, t\, \beta\colon \Gamma \vdash \Delta, B^\beta; A - B}$$

However, we shall see that the generalized rule we propose is needed to obtain a calculus which is closed under the μ-reduction (but from a logical standpoint these rules are equivalent).

3.5 Properties of the $\lambda\mu^{\to+\times-}$-calculus

We shall take advantage of the definability of subtration in classical logic to derive all the properties of the $\lambda\mu^{\to+\times-}$-calculus from the properties of the $\lambda\mu^{\to+\times}$-calculus. Actually, we shall need a $\lambda\mu^{\to+\times}$-calculus with an **abort** instruction [9] and pattern matching: we thus add a bottom $\perp$ and a new product $\otimes$ in our type system.

3.5.1 The $\lambda\mu^{\to+\times\otimes\perp}$-calculus

We add two term constructors to the syntax of the raw $\lambda\mu^{\to+\times}$-calculus:

$$M \ ::= \ ... \ | \quad (M,N) \ | \quad \textbf{match } M \textbf{ with } (x,y) \mapsto N$$

We define the simple $\lambda\mu^{\to+\times\otimes\perp}$-contexts by extending the grammar of simple $\lambda\mu^{\to+\times}$-contexts:

$$C \ ::= \ ... \ | \quad \textbf{match } M \textbf{ with } (x,y) \mapsto N$$

Remark 3.6. It is well-know that an instruction of the form **abort** t can be defined in the $\lambda\mu$-calculus as **set-context** ε t where ε is a free μ-variable. Consequently, we shall not consider here **abort** as a primitive instruction.

3.5.2 Reduction rules

We extend the reduction rules of the $\lambda\mu^{\to+\times}$-calculus with this new detour-reduction rule:

f) $\textbf{match } (u,v) \textbf{ with } (x,y) \mapsto t \rightsquigarrow t\{u/x, y/v\}]$

and the structural reduction is now defined by the following rules:

a) $C[\mu\alpha.u] \ \rightsquigarrow \ \mu\alpha.u\{[\alpha]C[t]/[\alpha]t\}$

b) $C[\textbf{case } t \textbf{ of } x \mapsto u \mid y \mapsto v] \ \rightsquigarrow \ \textbf{case } t \textbf{ of } x \mapsto C[u] \mid y \mapsto C[v]$

c) $C[\textbf{match } t \textbf{ with } (x,y) \mapsto u] \ \rightsquigarrow \ \textbf{match } t \textbf{ with } (x,y) \mapsto C[u]$

where $C[\,]$ ranges over simple $\lambda\mu^{\to+\times\otimes\perp}$-contexts.

3.5.3 Typing rule for $\perp$

$$\frac{t\colon \Gamma \vdash \Delta; \perp}{\textbf{abort } t\colon \Gamma \vdash \Delta; C}$$

where **abort** t is defined as **set-context** ε t and $\perp^{\varepsilon}$ is not represented in the conclusions of sequents. As usual, we also define $\neg A$ as $(A \to \perp)$.

3.5.4 Typing rules for $\otimes$

$$\frac{t\colon \Gamma \vdash \Delta; A \qquad u\colon \Gamma \vdash \Delta; B}{(t,u)\colon \Gamma \vdash \Delta; A \otimes B} \qquad \frac{t\colon \Gamma \vdash \Delta; A \otimes B \qquad u\colon \Gamma, A^x, B^y \vdash \Delta; C}{\textbf{match } t \textbf{ with } (x,y) \mapsto u\colon \Gamma \vdash \Delta; C}$$

The $\lambda\mu^{\to+\times\otimes\perp}$-calculus enjoys all the properties of $\lambda\mu^{\to+\times}$-calculus (listed in the introduction). We show in appendix B how to adapt de Groote's techniques to prove the strong normalization of the $\lambda\mu^{\to+\times\otimes\perp}$-calculus.

3.5.5 Defining $A - B$ as $A \otimes \neg B$

The introduction/elimination rules of $A - B$ are derivable in classical logic when $A - B$ is defined as $A \otimes \neg B$. Indeed:

Derivation of the introduction rule

$$\dfrac{t\colon \Gamma \vdash \Delta; A \qquad \dfrac{\dfrac{C[x]\colon \Gamma, D^x \vdash \Delta; B}{\textbf{set-context}\ \beta\ C[x]\colon \Gamma \vdash \Delta, B^\beta; \bot}}{\lambda x.\textbf{set-context}\ \beta\ C[x]\colon \Gamma \vdash \Delta, B^\beta; \neg D}}{(t, \lambda x.\textbf{set-context}\ \beta\ C[x])\colon \Gamma \vdash \Delta, B^\beta; A - D}$$

Derivation of the elimination rule

$$\dfrac{t\colon \Gamma \vdash \Delta; A \otimes \neg B \qquad \dfrac{\dfrac{k\colon \neg B^k \vdash \neg B \qquad u\colon \Gamma, A^x \vdash \Delta; B}{(k\, u)\colon \Gamma, A^x, \neg B^k \vdash \Delta; \bot}}{\textbf{abort}\ (k\, u)\colon \Gamma, A^x, \neg B^k \vdash \Delta; C}}{\textbf{match}\ t\ \textbf{with}\ (x,k) \mapsto \textbf{abort}\ (k\ u)\colon \Gamma \vdash \Delta; C}$$

We thus obtained the following macro-definitions:

$$\begin{array}{rcl}\textbf{make-coroutine}\, t\, C_\alpha & \equiv & (t, \lambda z.\textbf{set-context}\ \beta\ C[z]) \\ \textbf{resume}\ t\ \textbf{with}\ x \mapsto u & \equiv & \textbf{match}\ t\ \textbf{with}\ (x,k) \mapsto\ \textbf{abort}\ (k\ u)\end{array}$$

If we denote by Φ the translation from the $\lambda\mu^{\to+\times-}$-calculus into the $\lambda\mu^{\to+\times\otimes\bot}$-calculus defined by these macro-definitions, we have to check the following properties:

1. Φ is a morphism for the reduction: if $u \rightsquigarrow v$ then $\Phi(u) \rightsquigarrow^+ \Phi(v)$

2. Φ preserves normal forms: if u is a normal $\lambda\mu^{\to+\times-}$-term then $\Phi(u)$ is a normal $\lambda\mu^{\to+\times\otimes\bot}$-term.

3. Φ is injective on normal forms.

From the first property, we can derive the strong normalization of the $\lambda\mu^{\to+\times-}$-calculus from the strong normalization of the $\lambda\mu^{\to+\times\otimes\bot}$-calculus. From the second and third properties, we obtain the uniqueness of the normal form (*i.e.* the Church-Rosser property). These last properties are easy to check. Let us verify that Φ is a morphism:

- $\textbf{resume}\ (\textbf{make-coroutine}\, t\, C_\alpha)\ \textbf{with}\ x \mapsto u$
  $\equiv \textbf{match}\ (t, C_\alpha)\ \textbf{with}\ (x,k) \mapsto \textbf{abort}\ (k\ u)$
  $\rightsquigarrow \textbf{abort}\ (C_\alpha\ u\{t/x\})$
  $\equiv \textbf{abort}\ (\lambda z.\textbf{set-context}\ \alpha\ C[z]\ u\{t/x\})$
  $\rightsquigarrow \textbf{abort}\ (\textbf{set-context}\ \alpha\ C[u\{t/x\}])$
  $\equiv \textbf{set-context}\ \varepsilon\ \textbf{set-context}\ \alpha\ C[u\{t/x\}]$
  $\rightsquigarrow \textbf{set-context}\ \alpha\ C[u\{t/x\}]$

- $C[\textbf{resume}\ t\ \textbf{with}\ x \mapsto u]$
  $\equiv C[\textbf{match}\ t\ \textbf{with}\ (x,k) \mapsto \textbf{abort}\ (k\ u)]$
  $\rightsquigarrow \textbf{match}\ t\ \textbf{with}\ (x,k) \mapsto C[\textbf{abort}\ (k\ u)]$
  $\equiv \textbf{match}\ t\ \textbf{with}\ (x,k) \mapsto C[\textbf{set-context}\ \varepsilon\ (k\ u)]$
  $\rightsquigarrow \textbf{match}\ t\ \textbf{with}\ (x,k) \mapsto \textbf{set-context}\ \varepsilon\ (k\ u)$
  $\equiv \textbf{resume}\ t\ \textbf{with}\ x \mapsto u$

The remaining rules are straightforward to deal with.

3.5.6 Normal forms and the subformula property

Let us call *applicative contexts* the contexts defined by the following grammar:

$$A\ ::=\ [\,]\ \mid\ (A\ t)\ \mid\ \textbf{fst}\ A\ \mid\ \textbf{snd}\ A$$

In a normal well-typed $\lambda\mu^{\to+\times-}$-term, any occurrence of an application or a projection has the form $A[x]$ (where x is a variable). Indeed, in an application $(u\,v)$ and in a projection $\mathbf{fst}\ u$ or $\mathbf{snd}\ u$, the term u must be either a variable, a projection or again an application (otherwise we obtain a detour-redex or a redex for one of the structural rules). Consequently, in a normal derivation, any occurrence of an elimination rule for $\to$ (resp. $\wedge$) is isomorphic to the following left-hand side introduction rule for $\to$ (resp. $\wedge$):

Left-hand side introduction rules for $\wedge$

$$\frac{A[x]\colon \Gamma, B^x \vdash \Delta; D}{A[\mathbf{fst}\ z]\colon \Gamma, B\wedge C^z \vdash \Delta; D} \qquad\qquad \frac{A[y]\colon \Gamma, C^y \vdash \Delta; D}{A[\mathbf{snd}\ z]\colon \Gamma, B\wedge C^z \vdash \Delta; D}$$

Left-hand side introduction rule for $\to$

$$\frac{A[x]\colon \Gamma, C^x \vdash \Delta; D \qquad t\colon \Gamma \vdash \Delta; B}{A[(z\ t)]\colon \Gamma, (B\to C)^z \vdash \Delta; D}$$

By induction, we show that in the typing judgement of an applicative context $A[x]\colon \Gamma, F^x \vdash \Delta; D$, the formula D is a subformula of F. Moreover, in a normal well-typed $\lambda\mu^{\to+\times-}$-term, in any subterm of the form $\mathbf{case}\ t\ \mathbf{of}\ x\mapsto u \mid y\mapsto v$ or $\mathbf{resume}\ t\ \mathbf{with}\ x\mapsto u$ the term t has the forms $A[z]$ (where z is a variable). Indeed, otherwise we obtain a detour-redex or redex for one of the structural rules. Consequently, any occurrence of an elimination rule for $\vee$ (resp. $-$) has the following form:

Elimination rule for $\vee$

$$\frac{A[z]\colon \Gamma, F^z \vdash \Delta; B\vee C \qquad u\colon \Gamma, B^x \vdash \Delta; D \qquad v\colon \Gamma; C^y \vdash \Delta; D}{\mathbf{case}\ A[z]\ \mathbf{of}\ x\mapsto u \mid y\mapsto v\colon \Gamma, F^z \vdash \Delta; D}$$

Elimination rule for $-$

$$\frac{A[z]\colon \Gamma, F^z \vdash \Delta; A-B \qquad u\colon \Gamma, A^x \vdash \Delta; B}{\mathbf{resume}\ A[z]\ \mathbf{with}\ x\mapsto u\colon \Gamma, F^z \vdash \Delta; C}$$

Then one can easily check that the subformula property hold for these rules.

4 The safe $\lambda\mu$-calculus

From a computational standpoint, the restriction of the introduction rule for $\to$ to at most one conclusion amounts to requiring that in any subterm $\lambda x.u$ of a term t, u is μ-closed (there is no free μ-variable in u). This restriction is clearly not closed under substitution. For instance:

$$\mu\alpha[\alpha](\lambda y\lambda x.y\ \mu\beta[\alpha]\lambda z.z) \rightsquigarrow \mu\alpha[\alpha]\lambda x.\mu\beta[\alpha]\lambda z.z$$

where α is free in $\lambda x.\mu\beta[\alpha]\lambda z.z$ (and thus the contractum is not μ-closed). We define in this section a weaker restriction on $\lambda\mu^{\to+\times}$-terms which is closed under reduction. This restriction as actually been derived from the restriction on sequents given in section 1.1.3. The formal relation between the two definitions is stated by theorem 4.8.

4.1 The safe $\lambda\mu^{\to+\times}$-calculus

In this section we introduce the concept of "scope of a μ-variable with respect to (λ-)variables". This leads to the formal definition of a $\lambda\mu^{\to+\times}$-term in which a coroutine does not access the local environment of another coroutine (see remark 4.7). We shall say that such a $\lambda\mu^{\to+\times}$-term is *safe with respect to coroutine contexts*. The remainding of the section is devoted to the proof of theorem 4.20 which states that the safe $\lambda\mu^{\to+\times}$-calculus is closed under reduction.

Definition 4.1. *We define by induction on t the set $\mathcal{S}_{[]}(t)$ of free λ-variables that occur out of the scope of any μ-variable in t and the set $\mathcal{S}_\delta(t)$ of the λ-variables that occur in the scope of a free μ-variable δ in t:*

- $\mathcal{S}_{[]}(x) = \{x\}$
  $\mathcal{S}_\delta(x) = \emptyset$
- $\mathcal{S}_{[]}(\lambda x.u) = \mathcal{S}_{[]}(u) \backslash \{x\}$
  $\mathcal{S}_\delta(\lambda x.u) = \mathcal{S}_\delta(u) \backslash \{x\}$
- $\mathcal{S}_{[]}(u\, v) = \mathcal{S}_{[]}(u) \cup \mathcal{S}_{[]}(v)$
  $\mathcal{S}_\delta(u\, v) = \mathcal{S}_\delta(u) \cup \mathcal{S}_\delta(v)$
- $\mathcal{S}_{[]}([\alpha]u) = \emptyset$
  $\mathcal{S}_\delta([\alpha]u) = \mathcal{S}_\delta(u)$ *for any* $\delta \neq \alpha$ *and* $\mathcal{S}_\alpha([\alpha]u) = \mathcal{S}_\alpha(u) \cup \mathcal{S}_{[]}(u)$
- $\mathcal{S}_{[]}(\mu\alpha.u) = \mathcal{S}_\alpha(u)$
  $\mathcal{S}_\delta(\mu\alpha.u) = \mathcal{S}_\delta(u)$
- $\mathcal{S}_{[]}(\mathbf{inl}\, u) = \mathcal{S}_{[]}(u)$ *and* $\mathcal{S}_{[]}(\mathbf{inr}\, u) = \mathcal{S}_{[]}(u)$
  $\mathcal{S}_\delta(\mathbf{inl}\, u) = \mathcal{S}_\delta(u)$ *and* $\mathcal{S}_\delta(\mathbf{inr}\, u) = \mathcal{S}_\delta(u)$
- $\mathcal{S}_{[]}(\mathbf{case}\, w\, \mathbf{of}\, (\mathbf{inl}\, x) \mapsto u \mid (\mathbf{inr}\, y) \mapsto v) = \mathcal{S}_{[]}(u)[\mathcal{S}_{[]}(w)/x] \cup \mathcal{S}_{[]}(v)[\mathcal{S}_{[]}(w)/y]$
  $\mathcal{S}_\delta(\mathbf{case}\, w\, \mathbf{of}\, (\mathbf{inl}\, x) \mapsto u \mid (\mathbf{inr}\, y) \mapsto v) = \mathcal{S}_\delta(u)[\mathcal{S}_{[]}(w)/x] \cup \mathcal{S}_\delta(v)[\mathcal{S}_{[]}(w)/y] \cup \mathcal{S}_\delta(w)$
- $\mathcal{S}_{[]}(\langle t, u\rangle) = \mathcal{S}_{[]}(t) \cup \mathcal{S}_{[]}(u)$
  $\mathcal{S}_\delta(\langle t, u\rangle) = \mathcal{S}_\delta(t) \cup \mathcal{S}_\delta(u)$
- $\mathcal{S}_{[]}(\mathbf{fst}\, u) = \mathcal{S}_{[]}(u)$ *and* $\mathcal{S}_{[]}(\mathbf{fst}\, u) = \mathcal{S}_{[]}(u)$
  $\mathcal{S}_\delta(\mathbf{snd}\, u) = \mathcal{S}_\delta(u)$ *and* $\mathcal{S}_\delta(\mathbf{snd}\, u) = \mathcal{S}_\delta(u)$
- $\mathcal{S}_{[]}(\mathbf{let}\, x = v\, \mathbf{in}\, u) = \mathcal{S}_{[]}(u)[\mathcal{S}_{[]}(v)/x]$
  $\mathcal{S}_\delta(\mathbf{let}\, x = v\, \mathbf{in}\, u) = \mathcal{S}_\delta(u)[\mathcal{S}_{[]}(v)/x] \cup \mathcal{S}_\delta(v)$

