Syntax and Models of a non-Associative Composition of Programs and Proofs

Guillaume Munch-Maccagnoni

▶ **To cite this version:**

Guillaume Munch-Maccagnoni. Syntax and Models of a non-Associative Composition of Programs and Proofs. Logic in Computer Science [cs.LO]. Université Paris-Diderot - Paris VII, 2013. English. NNT : . tel-00918642

HAL Id: tel-00918642

https://theses.hal.science/tel-00918642

Submitted on 13 Dec 2013

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Université Paris Diderot – Paris 7
Sorbonne Paris Cité

École doctorale Sciences Mathématiques de Paris Centre

THÈSE
en vue d'obtenir le grade de
DOCTEUR DE L'UNIVERSITÉ PARIS DIDEROT
Spécialité Informatique fondamentale

SYNTAXE ET MODÈLES D'UNE COMPOSITION NON-ASSOCIATIVE DES PROGRAMMES ET DES PREUVES

Présentée et soutenue publiquement par
Guillaume Munch
le 10 décembre 2013

devant Messieurs les membres du jury

Pierre-Louis Curien Thomas Ehrhard	*Directeurs*
Andrzej Filinski Thomas Streicher	*Rapporteurs*
Hugo Herbelin Andrew Pitts	*Examinateurs*
Jean-Yves Girard	*Président*

Syntax and Models of a non-Associative Composition

of Programs and Proofs

Guillaume Munch-Maccagnoni

11.12.2013 17:57

Remerciements

Mes premiers remerciements vont à Pierre-Louis Curien, le directeur de ma thèse. Au-delà de l'inspiration que suscitent ses travaux, merci pour son intérêt constant et pour les opportunités qu'il m'a offertes, notamment à Cambridge. Je lui suis reconnaissant d'avoir consacré une partie de ses propres recherches à ce sujet, grâce à quoi il a quelquefois pu apporter une aide essentielle: ce n'est pas facile d'encadrer un jeune chercheur qui débarque avec son propre sujet de recherche, et en cela il a dépassé mes espérances. Merci d'avoir parlé de mon travail.

Avant Pierre-Louis, c'est avec Hugo Herbelin que j'ai eu mon premier contact avec le monde de la recherche. Beaucoup d'idées contenues dans cette thèse n'auraient pas émergé sans son travail et sa vision de la logique. Je le remercie pour sa dévotion envers ses étudiants, les nombreuses discussions et débats, son intérêt et son soutien. C'est tout naturellement qu'il devait faire partie de mon jury, et qu'il ait accepté m'honore.

C'est Daniel Hirschkoff qui, lorsque j'étais un jeune élève de l'ENS de Lyon, m'a mis sur la piste de la « dualité du calcul ». C'est le premier qui m'a parlé de la machine de Krivine (« c'est facile, ça consiste juste à faire plic-plic » avec un geste des doigts). Par la suite, il m'a envoyé chez Hugo Herbelin. Pour tout cela, je lui suis reconnaissant.

Cette thèse ne serait rien sans l'inspiration que beaucoup d'entre nous tirent des travaux de Jean-Yves Girard et de Jean-Louis Krivine. Leur vision décloisonnée de la théorie de la démonstration rétablit les ponts avec les autres disciplines. Merci à Jean-Yves, en particulier, d'être un modèle pour la radicalité de sa pensée. Merci également pour ses idées et pour son soutien. C'est un grand honneur qu'il me fait en présidant mon jury.

Je tiens à remercier Thomas Ehrhard, qui m'a accordé beaucoup de temps au tout début, avant de devoir se consacrer entièrement à la direction du laboratoire. Ma thèse a pris une autre direction, mais j'ai toujours eu la plus haute estime pour ses travaux et sa présence dans mon jury m'honore.

À l'Université de Cambridge, je tiens à remercier tout d'abord Tim Griffin pour ses encouragements et Marcelo Fiore pour son intérêt et pour le long travail de défrichage de nos futures (je le souhaite) pistes de recherche. Travailler avec Marcelo m'a permis de développer un certain point de vue. Il n'y aurait pas de chapitre dans le langage des catégories sans son influence: je lui en suis reconnaissant. Merci à Andrew Pitts pour ses encouragements, et pour l'honneur qu'il me fait d'être membre de mon jury. Merci aux thésards de Cambridge.

Je suis reconnaissant envers Thomas Streicher et Andrzej Filinski d'avoir accepté de rapporter cette thèse, ce qu'ils ont accompli avec rigueur et célérité. Merci à Thomas en particulier, qui a fourni un effort extraordinaire pour être convaincu de tous les détails, y compris dans les domaines qui ne lui étaient pas familiers. Merci à Andrzej pour avoir travaillé jusqu'au dernier moment.

Olivier Danvy est un modèle pour son dévouement envers les jeunes chercheurs: qu'il en soit remercié. Je lui suis reconnaissant d'avoir posé une question qui, *dûment reformulée*, revenait à demander si l'implémentation de l'appel par valeur dans l'appel par nom d'Algol 60 pouvait s'expliquer avec une monade exécutable. Je n'avais pas pensé à celles-ci comme structure de calcul avant cela, et de toute évidence je n'étais pas le seul. Mais mes remerciements les plus appuyés concernent le fait qu'il ait assuré, avec beaucoup d'attention, la relecture de dernière minute de ma thèse.

Merci à Paul-André Melliès. Il a toujours su trouver du temps pour expliquer, inlassablement, sa « *vision atomiste de la logique* ». Merci aussi pour son nouvel enthousiasme envers cette « *période merveilleuse* » pour nos domaines.

La période ne m'était pas toujours apparue merveilleuse, et la rencontre, en 2011, des gens du Groupe de Travail Logique de la rue d'Ulm fut un tournant heureux pour le contexte de ma thèse. Le *GdT* réunis-

sait tous ceux à Paris qui se reconnaissaient dans l'idée que l'on ne peut plus faire de théorie de la démonstration sans comprendre la programmation. Sa richesse et sa diversité ont fourni un environnement scientifique qui a nourri ma réflexion. Un grand merci à Marc Bagnol, Aloïs Brunel, Guillaume Brunerie, Charles Grellois, Adrien Guatto[1], Baptiste Mélès, Pierre-Marie Pédrot, Maël Pégny, Silvain Rideau, Gabriel Scherer, Anne-Sophie de Suzzoni, et bien d'autres.

Enfin, merci à Beniamino Accattoli, Emmanuel Beffara, Danko Ilik, Delia Kesner, Stéphane Lengrand, Alexandre Miquel[2], Noam Zeilberger et bien d'autres pour les nombreuses discussions. Merci à Alexis Saurin. Pour le plaisir qu'il y a à enseigner à leurs côtés, merci à Inès Klimann, Christophe Prieur, Yann Régis-Gianas, Christine Tasson et Ralf Treinen. Merci à Odile Ainardi et son aide précieuse, que les autres laboratoires doivent bien nous envier. Merci à Yves Legrandgérard.

Merci aux thésards de PPS et en tout premier lieu à ceux que j'ai le plus côtoyés et dont j'ai partagé le bureau: Alexis Goyet, Fabien Renaud, Giulio Guerrieri, Marie Kerjean et Shahin Amini. Merci à Pierre-Évariste Dagand pour sa relecture attentive et enthousiaste.

Une pensée particulière pour mes proches et en particulier pour mes amis de l'ENS de Lyon (Charlotte, Marc, Raphaël) et d'avant (Antoine, Julien). Merci à Jehanne.

[1] Aux jeux de mots mémorables. (« *Highway to L* », hélas, ne fonctionne pas en anglais. Il m'a fallu trouver un autre titre.)

[2] Merci pour avoir souligné l'analogie entre les termes positifs et les distributions.

Contents

Extended abstract in French

L'hypothèse de la polarisation

Le λ-calcul est à la base d'une correspondance célèbre entre une partie de la théorie des catégories, la programmation fonctionnelle et la théorie de la démonstration intuitionniste. Au début des années 90, des liens supplémentaires sont apparus entre:

- la modélisation catégorielle des effets;
- la modélisation de l'ordre d'évaluation et du contrôle au sein des langages de programmation;
- la théorie de la démonstration classique.

Depuis, ces trois domaines fournissent une vision interactive du calcul. Mais il reste à établir les principes qui régissent l'extension de la correspondance à ce point de vue interactif.

Les polarités, qui distinguent les connecteurs négatifs ($\forall$, $\to \dots$) des connecteurs positifs ($\exists$, $\vee \dots$), sont héritées de l'intuitionnisme. La reconnaissance de leur importance théorique est l'un des fruits du tournant interactif, à la fois pour la théorie des langages de programmation et pour la théorie de la démonstration.

La polarisation correspond à l'hypothèse que les polarités doivent être prises en compte de manière formelle:

- pour les modèles catégoriels, la polarisation correspond à relâcher l'hypothèse que la composition est associative *a priori*, sur certaines polarités;

- dans les modèles des langages de programmation à base de continuations, les polarités déterminent si une continuation est destinée à être appliquée ou à être passée;
- dans la théorie de la démonstration classique, les polarités déterminent l'élimination des coupures.

Ainsi, la polarisation est responsable de l'ordre d'évaluation des programmes, et de la constructivité des démonstrations. Les polarités apparaissent derrière de nombreux phénomènes:

- le problème de Blass dans les sémantiques de jeux décrit par Abramsky [Abr03] et par Melliès [Mel05];
- la présence de paresse en appel par valeur décrite par Hatcliff et Danvy [HD97] et par Führmann [Füh99];
- l'incorrection du polymorphisme libre en appel par valeur en présence d'effets de bord (Harper et Lillibridge [HL91]);
- la hiérarchie arithmétique dans l'étude de l'arithmétique de Peano;
- la focalisation dans la recherche de démonstrations (Andreoli [And92]);
- les relations logiques à base d'orthogonalité (voir Girard [Gir87], Krivine [Kri09], Pitts [Pit05], et, en ce qui concerne la polarisation, M.-M. [Mun09]);

et certainement davantage.

La thèse contribue à la compréhension de la nature, du rôle et des mécanismes de la polarisation dans les trois domaines. Notre approche est basée sur une représentation interactive des démonstrations et des programmes à base de termes, qui met en évidence la structure des polarités. Par *interactif* on entend que la nouveauté par rapport au λ-calcul est le rôle explicite accordé au contexte d'évaluation (pour les programmes) ou à l'opposant (pour les démonstrations). Elle est basée sur les lieurs μ et $\tilde{\mu}$ et sur la relation entre les machines abstraites et les calculs de séquents, introduits

par Curien et Herbelin avec le calcul $\bar{\lambda}\mu\tilde{\mu}$ [CH00]. En l'honneur de Gentzen [Gen35] qui a introduit les calculs de séquents **LJ** et **LK**, et suivant une suggestion d'Herbelin, on utilise la lettre "**L**" pour nommer les calculs basés sur les lieurs μ et $\tilde{\mu}$.

La syntaxe **L** a pour champ l'étude de la structure des démonstrations selon Gentzen et la modélisation des langages de programmation d'ordre supérieur selon Landin [Lan64, Lan65]. Le but est de fournir les moyens d'une unification de directions de recherche récentes: la modélisation des effets à la suite de Moggi [Mog89], la quête d'un lien entre la dualité catégorielle et les continuations à la suite de Filinski [Fil89], et la notion interactive de construction selon Girard [Gir87, Gir91, Gir01] et Krivine [Kri09].