Definition 4.2. *We extend this definition to arbitrary contexts as follows:*

- $\mathcal{S}_{[]}([\,]) = \emptyset$
  $\mathcal{S}_\delta([\,]) = \emptyset$

Remark 4.3. In the particular case of **set-context** and **get-context**, the previous definition may be rephrased as follows:

- If the $\lambda\mu$-term is **get-context** $\alpha\, u$ ($i.e.$ $\mu\alpha[\alpha]u$) then:
  $\mathcal{S}_{[]}(\mu\alpha[\alpha]u) = \mathcal{S}_\alpha([\alpha]u) = \mathcal{S}_{[]}(u) \cup \mathcal{S}_\alpha(u)$
  $\mathcal{S}_\delta(\mu\alpha[\alpha]u) = \mathcal{S}_\delta([\alpha]u) = \mathcal{S}_\delta(u)$
- If the $\lambda\mu$-term is **set-context** $\alpha\, u$ ($i.e.$ $\mu\beta[\alpha]u$ where β does not occur in $[\alpha]u$) then:
  $\mathcal{S}_{[]}(\mu\beta[\alpha]u) = \mathcal{S}_\beta([\alpha]u) = \emptyset$
  $\mathcal{S}_\delta(\mu\beta[\alpha]u) = \mathcal{S}_\delta([\alpha]u) = \mathcal{S}_\delta(u)$for any $\delta \neq \alpha$ and $\mathcal{S}_\alpha(\mu\beta[\alpha]u) = \mathcal{S}_\alpha([\alpha]u) = \mathcal{S}_\alpha(u) \cup \mathcal{S}_{[]}(u)$

Remark 4.4. Given a $\lambda\mu^{\rightarrow+\times}$-term t, a λ-variable may occur out of the scope of any free μ-variable in t *and also* in the scope of *several* free μ-variables in t. Such a λ-variable is *shared* between several coroutines and the current coroutine.

Definition 4.5. *A $\lambda\mu^{\rightarrow+\times}$-term t is safe with respect to coroutine contexts iff for any subterm of t which has the form $\lambda x.u$, for any free μ-variable δ of u, $x \notin \mathcal{S}_\delta(u)$.*

Remark 4.6. In safe $\lambda\mu^{\rightarrow+\times}$-terms, the usual abbreviation $(\lambda x.t\, u)$ is no longer equivalent to $\mathbf{let}\, x = u\, \mathbf{in}\, t$ since in $(\lambda x.t\, u)$, the λ-variable x may not occur in the scope of some μ-variable in t: this declaration of x is local to the current coroutine.

Remark 4.7. (Krivine's abstract machine for the safe $\lambda\mu^{\rightarrow}$-calculus). In order to justify our terminology, we present briefly how to adapt Krivine's abstract machine [11, 47] to the safe $\lambda\mu$-calculus. However, the properties of this machine and its extension with product, coproduct and first-class coroutines will not be considered in this paper. Our purpose here is just to give the reader some intuition about the kind of optimizations that are enabled by the safe $\lambda\mu^{\rightarrow}$-calculus (from the implementation perspective).

We first recall Krivine's abstract machine for the $\lambda\mu$-calculus (we consider only the version which performs weak-head reduction). We follow the presentation given in [47]. The states of the machine are triples $[t, E, S]$ where t is a term, E is an environment and S is a stack of closures. An environment E is actually a pair $\langle E_\lambda, E_\mu\rangle$, where E_λ is a finite map sending variables to closures, and E_μ is a finite map sending μ-variables to stacks. Formally, the sets **term** of terms, **env** of environments, **clos** of closures, **stack** of stacks (of closures) and **state** of machine states are defined inductively as follows:

$$\begin{array}{lll}
\textbf{(term)} & M & ::= \; x \;\mid\; \lambda x.M \;\mid\; (M\, N) \;\mid\; \mu\alpha M \;\mid\; [\beta]M \\
\textbf{(env)} & E & \in \; (\textbf{var} \rightarrow \textbf{clos}) \times (\mu\text{-}\textbf{var} \rightarrow \textbf{stack}) \\
\textbf{(clos)} & c & ::= \; \langle M, E\rangle \\
\textbf{(stack)} & S & ::= \; () \;\mid\; c :: S \\
\textbf{(state)} & \sigma & ::= \; [M, E, S]
\end{array}$$

The binary transition relation $\rightarrow$ on **state** is given by the following transition rules:

$$\begin{array}{llcll}
\textbf{(var)} & [x, E, S] & \rightarrow & [t, E', S] & \text{where } \langle t, E'\rangle = E(x) \\
\textbf{(fun)} & [\lambda x.t, E, c :: S] & \rightarrow & [\lambda x.t, E\{x := c\}, S] & \\
\textbf{(app)} & [(t\, u), E, S] & \rightarrow & [t, E, \langle u, E\rangle :: S] & \\
\textbf{(swap)} & [\mu\alpha[\beta]t, E, c :: S] & \rightarrow & [t, E, S'] & \text{where } S' = E\{\alpha := S\}(\beta)
\end{array}$$

Towards an abstract machine for coroutines

We now adapt the previous machine to allow for coroutine contexts switching. The resulting machine can be proved to be valid for safe $\lambda\mu^{\rightarrow}$-terms (but we shall not prove this result here). A coroutine context is a pair consisting of the local environment of the coroutine together with a continuation. The second component E_μ of an environment E is now a finite map sending μ-variables to coroutine contexts. To reflect this, we just have to change the definition of the set **env** of environments:

$$\textbf{(env)} \quad E \;\in\; (\textbf{var} \rightarrow \textbf{clos}) \times (\mu\text{-}\textbf{var} \rightarrow (\textbf{env} \times \textbf{stack}))$$

We expect our machine to perform context-switching, which the new intended meaning of the **swap** rule. We thus modify this rule accordingly:

$$\textbf{(swap)} \quad [\mu\alpha[\beta]t, E, c :: S] \;\rightarrow\; [t, E', S'] \;\; \text{where } \langle E', S'\rangle = E\{\alpha := \langle E, S\rangle\}(\beta)$$

One can also derive the rules for **get-context** and **set-context** from the **swap** rule:

$$\begin{array}{lcll}
[\textbf{get-context}\, \alpha\, u, E, S] & \rightarrow & [t, E', S] & \text{where } E' = E\{\alpha := \langle E, S\rangle\} \\
[\textbf{set-context}\, \alpha\, u, E, S] & \rightarrow & [t, E', S'] & \text{where } \langle E', S'\rangle = E(\alpha)
\end{array}$$

Note that the current environment is saved (together with the current continuation) whenever **get-context** is performed. Similarly, the new environment is restored together with the new continuation whenever **set-context** is performed (while both the current environment and the current continuation are discarded).

4.2 Typing safe $\lambda\mu^{\to+\times}$-terms in $\mathrm{SND}_{\to\vee\wedge}$

Proofs of $\mathrm{SND}_{\to\vee\wedge}$ (*i.e.* constructive proofs of $\mathrm{CND}_{\to\vee\wedge}$) and safe $\lambda\mu^{\to+\times}$-terms are related by this theorem (and its corollary):

Theorem 4.8. *Given an annotated derivation, in system* $\mathrm{CND}_{\to\vee\wedge}$*, of some typing judgement* $t:\Gamma_1^{x_1},\ldots,\Gamma_n^{x_n}\vdash\Delta_1^{\alpha_1},\ldots,\Delta_m^{\alpha_m};A$*, if* $x_i\in\mathcal{S}_{\alpha_j}(t)$ *then there is a link between* $\Gamma_i^{x_i}$ *and* $\Delta_j^{\alpha_j}$ *and if* $x_i\in\mathcal{S}_{[]}(t)$ *then there is a link between* $\Gamma_i^{x_i}$ *and* A.

Proof. Just check that we have two inductive definitions of the same relation between hypothesis and conclusions of a sequent (think of [] as the name of the formula which is on the right-hand side of the semi-colon). Take for instance the elimination rule for $\to$:

$$\frac{t:\Gamma,A^{x_i}\vdash\Delta,S_1:C^{\alpha_j},U;(A\to B)\qquad u:\Gamma',A^{x_i}\vdash\Delta',S_2:C^{\alpha_j};V}{(t\,u):\Gamma,\Gamma',A^{x_i}\vdash\Delta,\Delta',S_1\cup S_2:C^{\alpha_j},U\cup V;B}$$

If $x_i\in\mathcal{S}_{\alpha_j}(t\,u)$ then, by definition of $\mathcal{S}$, $x_i\in\mathcal{S}_{\alpha_j}(t)$ or $x_i\in\mathcal{S}_{\alpha_j}(u)$. Thus by induction hypothesis, $x_i\in S_1$ or $x_i\in S_2$. Then $x_i\in S_1\cup S_2$ and there is link between A^{x_i} and C^{α_j}. Similarly, if $x_i\in\mathcal{S}_{[]}(t\,u)$ then, by definition of $\mathcal{S}$, $x_i\in\mathcal{S}_{[]}(t)$ or $x_i\in\mathcal{S}_{[]}(u)$. Thus by induction hypothesis, $x_i\in U$ or $x_i\in V$. Then $x_i\in U\cup V$ and there is a link between A^{x_i} and B. □

Corollary 4.9. *Given a derivation of the typing judgement* $t:\Gamma\vdash\Delta$ *in* $\mathrm{CND}_{\to\vee\wedge}$*, if* t *is safe then the derivation of* $t:\Gamma\vdash\Delta$ *belongs to* $\mathrm{SND}_{\to\vee\wedge}$.

Proof. Let us consider an occurrence of the introduction rule for implication:

$$\frac{u:\Gamma,A^x\vdash\Delta_1^{\alpha_1},\ldots,\Delta_m^{\alpha_m};B}{\lambda x.u:\Gamma\vdash\Delta_1^{\alpha_1},\ldots,\Delta_m^{\alpha_m};A\to B}$$

Since t is safe with respect to coroutine contexts, for any α_j, $x\notin\mathcal{S}_{\alpha_j}(u)$ and then by theorem 5.4, there is no link between A^x and Δ, thus this occurrence of the introduction rule for implication is constructive. □

4.3 Closure under reduction

In the remaining of this section, we prove that the subset of safe $\lambda\mu^{\to+\times}$-terms is closed under the reduction rules of the $\lambda\mu^{\to+\times}$-calculus. We first need a few technical lemmas.

Lemma 4.10. $\mathcal{S}_\delta(t)=\emptyset$ *if* δ *does not occur in* t.

Lemma 4.11. *If* u *and* v *are* $\lambda\mu^{\to+\times}$*-terms then:*

$$\begin{array}{rcl}\mathcal{S}_{[]}(u\{v/x\}) & \subseteq & \mathcal{S}_{[]}(u)[\mathcal{S}_{[]}(v)/x]\\ \mathcal{S}_\delta(u\{v/x\}) & \subseteq & \mathcal{S}_\delta(u)[\mathcal{S}_{[]}(v)/x]\cup\mathcal{S}_\delta(v)\end{array}$$

Lemma 4.12. *If* t *is a* $\lambda\mu^{\to+\times}$*-term then:*

$$\begin{array}{rcl}\mathcal{S}_{[]}(t\{\beta/\alpha\}) & \subseteq & \mathcal{S}_{[]}(t)\\ \mathcal{S}_\beta(t\{\beta/\alpha\}) & \subseteq & \mathcal{S}_\beta(t)\cup\mathcal{S}_\alpha(t)\\ \mathcal{S}_\delta(t\{\beta/\alpha\}) & \subseteq & \mathcal{S}_\delta(t)\text{ for any }\delta\neq\beta\end{array}$$

Lemma 4.13. *If* t *and* v *are* $\lambda\mu^{\to+\times}$*-terms such that* α *does not occur in* v *then:*

$$\begin{array}{rcl}\mathcal{S}_{[]}(t\{[\alpha]C[u]/[\alpha]u\}) & \subseteq & \mathcal{S}_{[]}(t)\\ \mathcal{S}_\alpha(t\{[\alpha]C[u]/[\alpha]u\}) & \subseteq & \mathcal{S}_\alpha(t)\cup\mathcal{S}_{[]}(C[\,])\\ \mathcal{S}_\delta(t\{[\alpha]C[u]/[\alpha]u\}) & \subseteq & \mathcal{S}_\delta(t)\cup\mathcal{S}_\delta(C[\,])\text{ for any }\delta\neq\alpha\end{array}$$

Lemma 4.14. *Let u be a $\lambda\mu^{\to+\times}$-term, let $C[\,]$ be an simple $\lambda\mu^{\to+\times}$-context, and let δ be a μ-variable free in $C[u]$:*

$$\mathcal{S}_{[]}(C[u]) = \mathcal{S}_{[]}(C[\,]) \cup \mathcal{S}_{[]}(u)$$
$$\mathcal{S}_{\delta}(C[u]) = \mathcal{S}_{\delta}(C[\,]) \cup \mathcal{S}_{\delta}(u)$$

Lemma 4.15. *Given an instance $r \rightsquigarrow s$ of a rule of the $\lambda\mu^{\to+\times}$-calculus, if r is safe then $\mathcal{S}_{[]}(s) \subseteq \mathcal{S}_{[]}(r)$ and $\mathcal{S}_{\delta}(s) \subseteq \mathcal{S}_{\delta}(r)$ for any a free μ-variable δ of s.*

Proof. Let us consider each reduction rule of the $\lambda\mu^{\to+\times}$-calculus. Let y be a free λ-variable of s (and thus a free λ-variable of r) and let δ be a free μ-variable δ in s (and thus a free μ-variable δ in r):

Detour-reduction

a) $r = (\lambda x.u\,v)$ and $s = u\{v/x\}$. By lemma 4.11:
$$\mathcal{S}_{[]}(u\{v/x\}) \subseteq \mathcal{S}_{[]}(u)[\mathcal{S}_{[]}(v)/x] \subseteq \mathcal{S}_{[]}(u)\backslash\{x\} \cup \mathcal{S}_{[]}(v) = \mathcal{S}_{[]}(\lambda x.u\,v)$$
$$\mathcal{S}_{\delta}(u\{v/x\}) \subseteq \mathcal{S}_{\delta}(u)[\mathcal{S}_{[]}(v)/x] \cup \mathcal{S}_{\delta}(v) \subseteq \mathcal{S}_{\delta}(u)\backslash\{x\} \cup \mathcal{S}_{\delta}(v) = \mathcal{S}_{\delta}(\lambda x.u\,v)$$
(since r is safe and thus $x \notin \mathcal{S}_{\delta}(u)$)

b) $r = \mathbf{fst}\ \langle t, u\rangle$ and $s = t$.
$$\mathcal{S}_{[]}(t) = \mathcal{S}_{[]}(t) \subseteq \mathcal{S}_{[]}(t) \cup \mathcal{S}_{[]}(u) = \mathcal{S}_{[]}(\mathbf{fst}\ \langle t, u\rangle)$$
$$\mathcal{S}_{\delta}(t) = \mathcal{S}_{\delta}(t) \subseteq \mathcal{S}_{\delta}(t) \cup \mathcal{S}_{\delta}(u) = \mathcal{S}_{\delta}(\mathbf{fst}\ \langle t, u\rangle)$$

c) Similar to b).

d) $r = \mathbf{case}\,(\mathbf{inl}\,t)\,\mathbf{of}\,(\mathbf{inl}\,x) \mapsto u \mid (\mathbf{inr}\ y) \mapsto v$ and $s = u\{t/x\}$. By lemma 4.11:
$$\mathcal{S}_{[]}(u\{t/x\}) \subseteq \mathcal{S}_{[]}(u)[\mathcal{S}_{[]}(t)/x] \subseteq \mathcal{S}_{[]}(u)[\mathcal{S}_{[]}(t)/x] \cup \mathcal{S}_{[]}(v)[\mathcal{S}_{[]}(t)/y] = \mathcal{S}_{[]}(r)$$
$$\mathcal{S}_{\delta}(u\{t/x\}) \subseteq \mathcal{S}_{\delta}(u)[\mathcal{S}_{[]}(t)/x] \cup \mathcal{S}_{\delta}(t) \subseteq \mathcal{S}_{\delta}(u)[\mathcal{S}_{[]}(t)/x] \cup \mathcal{S}_{\delta}(v)[\mathcal{S}_{[]}(t)/y] \cup \mathcal{S}_{\delta}(t) = \mathcal{S}_{\delta}(r)$$

e) Similar to d).

Structural reduction

a) $r = C[\mu\alpha.u]$ and $s = \mu\alpha.u\{[\alpha]C[t]/[\alpha]t\}$. By lemma 4.13:
$$\mathcal{S}_{[]}(\mu\alpha.u\{[\alpha]C[t]/[\alpha]t\}) = \mathcal{S}_{\alpha}(u\{[\alpha]C[t]/[\alpha]t\}) \cup \mathcal{S}_{[]}(u\{[\alpha]C[t]/[\alpha]t\})$$
$$\subseteq \mathcal{S}_{\alpha}(u) \cup \mathcal{S}_{\alpha}(C[\,]) \cup \mathcal{S}_{[]}(u) = \mathcal{S}_{\alpha}(C[\mu\alpha.u])$$
$$\mathcal{S}_{\delta}(\mu\alpha.u\{[\alpha]C[t]/[\alpha]t\}) \subseteq \mathcal{S}_{\delta}(u\{[\alpha]C[t]/[\alpha]t\}) \subseteq \mathcal{S}_{\delta}(u) \cup \mathcal{S}_{\delta}(C[\,]) = \mathcal{S}_{\delta}(C[\mu\alpha.u])$$

b) $r = C[\mathbf{case}\ t\ \mathbf{of}\ x \mapsto u \mid y \mapsto v]$ and $s = \mathbf{case}\ t\ \mathbf{of}\ x \mapsto C[u] \mid y \mapsto C[v]$
$$\mathcal{S}_{[]}(\mathbf{case}\ t\ \mathbf{of}\ x \mapsto C[u] \mid y \mapsto C[v]) = \mathcal{S}_{[]}(C[u])[\mathcal{S}_{[]}(t)/x] \cup \mathcal{S}_{[]}(C[v])[\mathcal{S}_{[]}(t)/y]$$
$$= \mathcal{S}_{[]}(C[\,]) \cup \mathcal{S}_{[]}(u)[\mathcal{S}_{[]}(t)/x] \cup \mathcal{S}_{[]}(v)[\mathcal{S}_{[]}(t)/y]$$
$$= \mathcal{S}_{[]}(C[\mathbf{case}\ t\ \mathbf{of}\ x \mapsto u \mid y \mapsto v])$$
$$\mathcal{S}_{\delta}(\mathbf{case}\ t\ \mathbf{of}\ x \mapsto C[u] \mid y \mapsto C[v]) = \mathcal{S}_{\delta}(C[u])[\mathcal{S}_{\delta}(t)/x] \cup \mathcal{S}_{\delta}(C[v])[\mathcal{S}_{\delta}(t)/y]$$
$$= \mathcal{S}_{\delta}(C[\,]) \cup \mathcal{S}_{\delta}(u)[\mathcal{S}_{\delta}(t)/x] \cup \mathcal{S}_{\delta}(v)[\mathcal{S}_{\delta}(t)/y]$$
$$= \mathcal{S}_{\delta}(C[\mathbf{case}\ t\ \mathbf{of}\ x \mapsto u \mid y \mapsto v])$$

Simplification

a) $r = \mu\alpha[\alpha]u$ and $s = u$ where α does not occur free in u.
$$\mathcal{S}_{[]}(u) = \mathcal{S}_{[]}(u) \cup \mathcal{S}_{\alpha}(u) = \mathcal{S}_{\alpha}([\alpha]u) = \mathcal{S}_{[]}(\mu\alpha[\alpha]u)\ \text{(since } \mathcal{S}_{\alpha}(u) = \emptyset)$$
$$\mathcal{S}_{\delta}(u) = \mathcal{S}_{\delta}([\alpha]u) = \mathcal{S}_{\delta}(\mu\alpha[\alpha]u)$$

b) $r = [\beta]\mu\alpha.t$ and $s = t\{\beta/\alpha\}$. By lemma 4.12:
$$\mathcal{S}_{[]}(t\{\beta/\alpha\}) \subseteq \mathcal{S}_{[]}(t) = \emptyset = \mathcal{S}_{\alpha}([\beta]\mu\alpha.t)\ \text{(since } t \text{ has the form } [\delta]v)$$

$$\mathcal{S}_\beta(t\{\beta/\alpha\}) \subseteq \mathcal{S}_\beta(t) \cup \mathcal{S}_\alpha(t) = \mathcal{S}_\beta(\mu\alpha.t) \cup \mathcal{S}_{[]}(\mu\alpha.t) = \mathcal{S}_\beta([\beta]\mu\alpha.t)$$
$$\mathcal{S}_\delta(t\{\beta/\alpha\}) \subseteq \mathcal{S}_\delta(t) = \mathcal{S}_\delta(\mu\alpha.t) = \mathcal{S}_\delta([\beta]\mu\alpha.t) \text{ for any } \delta \neq \beta$$

□

Lemma 4.16. *Given two $\lambda\mu^{\to+\times}$-term t, u and an arbitrary context $C[\,]$, if $\mathcal{S}_{[]}(u) \subseteq \mathcal{S}_{[]}(t)$ and $\mathcal{S}_\delta(u) \subseteq \mathcal{S}_\delta(t)$ then $\mathcal{S}_{[]}(C[u]) \subseteq \mathcal{S}_{[]}(C[t])$ and $\mathcal{S}_\delta(C[u]) \subseteq \mathcal{S}_\delta(C[t])$ for any a free μ-variable δ of $C[t]$.*

Proof. By induction on the context $C[\,]$. □

Proposition 4.17. *Given a $\lambda\mu^{\to+\times}$-term t, if t is safe and $t \rightsquigarrow u$ then $\mathcal{S}_{[]}(u) \subseteq \mathcal{S}_{[]}(t)$ and $\mathcal{S}_\delta(u) \subseteq \mathcal{S}_\delta(t)$ for any a free μ-variable δ of t.*

Proof. By lemma 4.15 and lemma 4.16. □

Lemma 4.18. *Given two $\lambda\mu^{\to+\times}$-terms u, v, if u and v are safe then $u\{v/x\}$ is safe.*

Proof. Let $\lambda y.t$ be a subterm of $u\{v/x\}$. Either $\lambda y.t$ is a subterm of v or y does not occur in v. Consequently, $y \notin \mathcal{S}_\delta(t)$ since v is safe in the former case and since u is safe in the latter case. □

Lemma 4.19. *Given an instance $r \rightsquigarrow s$ of a rule of the $\lambda\mu^{\to+\times}$-calculus, if r is safe then s is safe.*

Proof. Again, we consider each rule of the $\lambda\mu^{\to+\times}$-calculus:

Detour-reduction

a) $r = (\lambda x.u\, v)$ and $s = u\{v/x\}$. Apply lemma 4.18.

b) $r = \textbf{fst}\ \langle t, u\rangle$ and $s = t$. Then $s = t$ is safe.

c) Similar to b).

d) $r = \textbf{case}\,(\textbf{inl}\, t)\,\textbf{of}\,(\textbf{inl}\, x) \mapsto u \mid (\textbf{inr}\ y) \mapsto v$ and $s = u\{t/x\}$. Apply lemma 4.18.

e) Similar to d).

Structural reduction

a) $r = C[\mu\alpha w]$ and $s = \mu\alpha w\{[\alpha]C[u]/[\alpha]u\}$. Let $\lambda y.t$ be a subterm of $\mu\alpha w\{[\alpha]C[u]/[\alpha]u\}$, either $\lambda y.t$ is a subterm of $C[\,]$ or y does not occur in $C[\,]$. Consequently, $y \notin \mathcal{S}_\delta(t)$ since $C[\,]$ is safe in the former case and since w is safe in the latter case.

b) $r = C[\textbf{case}\ w\ \textbf{of}\,(\textbf{inl}\, x) \mapsto u \mid (\textbf{inr}\ y) \mapsto v]$ and $s = \textbf{case}\ w\ \textbf{of}\,(\textbf{inl}\, x) \mapsto C[u] \mid (\textbf{inr}\ y) \mapsto C[v]$. Let $\lambda y.t$ be a subterm of $\textbf{case}\ w\ \textbf{of}\,(\textbf{inl}\, x) \mapsto C[u] \mid (\textbf{inr}\ y) \mapsto C[v]$, either $\lambda y.t$ is a subterm of $C[\,]$ or y does not occur in $C[\,]$. Consequently, $y \notin \mathcal{S}_\delta(t)$ since $C[\,]$ is safe in the former case and since u, v and w are safe in the latter case.

Simplification

a) $r = \mu\alpha[\alpha]u$ and $s = u$ where α does not occur free in u. Then $s = u$ is safe.

b) $r = [\beta]\mu\alpha.u$ and $s = u\{\beta/\alpha\}$. Let $\lambda y.t$ be a subterm of $u\{\beta/\alpha\}$ and let $\lambda y.v$ be the subterm of u such as $\lambda y.v = \lambda y.t\{\beta/\alpha\}$. Then $y \notin \mathcal{S}_\delta(t) = \mathcal{S}_\delta(v)$ and $y \notin \mathcal{S}_\beta(t) = \mathcal{S}_\beta(v) \cup \mathcal{S}_\alpha(v)$ since u is safe.

□

Theorem 4.20. *Given a $\lambda\mu^{\to+\times}$-term t, if t is safe and $t \rightsquigarrow u$ then u is safe.*

Proof. Let r be the redex of t which is reduced in $t \rightsquigarrow u$ and let r' be the contractum. Now, let us consider a subterm of u which has the form $\lambda y.s'$ and where δ is free μ-variable of s'. We have to prove that $y \notin \mathcal{S}_\delta(s')$. Three cases may occur: (1) r' and $\lambda y.s'$ do not overlap, (2) r' is a subterm of s' and (3) $\lambda y.s'$ is a subterm of r'. The first case is trivial since $\lambda y.s'$ was already a subterm of t (which is safe). In the second case, since t is safe then r is safe and $y \notin \mathcal{S}_\delta(r)$, by proposition 4.17, $\mathcal{S}_\delta(r') \subseteq \mathcal{S}_\delta(r)$ and thus $y \notin \mathcal{S}_\delta(r')$. In the third case, just apply lemma 4.19. ☐

5 The safe $\lambda\mu^{\to+\times-}$-calculus

In this section, we extend the concept of "scope of a μ-variable with respect to (λ-)variables" to the syntax of the $\lambda\mu^{\to+\times-}$-calculus. Then we give the formal definition of a $\lambda\mu^{\to+\times-}$-term *safe with respect to first-class coroutines*. The remainding of the section is devoted to the proof of theorem 5.11 which states that the safe $\lambda\mu^{\to+\times-}$-calculus is closed under reduction.

Definition 5.1. *We extend the definitions of $\mathcal{S}_{[]}(t)$ and $\mathcal{S}_\delta(t)$ to any $\lambda\mu^{\to+\times-}$-term t (where δ is a free μ-variable of t):*

- $\mathcal{S}_{[]}(\textbf{make-coroutine}\, t\, C_\alpha) = \mathcal{S}_{[]}(t)$
  $\mathcal{S}_\alpha(\textbf{make-coroutine}\, t\, C_\alpha) = \mathcal{S}_\alpha(t) \cup \mathcal{S}_\alpha(C[\,]) \cup \mathcal{S}_{[]}(t) \cup \mathcal{S}_{[]}(C[\,])$
  $\mathcal{S}_\delta(\textbf{make-coroutine}\, t\, C_\alpha) = \mathcal{S}_\delta(t) \cup \mathcal{S}_\delta(C[\,])$ *for any* $\delta \neq \alpha$

- $\mathcal{S}_{[]}(\textbf{resume}\, c\, \textbf{with}\, x \mapsto u) = \mathcal{S}_{[]}(u) \backslash \{x\} \cup \mathcal{S}_{[]}(c)$
  $\mathcal{S}_\delta(\textbf{resume}\, c\, \textbf{with}\, x \mapsto u) = \mathcal{S}_\delta(u)[\mathcal{S}_{[]}(c)/x] \cup \mathcal{S}_\delta(c)$

Definition 5.2. *A $\lambda\mu^{\to+\times-}$-term t is safe with respect to first-class coroutines (or just "safe" for short) iff:*

1. *for any subterm of t which has the form $\lambda x.u$, for any free μ-variable δ of u, $x \notin \mathcal{S}_\delta(u)$ (i.e. t is safe with respect to coroutine contexts)*
2. *for any subterm of t which has the form* $\textbf{resume}\, c\, \textbf{with}\, x \mapsto u$, $\mathcal{S}_{[]}(u) \subseteq \{x\}$

Remark 5.3. If we violate one of these restrictions, we reobtain the (unsafe) $\lambda\mu^{\to+\times}$-calculus (and classical logic as a type system). For the first restriction, it is obvious. For the second restriction, consider the following $\lambda\mu^{\to+\times-}$-term:

$$\lambda y.(\textbf{resume}\ (\textbf{make-coroutine}\, e\, \beta)\ \textbf{with}\ x \mapsto y) \rightsquigarrow \lambda y.\textbf{set-context}\ \beta\ y$$

This term is a redex and its contractum is a first-class continuation.

5.1 Typing safe $\lambda\mu^{\to+\times-}$-terms in $\mathrm{SND}_{\to\vee\wedge-}$

Proofs of $\mathrm{SND}_{\to\vee\wedge-}$ (*i.e.* constructive proofs of $\mathrm{CND}_{\to\vee\wedge-}$) and safe $\lambda\mu^{\to+\times-}$-terms are related by this theorem (and its corollary):

Theorem 5.4. *Given an annotated derivation, in system $\mathrm{CND}_{\to\vee\wedge-}$, of some typing judgement $t : \Gamma_1^{x_1}, \ldots, \Gamma_n^{x_n} \vdash \Delta_1^{\alpha_1}, \ldots, \Delta_m^{\alpha_m}; A$, if $x_i \in \mathcal{S}_{\alpha_j}(t)$ then there is a link between $\Gamma_i^{x_i}$ and $\Delta_j^{\alpha_j}$ and if $x_i \in \mathcal{S}_{[]}(t)$ then there is a link between $\Gamma_i^{x_i}$ and A.*

Proof. Complete the proof of theorem 4.8 with the rules for subtraction. For instance, consider the introduction rule for subtraction (since $C[\,]$ is a continuation context, we know that y occurs in T'' and nowhere else):

$$\frac{t\colon \Gamma \vdash S_1\colon \Delta_1^{\alpha_1}, \ldots, S_p\colon \Delta_p^{\alpha_p}, T\colon B^\beta; S\colon A \qquad C[y]\colon \Gamma, D^y \vdash S_1'\colon \Delta_1^{\alpha_1}, \ldots, S_p'\colon \Delta_p^{\alpha_p}, T'\colon B^\beta; T''\colon B}{\mathbf{make\text{-}coroutine}\; t\, C_\beta\colon \Gamma \vdash S_1 \cup S_1'\colon \Delta_1^{\alpha_1}, \ldots, S_p \cup S_p'\colon \Delta_p^{\alpha_p}, T \cup T' \cup T''[S/y]\colon B^\beta; S\colon A - D}$$

By definition of $\mathcal{S}$, $\mathcal{S}_{[]}(\mathbf{make\text{-}coroutine}\; t\, C_\beta) = \mathcal{S}_{[]}(t)$. Then, if $x_i \in \mathcal{S}_{[]}(\mathbf{make\text{-}coroutine}\; t\, C_\beta)$, by induction hypothesis, we have either $x_i \in S$ and there is link between $\Gamma_i^{x_i}$ and $A - D$.