Dans chacune des trois contributions qui suivent, correspondant chacune à un chapitre, nous avons trouvé que les calculs **L** fournissent la représentation canonique d'une structure intéressante:

- Dans le Chapitre II, on caractérise la polarisation à travers une structure catégorielle où la condition d'associativité de la composition est relâchée. La caractérisation détaille comment la polarisation est aux adjonctions ce que l'appel par valeur en programmation est aux monades. Un langage interne est fourni par le calcul $\mathbf{L}_{\mathrm{dup}}$.
- Dans le Chapitre III, on décompose les traductions par passage de continuation pour le contrôle délimité en trois étapes: 1) l'implémentation des opérations du langage comme solutions d'équations dans un calcul abstrait en style direct; 2) la traduction du style direct au style indirect; 3) le retour au λ-calcul, obscurcissant la traduction. Les calculs $\mathbf{L}_{\mathrm{pol},\hat{\mathrm{tp}}^{+}}$ et $\mathbf{L}_{\mathrm{exp}}$ sont utilisés comme étapes intermédiaires dans la traduction.
- Dans le Chapitre IV, on décrit une correspondance "formule comme type" pour une négation involutive en déduction naturelle. Pour cela on introduit le calcul $\lambda\ell$, qui implémente l'idée que les contextes capturés, contrairement aux continuations, permettent l'accès à leurs constituants. Le raisonnement extensionnel est permis à travers le calcul $\mathbf{L}_{\mathrm{pol},\hat{\mathrm{tp}}^{\ominus}}$.

Ces chapitres sont précédés d'une introduction aux calculs **L**, qui suppose du lecteur uniquement une connaissance élémentaire du λ-calcul simplement typé et de la réécriture.

Certaines hypothèses simplificatrices rendraient les polarités moins saillantes. Notre étude a laissé les phénomènes intéressants apparaître parce qu'on a cherché une description du calcul qui soit non-typée (à la Curry), extensionnelle et directe. L'approche directe des programmes et des démonstrations souligne que lorsqu'un système est étudié au moyen d'une traduction dans un système plus simple, comprendre la traduction est aussi important que comprendre sa cible. Ces trois contraintes nous permettent d'échapper aux pièges des hypothèses suivantes:

- Rechercher un modèle catégoriel qui met en jeu une catégorie unique — par exemple, si le but est d'obtenir une généralisation catégorielle des algèbres de Boole, pourquoi conserver l'associativité de la composition?
- Rechercher un cadre fortement normalisant *a priori* — par exemple, beaucoup de propriétés remarquables du Système F [Gir72], notamment d'extensionnalité et donc d'isomorphismes, sont une conséquence de la normalisation forte [GSS92]. Mais, outre qu'elle nous lie à une logique particulière, la normalisation *dépolarise* le système, c'est-à-dire que l'ordre d'évaluation n'a plus d'importance.
- Supposer que la transparence référentielle est essentielle pour la constructivité — par exemple, cette hypothèse n'est pas valide dans les interprétations directes de la logique classique à base d'opérateurs de contrôle. En présence de tels opérateurs, l'ordre d'évaluation est important pour la constructivité, même lorsque le cadre est fortement normalisant.
- Supposer que l'on peut se restreindre *a priori* à des démonstrations de certaines formes — par exemple, l'étude du calcul des prédicats classique est simplifiée si on se restreint aux démonstrations η-longues. Mais il est connu que l'astuce se limite au

premier ordre et correspond en outre à une traduction de Gödel-Gentzen (Laurent [Lau02]).

Nous nous sommes rendu compte que de telles hypothèses ne sont pas inoffensives.

On résume maintenant les contributions de chaque chapitre.

L'approche « L » des programmes et des démonstrations

Le Chapitre 1 introduit l'approche **L** et en particulier la polarisation. Cette introduction s'inspire des articles publiés: M.-M. [Mun09] et Curien et M.-M. [CM10]. La fin du chapitre présente par ailleurs en détail les contributions des trois chapitres qui suivent, dans leurs contextes respectifs.

On résume ici les arguments principaux du chapitre en prenant pour exemple le λC-calcul de Lafont, Reus et Streicher [LRS93]. La Figure 1 présente le λC-calcul ainsi que son évaluation dans une machine de Krivine décrite par Streicher et Reus [SR98]. Il s'agit d'un calcul pour la déduction naturelle classique inspiré de l'interprétation de Griffin [Gri90] qui type l'opérateur C de Felleisen par l'élimination de la double négation:

$$C : \neg\neg A \to A\,.$$

La Figure 2 présente le calcul $\mathbf{L}_{\mathrm{n}}$ (originellement $\bar{\lambda}\mu\tilde{\mu}_T$) qui est une syntaxe en appel par nom pour le calcul des séquents classique introduite par Curien et Herbelin [CH00]. Les règles de typage dans le calcul des séquents sont présentés dans la Figure 3.

L'idée qui relie les deux calculs est que les règles de transition des

$$t, u ::= x \mid \lambda x\, t \mid t\, u \mid C$$

(a) Termes quasi-preuves

$$N, M ::= X(\mathrm{t}_1, \ldots, \mathrm{t}_n) \mid N \to M \mid \forall \mathrm{x} N \mid \bot$$

(b) Formules

$$\frac{}{\Gamma, x : N \vdash x : N} \qquad \frac{}{\Gamma \vdash C : \neg\neg N \to N}$$

$$\frac{\Gamma, x : N \vdash t : M}{\Gamma \vdash \lambda x\, t : N \to M} \qquad \frac{\Gamma \vdash t : N \to M \qquad \Gamma \vdash u : N}{\Gamma \vdash t\, u : M}$$

$$\frac{\Gamma \vdash t : N}{\Gamma \vdash t : \forall \mathrm{x} N}\ (\mathrm{x} \notin \mathbf{fv}(\Gamma)) \qquad \frac{\Gamma \vdash t : \forall \mathrm{x} N}{\Gamma \vdash t : N[\mathrm{u}/\mathrm{x}]}$$

(c) Déduction naturelle classique

$$c ::= \langle t \parallel \pi \rangle$$
$$t, u ::= \ldots \mid \mathsf{k}_\pi$$
$$\pi ::= \mathsf{stop} \mid t \cdot \pi$$

(d) Machines, termes et piles

$$\begin{aligned}
\langle t\, u \parallel \pi \rangle &\succ_n \langle t \parallel u \cdot \pi \rangle \\
\langle \lambda x\, t \parallel u \cdot \pi \rangle &\succ_n \langle t[u/x] \parallel \pi \rangle \\
\langle C \parallel t \cdot \pi \rangle &\succ_n \langle t \parallel \mathsf{k}_\pi \cdot \mathsf{stop} \rangle \\
\langle \mathsf{k}_\pi \parallel t \cdot \pi' \rangle &\succ_n \langle t \parallel \pi \rangle
\end{aligned}$$

(e) Règles de réduction

Figure 1: Le λC-calcul en appel par nom

opérations du λC-calcul dans la machine abstraite:

$$\begin{aligned}
\langle t\, u \parallel \pi \rangle &\succ_n \langle t \parallel u \cdot \pi \rangle \\
\langle \lambda x\, t \parallel u \cdot \pi \rangle &\succ_n \langle t[u/x] \parallel \pi \rangle \\
\langle C \parallel t \cdot \pi \rangle &\succ_n \langle t \parallel \mathsf{k}_\pi \cdot \mathsf{stop} \rangle \\
\langle \mathsf{k}_\pi \parallel t \cdot \pi' \rangle &\succ_n \langle t \parallel \pi \rangle
\end{aligned}$$

constituent des équations que l'on peut résoudre dans $\mathbf{L}_\mathrm{n}$. Les opé-

$$t = t_{\ominus} ::= x \mid \mu\alpha.c \mid \lambda x.t$$
$$e = e_{\ominus} ::= \pi \mid \tilde{\mu}x.c$$
$$e_{\ominus} \supseteq \pi ::= \alpha \mid \mathsf{stop} \mid t \cdot \pi$$
$$c ::= \langle t \parallel e \rangle$$

(a) Termes, contextes et commandes

$$t\ u \stackrel{\text{def}}{=} \mu\alpha.\langle t \parallel u \cdot \alpha \rangle$$
$$\mathsf{k}_e \stackrel{\text{def}}{=} \lambda x.\mu\alpha.\langle x \parallel e \rangle$$
$$C \stackrel{\text{def}}{=} \lambda x.\mu\alpha.\langle x \parallel \mathsf{k}_\alpha \cdot \mathsf{stop} \rangle$$

(b) Plongement du λC-calcul

$$\langle t \parallel \tilde{\mu}x.c \rangle \ \triangleright_{\mathrm{R}_n}\ c[t/x]$$
$$\langle \mu\alpha.c \parallel \pi \rangle \ \triangleright_{\mathrm{R}_n}\ c[\pi/\alpha]$$
$$\langle \lambda x.t \parallel u \cdot \pi \rangle \ \triangleright_{\mathrm{R}_n}\ \langle t[u/x] \parallel \pi \rangle$$

(c) Règles de réduction

$$e \ \triangleright_{\mathrm{E}_n}\ \tilde{\mu}x.\langle x \parallel e \rangle$$
$$t \ \triangleright_{\mathrm{E}_n}\ \mu\alpha.\langle t \parallel \alpha \rangle$$
$$t \ \triangleright_{\mathrm{E}_n}\ \lambda x.\mu\alpha.\langle t \parallel x \cdot \alpha \rangle$$

(d) Règles d'expansion

Figure 2: Le calcul $\mathbf{L_n}$

rations du λC-calcul sont en effet caractérisées par leur action sur les piles:

$$\begin{aligned} \lambda x\, t : &\quad u \cdot \pi \mapsto \langle t[u/x] \parallel \pi \rangle \\ t\ u : &\quad \pi \mapsto \langle t \parallel u \cdot \pi \rangle \\ \mathsf{k}_\pi : &\quad t \cdot \pi' \mapsto \langle t \parallel \pi \rangle \\ C : &\quad u \cdot \pi \mapsto \langle u \parallel \mathsf{k}_\pi \cdot \mathsf{stop} \rangle \end{aligned}$$

Le lieur μ rend cette action interne au langage des termes, en liant des *co*-variables $(\alpha, \beta \dots)$ dénotant des piles:

$$\mu\alpha.c : \pi \mapsto c[\pi/\alpha]$$

On en déduit des solutions à ces équations, c'est-à-dire une définition des constructeurs du λC-calcul:

$$\begin{aligned} t\ u &= \mu\alpha.\langle t \parallel u \cdot \alpha \rangle \\ \mathsf{k}_\pi &= \lambda x.\mu\alpha.\langle x \parallel \pi \rangle \\ C &= \lambda x.\mu\alpha.\langle x \parallel \mathsf{k}_\alpha \cdot \mathsf{stop} \rangle \end{aligned}$$

$$\Gamma = \vec{x_i} : \vec{N_i} \qquad \Delta = \vec{\alpha_j} : \vec{M_j}$$
$$\Gamma \vdash t : N \mid \Delta \qquad \Gamma \mid e : N \vdash \Delta$$
$$c : (\Gamma \vdash \Delta)$$

(a) Jugements

$$\frac{}{\Gamma, x : N \vdash x : N \mid \Delta}\ (\vdash \mathrm{ax}) \qquad \frac{}{\Gamma \mid \alpha : N \vdash \alpha : N, \Delta}\ (\mathrm{ax} \vdash)$$

$$\frac{c : (\Gamma, x : N \vdash \Delta)}{\Gamma \mid \tilde{\mu} x.c : N \vdash \Delta}\ (\tilde{\mu} \vdash) \qquad \frac{c : (\Gamma \vdash \alpha : N, \Delta)}{\Gamma \vdash \mu \alpha.c : N \mid \Delta}\ (\vdash \mu)$$

$$\frac{\Gamma \vdash t : N \mid \Delta \qquad \Gamma \mid e : N \vdash \Delta}{\langle t \parallel e \rangle : (\Gamma \vdash \Delta)}\ (\mathrm{cut})$$

(b) Groupe identité et structure

$$\frac{\Gamma, x : N \vdash t : M \mid \Delta}{\Gamma \vdash \lambda x.t : N \to M \mid \Delta}\ (\vdash \to) \qquad \frac{\Gamma \vdash t : N \mid \Delta \qquad \Gamma \mid \pi : M \vdash \Delta}{\Gamma \mid t \cdot \pi : N \to M \vdash \Delta}\ (\to \vdash_f)$$

$$\frac{\Gamma \vdash t : N \mid \Delta}{\Gamma \vdash t : \forall \mathrm{x} N \mid \Delta}\ (\vdash \forall)^* \qquad \frac{\Gamma \mid e : N[\mathrm{t/x}] \vdash \Delta}{\Gamma \mid e : \forall \mathrm{x} N \vdash \Delta}\ (\forall \vdash)$$

$$(* : x \notin \mathbf{fv}(\Gamma, \Delta)) \qquad \frac{}{\Gamma \mid \mathsf{stop} : \bot \vdash \Delta}\ (\bot \vdash)$$

(c) Groupe logique

Figure 3: Typage dans le calcul des séquents en appel par nom

Le λC-calcul se plonge ainsi dans le calcul $\mathbf{L}_{\mathrm{n}}$ et hérite d'une relation d'équivalence contextuelle qui est extensionnelle.

De façon remarquable, considérer le λC-calcul à travers son implémentation dans le calcul $\mathbf{L}_{\mathrm{n}}$ entraîne trois simplifications importantes:

1. Les règles de typage sont en correspondance avec celles du calcul des séquents.
2. Grâce à l'ajout de contextes explicites, la théorie équationnelle

est plus simple que sa description directe dans le λC-calcul.

3. Les choix qui ont été faits concernant la stratégie d'évaluation apparaissent plus clairement, en particulier avec une symétrie entre l'appel par valeur et l'appel par nom, caractérisés par des règles de réduction qui donnent la priorité au terme ou au contexte.

On illustre cela dans ce qui suit.

Une correspondance avec le calcul des séquents

L'idée est d'associer à la coupure du calcul des séquents classique de Gentzen [Gen35]:

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

la règle de typage suivante:

$$\frac{\Gamma \vdash t : A \mid \Delta \qquad \Gamma \mid \pi : A \vdash \Delta}{\langle t \parallel \pi \rangle : (\Gamma \vdash \Delta)} \ .$$

Γ devient un environnement de variables de termes $(x_1 : A_1, \dots, x_n : A_n)$ et Δ devient un environnement de variables de piles $(\alpha_1 : B_1, \dots, \alpha_m : B_m)$.

$$\boxed{\begin{array}{c} \Gamma \vdash t : A \mid \Delta \\ \Gamma \mid \pi : A \vdash \Delta \\ \langle t \parallel \pi \rangle : (\Gamma \vdash \Delta) \end{array}}$$

Dans l'encadré ci-dessus, le premier des trois jugements est familier: le type d'un terme est donné par une conclusion du séquent. Le second jugement décrit le type d'une pile par une hypothèse du séquent, et peut se lire « π *est une réfutation de* A *dans le contexte* Γ, Δ ». Le troisième jugement attribue des types aux variables de la machine $\langle t \parallel \pi \rangle$, laquelle n'a pas de type propre. Le slogan est que la réduction des commandes correspond à l'élimination des coupures.

Adjoints

La notation abstraite $\mu/\tilde{\mu}$ résout le problème des coupures commutatives de la déduction naturelle. C'est expliqué en grande partie par le fait que les règles d'élimination sont retrouvées comme les adjointes des règles d'introduction à gauche, à travers la définition:

$$\boxed{\tau^*(t) \stackrel{\text{def}}{=} \mu\alpha.\langle t \,\|\, \tau(\alpha)\rangle}\,.$$

Ainsi, la dérivation de la règle d'élimination de $\to$ dans **LK**:

$$\dfrac{\Gamma \vdash A \to B, \Delta \qquad \dfrac{\Gamma \vdash A, \Delta \qquad \overline{\Gamma, B \vdash B, \Delta}}{\Gamma, A \to B \vdash B, \Delta}}{\Gamma \vdash B, \Delta}$$

donne bien $t\,u = (u \cdot -)^*(t)$, puisqu'elle est en correspondance avec la dérivation suivante:

$$\dfrac{\Gamma \vdash t : A \to B \mid \Delta \qquad \dfrac{\Gamma \vdash u : A \mid \Delta \qquad \overline{\Gamma \mid \alpha : B \vdash \alpha : B, \Delta}^{\,(\text{ax}\vdash)}}{\Gamma \mid u \cdot \alpha : A \to B \vdash \alpha : B, \Delta}^{\,(\to\vdash)}}{\dfrac{\langle t \,\|\, u \cdot \alpha\rangle : (\Gamma \vdash \alpha : B, \Delta)}{\Gamma \vdash (u \cdot -)^*(t) : B \mid \Delta}^{\,(\vdash\mu)}}^{\,(\text{cut})}$$

Les règles de typage du λC-calcul de la Figure 2 sont retrouvées de cette façon à travers les règles de la Figure 3.

Contextes non linéaires

Les piles (notation π) sont des contextes (notation e) linéaires, au sens de la logique linéaire [Lau02]; leur présence suffit si le but est de décrire la réduction de tête d'un terme (machines de Krivine).

Curien et Herbelin introduisent le lieur $\tilde{\mu}$ qui définit des contextes non-linéaires à travers leur interaction avec un terme. Celui-ci rend explicite les choix d'ordres d'évaluation, et permet d'obtenir une correspondance complète entre machines et calcul des séquents. Le

contexte $\tilde{\mu}x.\langle t \,\|\, e\rangle$ est introduit par la règle suivante:

$$\frac{\langle t \,\|\, e\rangle : (\Gamma, x : A \vdash \Delta)}{\Gamma \mid \tilde{\mu}x.\langle t \,\|\, e\rangle : A \vdash \Delta}\ (\tilde{\mu} \vdash)$$

On peut comprendre $\tilde{\mu}x.\ \langle\ t\ \|\ \pi\ \rangle$ comme le contexte du terme u au sein du terme du λC-calcul $(\lambda x.\mathrm{k}_{\pi} t)u$. La complétude des règles de typage vis-à-vis de la prouvabilité dans le fragment $\to$, $\forall$, $\bot$ du calcul des séquents est établie avec la dérivation suivante qui montre comment obtenir la règle $(\to \vdash)$ manquante.

$$\frac{\Gamma \vdash t : N \mid \Delta \qquad \Gamma \mid e : M \vdash \Delta}{\Gamma \mid \tilde{\mu}x.\langle x\, t \,\|\, e\rangle : N \to M \vdash \Delta}$$

Polarité négative

Les termes du calcul $\mathbf{L}_{\mathrm{n}}$ sont négatifs. Cela signifie que la réduction du μ est restreinte aux contextes qui sont des piles:

$$\langle t_{\ominus} \,\|\, \tilde{\mu}x.c\rangle \rhd_{\mathrm{R}_n} c[t_{\ominus}/x]$$
$$\langle \mu\alpha.c \,\|\, \pi\rangle \rhd_{\mathrm{R}_n} c[\pi/\alpha]$$

En conséquence, l'évaluation d'un terme négatif est retardée tant qu'il n'est pas en position de tête, dans la terminologie du λ-calcul. En ce sens-là, la polarité négative décrit une réduction en appel par nom. Dans le calcul $\mathbf{L}_{\mathrm{n}}$, la position de tête est déterminée par le fait que le contexte est une pile π.

Polarité positive

La polarité positive, non représentée dans le calcul $\mathbf{L}_{\mathrm{n}}$, restreint la réduction du lieur $\tilde{\mu}$ aux termes qui sont des valeurs (V_+):

$$\langle \mu\alpha^+.c \,\|\, e_+\rangle \rhd_{\mathrm{R}_p} c[e_+/\alpha^+]$$
$$\langle V_+ \,\|\, \tilde{\mu}x^+.c\rangle \rhd_{\mathrm{R}_p} c[V_+/x^+]$$

Autrement dit un terme positif est appelé par valeur.

La polarité négative donne la priorité, dans $\langle\ \mu\alpha.c \parallel \tilde{\mu}x.c'\ \rangle$, à l'évaluation du contexte $\tilde{\mu}x.c'$, tandis que la polarité positive revient à donner la priorité à l'évaluation du terme $\mu\alpha.c$. Le lieur $\tilde{\mu}$ permet de décomposer la liaison $\mathsf{let}\, x\, \mathsf{be}\, t\, \mathsf{in}\, u$, qui est standard depuis Moggi [Mog91] pour modéliser l'appel par valeur. Le terme $\mathsf{let}\, x\, \mathsf{be}\, t\, \mathsf{in}\, u$, qui a la même polarité que u, est obtenu à travers la définition suivante:

$$\boxed{\mathsf{let}\, x\, \mathsf{be}\, t\, \mathsf{in}\, u_\varepsilon \overset{\text{def}}{=} \mu\alpha^\varepsilon.\langle t \,\|\, \tilde{\mu}x.\langle u_\varepsilon \,\|\, \alpha^\varepsilon\rangle\rangle}\ .$$

Dans ce cadre, un terme de la forme suivante:

$$\mathsf{let}\, y^\ominus\, \mathsf{be}\, (\mathsf{let}\, x^+\, \mathsf{be}\, t_+\, \mathsf{in}\, u_\ominus)\, \mathsf{in}\, v$$

commence par évaluer v, tandis que le suivant:

$$\mathsf{let}\, x^+\, \mathsf{be}\, t_+\, \mathsf{in}\, \mathsf{let}\, y^\ominus\, \mathsf{be}\, u_\ominus\, \mathsf{in}\, v$$

commence par évaluer t_+. Ces deux termes dénotent les deux façons de composer des termes t_+, $u_\ominus$ et v. Cette différence d'ordre d'évaluation illustre pourquoi la polarisation peut être décrite négativement comme le rejet, direct ou indirect, de l'hypothèse que la composition est associative *a priori*.

Les duploïdes, modèles d'une composition non-associative

Dans le Chapitre II, on introduit une description positive et directe de la polarisation. On caractérise l'ordre d'évaluation polarisé à travers une structure catégorielle où certaines associativités ne sont pas vérifiées. *Duploïde* est le nom de la structure, en référence aux algèbres dupliciales de Loday [Lod06].

Le résultat principal relie les duploïdes aux adjonctions. Pour mieux comprendre ce lien, rappelons tout d'abord la correspondance entre les modèles directs de l'appel par valeur et les modèles

indirects de Moggi.

Modèles directs

Dans un modèle dénotationnel *direct*, il doit y avoir une correspondance précise entre les opérations du modèle et les constructions du langage. Les opérations sur les types et sur les programmes doivent essentiellement correspondre aux opérations sur les objets et sur les morphismes d'une catégorie. En particulier, il doit être possible de raisonner sur une instance du modèle au sein du langage.[3] Un exemple de modèle direct du λ-calcul simplement typé est donné par les catégories cartésiennes fermées.