Similarly, by definition of $\mathcal{S}$, $\mathcal{S}_{\alpha_j}(\mathbf{make\text{-}coroutine}\; t\, C_\beta) = \mathcal{S}_{\alpha_j}(t) \cup \mathcal{S}_{\alpha_j}(C[\,])$. Consequently, if we know that $x_i \in \mathcal{S}_{\alpha_j}(\mathbf{make\text{-}coroutine}\; t\, C_\beta)$, by induction hypothesis, we have either $x_i \in S_j$ or $x_i \in S_j'$ and there is link between $\Gamma_i^{x_i}$ and Δ^{α_j}.

Finally, by definition of $\mathcal{S}$, $\mathcal{S}_\beta(\mathbf{make\text{-}coroutine}\; t\, C_\beta) = \mathcal{S}_\beta(t) \cup \mathcal{S}_\beta(C[\,]) \cup \mathcal{S}_{[]}(t) \cup \mathcal{S}_{[]}(C[\,])$. Consequently, if we know that $x_i \in \mathcal{S}_\beta(\mathbf{make\text{-}coroutine}\; t\, C_\beta)$, by induction hypothesis, we have respectively $x_i \in T$, $x_i \in T'$, $x_i \in S$ (and thus $x_i \in T''[S/y]$ since $y \in T''$) and $x_i \in T''$. In all cases, there is a link between $\Gamma_i^{x_i}$ and B^β. □

Corollary 5.5. *Given a derivation of the typing judgement $t\colon \Gamma \vdash \Delta$ in $\mathrm{CND}_{\to\vee\wedge-}$, if t is safe with respect to first-class coroutines then the derivation of $t\colon \Gamma \vdash \Delta$ belongs to $\mathrm{SND}_{\to\vee\wedge-}$.*

Proof. We already know by corollary 4.9 that any occurrence of the introduction rule for implication is constructive. Let us now consider an occurrence of the elimination rule for subtraction:

$$\frac{t\colon \Gamma \vdash \Delta; A - B \qquad u\colon \Gamma', A^x \vdash \Delta'; B}{\mathbf{resume}\; t\; \mathbf{with}\; x \mapsto u\colon \Gamma, \Gamma' \vdash \Delta, \Delta'; C}$$

Since t is safe with respect to first-class coroutines, we know that $\mathcal{S}_{[]}(u) \subseteq \{x\}$ and then by theorem 5.4, there is no link between Γ' and B, thus this occurrence of the elimination rule for subtraction is constructive.

□

5.2 Closure under reduction

In the remaining of this section, we prove that the subset of safe $\lambda\mu^{\to+\times-}$-terms is closed under the reduction rules of the $\lambda\mu^{\to+\times-}$-calculus. We first need the following additional lemmas:

Lemma 5.6. *Let u be a $\lambda\mu^{\to+\times-}$-term, let $C[\,]$ be a continuation context, and let δ be a μ-variable free in $C[u]$:*

$$\mathcal{S}_{[]}(C[u]) = \mathcal{S}_{[]}(C[\,]) \cup \mathcal{S}_{[]}(u)$$
$$\mathcal{S}_\delta(C[u]) = \mathcal{S}_\delta(C[\,]) \cup \mathcal{S}_\delta(u)$$

Lemma 5.7. *Given an instance $r \rightsquigarrow s$ of a rule of the $\lambda\mu^{\to+\times-}$-calculus, if r is safe then $\mathcal{S}_{[]}(s) \subseteq \mathcal{S}_{[]}(r)$ and $\mathcal{S}_\delta(s) \subseteq \mathcal{S}_\delta(r)$ for any a free μ-variable δ of s.*

Proof. The rules of the $\lambda\mu^{\to+\times}$-calculus have been delt with in section 4.1. The only remaining rules are:

Detour reduction

f) $r = \mathbf{resume}\; (\mathbf{make\text{-}coroutine}\; v\, C_\alpha)\; \mathbf{with}\; x \mapsto u$ and $s = \mathbf{set\text{-}context}\; \beta\; C[u\{v/x\}]$

Let δ be a free μ-variable of s. We have:

- $\mathcal{S}_{[]}(\mathbf{set\text{-}context}\; \beta\; C[u\{v/x\}]) = \emptyset \subseteq \mathcal{S}_{[]}(\mathbf{resume}\; (\mathbf{make\text{-}coroutine}\; v\, C_\beta) \mathbf{with}\; x \mapsto u)$

- $\mathcal{S}_\beta(\textbf{set-context}\ \beta\, C[u\{v/x\}]) = \mathcal{S}_\beta(C[u\{v/x\}]) \cup \mathcal{S}_{[]}(C[u\{v/x\}])$
  $= \mathcal{S}_\beta(C[\,]) \cup \mathcal{S}_\beta(u\{v/x\}) \cup \mathcal{S}_{[]}(C[\,]) \cup \mathcal{S}_{[]}(u\{v/x\})$
  $\subseteq \mathcal{S}_\beta(C[\,]) \cup \mathcal{S}_\beta(u)[\mathcal{S}_{[]}(v)/x] \cup \mathcal{S}_\beta(v) \cup \mathcal{S}_{[]}(C[\,]) \cup \mathcal{S}_{[]}(u)[\mathcal{S}_{[]}(v)/x]$
  $\subseteq \mathcal{S}_\beta(C[\,]) \cup \mathcal{S}_\beta(u)[\mathcal{S}_{[]}(v)/x] \cup \mathcal{S}_\beta(v) \cup \mathcal{S}_{[]}(C[\,]) \cup \mathcal{S}_{[]}(v)$ since $\mathcal{S}_{[]}(u) \subseteq \{x\}$
  $= \mathcal{S}_\beta(u)[\mathcal{S}_{[]}(v)/x] \cup \big(\mathcal{S}_\beta(v) \cup \mathcal{S}_\beta(C[\,]) \cup \mathcal{S}_{[]}(v) \cup \mathcal{S}_{[]}(C[\,])\big)$
  $= \mathcal{S}_\beta(u)[\mathcal{S}_{[]}(v)/x] \cup \mathcal{S}_\beta(\textbf{make-coroutine}\, v\, C_\beta)$
  $= \mathcal{S}_\beta(u)[\mathcal{S}_{[]}(\textbf{make-coroutine}\, v\, C_\beta)/x] \cup \mathcal{S}_\beta(\textbf{make-coroutine}\, v\, C_\beta)$
  $= \mathcal{S}_\beta(\textbf{resume}\ (\textbf{make-coroutine}\, v\, C_\beta)\ \textbf{with}\, x \mapsto u)$

- $\mathcal{S}_\delta(\textbf{set-context}\,\beta\, C[u\{v/x\}]) = \mathcal{S}_\delta(\mathcal{C}[u\{v/x\}])$
  $= \mathcal{S}_\delta(C[\,]) \cup \mathcal{S}_\delta(u\{v/x\})$
  $\subseteq \mathcal{S}_\delta(C[\,]) \cup \mathcal{S}_\delta(u)[\mathcal{S}_{[]}(v)/x] \cup \mathcal{S}_\delta(v)$
  $= \mathcal{S}_\delta(u)[\mathcal{S}_{[]}(v)/x] \cup (\mathcal{S}_\delta(v) \cup \mathcal{S}_\delta(C[\,]))$
  $= \mathcal{S}_\delta(u)[\mathcal{S}_{[]}(v)/x] \cup \mathcal{S}_\delta(\textbf{make-coroutine}\, v\, C_\beta)$
  $= \mathcal{S}_\delta(u)[\mathcal{S}_{[]}(\textbf{make-coroutine}\, v\, C_\beta)/x] \cup \mathcal{S}_\delta(\textbf{make-coroutine}\, v\, C_\beta)$
  $= \mathcal{S}_\delta(\textbf{resume}\ (\textbf{make-coroutine}\, v\, C_\beta)\ \textbf{with}\, x \mapsto u)$

Structural reduction

c) $r = C[\textbf{resume}\ t\ \textbf{with}\ x \mapsto u]$ and $s = \textbf{resume}\ t\ \textbf{with}\ x \mapsto u$

Let δ be a free μ-variable of s. We have:

- $\mathcal{S}_{[]}(\textbf{resume}\ t\ \textbf{with}\ x \mapsto C[u]) = \mathcal{S}_{[]}(C[u])[\mathcal{S}_{[]}(t)/x]$
  $= \mathcal{S}_{[]}(C[\,]) \cup \mathcal{S}_{[]}(u)[\mathcal{S}_{[]}(t)/x]$
  $\subseteq \mathcal{S}_{[]}(u)[\mathcal{S}_{[]}(t)/x]$
  $= \mathcal{S}_{[]}(\textbf{resume}\ t\ \textbf{with}\ x \mapsto u)$

- $\mathcal{S}_\delta(\textbf{resume}\ t\ \textbf{with}\ x \mapsto C[u]) = \mathcal{S}_\delta(C[u])[\mathcal{S}_{[]}(t)/x] \cup \mathcal{S}_\delta(t)$
  $= \mathcal{S}_\delta(C[\,]) \cup \mathcal{S}_\delta(u)[\mathcal{S}_{[]}(t)/x] \cup \mathcal{S}_\delta(t)$
  $\subseteq \mathcal{S}_\delta(u)[\mathcal{S}_{[]}(t)/x] \cup \mathcal{S}_\delta(t)$
  $= \mathcal{S}_\delta(\textbf{resume}\ t\ \textbf{with}\ x \mapsto u)$

□

Lemma 5.8. *Given an instance $r \rightsquigarrow s$ of a rule of the $\lambda\mu^{\to + \times -}$-calculus, if r is safe w.r.t. coroutine contexts then s is safe w.r.t. coroutine contexts.*

Proof. Again, the rules of the $\lambda\mu^{\to + \times}$-calculus have been delt with in section 4.1. The only remaining rules are:

Detour-reduction

f) $r = \textbf{resume}\ (\textbf{make-coroutine}\, v\, C_\alpha)\ \textbf{with}\ x \mapsto u$ and $s = \textbf{set-context}\ \beta\ C[u\{v/x\}]$
By lemma 4.18, $u\{v/x\}$ is safe w.r.t. coroutine contexts, and since $C[\,]$ is safe w.r.t. coroutine contexts, we know that s is also safe w.r.t. coroutine contexts.

Structural reduction

c) $r = C[\textbf{resume}\ c\ \textbf{with}\ x \mapsto u]$ and $s = \textbf{resume}\ c\ \textbf{with}\ x \mapsto u$
Obviously s is safe w.r.t. coroutine contexts since it is a subterm of r.

□

Lemma 5.9. *Given two $\lambda\mu^{\to + \times -}$-term t, u and an arbitrary context $C[\,]$, if $\mathcal{S}_{[]}(u) \subseteq \mathcal{S}_{[]}(t)$ and $\mathcal{S}_\delta(u) \subseteq \mathcal{S}_\delta(t)$ then $\mathcal{S}_{[]}(C[u]) \subseteq \mathcal{S}_{[]}(C[t])$ and $\mathcal{S}_\delta(C[u]) \subseteq \mathcal{S}_\delta(C[t])$ for any a free μ-variable δ of $C[t]$.*

Proof. By induction on the context $C[\,]$. □

Lemma 5.10. *Given an instance $r \rightsquigarrow s$ of a rule of the $\lambda\mu^{\rightarrow+\times-}$-calculus, if r is safe w.r.t. first-class coroutines then s is safe w.r.t. first-class coroutines.*

Proof. We already know by lemma 5.8 that s is safe w.r.t. coroutine contexts. We just have to check that for any subterm of s which has the form $\textbf{resume}\, c\, \textbf{with}\, x \mapsto u$, $\mathcal{S}_{[]}(u) \subseteq \{x\}$. This property follows from lemma 5.7 and lemma 5.9. □

Theorem 5.11. *Given a $\lambda\mu^{\rightarrow+\times-}$-term t, if t is safe and $t \rightsquigarrow u$ then u is safe.*

Proof. Using the same argument as in the proof of theorem 4.20, we deduce that u is safe w.r.t. coroutine contexts. Now, let r be the redex of t which is reduced in $t \rightsquigarrow u$ and let r' be the contractum. Consider also a subterm v' of u which has the form $\textbf{resume}\, c'\, \textbf{with}\, x \mapsto u'$. Again, three cases may occur:

1. r' and v' do not overlap: v' is safe since it was already a subterm of t (which is safe).
2. v' is a subterm of c' or v' is a subterm of u': just apply lemma 5.7 and lemma 5.9.
3. v' is a subterm of r': we know that r' comes from a subterm r of t with the same form $\textbf{resume}\, c\, \textbf{with}\, x \mapsto u$ and such that $\mathcal{S}_{[]}(u) \subseteq \{x\}$. Moreover, u' is obtained from u by applying either the usual substitution (to some variable different from x) or a structural substitution. Then we know by lemma 4.11 and lemma 4.13 that $\mathcal{S}_{[]}(u') \subseteq \{x\}$.

□

6 Directions for extension

We have defined the safe $\lambda\mu^{\rightarrow+\times}$-calculus, which is closed under reduction and whose type system corresponds to intuitionnistic logic (resp. CDL). In this calculus, continuations are not first-class objects but the ability of context-switching remains. The safe $\lambda\mu^{\rightarrow+\times-}$-calculus is an extension of the $\lambda\mu^{\rightarrow+\times}$-calculus with first-class coroutines. First-class coroutines are striclty less powerful than first-class continuations: the type system of the $\lambda\mu^{\rightarrow+\times-}$-calculus corresponds to subtractive logic, which is conservative over the constructive logic CDL.

We have proved that first-class coroutines disappear during the normalization process (as a corollary of the subformula property) whenever the type of the term contains no subtraction. The normal form belongs to the safe $\lambda\mu^{\rightarrow+\times}$-calculus. Since the existence property holds in CDL, we expect to be able to extract witnesses from normal proofs of existential formulas. A first attempt could consist in exploiting the proof of conservativity given in appendix C. A forthcoming paper will be devoted to this issue.

Applications of first-class coroutines were barely mentioned in this paper. In fact, practical applications of coroutines often uses other extensions such as imperative features which do not easily fit in the formulae-as-types framework. However, purely functional examples have to be explored. On the other hand, the abstract machine we briefly described in section 4 seems quite promising. The extension of this machine to a realistic programming language seems also worthy to be investigated.

Eventually, the duality call-by-name/call-by-value (from the classical $\lambda\mu$-calculus) should be revisited in our framework, where the duality is likely to exchange functions and coroutines. We already know by construction of the safe $\lambda\mu^{\rightarrow+\times-}$-calculus that the dual of a safe $\lambda\mu^{\rightarrow+\times-}$-term is also safe.

Appendix A. Existence property for CDL

Theorem A.1. *If a formula $\exists x\varphi$ is valid in* CDL *then there exists a term t such that $\varphi(t)$ is valid in* CDL*.*

Proof. We only sketch a semantical proof of this theorem using Kripke models. Recall that Kripke models with constant domains are sound and complete for CDL (C. Rauszer [42]). Let us first extend the signature with an enumerable set of constants $\{c_i/i \in N\}$. We know that a formula φ with some free variables $x_1, ..., x_n$ is valid iff the closed formula $\varphi[c_1/x_1, ..., c_n/x_n]$ is valid. We will thus consider (without restriction) some closed theorem $\exists x\varphi$. We have to prove that there exists a closed term t such that $\vdash \varphi(t)$.