Dans un modèle tel que le λ_C de Moggi [Mog89], ou le modèle en appel par nom de Lafont, Reus et Streicher [LRS93], cependant, le langage n'est pas interprété directement mais à travers la construction de Kleisli d'une monade ou d'une co-monade. Führmann [Füh99] a décrit précisément le lien qui existe entre les modèles directs et les modèles indirects. Les catégories qui modélisent l'appel par valeur directement sont caractérisées par la présence d'un *thunk*, la structure bien connue qui permet d'implémenter la paresse en appel par valeur [HD97].

La caractérisation prend la forme suivante: tout modèle direct s'obtient par la construction de Kleisli à partir d'un λ_C modèle. Cependant, en partant d'un modèle direct on ne peut retrouver qu'un λ_C modèle spécifique: il est fait de valeurs sémantiques consistant en toutes les expressions pures. Plus précisément, la construction de Kleisli est une réflexion qui réunit toute paire de valeurs identifiées du point de vue de la monade, et donne le statut de valeur à toute expression *thunkable*. Une expression est thunkable si elle se comporte similairement à une valeur en un sens déterminé par la monade.

Selinger [Sel01] a démontré un lien similaire entre les modèles directs du λμ-calcul en appel par nom et les modèles de Lafont, Reus et Streicher [LRS93].

[3] Führmann [Füh99], Selinger [Sel99].

Ordre d'évaluation	Par valeur	Par nom	Polarisé
Modèle direct	Thunk	Monade exécutable	Duploïde
Modèle indirect	Monade T	Co-monade L	Adjonction $F \dashv G$
Programmes	Morphisme de Kleisli $P \to TQ$	Morphisme de co-Kleisli $LN \to M$	Morphisme oblique $FP \to N$ $\simeq P \to GN$
Données syntaxiques	Valeurs	Piles	Les deux
Complétées en	Expressions thunkables	Contextes d'évaluation linéaires	Les deux

Table 4: Comparaison des structures sous-jacentes à divers modèles directs du calcul

Modèles à base d'adjonction

Le chapitre traite des structures algébriques sous-jacentes de ces modèles: une monade sur une catégorie de valeurs pour l'appel par valeur, une co-monade sur une catégorie de piles pour l'appel par nom. Les duploïdes généralisent cette structure sous-jacente à une adjonction entre une catégorie de valeurs et une catégorie de piles. (Voir le Tableau 4.)

La relation avec les polarités est donnée par la *traduction polarisée* de la logique classique de Girard [Gir91, DJS97, Lau02]. Notre construction de duploïde étend (le squelette de) la traduction polarisée à toute adjonction. (Essentiellement, on n'a pas besoin de supposer qu'il existe une opération de négation involutive sur les formules.)

Il est connu qu'il y a un intérêt pratique à décomposer les monades, lorsqu'elles sont vues comme des notions de calcul, en adjonctions, grâce à Levy [Lev99, Lev04, Lev05]. Les adjonctions de Levy englobent les modèles de l'appel par valeur et de l'appel par nom.

Cependant le modèle est indirect, et manque encore d'une notion correspondante de modèle direct.

La construction de duploïde

Avec la construction de duploïde, on définit, en partant d'une adjonction $F \dashv G : \mathscr{C}_1 \to \mathscr{C}_2$, des morphismes $A \to B$ pour A et B des objets de n'importe laquelle des catégories $\mathscr{C}_1$ and $\mathscr{C}_2$.

On définit tout d'abord les morphismes obliques $P \to N$, avec $P \in |\mathscr{C}_2|$ et $N \in |\mathscr{C}_1|$, de façon équivalente comme des morphismes $P \to GN$ ou $FP \to N$. Considérons maintenant $\bullet$ la composition dans $\mathscr{C}_1$ et $\circ$ la composition dans $\mathscr{C}_2$. On observe que les morphismes obliques peuvent être composés ou bien dans $\mathscr{C}_1$:

$$\dfrac{\dfrac{f : P \to FQ}{f : FP \to FQ}\,(\simeq) \qquad \dfrac{g : Q \to N}{g : FQ \to N}\,(\simeq)}{\dfrac{g \bullet f : FP \to N}{g \bullet f : P \to N}\,(\simeq)}\,(\bullet)$$

ou bien dans $\mathscr{C}_2$:

$$\dfrac{\dfrac{f : P \to N}{f : P \to GN}\,(\simeq) \qquad \dfrac{g : GN \to M}{g : GN \to GM}\,(\simeq)}{\dfrac{g \circ f : P \to GM}{g \circ f : P \to M}\,(\simeq)}\,(\circ)$$

On définit donc un morphisme $A \to B$ comme un morphisme oblique:

$$\boxed{A^+ \to B^\ominus}\,,$$

où:

$$P^+ \stackrel{\text{def}}{=} P \qquad\qquad P^\ominus \stackrel{\text{def}}{=} FP$$
$$N^+ \stackrel{\text{def}}{=} GN \qquad\qquad N^\ominus \stackrel{\text{def}}{=} N$$

La raison pour laquelle cela ne définit pas forcément une catégorie

est qu'on peut avoir:

$$f \circ (g \bullet h) \neq (f \circ g) \bullet h$$

lorsque l'adjonction n'est pas idempotente.

Contribution

Dans Chapter II, on démontre qu'il existe une réflexion:

$$\mathcal{Dupl} \triangleleft \mathcal{Adj}$$

où $\mathcal{Dupl}$ est une catégorie des duploïdes et des *foncteurs de duploïdes*, et où $\mathcal{Adj}$ est la catégorie des adjonctions et des preudomorphismes d'adjonctions. En d'autres termes, la construction de duploïde ci-dessus se prolonge en un foncteur $\mathcal{Adj} \to \mathcal{Dupl}$ qui admet un adjoint à droite plein et fidèle. En particulier, tout duploïde est obtenu à partir d'une adjonction.

En conséquence, les duploïdes rendent compte de beaucoup de modèles de calcul, comme on le verra dans divers exemples de sémantique dénotationnelle. Cela suggère que les différents biais de la sémantique dénotationnelle: indirect, appel par valeur, appel par nom… sont des façons de cacher le fait que la composition n'est pas toujours associative.

Par ailleurs, en raison de la réflexion, $\mathcal{Dupl}$ est équivalent à une sous-catégorie pleine de $\mathcal{Adj}$, ce qui signifie que l'adjonction associée à un duploïde possède des propriétés additionnelles. On caractérise lesquelles. On montre qu'il existe une équivalence de catégories:

$$\mathcal{Dupl} \simeq \mathcal{Adj}_{eq}$$

où $\mathcal{Adj}_{eq}$ est la sous-catégorie pleine des adjonctions satisfaisant la *condition d'égalisation*: l'unité et la co-unité de l'adjonction sont respectivement des égalisateurs et des co-égalisateurs. Cette condition signifie que:

- la catégorie des valeurs est complétée avec toutes les expressions thunkables;

$$
\begin{array}{lcl}
x^{+} & \stackrel{\text{def}}{=} & \lambda k.(k\,x) \\
(\lambda x.t)^{+} & \stackrel{\text{def}}{=} & \lambda k.(k\,\lambda x.t^{+}) \\
(t\,u)^{+} & \stackrel{\text{def}}{=} & \lambda k.(t^{+}\,\lambda x.(u^{+}\,\lambda y.(x\,y\,k)))
\end{array}
$$

(a) Appel par valeur (Plotkin [Plo75])

$$
\begin{array}{lcl}
x^{\ddagger} & \stackrel{\text{def}}{=} & x \\
(\lambda x.t)^{\ddagger} & \stackrel{\text{def}}{=} & \lambda y.(t^{\ddagger}[\pi_1(y)/x]\,\pi_2(y)) \\
(t\,u)^{\ddagger} & \stackrel{\text{def}}{=} & \lambda k.(t^{\ddagger}\,(u^{\ddagger}, k))
\end{array}
$$

(b) Appel par nom (Lafont, Reus et Streicher [LRS93])

Figure 5: Traductions par passage de continuation dans le λ-calcul

- la catégorie des piles est complétée avec tous les contextes d'évaluation linéaires;
- toute paire de valeurs et toute paire de piles qui ne sont pas distinguables dans le modèle de calcul sont identifiées.

En d'autres termes, le duploïde exprime le point de vue interne du modèle de calcul défini par l'adjonction.

Le chapitre introduit le calcul $\mathbf{L}_{\text{dup}}$ qui est un langage interne pour les duploïdes. On explique avec lui le contenu calculatoire de ces derniers.

Une décomposition des traductions CPS délimitées

La polarisation et le calcul **L** permettent d'expliquer la structure des continuations de première classe et des traductions par passage de continuation (*continuation-passing style* ou CPS) à travers une décomposition en trois étapes. C'est l'objet du Chapitre III.

Traductions CPS et polarités

Grâce à l'étude de la théorie de la démonstration classique [Gir91, DJS95, DJS97, Lau02], nous savons que la traduction CPS en appel par valeur standard (voir Figure 5) correspond à une polarisation positive des formules tandis que la traduction en appel par nom de Lafont, Reus et Streicher correspond à une polarisation négative.

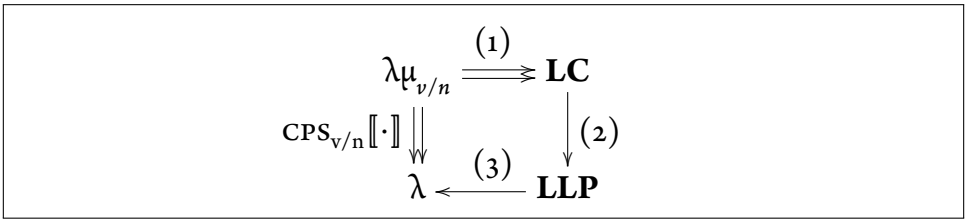


Figure 6: La décomposition polarisée des traductions CPS en trois étapes

(On considère les variantes qui prennent en compte l'opérateur $\mathcal{C}$.) La symétrie entre les polarités se traduit en une dualité catégorielle décrite par Selinger [Sel01] et Curien et Herbelin [CH00].

Le chapitre est pour commencer une description de comment, grâce aux outils fournis par la notion de polarité, par la focalisation et par le calcul des séquents, on peut décomposer en trois étapes les traductions CPS en appel par nom et en appel par valeur (Figure 6). Ce faisant, on reformule des traductions existantes et bien connues pour le λμ calcul [Lau02].

L'ÉTAPE (1) encode le λμ-calcul en appel par valeur ou en appel par nom avec des dérivations entièrement positives ou négatives dans les logiques classiques polarisées **LC** [Gir91] ou $\mathbf{LK}^{\eta}_{p}$ [DJS97]; on choisit ainsi une polarisation des formules.

L'ÉTAPE (2) est la traduction de Laurent de **LC** vers la « logique linéaire polarisée » **LLP** [Lau02]. Cette traduction est un cas particulier de la construction de duploïde du Chapitre II, et par conséquent l'étape (2) est celle où on perd le style direct. **LLP** se comprend comme le fragment $\wedge, \vee, \neg$ de **LJ**, rendu symétrique par l'introduction d'une opération de négation involutive sur les formules. En conséquence, la négation:

$$\frac{\Gamma, P \vdash}{\Gamma \vdash \neg P}\,(\vdash \neg) \qquad\qquad \frac{\Gamma \vdash P}{\Gamma, \neg P \vdash}\,(\neg \vdash)$$

est dédoublée en deux modalités covariantes duales notées ! et ?:

$$\frac{\vdash N, \mathcal{N}}{\vdash !N, \mathcal{N}}\ (\text{prom}) \qquad\qquad \frac{\vdash \mathcal{N}, P}{\vdash \mathcal{N}, ?P}\ (\text{der})$$

où $\mathcal{N}$ ne contient que des formules négatives. Ces modalités reflètent les changements de polarité; pour cette raison elles distinguent la formule pour la flèche en appel par valeur $(!(P \multimap ?Q))$ de celle en appel par nom $(!N \multimap M)$.

L'étape (3) traduit la logique précédente dans le λ-calcul avec uniquement des appels terminaux, c'est-à-dire dans un calcul de continuations. Elle procède comme suit:

- Les séquents sont formatés de sorte à n'avoir qu'une seule conclusion, par exemple $\vdash \mathcal{N}, P$ est transformé en $\mathcal{N}^{\perp} \vdash P$. C'est là que les polarités disparaissent.
- Le calcul de séquents est converti en déduction naturelle: en termes de calcul **L**, un contexte $\Gamma \mid E : A \vdash \star : B$ est transformé en le λ-terme $\Gamma, k : A \vdash E^* : B$ où k est linéaire dans E^*. C'est là que les réductions administratives doivent être traitées, comme on le verra.

En d'autres termes, cette étape cache ce qu'il y a d'intéressant dans la structure pour rentrer dans le moule du λ-calcul.

Remarquons que les étapes (2) et (3) sont les mêmes indépendamment du calcul de départ.

Contribution

Notre contribution est en deux parties: on souligne les différences entre les étapes (1) et (2) grâce à un calcul **L** qui prend la place dans l'angle supérieur droit de la Figure 6, et on étend la décomposition en trois étapes aux opérateurs de contrôle délimité *shift* et *reset* de Danvy et Filinski [DF90]. Notre approche permet la reconstruction rationnelle en style direct de quatre variantes en appel par nom de *shift* et *reset*, l'une étant nouvelle.

Calcul	Réf.	Stratégie	Style	Réd. & Éq.	
$\mathbf{L}_{\mathrm{pol},\widehat{\mathfrak{tp}}^+}$	Fig. III.2	Polarisée	Calc. seq.	$\triangleright_{\mathrm{R}_p}, \simeq_{\mathrm{RE}_p}$	
$\lambda_{\mathrm{v}}^{\times}$	Fig. III.3	Indirecte	Déd. nat.	$\triangleright_{\mathrm{R}_\lambda}, \simeq_{\mathrm{RE}_\lambda}$	
$\lambda\mu\widehat{\mathfrak{tp}}_v$	Fig. III.5	Par valeur	Déd. nat.	$\triangleright_{\mathrm{R}_v}, \simeq_{\mathrm{RE}_v}$	[AHS04]
$\lambda\mu\widehat{\mathfrak{tp}}_n$	Fig. III.7	Par nom	Déd. nat.	$\triangleright_{\mathrm{R}_n}, \simeq_{\mathrm{RE}_n}$	[HG08]
$\mathbf{L}_{\mathrm{exp}}$	Fig. III.11	Indirecte	Calc. seq.	$\triangleright_{\mathrm{R}_e}, \simeq_{\mathrm{RE}_e}$	

Table 7: Récapitulatif des calculs considérés dans le Chapitre III

Sur l'interprétation constructive d'une négation involutive

Dans le Chapitre IV, on retourne à la source de la polarisation en théorie de la démonstration, que l'on souhaite expliquer: la description par Girard [Gir91] d'une négation involutive en logique classique. Cependant on ne suppose pas le lecteur familier avec les travaux de Girard et on adopte une approche différente. Notre but est d'étudier l'interprétation constructive d'une négation involutive, ou de façon équivalente du raisonnement par contraposée, en donnant à ces mots un sens spécifique.

Hypothèse 1. *Par* constructif*, on entend une interprétation qui suit le principe de la* formule comme type*, dans laquelle les formules sont traduites en les types d'un langage de programmation.*

Cette notion de construction n'est pas basée sur une limitation a priori des techniques de démonstration, telles que le rejet du principe du tiers exclu $\forall A\,(A \vee \neg A)$. Ainsi, Griffin a montré que la variante C de Felleisen [FFKD87] de l'opérateur de contrôle *call/cc* de Scheme pouvait être typée par l'élimination de la double négation [Gri90], et par conséquent peut être utilisée pour dériver le tiers exclu. Mais la démonstration ne fournit pas de procédure de décision pour A quelconque: le comportement dépend en général du contexte dans lequel le principe est invoqué. En d'autres termes, cela ne doit pas être vu comme une contradiction de l'intuitionnisme:

celui-ci suppose que le comportement des preuves est référentiellement transparent, hypothèse que nous ne faisons pas.

Cette notion de construction ne doit pas non plus être vue comme une restriction à des calculs typés a la Church. De la même façon que les systèmes de types approximent la notion de programme correct, les systèmes logiques sont vus comme une approximation de la notion de construction. Il s'agit du point de vue de la réalisabilité de Krivine [Kri09], par exemple. C'est pourquoi cela ne contredit pas notre intérêt premier pour les calculs non typés.

Dans ce contexte, le caractère constructif s'observe en particulier pour des formules héréditairement positives (Girard [Gir91]). En arithmétique, la notion recouvre les formules Σ_1^0. Les formules purement positives doivent satisfaire le même critère que la logique intuitionniste: la propriété de la disjonction, de l'existence… En conséquence, les démonstrations d'une formule $\forall \vec{x}(P_0(\vec{x}) \to Q_0(\vec{x}))$ où P_0, Q_0 sont purement positives (c'est à dire des formules Π_2^0 en arithmétique) correspondent à des algorithmes (Murthy [Mur91]).

Une telle notion de construction peut se comprendre dans une comparaison avec les langages de programmation: en présence d'effets de bord (opérateurs de contrôle, états, entrées-sorties…), certains types sont opaques lors de l'exécution.

Hypothèse 2. *Par* involutif*, on entend une négation qui vérifie un isomorphismes de types entre* $\neg\neg A$ *et* A*.*

La raison pour demander davantage qu'une simple équivalence entre A et $\neg\neg A$ est qu'il y a trop de choix possibles pour la contraposée d'une formule comme la suivante:

$$\forall \mathrm{x}, \mathrm{y} \in A, (P(\mathrm{x}) \vee Q(\mathrm{y})) \to (\forall \mathrm{x} \in A, P(\mathrm{x})) \vee (\forall \mathrm{y} \in A, Q(\mathrm{y}))$$

par exemple:

$$\begin{array}{rcl}
\neg\big((\forall \mathrm{x} \in A, P(\mathrm{x})) \vee (\forall \mathrm{y} \in A, Q(\mathrm{y}))\big) & \to & \neg \forall \mathrm{x}, \mathrm{y} \in A, (P(\mathrm{x}) \vee Q(\mathrm{y})) \\
(\neg \forall \mathrm{x} \in A, P(\mathrm{x})) \vee (\neg \forall \mathrm{y} \in A, Q(\mathrm{y})) & \to & \exists \mathrm{x}, \mathrm{y} \in A, \neg(P(\mathrm{x}) \vee Q(\mathrm{y})) \\
(\exists \mathrm{x} \in A, \neg P(\mathrm{x})) \wedge (\exists \mathrm{y} \in A, \neg Q(\mathrm{y})) & \to & \exists \mathrm{x}, \mathrm{y} \in A, (\neg P(\mathrm{x}) \wedge \neg Q(\mathrm{y}))
\end{array}$$

L'embarras du choix disparaît lorsque les loi de De Morgan sont des isomorphismes de types: s'il y a trop de démonstrations, alors il faut être capable d'en choisir une canonique, qui préserve la signification.

En l'absence de tels isomorphismes, on suppose que la contraposée est obtenue en poussant les négations au plus proche des feuilles. Dans l'exemple ci-dessus il s'agit de la troisième proposition. Des travaux comme ceux de Krivine [Kri09] montrent d'un point de vue technique l'importance d'un tel raisonnement par contraposée; par exemple l'axiome du choix dénombrable n'est réalisé qu'à travers sa contraposée. Pourtant, raisonner par contraposée n'est pas trivial dans le contexte du λC-calcul employé par l'auteur, comme le montre l'exemple suivant:

Exemple 1. On se place dans le cadre du λC-calcul de la Figure 1, étendu au second ordre.[4] Cela nous permet de définir les quantificateurs de la façon standard: $\forall X \in A, B \stackrel{\text{def}}{=} \forall X(A[X] \to B)$ et $\exists X \in A, B \stackrel{\text{def}}{=} \forall Y(\forall X\,(A[X] \to B \to Y) \to Y)$, et similairement pour le premier ordre.

Considérons maintenant les formules suivantes, qui ont la même complexité que l'axiome du choix dénombrable et que de sa contraposée:

$$[C]: \qquad \forall \mathrm{x} \in \mathbb{N}, \exists \mathrm{y} \in E, A_1(\mathrm{x},\mathrm{y}) \to \exists Y \in F, \forall \mathrm{x} \in \mathbb{N}, A_2(\mathrm{x}, Y)$$
$$[\bar{C}]: \quad \forall Y \in F, \exists \mathrm{x} \in \mathbb{N}, \neg A_2(\mathrm{x}, Y) \to \exists \mathrm{x} \in \mathbb{N}, \forall \mathrm{y} \in E, \neg A_1(\mathrm{x},\mathrm{y})$$

On peut démontrer $[C] \leftrightarrow [\bar{C}]$, et en le faisant directement, on obtient le terme suivant de type $[C] \to [\bar{C}]$:

$$\lambda axy.\Big(C\lambda k.\Big(a\,\lambda e.\big(C\,\lambda z.\big(k\,(y\,e\,\lambda np.(z\,\lambda k.(k\,n\,p)))\big)\big)\,\lambda np.\big(x\,n\,\lambda ez.(z\,(p\,e))\big)\Big)\Big)$$

[4] Les formules sont étendues avec des quantifications sur les atomes $(\forall X\,N)$, et les deux règles suivantes sont ajoutées:

$$\frac{\Gamma \vdash t : N}{\Gamma \vdash t : \forall X\,N}\ (X \notin \mathbf{fv}(\Gamma)) \qquad \frac{\Gamma \vdash t : \forall X\,N}{\Gamma \vdash t : N[M/X\mathrm{x}_1 \ldots \mathrm{x}_n]}$$

Chacune des quatre lois de De Morgan reliant $\forall$ et $\exists$ sont utilisées dans la démonstration. Parmi elles, seul le principe suivant n'est pas intuitionniste, et est responsable des deux occurrences de C:

$$\neg\forall \mathrm{x} \in \mathbb{N}, P \to \exists \mathrm{y} \in \mathbb{N}, \neg P\,.$$

Le squelette de sa démonstration est le suivant:

$$\boxed{\vdash \lambda xy.(C\,\lambda k.(x\,\lambda e.(C\,\lambda l.(k\,(y\,e\,l)))))) : \neg\forall \mathrm{x} \in \mathbb{N}, P(\mathrm{x}) \to \exists \mathrm{x} \in \mathbb{N}, \neg P(\mathrm{x})}$$

Bien qu'il s'agisse d'une tautologie classique élémentaire, le rôle calculatoire du terme n'est pas immédiat, en particulier en raison de la présence des deux C.