Let us assume that for each closed term t, we have a Kripke model $\mathcal{M}_t$ such that $\mathcal{M}_t \not\Vdash \varphi(t)$. Let $D(\mathcal{M})$ be the free algebra generated by the disjoint union D of the domains of all $\mathcal{M}_t$'s. Let $\mathcal{M}_t^c$ be a copy of $\mathcal{M}_t$ for each constant c that occurs in t. The domain of each $\mathcal{M}_t^c$ is extended to $D(\mathcal{M})$ in such a way that any new element is a copy of $\mathcal{M}_t(c)$. More specifically, let $\eta : D \to D(\mathcal{M}_t)$ be the mapping defined by $\eta(e) = e$ if $e \in D(\mathcal{M}_t)$ and $\eta(e) = \mathcal{M}_t(c)$ otherwise. This mapping is extended in the usual way to the free algebra $D(\mathcal{M})$. Now, for each node α of $\mathcal{M}_t^c$ and each n-ary predicate symbol P we define $\mathcal{M}(P)$ by:

$$(e_1, ..., e_n) \in_\alpha \mathcal{M}(P) \text{ iff } (\eta(e_1), ..., \eta(e_n)) \in_\alpha \mathcal{M}_t^c(P)$$

We build the model $\mathcal{M}$ by taking the disjoint union of the $\mathcal{M}_t^c$'s and adding a bottom node α_0.

Let e be an element of $D(\mathcal{M})$, let us show that $\mathcal{M} \not\Vdash \varphi(e)$. By definition, e has the form $t[d_1/x_1, ..., d_n/x_n]$ where t is a term without parameter (but with some free variables $x_1, ..., x_n$) and $d_1, ..., d_n$ are elements of D (if e is a closed term then $n = 0$). Now, let c be a constant such that for any $1 \leq i \leq n$ we have $d_i \notin D(\mathcal{M}_{\bar{t}})$ where $\bar{t}$ is the closed term $t[c/x_1, ..., c/x_n]$ (there are infinitely many such constants). By construction, $\mathcal{M}_{\bar{t}} \not\Vdash \varphi(\bar{t})$ and thus $\mathcal{M}_{\bar{t}}^c \not\Vdash \varphi(e)$ since for each for any $1 \leq i \leq n$, d_i is a copy of c in $\mathcal{M}_{\bar{t}}^c$. Consequently $\alpha_0 \not\Vdash \varphi(e)$. □

Appendix B. Properties of the $\lambda\mu^{\to+\times\otimes\perp}$-calculus

In [12], de Groote presents a proof of strong normalization of CND with primitive disjunction (and primitive conjunction with projections). We show here how to adapt this proof to take into account the rule for pattern matching. Although de Groote did not consider the simplication rules in his paper, it is easy to show that simplification (alone) is strongly normalizing and also that simplication may be postponed with respect with the other reduction relations.

De Groote's proof of strong normalization is two-fold: first he proves the strong normalization of the structural reductions (for untyped $\lambda\mu^{\to\wedge\vee\perp}$-terms) then he proves the strong normalization of typed $\lambda\mu^{\to\wedge\vee\perp}$-terms using a CPS-simulation. The latter proof is given in a propositional framework, but de Groote claims that since this CPS-translation is defined on the untyped $\lambda\mu$-terms, it may be raised to the second-order.

B.1 Strong normalization of the structural reductions

We provide $\lambda\mu^{\to+\times\otimes\perp}$-terms with a norm adapted from the norm introduced in [12] (we recall the full definitions but the new cases are only the two last of each definition):

Definition B.1. *The norm* $|\cdot|$ *assigned to the* $\lambda\mu^{\rightarrow+\times\otimes\perp}$*-terms is inductively defined as follows:*

a) $|x| = 1$

b) $|\lambda x.t| = |t|$

c) $|(t\, u)| = |t| + \#t \times |u|$

d) $|\langle t, u\rangle| = |t| + |u|$

e) $|\mathbf{fst}\, t| = |t| + \#t$

f) $|\mathbf{snd}\, t| = |t| + \#t$

g) $|\mathbf{inl}\, t| = |t|$

h) $|\mathbf{inr}\, t| = |t|$

i) $|\mathbf{case}\ t\ \mathbf{of}\ (\mathbf{inl}\, x) \mapsto u \mid (\mathbf{inr}\ y) \mapsto v| = |t| + \#t \times (|t| + |u|)$

j) $|\mu\alpha.t| = |t|$

k) $|[\alpha]t| = |t|$

l) $|(t, u)| = |t| + |u|$

m) $|\mathbf{match}\ t\ \mathbf{with}\ (x, y) \mapsto u| = |t| + 2 \times \#t \times |u|$

where:

a) $\#x = 1$

b) $\#\lambda x.t = 1$

c) $\#(t\, u) = \#t$

d) $\#\langle t,\ u\rangle = 1$

e) $\#\mathbf{fst}\, t = \#t$

f) $\#\mathbf{snd}\, t = \#t$

g) $\#\mathbf{inl}\, t = 1$

h) $\#\mathbf{inr}\ t = 1$

i) $\#\mathbf{case}\ t\ \mathbf{of}\ (\mathbf{inl}\, x) \mapsto u \mid (\mathbf{inr}\ y) \mapsto v = (2 \times \#t) \times (\#u + \#v)$

j) $\#\mu\alpha.t = \lfloor t \rfloor_\alpha$

k) $\#[\alpha]t = 1$

l) $\#(t,\ u) = 1$

m) $\#\mathbf{match}\ t\ \mathbf{with}\ (x, y) \mapsto u = (2 \times \#t) \times (2 \times \#u)$

and where:

a) $\lfloor x \rfloor_\alpha = 0$

b) $\lfloor \lambda x.t \rfloor_\alpha = \lfloor t \rfloor_\alpha$

c) $\lfloor (t\, u) \rfloor_\alpha = \lfloor t \rfloor_\alpha + \#t \times \lfloor u \rfloor_\alpha$

d) $\lfloor \langle t, u\rangle \rfloor_\alpha = \lfloor t \rfloor_\alpha + \lfloor u \rfloor_\alpha$

e) $\lfloor \mathbf{fst}\, t \rfloor_\alpha = \lfloor t \rfloor_\alpha$

f) $\lfloor \mathbf{snd}\, t \rfloor_\alpha = \lfloor t \rfloor_\alpha$

g) $\lfloor \mathbf{inl}\, t \rfloor_\alpha = \lfloor t \rfloor_\alpha$

h) $\lfloor \mathbf{inr}\, t \rfloor_\alpha = \lfloor t \rfloor_\alpha$

i) $\lfloor \mathbf{case}\ t\ \mathbf{of}\ (\mathbf{inl}\, x) \mapsto u \mid (\mathbf{inr}\ y) \mapsto v \rfloor_\alpha = \lfloor t \rfloor_\alpha + \#t \times (\lfloor u \rfloor_\alpha + \lfloor v \rfloor_\alpha)$

j) $\lfloor \mu\alpha.t \rfloor_\alpha = \lfloor t \rfloor_\alpha$

k) $\lfloor [\alpha]t \rfloor_\alpha = \lfloor t \rfloor_\alpha + \#t$

l) $\lfloor [\beta]t \rfloor_\alpha = \lfloor t \rfloor_\alpha$

m) $\lfloor (t, u) \rfloor_\alpha = \lfloor t \rfloor_\alpha + \lfloor u \rfloor_\alpha$

n) $\lfloor \mathbf{match}\ t\ \mathbf{with}\ (x, y) \mapsto u \rfloor_\alpha = \lfloor t \rfloor_\alpha + 2 \times \#t \times \lfloor u \rfloor_\alpha$

Lemma B.2. *If t and u are two $\lambda\mu^{\to + \times \otimes \bot}$-terms and $t \rightsquigarrow_S u$ (where $\rightsquigarrow_S$ denotes the structural reduction) then $|t| > |u|$.*

B.2 CPS-simulation

We adapt here de Groote's modified CPS-translation, which simulates the relation of detour-reduction by strict β-reduction and the relation of structural reduction by equality.

Definition B.3. *The modified CPS-translation $\bar{\bar{t}}$ of any $\lambda\mu^{\to + \times \otimes \bot}$-term is defined as:*

$$\bar{\bar{t}} = \lambda k.(t : k)$$

where k is a fresh variable and where the infix operator ":" obeys the following definition:

a) $x : k = (x\ k)$

b) $\lambda x.t : k = (k\ \lambda x.\bar{\bar{t}})$

c) $(t\, u) : k = t :\ \lambda m.m\ \bar{\bar{u}}\ k$

d) $\langle t, u \rangle : k = (k\ \lambda m.(m\ \bar{\bar{t}}\ \bar{\bar{u}}))$

e) $\mathbf{fst}\, t : k = t : \lambda m.(m\ \lambda i.\lambda j.(i\ k))$

f) $\mathbf{snd}\, t : k = t : \lambda m.(m\ \lambda i.\lambda j.(j\ k))$

g) $\mathbf{inl}\, t : k = (k\ \lambda i.\lambda j.(i\ \bar{\bar{t}}))$

h) $\mathbf{inr}\, t : k = (k\ \lambda i.\lambda j.(j\ \bar{\bar{t}}))$

i) $\mathbf{case}\ t\ \mathbf{of}\ (\mathbf{inl}\, x) \mapsto u \mid (\mathbf{inr}\ y) \mapsto v : k = t : \lambda m.(m\ \lambda x.(u\ k)\ \lambda y.(v\ k))$

j) $\mu\alpha.t : k = (t :\ \lambda k.k)\{k/\alpha\}$ *if α occurs free in t*

k) $\mu\alpha.t : k = \lambda\alpha.(t :\ \lambda k.k)\ k$ *otherwise*

l) $[\alpha]t : k = t\ \alpha$

m) $(t, u) : k = (k\ \lambda m.(m\ \bar{\bar{t}}\ \bar{\bar{u}}))$

n) $\mathbf{match}\ t\ \mathbf{with}\ (x, y) \mapsto u : k = t : \lambda m.(m\ \lambda x\lambda y.(u\ k)$

where m, i, j are fresh variables.

Lemma B.4. *Let t and u be two $\lambda\mu^{\to+\times\otimes\perp}$-terms. If $t \rightsquigarrow_D u$ (where $\rightsquigarrow_D$ denotes the structural reduction) then $\bar{\bar{t}} \rightsquigarrow_\beta \bar{\bar{u}}$.*

Lemma B.5. *Let t and u be two $\lambda\mu^{\to+\times\otimes\perp}$-terms. If $t \rightsquigarrow_S u$ then $\bar{\bar{t}} = \bar{\bar{u}}$.*

Theorem B.6. *The well-typed $\lambda\mu^{\to+\times\otimes\perp}$-terms are strongly normalizable.*

B.3 Church-Rosser property

The Church-Rosser property follows from the local confluence of the reductions, the strong normalization and Newman lemma.

Appendix C. Conservativity of SND over subtractive logic

In this appendix, we show that $\mathrm{SND}_{\to\vee\wedge-}$ is conservative over first-order subtractive logic (SL). As a corollary, we obtain that $\mathrm{SND}_{\to\vee\wedge}$ is conservative over CDL. The idea of the proof is the following one: any derivation of a sequent in $\mathrm{SND}_{\to\vee\wedge-}$ can be translated into a derivation which does not contain any (right-hand side) introduction rule of implication nor any left-hand side introduction rule of subtraction, but which depends only on axioms valid in SL. Then, we just have to apply the following lemma to conclude (if $S = \{S_1, ..., S_n\}$ then $S^\wedge$ and $S^\vee$ stand respectively for $S_1 \wedge ... \wedge S_n$ and $S_1 \vee ... \vee S_n$).

Lemma C.1. *For any sequent $\Gamma \vdash \Delta$ derivable in $\mathrm{SND}_{\to\vee\wedge-}$ from sequents valid in subtractive logic without the (right-hand side) introduction rule for implication and without the left-hand side introduction rule for subtraction, the sequent $\Gamma^\wedge \vdash \Delta^\vee$ is valid in subtractive logic.*

Proof. By induction on the derivation. □

The tricky part of the proof consists in showing that the constructive introduction rule of implication (resp. the constructive left-hand side introduction rule of subtraction) *commutes* with any other rule. In order to prove this property, we have first to generalize these rules :

- We denote by $\mathrm{hyp}(\Delta)$ the set of (occurrences of) hypothesis linked to at least one of (the occurrences of) a conclusion of Δ and let us define the following generalization of the constructive introduction rule of implication:

$$\frac{\Gamma \vdash \Delta}{\Gamma\backslash\{H\} \vdash \Delta\backslash S, H \Rightarrow S^\vee} \quad \text{where } H \notin \mathrm{hyp}(\Delta\backslash S)$$

  Note that H does not need to occur in Γ, and each occurence of hypothesis of S does not need to occur (and possibly none does) in Δ.

- Its dual rule, which generalizes the constructive left-hand side introduction rule for subtraction is the following (where $\mathrm{cncl}(\Gamma)$ denotes the set of (occurrences of) conclusions linked to at least one of (the occurrence of) an hypothesis of Γ)

$$\frac{\Gamma \vdash \Delta}{\Gamma\backslash S, S^\wedge - C \vdash \Delta\backslash\{C\}} \quad \text{where } C \notin \mathrm{cncl}(\Gamma\backslash S)$$

  Again, C does not need to occur in Γ, and each occurence of hypothesis of S does not need to occur (and possibly none does) in Γ.

Theorem C.2. *The system* $\mathrm{SND}_{\rightarrow\vee\wedge-}$ *is conservative over* SL.

Proof. We first deal with the case of axioms (section C.1). Then, the main part of the proof consists in showing that the generalized introduction rule of implication *commutes* with any other rule (section C.2). Eventually, we conclude by applying the duality: the generalized left-hand side introduction rule for subtraction also *commutes* with any other rule (by duality). $\square$

Corollary C.3. *The system* $\mathrm{SND}_{\rightarrow\vee\wedge}$ *is conservative over* CDL.

Proof. A sequent derivable in $\mathrm{SND}_{\rightarrow\vee\wedge}$ is valid in SL which is conservative over CDL. $\square$

C.1 Axioms

Proposition C.4. *In* $\mathrm{SND}_{\rightarrow\vee\wedge}$*, the set of sequents that belong to one of the three following collections:*

- $\Gamma, A \vdash \Delta, B$ *where* $A \vdash B$ *is valid in SL, and there is at most one link which annotates this sequent, and this link binds* A *and* B *together;*
- $\Gamma, Y \vdash \Delta$ *where* $Y \vdash \bot$ *is valid in SL;*
- $\Gamma \vdash \Delta, X$ *where* $\top \vdash X$ *is valid in SL.*

is closed under the generalized (right-hand side) introduction rule of implication (resp. left-hand side introduction rule for subtraction).

Proof. Let us consider the generalized introduction rule of implication for the three collections of sequents:

1. The upper sequent has the form Γ, $A \vdash \Delta$, B where $A \vdash B$ is valid in DIS-logic, and the unique link which annotates this sequent binds A and B together.

   - First case, $H \neq A$ and $B \notin S$.

     $$\frac{\Gamma, A \vdash \Delta, B}{\Gamma\backslash\{H\}, A \vdash \Delta\backslash S, B, H \Rightarrow S^{\vee}}$$

     the lower sequent is indeed of the first form.

   - Second case, $H \neq A$ and H is discharged on $S \cup \{B\}$

     $$\frac{\Gamma, A \vdash \Delta, B}{\Gamma\backslash\{H\}, A \vdash \Delta\backslash S, H \Rightarrow (S^{\vee} \vee B)}$$

     the lower sequent is indeed of the third form since if $A \vdash B$ is valid in SL and $B \in S$ then $A \vdash H \Rightarrow (S^{\vee} \vee B)$ is also valid.

   - Third case, $H = A$ and A is discharged on $S \cup \{B\}$

     $$\frac{\Gamma, A \vdash \Delta, B}{\Gamma \vdash \Delta\backslash S, A \Rightarrow (S^{\vee} \vee B)}$$

     the lower sequent is indeed of the third form since if $A \vdash B$ is valid in SL then $\top \vdash A \Rightarrow (S^{\vee} \vee B)$ is also valid.

   In the case $H = A$ and $B \notin S$, *i.e.* where A is discharged on another conclusion that B, the constructive constraint does not hold. Consequently this case has not to be considered.

2. The upper sequent has the form $\Gamma, Y \vdash \Delta$ where $Y \vdash \bot$ is valid in SL.

   - First case, $H \neq Y$ and $H \notin \mathrm{hyp}(\Delta \backslash S)$

   $$\frac{\Gamma, Y \vdash \Delta}{\Gamma \backslash \{H\}, Y \vdash \Delta \backslash S, H \Rightarrow S^{\vee}}$$

   the lower sequent is still of the second form.