Dans l'exemple ci-dessus, le caractère illisible est une idiosyncrasie du λC-calcul qui reflète l'absence, comme on le verra, d'une involution $\neg\neg N \simeq N$ et par conséquent d'un isomorphisme de types $\neg\forall \mathrm{x}\, N \simeq \exists \mathrm{x}\, \neg N$.

La négation involutive à travers le principe de la formule comme type

Girard interprète une négation involutive en lui faisant changer la polarité de la formule [Gir91]. On montre dans le chapitre comment une notion de négation involutive inspirée de Girard correspond à l'idée d'exposer une interface de haut niveau aux piles capturées, similairement à la suggestion de Felleisen [AH08, Note] et comme mis en œuvre par Clements [Cle06]. Dans notre cadre, cela se traduit par la présence d'un type positif $\sim A$ des *piles inspectables* et de constantes D qui permettent d'accéder au contenu de ces piles. Le type $\sim A$ est donc distinct du type négatif $A \to \bot$ des continuations. Cette distinction n'a de sens que dans un cadre où les deux polarités sont prises en compte.

La polarisation explique une technique de Krivine qui allège la complexité du raisonnement dans le λC-calcul en permettant à gauche des flèches certains pseudo-types de saveur positive [Kri09, Kri08]. La polarisation donne à ces pseudo-types un statut de première classe, car elle les définit également à droite des implications.

	Fig.	Style	Stratégie	$A \simeq \neg\neg A$	Réd. & Éq.	
λC	I.12	Déd. nat.	Par nom	Non	$\succ_n, \approx_n$	[HS02]
$\lambda\ell$	IV.1	Déd. nat.	Polarisée	Oui	$\succ_p, \approx_p$	
$\mathbf{L}_{\mathrm{n}}$	I.10	Seq. Calc.	Par nom	Non	$\triangleright_{\mathrm{R}_n}, \simeq_{\mathrm{RE}_n}$	[CH00]
$\mathbf{L}_{\mathrm{pol},\hat{\mathrm{tp}}^{\ominus}}$	IV.4	Seq. Calc.	Polarisée	Oui	$\triangleright_{\mathrm{R}_p}, \simeq_{\mathrm{RE}_p}$	

Table 8: Une comparaison des calculs apparaissant dans le Chapitre IV

Par ailleurs, le pseudo-type le plus important dans le travail de Krivine est $\mathcal{X}^{-}$, defini comme l'ensemble $\{\mathsf{k}_\pi \mid \pi \in \mathcal{X}\}$. Cela correspond à distinguer les piles capturées des continuations, comme on le fait de façon plus directe dans le chapitre.[5]

Dans le chapitre, les délimiteurs du contrôle fournissent une interprétation pour l'unité $\bot$, étendant ainsi l'interprétation en théorie de la démonstration du contrôle délimité par Herbelin et d'autres [HG08, AHS09, Her10, Ili10] qui généralise le principe de la A-traduction de Friedman [Fri78]. Ainsi, bien que le contrôle délimité ne prouve pas de nouvelle formule dans le cadre typé sans annotations par rapport au contrôle non-délimité, notre résultat montre qu'il donne de meilleures démonstrations sur le plan constructif.

Contributions

Dans le Chapitre IV, on introduit les calculs extensionnels et non-typés $\lambda\ell$ et $\mathbf{L}_{\mathrm{pol},\hat{\mathrm{tp}}^{\ominus}}$, dans lesquels la négation définie comme suit est involutive:

$$\neg P \stackrel{\text{def}}{=} P \to \bot$$
$$\neg N \stackrel{\text{def}}{=} {\sim}N$$

[5] Remarquons par ailleurs que le cadre de Krivine n'est pas constructif selon notre définition, car une étape externe est nécessaire pour convertir les réaliseurs en algorithmes (Krivine [Kri09], voir aussi Miquel [Miq09]). La polarisation enlève cette étape externe, puisque les termes qui calculent les témoins peuvent être identifiés avec des réaliseurs positifs.

Le calcul $\lambda\ell$ définit la notion de programme, ou terme quasi-preuve, et le calcul $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^{\ominus}}$ décompose $\lambda\ell$ pour permettre le raisonnement extensionnel. Les calculs contiennent le $\Lambda\mu$-calcul de De Groote-Saurin [dG94, Sau05] et (une variante) des opérateurs $\mathit{shift}_0/\mathit{reset}_0$ [DF90, Sha07, MB12].

Introduction

The polarisation hypothesis

The λ calculus is now the cornerstone of a celebrated correspondence between some of category theory, functional programming, and intuitionistic proof theory. At the turn of the 1990's, further links have emerged between:

- the categorical modelling of effects;
- the continuation-based modelling of evaluation order and control in programming languages;
- the proof theory of classical logic.

Since then, these three domains have provided an interactive view on computation, and we are waiting to discover the principles that extend the correspondence to this interactive point of view.

Polarities, which distinguish negative connectives ($\forall$, $\to \dots$) from positive ones ($\exists$, $\vee \dots$), are an empirical legacy of intuitionism. Recognising the importance of these polarities is already one fruit of the interactive turn, both for the theory of programming languages and for proof theory.

Polarisation corresponds to the hypothesis that polarities have to be taken into account formally:

- in terms of categorical models, polarisation amounts to relaxing the hypothesis that composition is *a priori* associative on certain polarities;

- in continuation-based models of programming languages, polarities determine whether a continuation is meant to be applied or meant to be passed;
- in the proof theory of classical logic, polarities determine cut-elimination.

Thus, polarities are responsible for evaluation order in programs, and constructiveness of proofs. Polarities appear behind many phenomena:

- the Blass problem in game semantics described by Abramsky [Abr03] and Melliès [Mel05];
- the presence of laziness in call by value described Hatcliff and Danvy [HD97] and Führmann [Füh99];
- the unsoundness of unrestricted polymorphism in call by value in the presence of effects (Harper and Lillibridge [HL91]);
- the arithmetical hierarchy in the study of Peano arithmetic;
- focalisation in proof search (Andreoli [And92]);
- orthogonality-based logical relations (see Girard [Gir87], Krivine [Kri09], Pitts [Pit05], and M.-M. [Mun09] in relationship with polarisation);

and probably more.

The thesis is a contribution to the understanding of the nature, role, and mechanisms of polarisation in the three domains. The cornerstone of our approach is an interactive term-based representation of proofs and programs which exposes the structure of polarities. By *interactive* we mean that the novelty compared to the λ calculus is the explicit role given to the evaluation context (for programs) or to the opponent (for proofs). It is based on binders μ and $\tilde{\mu}$ and on the correspondence between abstract machines and sequent calculi, introduced by Curien and Herbelin with the $\bar{\lambda}\mu\tilde{\mu}$ calculus [CH00]. In honour of Gentzen [Gen35] who introduced the sequent calculi **LJ** and **LK**, and following a suggestion of Herbelin,

we refer to calculi that are based on the binders μ and $\tilde{\mu}$ with the letter "**L**"[6].

The scope of the syntax is both the investigation of the structure of proofs, after Gentzen, and the modelling of higher-order programming languages, after Landin [Lan64, Lan65]. Its goal is to provide means of unification for recent trends in category theory, proof theory and the theory of programming languages: the modelling of effects after Moggi [Mog89], the quest for a relationship between categorical duality and continuations in computer science after Filinski [Fil89], and the interactive notion of construction after Girard [Gir87, Gir91, Gir01] and Krivine [Kri09].

In each of the following three contributions, each corresponding to a chapter, we found that **L** calculi provided the canonical representation for an interesting structure:

- In Chapter II, we positively characterise polarisation through a categorical structure where we relax the assumption that composition is associative. The characterisation proves, and details, how polarisation is to adjunctions what call by value in programming is to monads. An internal language is provided by the calculus $\mathbf{L}_{\mathrm{dup}}$.
- In Chapter III, we decompose continuation-passing-style translations for delimited control operators into three steps: 1) the implementation of the operations of the language as solutions of equations in an abstract direct-style calculus; 2) the translation from direct to indirect style; 3) a translation back into the λ calculus, which flattens the structure and makes continuation-passing style hard to understand. The calculi $\mathbf{L}_{\mathrm{pol},\hat{\wp}^+}$ and $\mathbf{L}_{\mathrm{exp}}$ are used as the intermediate steps in the decomposition.
- In Chapter IV, we describe a formulae-as-types correspondence for an involutive negation in classical logic by introducing the $\lambda\ell$ calculus in natural deduction. The involution is due to the ℓ control operator, which implements the idea that captured contexts,

[6] Not to be confused with Lindenmayer's L-systems.

unlike continuations, have accessors. Extensional reasoning is provided in the calculus $\mathbf{L}_{\mathrm{pol},\hat{\mathrm{tp}}^{\ominus}}$ and a corresponding sequent calculus.

Simplifying hypotheses would make polarities less salient. Our investigation let the interesting phenomena appear because we sought a description of computation through untyped (Curry-style), extensional and direct calculi. The *direct* approach to programs and proofs emphasises that, when a system is studied by means of a translation into a simpler system, understanding the translation is equally important as understanding its target. These three constraints save us from the pitfalls of the following hypotheses:

- Looking for a categorical model where a single category is involved — for instance, if the goal is to seek a categorical generalisation of boolean algebras, why should we keep the associativity of composition?
- Looking for an *a priori* strongly-normalising setting — for instance, many good properties of System F [Gir72], such as extensionality properties and therefore isomorphisms, follow from strong normalisation [GSS92]. But, in addition to tying us to a particular logic, normalisation *depolarises* the system, that is to say the order of evaluation no longer matters.
- Assuming that referential transparency is essential for constructiveness — for instance, this hypothesis does not hold for direct interpretations of classical logic with control operators. In the presence of such operators, the order of evaluation matters for constructiveness, even when the setting is strongly normalising.
- Assuming that we may *a priori* restrict to certain shapes of proofs — for instance, the study of classical predicate calculus is simplified if we restrict to proofs in η-long forms. But it is known that the trick is limited to the first order and moreover amounts to a Gödel-Gentzen translation (Laurent [Lau02]).

As a result of our investigations, we realised that such hypotheses are not innocuous.

An introductory chapter follows. We introduce **L** calculi, starting from the ground up and assuming from the reader only elementary knowledge of simply-typed λ calculus and rewriting. The contributions of the other three chapters, alluded to above, are summarized in more detail at the end of the chapter.

Notations and pre-requisites

The following terms are introduced with their specific meaning.

We assume familiarity with elementary notions from rewriting. If $\triangleright$ is a rewriting relation, then:

- $\triangleright^*$ denotes the reflexive and transitive closure of $\triangleright$;
- $\triangleright^+$ denotes the transitive closure of $\triangleright$;
- $\rightarrow$ denotes the *compatible* closure of $\triangleright$, that is to say the extension by application to a sub-term or a sub-command (when applicable);
- $\simeq$ denotes the compatible equivalence relation $(\leftarrow \cup \rightarrow)^*$.

If $\triangleright_{A_p}$ and $\triangleright_{B_p}$ are two relations on a calculus p, then $\triangleright_{AB_p}$ denotes $\triangleright_{A_p} \cup \triangleright_{B_p}$, etc.

Chapter II assumes familiarity with elementary notions from category theory (for instance Mac Lane [ML71], chapters I-VI). When $\mathscr{C}$ is a category, $|\mathscr{C}|$ denotes its set of objects. $1_{\mathscr{C}}$ denotes the identity functor on $\mathscr{C}$. The notation $\tau : F \dot{\rightarrow} G$ is for a natural transformation.

We use the notation $F \dashv_{(\eta,\varepsilon)} G : \mathscr{C} \to \mathscr{D}$ to denote an adjunction with $G : \mathscr{C} \to \mathscr{D}$, unit η and co-unit ε. Sometimes we use the notation $F \dashv_{(\sharp,\flat)} G : \mathscr{C} \to \mathscr{D}$ to denote an adjunction with the natural isomorphism:

$$\mathscr{C}(F-,=) \underset{\flat}{\overset{\sharp}{\rightleftarrows}} \mathscr{D}(-,G=)\,.$$

We assume that we are given a first-order language whose terms are written t, u and whose variables are written x, y.

Chapter I

The "L" approach to programs and proofs

In this introduction to **L** calculi, we start from a λ calculus with pairs and sums, for which we define an abstract machine leading to the intuitionistic calculus $\mathbf{L}_{\mathrm{i}}$. This allows us to introduce in a novel way the necessary concepts and techniques, which are not specific to **L** but which the **L** approach synthesises. Then we introduce polarities, and finally we recall the interpretation of classical logic with control operators. In the last section, we present the context for the remaining chapters as well as their respective contributions.

This exposition is adapted from the notes[1] of two courses given on October 24th 2011 and February 9th 2012 at the *Groupe de Travail Logique* organised by the graduate students of ENS Ulm. They are inspired by earlier published work (M.-M. [Mun09], Curien and M.-M. [CM10]).

I.1 NJ and the λ calculus

Gerhard Gentzen's system **NJ** is introduced in Figure I.1 on page 49. In Gentzen's naming scheme, **J** stands for *intuitionistic* and **N** stands for *natural deduction*.

An untyped λ calculus with pairs and sums, together with typing rules based on **NJ**, is introduced in Figure I.2 on page 50. It is based

[1] Available from my web page [Mun11, Mun12].

on the following grammar of expressions, or terms:

$$t, u, v ::= x \mid (t, u) \mid \iota_1(t) \mid \iota_2(t) \mid \lambda x.t \mid \\ \mathsf{fst}(t) \mid \mathsf{snd}(t) \mid \mathsf{match}\ t\ \mathsf{with}\ (u|v) \mid t\ u$$

The goal is for now to reduce terms according to the following naive call-by-name reduction relation:

$$\begin{array}{ll} (\lambda x.t)\ u & \succ\ t[u/x] \\ \mathsf{fst}(t, u) & \succ\ t \\ \mathsf{snd}(t, u) & \succ\ u \\ \mathsf{match}\ \iota_1(t)\ \mathsf{with}\ (u|v) & \succ\ u\ t \\ \mathsf{match}\ \iota_2(t)\ \mathsf{with}\ (u|v) & \succ\ v\ t \end{array}$$

Three questions that arise in this setting are:

1. How can we define the evaluation of a term of the following form?

$$\mathsf{match}\ (\lambda x.\iota_1(t))\ u\ \mathsf{with}\ (v|w)$$

2. How can we model the fact that the following term:

$$(\mathsf{match}\ x\ \mathsf{with}\ (\lambda y\lambda z.t|\lambda y\lambda z.u))\ z$$

   denotes the same proof or program as the following one?

$$\mathsf{match}\ x\ \mathsf{with}\ (\lambda y.t|\lambda y.u)$$

3. How can we model the fact that the following term:

$$\mathsf{match}\ t\ \mathsf{with}\ (\lambda y.u[\iota_1(y)/x]|\lambda z.u[\iota_2(z)/x])$$

   denotes the same proof or program as the following one?

$$u[t/x]$$

$$A, B ::= X(\mathrm{x}_1, \ldots, \mathrm{x}_n) \mid A \to B \mid A \wedge B \mid A \vee B \mid \forall \mathrm{x} A \mid \exists \mathrm{x} A \mid \bot$$

(a) Formulae

$$\frac{A \quad B}{A \wedge B} \qquad \frac{A \wedge B}{A} \quad \frac{A \wedge B}{B}$$

$$\frac{A}{A \vee B} \quad \frac{B}{A \vee B} \qquad \frac{A \vee B \quad A \to C \quad B \to C}{C}$$

$$\frac{A}{\forall \mathrm{x} A}\ {}^{*} \qquad \frac{\forall \mathrm{x} A(\mathrm{x})}{A(\mathrm{t})}$$

$$\frac{A(\mathrm{t})}{\exists \mathrm{x} A(\mathrm{x})} \qquad \frac{\exists \mathrm{y} A(\mathrm{y}) \quad A(\mathrm{x}) \to B}{B}\ {}^{*}$$

$$\frac{\begin{matrix}[A] \\ B\end{matrix}}{A \to B} \qquad \frac{A \to B \quad A}{B}$$

$$\frac{\bot}{A}$$

*: the variable x must not be free in the hypotheses.

(b) Rules

Figure I.1: Gentzen's **NJ** [Gen35, Gen69]

Almost 80 years ago, Gentzen embedded his "*Natural*" proof structures into "*Logistic*" ones to carry out the proof of consistency of arithmetic by means of reduction [Gen35]. We could say that the **L** calculi answer the above questions by extending Gentzen's insight to term calculi. (This would not, however, do justice to computer science and to its role in their discovery.)

$$t, u, v ::= x \mid (t, u) \mid \iota_1(t) \mid \iota_2(t) \mid \lambda x.t \mid$$
$$\mathsf{fst}(t) \mid \mathsf{snd}(t) \mid \mathsf{match}\ t\ \mathsf{with}\ (u|v) \mid t\ u$$

(a) Syntax

$$\frac{t : A \qquad u : B}{(t, u) : A \wedge B} \qquad \frac{t : A \wedge B}{\mathsf{fst}(t) : A} \qquad \frac{t : A \wedge B}{\mathsf{snd}(t) : B}$$

$$\frac{t : A}{\iota_1(t) : A \vee B} \qquad \frac{t : B}{\iota_2(t) : A \vee B} \qquad \frac{t : A \vee B \quad u : A \to C \quad v : B \to C}{\mathsf{match}\ t\ \mathsf{with}\ (u|v) : C}$$

$$\frac{t : A(\mathrm{x})}{t : \forall \mathrm{x}\, A(\mathrm{x})}\ {}^{*} \qquad \frac{t : \forall \mathrm{x}\, A(\mathrm{x})}{t : A(\mathrm{t})}$$

$$\frac{t : A(\mathrm{t})}{t : \exists \mathrm{x}\, A(\mathrm{x})} \qquad \frac{t : \exists \mathrm{y}\, A(\mathrm{y}) \qquad u : A(\mathrm{x}) \to B}{u\, t : B}\ {}^{*}$$

$$\frac{\begin{matrix}[x:A] \\ t : B\end{matrix}}{\lambda x.t : A \to B} \qquad \frac{t : A \to B \qquad u : A}{t\, u : B}$$

$$\frac{t : \bot}{t : A}$$

*: the variable x must not be free in the hypotheses.

(b) Rules

$$\begin{aligned}
(\lambda x.t)\, u &\succ t[u/x] \\
\mathsf{fst}(t, u) &\succ t \\
\mathsf{snd}(t, u) &\succ u \\
\mathsf{match}\ \iota_1(t)\ \mathsf{with}\ (u|v) &\succ u\, t \\
\mathsf{match}\ \iota_2(t)\ \mathsf{with}\ (u|v) &\succ v\, t
\end{aligned}$$

(c) Naive reduction relation

Figure I.2: λ calculus with pairs and sums

I.2 Abstract machines

In order to reduce "$\mathsf{match}\ (\lambda x.\iota_1(t))\ u\ \mathsf{with}\ (v|w)$", it is necessary to first reduce $(\lambda x.\iota_1(t))\,u$ — reduction must therefore happen in certain reduction contexts, here $\mathsf{match}\ \square\ \mathsf{with}\ (v|w)$.

For our purposes, an *abstract machine* is given by a grammar of expressions t, a grammar of *contexts* e, which are given a formal status[2], and rewriting rules on pairs $\langle\, t \parallel e \,\rangle$ (*commands*) that extend reduction to contexts.

Let us define an abstract machine for our λ calculus. First we define contexts as follows:

$$e ::= \star \mid \mathsf{fst}\cdot e \mid \mathsf{snd}\cdot e \mid (u|v)\cdot e \mid u\cdot e$$

In particular, $\star$ is a symbol that represents the empty context. Commands are defined with:

$$c ::= \langle t \parallel e \rangle$$

We now define the reduction relation $\rhd_{\mathrm{R}}$ on commands.

Main reductions Variables are substituted, pairs are projected, branches are chosen:

$$\begin{aligned}
\langle \lambda x.t \parallel u\cdot e\rangle &\rhd_{\mathrm{R}} \langle t[u/x] \parallel e\rangle\\
\langle (t,u) \parallel \mathsf{fst}\cdot e\rangle &\rhd_{\mathrm{R}} \langle t \parallel e\rangle\\
\langle (t,u) \parallel \mathsf{snd}\cdot e\rangle &\rhd_{\mathrm{R}} \langle u \parallel e\rangle\\
\langle \iota_1(t) \parallel (u|v)\cdot e\rangle &\rhd_{\mathrm{R}} \langle u \parallel t\cdot e\rangle\\
\langle \iota_2(t) \parallel (u|v)\cdot e\rangle &\rhd_{\mathrm{R}} \langle v \parallel t\cdot e\rangle
\end{aligned}$$

[2] Throughout the thesis we use the terminology *reduction context* or *evaluation context* to refer to terms t with a hole $\square$ (a distinguished variable that appears exactly once), and *context* alone to refer to the formal object e.

Adjoint reductions Projections, branches and function application build up the context.

$$\begin{aligned}
\langle \mathsf{fst}(u) \,\|\, e\rangle &\rhd_{\mathrm{R}} \langle u \,\|\, \mathsf{fst}\cdot e\rangle \\
\langle \mathsf{snd}(u) \,\|\, e\rangle &\rhd_{\mathrm{R}} \langle u \,\|\, \mathsf{snd}\cdot e\rangle \\
\langle \mathsf{match}\ t\ \mathsf{with}\ (u|v) \,\|\, e\rangle &\rhd_{\mathrm{R}} \langle t \,\|\, (u|v)\cdot e\rangle \\
\langle t\ u \,\|\, e\rangle &\rhd_{\mathrm{R}} \langle t \,\|\, u\cdot e\rangle
\end{aligned}$$

We called adjoint reductions as such because they are all of the form:

$$\text{“}\langle \tau^*(t) \,\|\, e\rangle \rhd_{\mathrm{R}} \langle t \,\|\, \tau(e)\rangle\text{”}\,.$$

In other words, adjoint reductions state that the destructive operations of the λ calculus are, by analogy with linear algebra, adjoint to the constructions on contexts.

Main reductions and adjoint reductions define $\rhd_{\mathrm{R}}$ as a deterministic relation: if $c \rhd_{\mathrm{R}} c'$ and $c \rhd_{\mathrm{R}} c''$ then $c' = c''$.