   - Second case, $H = Y$ and $Y \notin \mathrm{hyp}(\Delta \backslash S)$

   $$\frac{\Gamma, Y \vdash \Delta}{\Gamma \vdash \Delta \backslash S, Y \Rightarrow S^{\vee}}$$

   Since $Y \vdash \bot$ is valid in SL, we infer that $Y \vdash S^{\vee}$ and thus $\top \vdash Y \Rightarrow S^{\vee}$ is also valid in SL, and the lower sequent is thus of the third form.

3. The upper sequent has the form $\Gamma \vdash \Delta, X$ where $\top \vdash X$ is valid in SL.

   - First case, $X \notin S$

   $$\frac{\Gamma \vdash \Delta, X}{\Gamma \backslash \{H\} \vdash \Delta \backslash S, X, H \Rightarrow S^{\vee}}$$

   the lower sequent is still of the third form.

   - Second case, H is discharged on $S \cup \{X\}$

   $$\frac{\Gamma \vdash \Delta, X}{\Gamma \backslash \{H\} \vdash \Delta \backslash S, H \Rightarrow (S^{\vee} \vee X)}$$

   Since $\top \vdash X$ is derivable in SL, we infer that $H \vdash S^{\vee} \vee X$ and thus $\top \vdash H \Rightarrow (S^{\vee} \vee X)$ are also valid in SL, and the lower sequent is thus still of the third form.

The closure under the left-hand side introduction rule for subtraction is obtained by duality. □

C.2 Rules

Left-hand side weakening rule

- First case, $H \neq A$:

$$\frac{\dfrac{\Gamma \vdash^1 \Delta}{\Gamma, A \vdash^2 \Delta}}{\Gamma \backslash \{H\}, A \vdash^3 \Delta \backslash S, A \Rightarrow S^{\vee}}$$

where $H \notin \mathrm{hyp}_2(\Delta \backslash S)$ and thus $H \notin \mathrm{hyp}_1(\Delta \backslash S)$. Replace with:

$$\frac{\dfrac{\Gamma \vdash \Delta}{\Gamma \backslash \{H\} \vdash \Delta \backslash S, A \Rightarrow S^{\vee}}}{\Gamma \backslash \{H\}, A \vdash \Delta \backslash S, A \Rightarrow S^{\vee}}$$

- Second case, $H = A$:

$$\frac{\dfrac{\Gamma \vdash^1 \Delta}{\Gamma, A \vdash^2 \Delta}}{\Gamma \vdash^3 \Delta \backslash S, A \Rightarrow S^{\vee}}$$

where $A \notin \mathrm{hyp}_2(\Delta \backslash S)$ and thus $A \notin \mathrm{hyp}_1(\Delta \backslash S)$. Replace with:

$$\frac{\Gamma \vdash \Delta}{\Gamma \vdash \Delta \backslash S, A \Rightarrow S^{\vee}}$$

Left-hand side contraction rule

- First case, $H = A^z$:

$$\dfrac{\dfrac{\Gamma, A^x, A^y \vdash^1 \Delta}{\Gamma, A^z \vdash^2 \Delta}}{\Gamma \vdash^3 \Delta \backslash S, A \Rightarrow S^{\vee}}$$

where $A^z \notin hyp_2(\Delta \backslash S)$ and thus $A^x \notin \mathrm{hyp}_1(\Delta \backslash S)$ and $A^y \notin \mathrm{hyp}_1(\Delta \backslash \{B^S\})$. Replace with:

$$\dfrac{\dfrac{\Gamma, A^x, A^y \vdash \Delta}{\Gamma, A^z \vdash \Delta \backslash S, A \Rightarrow S^{\vee}}}{\Gamma \vdash \Delta \backslash S, A \Rightarrow (A \Rightarrow S^{\vee})}$$

and then cut with the sequent $A \Rightarrow (A \Rightarrow S^{\vee}) \vdash A \Rightarrow S^{\vee}$ valid in SL.

- Second case, $H \neq A^z$:

$$\dfrac{\dfrac{\Gamma, A^x, A^y \vdash^1 \Delta}{\Gamma, A^z \vdash^2 \Delta}}{\Gamma \backslash \{H\}, A^z \vdash^3 \Delta \backslash S, H \Rightarrow S^{\vee}}$$

where $H \notin \mathrm{hyp}_2(\Delta \backslash S)$ and thus $H \notin \mathrm{hyp}_1(\Delta \backslash S)$. Replace with:

$$\dfrac{\dfrac{\Gamma, A^x, A^y, H \vdash \Delta}{\Gamma \backslash \{H\}, A^x, A^y \vdash \Delta \backslash S, H \Rightarrow S^{\vee}}}{\Gamma \backslash \{H\}, A^z \vdash \Delta \backslash S, H \Rightarrow S^{\vee}}$$

Right-hand side contraction rule

- First case, $B \notin S$:

$$\dfrac{\dfrac{\Gamma \vdash^1 \Delta, B^{\alpha}, B^{\beta}}{\Gamma \vdash^2 \Delta, B^{\gamma}}}{\Gamma \backslash \{H\} \vdash^3 \Delta \backslash S, H \Rightarrow S^{\vee}, B^{\gamma}}$$

where $H \notin \mathrm{hyp}_2(\Delta \backslash S, B^{\gamma})$ and thus $H \notin \mathrm{hyp}_1(\Delta \backslash S, B^{\alpha}, B^{\beta})$. Replace with:

$$\dfrac{\dfrac{\Gamma \vdash \Delta, B, B}{\Gamma \backslash \{H\} \vdash \Delta \backslash S, H \Rightarrow S^{\vee}, B, B}}{\Gamma \backslash \{H\} \vdash \Delta \backslash S, H \Rightarrow S^{\vee}, B}$$

- Second case, H is discharged on $S \cup \{B\}$:

$$\dfrac{\dfrac{\Gamma \vdash^1 \Delta, B^{\alpha}, B^{\beta}}{\Gamma \vdash^2 \Delta, B^{\gamma}}}{\Gamma \backslash \{H\} \vdash^3 \Delta \backslash S, H \Rightarrow (S^{\vee} \vee B)}$$

where $H \notin \mathrm{hyp}_2(\Delta \backslash S, B^{\gamma})$ and thus $H \notin \mathrm{hyp}_1(\Delta \backslash S, B^{\alpha}, B^{\beta})$. Replace with:

$$\dfrac{\Gamma \vdash \Delta, B, B}{\Gamma \backslash \{H\} \vdash \Delta \backslash S, H \Rightarrow (S^{\vee} \vee B \vee B)}$$

and then cut with the sequent $H \Rightarrow (S^{\vee} \vee B \vee B) \vdash H \Rightarrow (S^{\vee} \vee B)$ valid in SL.

Right-hand side weakening rule

- First case, $B \notin S$:

$$\dfrac{\dfrac{\Gamma \vdash^1 \Delta}{\Gamma \vdash^2 \Delta, B}}{\Gamma \backslash \{H\} \vdash^3 \Delta \backslash S, H \Rightarrow S^{\vee}, B}$$

where $H \notin \mathrm{hyp}_2(\Delta\backslash S, B)$ and thus $H \notin \mathrm{hyp}_1(\Delta\backslash S)$. Replace with:

$$\dfrac{\dfrac{\Gamma \vdash \Delta}{\Gamma\backslash\{H\} \vdash \Delta\backslash S, H \Rightarrow S^\vee}}{\Gamma\backslash\{H\} \vdash \Delta\backslash S, H \Rightarrow S^\vee, B}$$

- Second case, H is discharged on $S \cup \{B\}$:

$$\dfrac{\dfrac{\Gamma \vdash^1 \Delta}{\Gamma \vdash^2 \Delta, B^\alpha}}{\Gamma\backslash\{H\} \vdash^3 \Delta\backslash S, H \Rightarrow (S^\vee \vee B)}$$

where $H \notin \mathrm{hyp}_2(\Delta\backslash S, B^\alpha)$ and thus $H \notin \mathrm{hyp}_1(\Delta\backslash S)$. Replace with:

$$\dfrac{\Gamma \vdash \Delta}{\Gamma\backslash\{H\} \vdash \Delta\backslash S, H \Rightarrow S^\vee}$$

and then cut with the sequent $H \Rightarrow S^\vee \vdash H \Rightarrow (S^\vee \vee B)$ valid in SL.

Cut rule

$$\dfrac{\dfrac{\Gamma' \vdash^1 A, \Delta' \qquad \Gamma'', A \vdash^4 \Delta''}{\Gamma', \Gamma'' \vdash^2 \Delta', \Delta''}}{(\Gamma', \Gamma'')\backslash\{H\} \vdash^3 (\Delta', \Delta'')\backslash S, H \Rightarrow S^\vee}$$

where $H \notin \mathrm{hyp}_2((\Delta', \Delta'')\backslash S)$ and thus, by setting $S' = S \cap \Gamma'$ and $S'' = S \cap \Gamma''$:

- First case, $H \notin \Gamma'$. Replace with:

$$\dfrac{\dfrac{\dfrac{\Gamma' \vdash \Delta', A}{\Gamma' \vdash \Delta'\backslash S', H \Rightarrow S'^\vee, A} \qquad \dfrac{\Gamma'', A \vdash \Delta''}{\Gamma''\backslash\{H\}, A \vdash \Delta''\backslash S', H \Rightarrow S''^\vee}}{(\Gamma', \Gamma'')\backslash\{H\} \vdash (\Delta', \Delta'')\backslash S, H \Rightarrow S'^\vee, H \Rightarrow S''^\vee}}{(\Gamma', \Gamma'')\backslash\{H\} \vdash (\Delta', \Delta'')\backslash S, (H \Rightarrow S'^\vee) \vee (H \Rightarrow S''^\vee)}$$

  then cut with the sequent $(H \Rightarrow S'^\vee) \vee (H \Rightarrow S''^\vee) \vdash H \Rightarrow (S'^\vee \vee S''^\vee)$ valid in SL.

- Second case, $H \notin \Gamma''$ and $H \notin \mathrm{hyp}_1(A)$ since $H \notin \mathrm{hyp}_1(A, \Delta'\backslash S')$. Replace with:

$$\dfrac{\dfrac{\dfrac{\Gamma' \vdash \Delta', A}{\Gamma'\backslash\{H\} \vdash \Delta'\backslash S', H \Rightarrow S'^\vee, A} \qquad \dfrac{\Gamma'', A \vdash \Delta''}{\Gamma'', A \vdash \Delta''\backslash S', H \Rightarrow S''^\vee}}{(\Gamma', \Gamma'')\backslash\{H\} \vdash (\Delta', \Delta'')\backslash S, H \Rightarrow S'^\vee, H \Rightarrow S''^\vee}}{(\Gamma', \Gamma'')\backslash\{H\} \vdash (\Delta', \Delta'')\backslash S, (H \Rightarrow S'^\vee) \vee (H \Rightarrow S''^\vee)}$$

  then cut with the sequent $(H \Rightarrow S'^\vee) \vee (H \Rightarrow S''^\vee) \vdash H \Rightarrow (S'^\vee \vee S''^\vee)$ valid in SL.

- Third case, $H \in \Gamma'$ and $H \in \mathrm{hyp}_1(A)$ and since $H \notin \mathrm{hyp}_1(\Delta''\backslash S'')$ we have $A \notin \mathrm{hyp}_1(\Delta''\backslash S')$. Replace with:

$$\dfrac{\dfrac{\Gamma' \vdash \Delta', A}{\Gamma'\backslash\{H\} \vdash \Delta'\backslash S', H \Rightarrow (S'^\vee \vee A)} \qquad \dfrac{\Gamma'', A \vdash \Delta''}{\Gamma'' \vdash \Delta'\backslash S', A \Rightarrow S''^\vee}}{(\Gamma', \Gamma'')\backslash\{H\} \vdash (\Delta', \Delta'')\backslash S, (H \Rightarrow (S'^\vee \vee A)) \wedge (A \Rightarrow S''^\vee)}$$

  then cut with the sequent $(H \Rightarrow (S'^\vee \vee A)) \wedge (A \Rightarrow S''^\vee) \vdash H \Rightarrow (S'^\vee \vee S''^\vee)$ valid in SL.

Right-hand side elimination rule for the implication

- First case, $B \notin S$:

$$\dfrac{\dfrac{\Gamma' \vdash^1 \Delta', A \Rightarrow B \qquad \Gamma'' \vdash^4 \Delta'', A}{\Gamma', \Gamma'' \vdash^2 \Delta', \Delta'', B}}{(\Gamma', \Gamma'')\backslash\{H\} \vdash^3 (\Delta', \Delta'')\backslash S, B, H \Rightarrow S^\vee}$$

where $H \notin \mathrm{hyp}_2(B, (\Delta', \Delta'')\backslash S)$ and thus, by setting $S' = S \cap \Gamma'$ and $S'' = S \cap \Gamma''$, we have $H \notin \mathrm{hyp}_1(A \Rightarrow B, \Delta\backslash S')$ and $H \notin \mathrm{hyp}_4(A, \Delta\backslash S')$. Replace with:

$$\dfrac{\dfrac{\dfrac{\Gamma' \vdash \Delta', A \Rightarrow B}{\Gamma'\backslash\{H\} \vdash \Delta'\backslash S', H \Rightarrow S'^{\vee}, A \Rightarrow B} \qquad \dfrac{\Gamma'' \vdash \Delta'', A}{\Gamma''\backslash\{H\} \vdash \Delta''\backslash S'', H \Rightarrow S''^{\vee}, A}}{(\Gamma', \Gamma'')\backslash\{H\} \vdash (\Delta', \Delta'')\backslash S, H \Rightarrow S'^{\vee}, H \Rightarrow S''^{\vee}, B}}{(\Gamma', \Gamma'')\backslash\{H\} \vdash (\Delta', \Delta'')\backslash S, (H \Rightarrow S'^{\vee}) \vee (H \Rightarrow S''^{\vee}), B}$$

and then cut with the sequent $(H \Rightarrow S'^{\vee}) \vee (H \Rightarrow S''^{\vee}) \vdash H \Rightarrow (S'^{\vee} \vee S''^{\vee})$ valid in SL.

- Second case, H is discharged on $S \cup \{B\}$:

$$\dfrac{\dfrac{\Gamma' \vdash^1 \Delta', A \Rightarrow B \qquad \Gamma'' \vdash^4 \Delta'', A}{\Gamma', \Gamma'' \vdash^2 \Delta', \Delta'', B}}{(\Gamma', \Gamma'')\backslash\{H\} \vdash^3 (\Delta', \Delta'')\backslash S, H \Rightarrow (S^{\vee} \vee B)}$$

where $H \notin \mathrm{hyp}_2((\Delta', \Delta'')\backslash S)$ and thus, by setting $S' = S \cap \Gamma'$ and $S'' = S \cap \Gamma''$, there is $H \notin \mathrm{hyp}_1(\Delta\backslash S')$ and $H \notin \mathrm{hyp}_4(\Delta\backslash S')$. Replace with:

$$\dfrac{\dfrac{\Gamma' \vdash \Delta', A \Rightarrow B}{\Gamma'\backslash\{H\} \vdash \Delta'\backslash S', H \Rightarrow (S'^{\vee} \vee (A \Rightarrow B))} \qquad \dfrac{\Gamma'' \vdash \Delta'', A}{\Delta''\backslash S', H \Rightarrow (S''^{\vee} \vee A)}}{(\Gamma', \Gamma'')\backslash\{H\} \vdash (\Delta', \Delta'')\backslash S, (H \Rightarrow (S'^{\vee} \vee (A \Rightarrow B))) \wedge (H \Rightarrow (S''^{\vee} \vee A))}$$

and then cut with the sequent $(H \Rightarrow (S'^{\vee} \vee (A \Rightarrow B))) \wedge (H \Rightarrow (S''^{\vee} \vee A)) \vdash H \Rightarrow (S^{\vee} \vee B)$ valid in SL.

Left-hand side introduction rule of disjunction

- First case, $H \neq A \vee B$:

$$\dfrac{\dfrac{\Gamma', A \vdash^1 \Delta' \qquad \Gamma'', B \vdash^4 \Delta''}{\Gamma', \Gamma'', A \vee B \vdash^2 \Delta', \Delta''}}{(\Gamma', \Gamma'')\backslash\{H\}, A \vee B \vdash^3 (\Delta', \Delta'')\backslash S, H \Rightarrow S^{\vee}}$$

where $H \notin \mathrm{hyp}_2((\Delta', \Delta'')\backslash S)$ and thus, by setting $S' = S \cap \Gamma'$ and $S'' = S \cap \Gamma''$, we have $H \notin \mathrm{hyp}_1(\Delta\backslash S')$ and $H \notin \mathrm{hyp}_4(\Delta\backslash S')$. Replace with:

$$\dfrac{\dfrac{\dfrac{\Gamma', A \vdash \Delta'}{\Gamma'\backslash\{H\}, A \vdash \Delta'\backslash S', H \Rightarrow S'^{\vee}} \qquad \dfrac{\Gamma'', B \vdash \Delta''}{\Gamma''\backslash\{H\}, B \vdash \Delta'', H \Rightarrow S''^{\vee}}}{(\Gamma', \Gamma'')\backslash\{H\}, A \vee B \vdash (\Delta', \Delta'')\backslash S, H \Rightarrow S'^{\vee}, H \Rightarrow S''^{\vee}}}{(\Gamma', \Gamma'')\backslash\{H\}, A \vee B \vdash (\Delta', \Delta'')\backslash S, (H \Rightarrow S'^{\vee}) \vee (H \Rightarrow S''^{\vee})}$$

and then cut with the sequent $(H \Rightarrow S'^{\vee}) \vee (H \Rightarrow S''^{\vee}) \vdash H \Rightarrow (S'^{\vee} \vee S''^{\vee})$ valid in SL.

- Second case, $H = A \vee B$:

$$\dfrac{\dfrac{\Gamma', A \vdash^1 \Delta' \qquad \Gamma'', B \vdash^4 \Delta''}{\Gamma', \Gamma'', A \vee B \vdash^2 \Delta', \Delta''}}{\Gamma', \Gamma'' \vdash^3 (\Delta', \Delta'')\backslash S, A \vee B \Rightarrow S^{\vee}}$$

where $A \vee B \notin \mathrm{hyp}_2((\Delta', \Delta'')\backslash S)$ and thus, by setting $S' = S \cap \Gamma'$ and $S'' = S \cap \Gamma''$, we have $A \notin \mathrm{hyp}_1(\Delta\backslash S')$ and $B \notin \mathrm{hyp}_4(\Delta\backslash S')$. Replace with:

$$\dfrac{\dfrac{\Gamma', A \vdash \Delta'}{\Gamma'\backslash\{H\} \vdash \Delta'\backslash S', A \Rightarrow S'^{\vee}} \qquad \dfrac{\Gamma'', B \vdash \Delta''}{\Delta''\backslash S', B \Rightarrow S''^{\vee}}}{(\Gamma', \Gamma'')\backslash\{H\} \vdash (\Delta', \Delta'')\backslash S, (A \Rightarrow S'^{\vee}) \wedge (B \Rightarrow S''^{\vee})}$$

and then cut with the sequent $(A \Rightarrow S'^{\vee}) \wedge (B \Rightarrow S''^{\vee}) \vdash (A \vee B) \Rightarrow (S'^{\vee} \vee S''^{\vee})$ valid in SL.

Left-hand side elimination rule of disjunction

We consider only the case of the first *injection*, the second may be treated in a similar way.

- First case, $H \neq A$:

$$\dfrac{\dfrac{\Gamma, A \vee B \vdash^1 \Delta}{\Gamma, A \vdash^2 \Delta}}{\Gamma \backslash \{H\}, A \vdash^3 \Delta \backslash S, H \Rightarrow S^\vee}$$

where $H \notin \mathrm{hyp}_2(\Delta \backslash S)$ and thus $H \notin \mathrm{hyp}_1(\Delta \backslash S)$. Replace with:

$$\dfrac{\dfrac{\Gamma, A \vee B \vdash \Delta}{\Gamma \backslash \{H\}, A \vee B \vdash \Delta \backslash S, H \Rightarrow S^\vee}}{\Gamma \backslash \{H\}, A \vdash \Delta \backslash S, H \Rightarrow S^\vee}$$

- Second case, $H = A$:

$$\dfrac{\dfrac{\Gamma, A \vee B \vdash^1 \Delta}{\Gamma, A \vdash^2 \Delta}}{\Gamma \backslash \{H\} \vdash^3 \Delta \backslash S, A \Rightarrow S^\vee}$$

where $H \notin \mathrm{hyp}_2(\Delta \backslash S)$ and thus $H \notin \mathrm{hyp}_1(\Delta \backslash S)$ and $A \notin \mathrm{hyp}_4(\Delta \backslash S)$. Replace with:

$$\dfrac{\Gamma, A \vee B \vdash \Delta}{\Gamma \backslash \{H\} \vdash \Delta \backslash S, (A \vee B) \Rightarrow S^\vee}$$

and then cut with the sequent $(A \vee B) \Rightarrow S^\vee \vdash A \Rightarrow S^\vee$ valid in SL.

Introduction rule for conjunction

- First case, $A \wedge B \notin S$:

$$\dfrac{\dfrac{\Gamma' \vdash^1 \Delta', A \qquad \Gamma'' \vdash^4 \Delta'', B}{\Gamma', \Gamma'' \vdash^2 \Delta', \Delta'', A \wedge B}}{(\Gamma', \Gamma'') \backslash \{H\} \vdash^3 (\Delta', \Delta'') \backslash S, A \wedge B, H \Rightarrow S^\vee}$$

where $H \notin \mathrm{hyp}_2(A, B, (\Delta', \Delta'') \backslash S)$ and thus by setting $S' = S \cap \Gamma'$ et $S'' = S \cap \Gamma''$, we have $H \notin \mathrm{hyp}_1(A, \Delta \backslash S')$ and $H \notin \mathrm{hyp}_4(B, \Delta \backslash S')$. Replace with:

$$\dfrac{\dfrac{\dfrac{\Gamma' \vdash \Delta', A}{\Gamma' \backslash \{H\} \vdash \Delta' \backslash S', H \Rightarrow S'^\vee, A} \qquad \dfrac{\Gamma'' \vdash \Delta'', B}{\Gamma'' \backslash \{H\} \vdash \Delta'' \backslash S'', H \Rightarrow S''^\vee, B}}{(\Gamma', \Gamma'') \backslash \{H\} \vdash (\Delta', \Delta'') \backslash S, H \Rightarrow S'^\vee, H \Rightarrow S''^\vee, A \wedge B}}{(\Gamma', \Gamma'') \backslash \{H\} \vdash (\Delta', \Delta'') \backslash S, (H \Rightarrow S'^\vee) \vee (H \Rightarrow S''^\vee), A \wedge B}$$

then cut with the sequent $(H \Rightarrow S'^\vee) \vee (H \Rightarrow S''^\vee) \vdash H \Rightarrow (S'^\vee \vee S''^\vee)$ valid in SL.

- Second case, H is discharged over $S \cup \{A \wedge B\}$:

$$\dfrac{\dfrac{\Gamma' \vdash^1 \Delta', A \qquad \Gamma'' \vdash^4 \Delta'', B}{\Gamma', \Gamma'' \vdash^2 \Delta', \Delta'', A \wedge B}}{(\Gamma', \Gamma') \backslash \{H\} \vdash^3 (\Delta', \Delta'') \backslash S, H \Rightarrow (S^\vee \vee (A \wedge B))}$$

where $H \notin \mathrm{hyp}_2((\Delta',\ \Delta'') \backslash S)$ and thus by setting $S' = S \cap \Gamma'$ et $S'' = S \cap \Gamma''$, on a $H \notin \mathrm{hyp}_1(\Delta \backslash S')$ and $H \notin \mathrm{hyp}_4(\Delta \backslash S')$. Replace with:

$$\dfrac{\dfrac{\Gamma' \vdash \Delta', A}{\Gamma' \backslash \{H\} \vdash \Delta' \backslash S', H \Rightarrow (S'^\vee \vee A)} \qquad \dfrac{\Gamma'' \vdash \Delta'', B}{\Delta'' \backslash S', H \Rightarrow (S''^\vee \vee B)}}{(\Gamma', \Gamma'') \backslash \{H\} \vdash (\Delta', \Delta'') \backslash S, (H \Rightarrow (S'^\vee \vee A)) \wedge (H \Rightarrow (S''^\vee \vee B))}$$

then cut with the sequent $(H \Rightarrow (S'^\vee \vee A)) \wedge (H \Rightarrow (S''^\vee \vee B)) \vdash H \Rightarrow (S^\vee \vee (A \wedge B))$ valid in SL.

Elimination rule for conjunction

We deal only with the first projection, the other one is similar.

- First case, $A \notin S$:

$$\dfrac{\dfrac{\Gamma \vdash^1 \Delta, A \wedge B}{\Gamma \vdash^2 \Delta, A}}{\Gamma \backslash \{H\} \vdash^3 \Delta \backslash S, H \Rightarrow S^\vee, A}$$

where $H \notin \mathrm{hyp}_2(A, \Delta\backslash S)$ and thus $H \notin \mathrm{hyp}_1(A \wedge B, \Delta \backslash S)$. Replace with:

$$\dfrac{\dfrac{\Gamma \vdash \Delta, A \wedge B}{\Gamma \backslash \{H\} \vdash \Delta \backslash S, H \Rightarrow S^\vee, A \wedge B}}{\Gamma \backslash \{H\} \vdash \Delta \backslash S, H \Rightarrow S^\vee, A}$$

- Second case, H is discharged over $S \cup \{A\}$:

$$\dfrac{\dfrac{\Gamma \vdash^1 \Delta, A \wedge B}{\Gamma \vdash^2 \Delta, A}}{\Gamma \backslash \{H\} \vdash^3 \Delta \backslash S, H \Rightarrow (S^\vee \vee A)}$$

where $H \notin \mathrm{hyp}_2(\Delta \backslash S)$ and thus $H \notin \mathrm{hyp}_1(\Delta \backslash S)$ and $A \notin \mathrm{hyp}_4(\Delta \backslash S)$. Replace with:

$$\dfrac{\Gamma \vdash \Delta, A \wedge B}{\Gamma \backslash \{H\} \vdash \Delta \backslash S, H \Rightarrow (S^\vee \vee (A \wedge B))}$$

then cut with the sequent $H \Rightarrow (S^\vee \vee (A \wedge B)) \vdash H \Rightarrow (S^\vee \vee A)$ valid in SL.

Left-hand side elimination rule for subtraction

- First case, $H \neq A$:

$$\dfrac{\dfrac{\Gamma', A - B \vdash^1 \Delta' \qquad \Gamma'', B \vdash^4 \Delta''}{\Gamma', \Gamma'', A \vdash^2 \Delta', \Delta''}}{(\Gamma', \Gamma'') \backslash \{H\}, A \vdash^3 (\Delta', \Delta'') \backslash S, H \Rightarrow S^\vee}$$

where $H \notin \mathrm{hyp}_2((\Delta', \Delta'') \backslash S)$ and thus by setting $S' = S \cap \Gamma'$ et $S'' = S \cap \Gamma''$, we have $H \notin \mathrm{hyp}_1(\Delta \backslash S')$ and $H \notin \mathrm{hyp}_4(\Delta \backslash S')$. Replace with:

$$\dfrac{\dfrac{\dfrac{\Gamma', A - B \vdash \Delta'}{\Gamma' \backslash \{H\}, A - B \vdash \Delta' \backslash S', H \Rightarrow S'^\vee} \qquad \dfrac{\Gamma'', B \vdash \Delta''}{\Gamma'' \backslash \{H\}, B \vdash \Delta'', H \Rightarrow S''^\vee}}{(\Gamma', \Gamma'') \backslash \{H\}, A \vdash (\Delta', \Delta'') \backslash S, H \Rightarrow S'^\vee, H \Rightarrow S''^\vee}}{(\Gamma', \Gamma'') \backslash \{H\}, A \vdash (\Delta', \Delta'') \backslash S, (H \Rightarrow S'^\vee) \vee (H \Rightarrow S''^\vee)}$$

then cut with the sequent $(H \Rightarrow S'^\vee) \vee (H \Rightarrow S''^\vee) \vdash H \Rightarrow (S'^\vee \vee S''^\vee)$ valid in SL.

- Second case, $H = A$:

$$\dfrac{\dfrac{\Gamma', A - B \vdash^1 \Delta' \qquad \Gamma'', B \vdash^4 \Delta''}{\Gamma', \Gamma'', A \vdash^2 \Delta', \Delta''}}{\Gamma', \Gamma'' \vdash^3 (\Delta', \Delta'') \backslash S, A \Rightarrow S^\vee}$$

where $A \notin \mathrm{hyp}_2((\Delta', \Delta'') \backslash S)$ and thus by setting $S' = S \cap \Gamma'$ and $S'' = S \cap \Gamma''$, we have $A - B \notin \mathrm{hyp}_1(\Delta \backslash S')$ and $B \notin \mathrm{hyp}_4(\Delta \backslash S')$. Replace with:

$$\dfrac{\dfrac{\Gamma', A - B \vdash \Delta'}{\Gamma' \backslash \{H\} \vdash \Delta' \backslash S', (A - B) \Rightarrow S'^\vee} \qquad \dfrac{\Gamma'', B \vdash \Delta''}{\Delta'' \backslash S', B \Rightarrow S''^\vee}}{(\Gamma', \Gamma'') \backslash \{H\} \vdash (\Delta', \Delta'') \backslash S, ((A - B) \Rightarrow S'^\vee) \wedge (B \Rightarrow S''^\vee)}$$

then cut with the sequent $((A - B) \Rightarrow S'^\vee) \wedge (B \Rightarrow S''^\vee) \vdash (A \Rightarrow (S'^\vee \vee S''^\vee)$ valid in SL.

Left-hand side introduction rule for subtraction

- First case, $H \neq A - B$:

$$\dfrac{\dfrac{\Gamma, A \vdash^1 \Delta, B}{\Gamma, A - B \vdash^2 \Delta}}{\Gamma \backslash \{H\}, A - B \vdash^3 \Delta \backslash S, H \Rightarrow S^\vee}$$

where $H \notin \mathrm{hyp}_2(\Delta\backslash S)$ and $H \notin \mathrm{hyp}_1(B)$ and thus $H \notin \mathrm{hyp}_1(\Delta\backslash S, B)$. Replace with:

$$\dfrac{\dfrac{\Gamma, A \vdash \Delta, B}{\Gamma\backslash\{H\}, A \vdash \Delta\backslash S, H \Rightarrow S^{\vee}, B}}{\Gamma\backslash\{H\}, A - B \vdash \Delta\backslash S, H \Rightarrow S^{\vee}}$$

- Second case, $H = A - B$:

$$\dfrac{\dfrac{\Gamma, A \vdash^1 \Delta, B}{\Gamma, A - B \vdash^2 \Delta}}{\Gamma \vdash^3 \Delta\backslash S, (A - B) \Rightarrow S^{\vee}}$$

where $H \notin \mathrm{hyp}_2(\Delta\backslash S)$ and $H \notin \mathrm{hyp}_1(B)$ and thus $H \notin \mathrm{hyp}_1(\Delta\backslash S, B)$. Replace with:

$$\dfrac{\Gamma, A \vdash \Delta, B}{\Gamma \vdash \Delta\backslash S, A \Rightarrow (S^{\vee} \vee B)}$$

then cut with the sequent $A \Rightarrow (S^{\vee} \vee B) \vdash (A - B) \Rightarrow S^{\vee}$ valid in SL.

C.3 Quantifiers

Right-hand side introduction rule for $\forall$ (where x does not occur in Γ, Δ)

- First case, $\forall x\, A \notin S$:

$$\dfrac{\dfrac{\Gamma \vdash^1 \Delta, A}{\Gamma \vdash^2 \Delta, \forall x\, A}}{\Gamma\backslash\{H\} \vdash^3 \Delta\backslash S, H \Rightarrow S^{\vee}, \forall x\, A}$$

where $H \notin \mathrm{hyp}_2(\Delta\backslash S, \forall x\, A)$ and thus $H \notin \mathrm{hyp}_1(\Delta\backslash S, A)$. Replace with:

$$\dfrac{\dfrac{\Gamma \vdash^1 \Delta, A}{\Gamma\backslash\{H\} \vdash^3 \Delta\backslash S, H \Rightarrow S^{\vee}, A}}{\Gamma\backslash\{H\} \vdash^3 \Delta\backslash S, H \Rightarrow S^{\vee}, \forall x\, A}$$

- Second case, H is discharged on $S \cup \{\forall x\, A\}$:

$$\dfrac{\dfrac{\Gamma \vdash^1 \Delta, A}{\Gamma \vdash^2 \Delta, \forall x\, A}}{\Gamma\backslash\{H\} \vdash^3 \Delta\backslash S, H \Rightarrow (S^{\vee} \vee \forall x\, A)}$$

where $H \notin \mathrm{hyp}_2(\Delta\backslash S)$ and thus $H \notin \mathrm{hyp}_1(\Delta\backslash S)$ and $A \notin \mathrm{hyp}_4(\Delta\backslash S)$. Replace with:

$$\dfrac{\dfrac{\Gamma \vdash \Delta, A}{\Gamma\backslash\{H\} \vdash \Delta\backslash S, H \Rightarrow (S^{\vee} \vee A)}}{\Gamma\backslash\{H\} \vdash \Delta\backslash S, \forall x(H \Rightarrow (S^{\vee} \vee A))}$$

then cut with the sequent $\forall x(H \Rightarrow (S^{\vee} \vee A)) \vdash H \Rightarrow (S^{\vee} \vee \forall x\, A)$ valid in SL since x does not occur in $H, S^{\vee}$. Notice that DIS is actually needed to prove this sequent.

Right-hand side elimination rule for $\forall$

- First case, $A \notin S$:

$$\dfrac{\dfrac{\Gamma \vdash^1 \Delta, \forall x\, A}{\Gamma \vdash^2 \Delta, A[t/x]}}{\Gamma\backslash\{H\} \vdash^3 \Delta\backslash S, H \Rightarrow S^{\vee}, A}$$

where $H \notin \mathrm{hyp}_2(A[t/x], \Delta\backslash S)$ and thus $H \notin \mathrm{hyp}_1(\forall x\, A, \Delta\backslash S)$. Replace with:

$$\dfrac{\dfrac{\Gamma \vdash^1 \Delta, \forall x\, A}{\Gamma\backslash\{H\} \vdash^3 \Delta\backslash S, H \Rightarrow S^{\vee}, \forall x\, A}}{\Gamma\backslash\{H\} \vdash^3 \Delta\backslash S, H \Rightarrow S^{\vee}, A[t/x]}$$

- Second case, H is discharged on $S \cup \{\forall x\, A\}$:

$$\dfrac{\dfrac{\Gamma \vdash^1 \Delta, \forall x\, A}{\Gamma \vdash^2 \Delta, A[t/x]}}{\Gamma \backslash \{H\} \vdash^3 \Delta \backslash S, H \Rightarrow (S^{\vee} \vee A[t/x])}$$

where $H \notin \mathrm{hyp}_2(\Delta \backslash S)$ and thus $H \notin \mathrm{hyp}_1(\Delta \backslash S)$ and $A \notin \mathrm{hyp}_4(\Delta \backslash S)$. Replace with:

$$\dfrac{\Gamma \vdash \Delta, \forall x\, A}{\Gamma \backslash \{H\} \vdash \Delta \backslash S, H \Rightarrow (S^{\vee} \vee A)}$$

then cut with the sequent $H \Rightarrow (S^{\vee} \vee \forall x\, A) \vdash H \Rightarrow (S^{\vee} \vee A[t/x])$ valid in SL.

Left-hand side introduction rule for $\exists$ (where x does not occur in Γ, Δ)

- First case, $H \neq \exists x\, A$:

$$\dfrac{\dfrac{\Gamma, A \vdash^1 \Delta}{\Gamma, \exists x\, A \vdash^2 \Delta}}{\Gamma \backslash \{H\} \vdash^3 \Delta \backslash S, H \Rightarrow S^{\vee}}$$

where $H \notin \mathrm{hyp}_2(\Delta \backslash S)$ and thus $H \notin \mathrm{hyp}_1(\Delta \backslash S)$. Replace with:

$$\dfrac{\dfrac{\Gamma, A \vdash \Delta, A}{\Gamma \backslash \{H\}, A \vdash \Delta \backslash S, H \Rightarrow S^{\vee}}}{\Gamma \backslash \{H\}, \exists x\, A \vdash \Delta \backslash S, H \Rightarrow S^{\vee}}$$

- Second case, $H = \exists x\, A$:

$$\dfrac{\dfrac{\Gamma, A \vdash^1 \Delta}{\Gamma, \exists x\, A \vdash^2 \Delta}}{\Gamma \vdash^3 \Delta \backslash S, \exists x\, A \Rightarrow S^{\vee}}$$

where $\exists x\, A \notin \mathrm{hyp}_2(\Delta \backslash S)$ and thus $A \notin \mathrm{hyp}_1(\Delta \backslash S)$. Replace with:

$$\dfrac{\dfrac{\Gamma \vdash \Delta, A}{\Gamma \backslash \{H\} \vdash \Delta \backslash S, A \Rightarrow S^{\vee}}}{\Gamma \backslash \{H\} \vdash \Delta \backslash S, \forall x (A \Rightarrow S^{\vee})}$$

then cut with the sequent $\forall x (A \Rightarrow S^{\vee}) \vdash (\exists x\, A) \Rightarrow S^{\vee}$ valid in SL (since x does not occur $S^{\vee}$).

Left-hand side elimination rule for $\exists$

- First case $H \neq \exists x\, A$:

$$\dfrac{\dfrac{\Gamma, \exists x\, A \vdash^1 \Delta}{\Gamma, A[t/x] \vdash^2 \Delta}}{\Gamma \backslash \{H\}, A[t/x] \vdash^3 \Delta \backslash S, H \Rightarrow S^{\vee}}$$

where $H \notin \mathrm{hyp}_2(\Delta \backslash S)$ and thus $H \notin \mathrm{hyp}_1(\Delta \backslash S)$. Replace with:

$$\dfrac{\dfrac{\Gamma, \exists x\, A \vdash \Delta}{\Gamma \backslash \{H\}, \exists x\, A \vdash \Delta \backslash S, H \Rightarrow S^{\vee}}}{\Gamma \backslash \{H\}, A[t/x] \vdash \Delta \backslash S, H \Rightarrow S^{\vee}}$$

- Second case $H = \exists x\, A$:

$$\dfrac{\dfrac{\Gamma, \exists x\, A \vdash^1 \Delta}{\Gamma, A[t/x] \vdash^2 \Delta}}{\Gamma \vdash^3 \Delta \backslash S, A[t/x] \Rightarrow S^{\vee}}$$

where $A[t/x] \notin \mathrm{hyp}_2(\Delta \backslash S)$ and thus $\exists x\, A \notin \mathrm{hyp}_1(\Delta \backslash S)$. Replace with:

$$\dfrac{\Gamma, \exists x\, A \vdash \Delta}{\Gamma \backslash \{H\} \vdash \Delta \backslash S, \exists x\, A \Rightarrow S^{\vee}}$$

Bibliography

[1] F. Barbanera and S. Berardi. Extracting constructive content from classical logic via control-like reductions. In *LNCS*, volume 662, pages 47–59. Springer-Verlag, 1994.

[2] E. W. Beth. Semantic construction of intuitionistic logic. *Koninklijke Nederlandse Akademie van Wetenschappen, Mededelingen, Nieuwe Reeks*, 19(11):357–388, 1956.

[3] E. C. Cooper and J. G. Morrisett. Adding threads to standard ML. Report CMU-CS-90-186, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, 1990.

[4] T. Crolard. Extension de l'isomorphisme de Curry-Howard au traitement des exceptions (application d'une étude de la dualité en logique intuitionniste). Thèse de Doctorat. Université Paris 7, 1996.

[5] T. Crolard. A confluent lambda-calculus with a catch/throw mechanism. *Journal of Functional Programming*, 9(6):625–647, 1999.

[6] T. Crolard. Typage des coroutines en logique soustractive. In *Journées Francophones des Langages Applicatifs*, Collection Didactique de l'INRIA, Février 1999.

[7] T. Crolard. Subtractive Logic. *Theoretical Computer Science*, 254(1–2):151–185, 2001.

[8] P.-L. Curien and H. Herbelin. The duality of computation. In *Proceedings of the ACM Sigplan International Conference on Functional Programming (ICFP-00)*, volume 35.9 of *ACM Sigplan Notices*, pages 233–243, N.Y., September 18–21 2000. ACM Press.

[9] P. De Groote. On the relation between the lambda-calculus and the syntactic theory of sequential control. *Lecture Notes in Computer Science*, 822:31–??, 1994.

[10] P. de Groote. A simple calculus of exception handling. In *Second International Conference on Typed Lambda Calculi and Applications*, LNCS, pages 201–215, Edinburgh, United Kingdom, 1995.

[11] P. de Groote. An environment machine for the lambda-mu-calculus. *Mathematical Structure in Computer Science*, 8:637–669, 1998.

[12] P. de Groote. Strong normalization of classical natural deduction with disjunction. In Typed Lambda Calculi and Applications, editors, *LNCS*, pages LNCS 2044, p. 182 ff.1–13. Springer-Verlag, 2001.

[13] A. G. Dragalin. Mathematical intuitionism: introduction to proof theory. In *Translations of Mathematical Monographs*, volume 67. American Matematical Society, Providence, Rhode Island, 1988.

[14] Bruce Duba, Robert Harper, and David MacQueen. Typing first-class continuations in ML. In ACM-SIGPLAN ACM-SIGACT, editor, *Conference Record of the 18th Annual ACM Symposium on Principles of Programming Languages (POPL '91)*, pages 163–173, Orlando, FL, USA, January 1991. ACM Press.

[15] R. Dyckhoff. Contraction-free sequent calculi for intuitionistic logic. *The Journal of Symbolic Logic*, 57(3):795–807, 1992.

[16] A. Filinski. Declarative Continuations: An Investigation of Duality in Programming Language Semantics. In *Category Theory and Comp. Sci.*, volume 389 of *LNCS*, pages 224–249. Springer-Verlag, 1989.

[17] D. P. Friedman, C. T. Haynes, and M. Wand. Continuations and coroutines: An exercise in metaprogramming. In *Proc. 1984 ACM Symposium on Lisp and Functional Programming*, pages 293–298, August 1984.

[18] D. P. Friedman, C. T. Haynes, and M. Wand. Obtaining coroutines with continuations. *Journal of Computer Languages*, 11(3/4):143–153, 1986.

[19] D. M. Gabbay. *Semantical Investigations in Heyting's Intuitionistic Logic*. Reidel, Dordrecht, 1981.

[20] R. Goré. Dual intuitionistic logic revisited. In Roy Dyckhoff, editor, *Automated Reasoning with Analytic Tableaux and Related Methods, International Conference, TABLEAUX 2000, St Andrews, Scotland, UK, July 3-7, 2000, Proceedings*, volume 1847 of *Lecture Notes in Computer Science*, pages 252–267. Springer, 2000.

[21] S. Görnemann. A logic stronger than intuitionism. *The Journal of Symbolic Logic*, 36:249–261, 1971.

[22] T. G. Griffin. A formulæ-as-type notion of control. In *Conference Record of the Seventeenth Annual ACM Symposium on Principles of Programming Langages*, pages 47–58, 1990.

[23] The Open Group. The Single UNIX Specification, Version 2, 1997. (http://www.UNIX-systems.org/online.html).

[24] Y. Kameyama. A new formulation of the catch/throw mechanism. In T. Ida, A. Ohori, and M. Takeichi, editors, *Second Fuji International Workshop on Functional and Logic Programming*, Word Scientific, pages 106–122, 1997.

[25] Y. Kameyama and M. Sato. A classical catch/throw calculus with tag abstraction and its strong normalizability. In X. Lin, editor, *Proc. the 4th Australasian Theory Symposium*, volume 20-3 of *Australian Computer Science Communications*, pages 183–197. Springer-Verlag, 1998.

[26] Y. Kameyama and M. Sato. Strong normalizability of the non-deterministic catch/throw calculi. *Theoretical Computer Science*, 272(1–2):223–245, 2002.

[27] J.-L. Krivine. Classical logic, storage operators and second order λ-calculus. *Ann. of Pure and Appl. Logic*, 68:53–78, 1994.

[28] C. R. Murthy. Classical proofs as programs: How, when, and why. Technical Report 91-1215, Cornell University, Department of Computer Science, 1991.

[29] H. Nakano. A constructive logic behind the catch and throw mechanism. *Annals of Pure and Applied Logic*, 69(3):269–301, 1994.

[30] H. Nakano. The non-deterministic catch and throw mechanism and its subject reduction property. In *Logic, Language and Computation*, volume 592 of *LNCS*, pages 61–72. Springer-Verlag, 1994.

[31] H. Nakano. *The Logical Structures of the Catch and Throw Mechanism*. PhD thesis, The University of Tokyo, 1995.

[32] H. Ono. Model extension theorem and Graigs interpolation theorem for intermediate predicate logics. *Rep. Math. Logic*, 15:41–57, 1983.

[33] M. Parigot. $\lambda\mu$-calculus: an algorithmic interpretation of classical natural deduction. In *Proc. Logic Prog. and Autom. Reasoning*, volume 624 of *LNCS*, pages 190–201, 1992.

[34] M. Parigot. Classical proofs as programs. In *Computationnal logic and theory*, volume 713 of *LNCS*, pages 263–276. Springer-Verlag, 1993.

[35] M. Parigot. Strong normalisation for second order classical natural deduction. In *Proceedings of the eighth annual IEEE symposium on logic in computer science*, 1993.

[36] D. Pym and E. Ritter. On the intuitionistic force of classical search. *Theoretical Computer Science*, 232(1-2):299–333, 2000.

[37] D. Pym and E. Ritter. On the semantics of classical disjunction. *Journal of Pure and Applied Algebra*, 159:315–338, 2001.

[38] D. Pym, E. Ritter, and L. Wallen. Proof-terms for classical and intuitionistic resolution. In M. A. McRobbie and J. K. Slaney, editors, *Proceedings of the Thirteenth International Conference on Automated Deduction (CADE-96)*, volume 1104 of *LNAI*, pages 17–31, Berlin, July 30–August 3 1996. Springer.

[39] N. Ramsey. Concurrent programming in ML. Technical Report CS-TR-262-90, Department of Computer Science, Princeton University, Princeton, NJ, 1990.

[40] C. Rauszer. A formalization of the propositional calculus of H-B logic. *Studia Logica*, 33(1):23–34, 1974.

[41] C. Rauszer. Semi-boolean algebras and their applications to intuitionistic logic with dual operations. In *Fundamenta Mathematicae*, volume 83, pages 219–249, 1974.

[42] C. Rauszer. An algebraic and Kripke-style approach to a certain extension of intuitionistic logic. In *Dissertationes Mathematicae*, volume 167. Institut Mathématique de l'Académie Polonaise des Sciences, 1980.

[43] N. J. Rehof and M. H. S⊘rensen. The λ_Δ-calculus. In *Theoritical Aspects of Computer Software*, volume 542 of *LNCS*, pages 516–542. Springer-Verlag, 1994.

[44] J. H. Reppy. Asynchronous signals is standard ML. Technical Report TR90-1144, Cornell University, Computer Science Department, August 1990.

[45] J. H. Reppy. First-class synchronous operations. *Lecture Notes in Computer Science*, 907:235–252, 1995.

[46] P. Selinger. Control categories and duality: On the categorical semantics of the lambda-mu calculus. *Mathematical Structures in Computer Science*, 11(2):207–260, 2001.

[47] T. Streicher and B. Reus. Classical logic, continuation semantics and abstract machines. *Journal of Functional Programming*, 8(6):543–572, 1998.

[48] M. E. Szabo. *Gentzen Collected work*. North-Holland, Amsterdam, 1969.

[49] M. Wand. Continuation-based multiprocessing. In J. Allen, editor, *Conference Record of the 1980 LISP Conference*, pages 19–28, Palo Alto, CA, 1980. The Lisp Company. Republished by ACM.