Example 1.1. The term "$\mathsf{match}\ (\lambda x.\iota_1(t))\ u\ \mathsf{with}\ (v|w)$" reduces as follows:

$$\begin{aligned}
&\langle \mathsf{match}\ (\lambda x.\iota_1(t))\ u\ \mathsf{with}\ (v|w) \,\|\, e\rangle \\
&\rhd_{\mathrm{R}} \langle (\lambda x.\iota_1(t))\ u \,\|\, (v|w)\cdot e\rangle \\
&\rhd_{\mathrm{R}} \langle \lambda x.\iota_1(t) \,\|\, u\cdot(v|w)\cdot e\rangle \\
&\rhd_{\mathrm{R}} \langle \iota_1(t[u/x]) \,\|\, (v|w)\cdot e\rangle \\
&\rhd_{\mathrm{R}} \langle v \,\|\, t[u/x]\cdot e\rangle
\end{aligned}$$

In particular, the original term is equivalent to $v\ t[u/x]$.

In an informal way, and despite the absence of ambiguity in the grammar, we find useful to write $|e\rangle$ to indicate that e is a context and $\langle t|$ to indicate that t is an expression.[3] (The convention is important

[3] We will see in Section IV.4.2 that the bracket notation is justified by an analogy with distributions and test functions in mathematics.

in Chapter III.)

Formal adjoints

In Figure 1.3 on the following page we summarise the machine, where we add one more ingredient. In Figure 1.3 we removed projection, branching and application, and we replaced them with a formal *adjoint operation*:

$$t, u ::= x \mid (t, u) \mid \iota_1(t) \mid \iota_2(t) \mid \lambda x.t \mid c^*.$$

The adjoint has the following reduction rule:

$$\boxed{\langle c^* \,\|\, e\rangle \rhd_{\mathrm{R}} c[e/\star]}\,.$$

With $c[e/\star]$ we replace with e any free $\star$ in c. Free occurrences of $\star$ are the ones that are not under $\cdot^*$, that is, we consider $\star$ to be bound in c^*.

Solving equations

The adjoint can now be used to *solve* the rules of adjoint reductions. For instance we consider the following rule:

$$\langle t\, u \,\|\, e\rangle \rhd_{\mathrm{R}} \langle t \,\|\, u{\cdot}e\rangle$$

as an (in)equation where the unknown is $t\, u$. Since we have:

$$\big\langle \langle t \,\|\, u{\cdot}\star\rangle^* \,\big\|\, e\big\rangle \rhd_{\mathrm{R}} \langle t \,\|\, u{\cdot}e\rangle\,,$$

we can define:

$$\boxed{t\, u \stackrel{\text{def}}{=} \langle t \,\|\, u{\cdot}\star\rangle^*}\,.$$

We may also define similarly:

$$\begin{aligned} \mathsf{fst}(t) &\stackrel{\text{def}}{=} \langle t \,\|\, \mathsf{fst}{\cdot}\star\rangle^* \\ \mathsf{snd}(t) &\stackrel{\text{def}}{=} \langle t \,\|\, \mathsf{snd}{\cdot}\star\rangle^* \\ \mathsf{match}\, t\, \mathsf{with}\, (u|v) &\stackrel{\text{def}}{=} \langle t \,\|\, (u|v){\cdot}\star\rangle^* \end{aligned}$$

$$t, u ::= x \mid (t, u) \mid \iota_1(t) \mid \iota_2(t) \mid \lambda x.t \mid c^*$$
$$e ::= \star \mid \mathsf{fst}\cdot e \mid \mathsf{snd}\cdot e \mid (u|v)\cdot e \mid u\cdot e$$
$$c ::= \langle t \parallel e \rangle$$

(a) Terms, contexts, and commands

$$\begin{array}{lll}
\langle \lambda x.t \parallel u\cdot e\rangle & \rhd_{\mathrm{R}} & \langle t[u/x] \parallel e\rangle \\
\langle (t,u) \parallel \mathsf{fst}\cdot e\rangle & \rhd_{\mathrm{R}} & \langle t \parallel e\rangle \\
\langle (t,u) \parallel \mathsf{snd}\cdot e\rangle & \rhd_{\mathrm{R}} & \langle u \parallel e\rangle \\
\langle \iota_1(t) \parallel (u|v)\cdot e\rangle & \rhd_{\mathrm{R}} & \langle u \parallel t\cdot e\rangle \\
\langle \iota_2(t) \parallel (u|v)\cdot e\rangle & \rhd_{\mathrm{R}} & \langle v \parallel t\cdot e\rangle \\
\langle c^* \parallel e\rangle & \rhd_{\mathrm{R}} & c[e/\star]
\end{array}$$

(b) Reduction rules

$$\begin{array}{rcl}
\mathsf{fst}(t) & \stackrel{\text{def}}{=} & \langle t \parallel \mathsf{fst}\cdot\star\rangle^* \\
\mathsf{snd}(t) & \stackrel{\text{def}}{=} & \langle t \parallel \mathsf{snd}\cdot\star\rangle^* \\
\text{match } t \text{ with } (u|v) & \stackrel{\text{def}}{=} & \langle t \parallel (u|v)\cdot\star\rangle^* \\
t\ u & \stackrel{\text{def}}{=} & \langle t \parallel u\cdot\star\rangle^*
\end{array}$$

(c) Definition of destructive operations

Figure I.3: An abstract machine for the λ calculus

To sum up, the basic operations have a symmetry: some construct the terms, others construct the contexts. This brings to mind the symmetry of Gentzen's sequent calculus, which we recall in the next section.

I.3 The sequent calculus LJ

We introduce Gentzen's intuitionistic sequent calculus **LJ** on Figure I.4 on page 56. **L** meant *logistic* for Gentzen, that is to say easy to

manipulate formally — as opposed to natural to reason in.

An intuitionistic sequent is of the form:

$$\Gamma \vdash B$$

where $\Gamma = A_1, \ldots, A_n$ is an unordered list of hypotheses.[4] The sequent corresponds to the proposition:

$$(A_1 \wedge \ldots \wedge A_n) \to B$$

Example 1.2. The elimination of implication is derived as follows:

$$\dfrac{\Gamma \vdash A \to B \qquad \dfrac{\Gamma \vdash A \qquad \overline{\Gamma, B \vdash B}}{\Gamma, A \to B \vdash B}}{\Gamma \vdash B}$$

1.4 Typing of machines

We introduce in Figure 1.5 on page 59 the type system for the abstract machines. The connection between abstract machines and the sequent calculus comes up when we try to match Gentzen's *cut* rule:

$$\dfrac{\Gamma \vdash A \qquad \Gamma, A \vdash B}{\Gamma \vdash B}$$

with the following typing rule for commands:

$$\boxed{\dfrac{\Gamma \vdash t : A \qquad \Gamma \mid e : A \vdash \star : B}{\langle t \parallel e \rangle : (\Gamma \vdash \star : B)}}$$

[4] We formulated Gentzen's system with implicit structural rules (otherwise known as the additive style) because in the technical part of the thesis, the use of variables will provide the proper control over the typing contexts. Without variables, explicit structural rules and reordering of the hypotheses are crucial in Gentzen's main result [Gen35].

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

$$\frac{\Gamma \vdash A \qquad \Gamma \vdash B}{\Gamma \vdash A \wedge B} \qquad \frac{\Gamma, A \vdash C}{\Gamma, A \wedge B \vdash C} \qquad \frac{\Gamma, B \vdash C}{\Gamma, A \wedge B \vdash C}$$

$$\frac{\Gamma \vdash A}{\Gamma \vdash A \vee B} \qquad \frac{\Gamma \vdash B}{\Gamma \vdash A \vee B} \qquad \frac{\Gamma, A \vdash C \qquad \Gamma, B \vdash C}{\Gamma, A \vee B \vdash C}$$

$$\frac{\Gamma \vdash A}{\Gamma \vdash \forall \mathrm{x}\, A}\ ^{*} \qquad \frac{\Gamma, A(\mathrm{t}) \vdash B}{\Gamma, \forall \mathrm{x}\, A(\mathrm{x}) \vdash B}$$

$$\frac{\Gamma \vdash A(\mathrm{t})}{\Gamma \vdash \exists \mathrm{x}\, A(\mathrm{x})} \qquad \frac{\Gamma, A \vdash B}{\Gamma, \exists \mathrm{x}\, A \vdash B}\ ^{*}$$

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \to B} \qquad \frac{\Gamma \vdash A \qquad \Gamma, B \vdash C}{\Gamma, A \to B \vdash C}$$

$$\frac{}{\Gamma, \bot \vdash A}$$

*: the variable x must not be free in Γ, B

Figure I.4: Gentzen's LJ [Gen35, Gen69]

where Γ becomes a function from a finite set of variables to formulae $x_1 : A_1, \ldots, x_n : A_n$.

There are now three judgements. The first one is similar to the one from natural deduction:

$$\Gamma \vdash t : A$$

It reads "*t is a proof of A under hypotheses* Γ" or "*t has type A under the typing context* Γ". The next one is:

$$\Gamma \mid e : A \vdash \star : B$$

It reads "*e is a refutation of A, or we have B*" or "*e expects a program of type A and returns B*". This corresponds to a sequent $\Gamma, A \vdash B$, with the vertical separation | between the context *e* and the hypotheses Γ

being added for more legibility. In a sequent, the formula that types t on the right or e on the left is called *main*.

The third judgement deals with commands:

$$\langle t \parallel e \rangle : (\Gamma \vdash \star : B)$$

In other words, $\langle\, t \parallel e \,\rangle$ does not have a type on its own, and the sequent has no main formula.

There are two novelties compared to **LJ**. First, the axiom rule:

$$\frac{}{\Gamma, A \vdash A}$$

is split in two:

$$\frac{}{\Gamma, x : A \vdash x : A} \qquad \frac{}{\Gamma \mid \star : A \vdash \star : A}$$

Second, if we want to type $\langle\, c^* \parallel e \,\rangle$, say with a cut rule involving a formula A, then according to the reduction rule $\langle\, c^* \parallel e \,\rangle \rhd_{\mathrm{R}} c[e/\star]$, the command c must be typable with conclusion $\star : A$. Thus we introduce a new rule:

$$\frac{c : (\Gamma \vdash \star : A)}{\Gamma \vdash c^* : A}$$

Notice that the underlying sequent calculus rule:

$$\frac{\Gamma \vdash A}{\Gamma \vdash A}$$

is superfluous in **LJ**.

Example 1.3. The defining equation for $t\ u$ can be retrieved through

the derivation of the elimination of implication from example 1.2:

$$\dfrac{\dfrac{\Gamma \vdash t : A \to B \qquad \dfrac{\Gamma \vdash u : A \qquad \overline{\Gamma \mid \star : B \vdash \star : B}}{\Gamma \mid u \cdot \star : A \to B \vdash \star : B}}{\langle t \parallel u \cdot \star \rangle : (\Gamma \vdash \star : B)}}{\Gamma \vdash \langle t \parallel u \cdot \star \rangle^* : B}$$

Issues with abstract machines

There are two issues that prevent us so far from having a correspondence between abstract machines and sequent calculus; and therefore we still have more to discover.

1. It is not possible to perform an introduction rule on an arbitrary formula on the left. Indeed, our left-introduction rules are so far restricted to proofs obtained starting from $\star$ and applying only left-introduction rules on the distinguished spot in the sequent.
2. The rules for the introduction of $\vee$ and $\exists$ on the left are incredibly remote from the rules of sequent calculus.

The symmetry of sequent calculus, and in particular the one between conjunction and disjunction, suggests the following rule for the introduction of $\vee$ on the left:

$$\dfrac{\Gamma \mid e : A \vdash \star : D \qquad \Gamma \mid e' : B \vdash \star : D}{\Gamma \mid (e|e') : A \vee B \vdash \star : D}$$

This suggests that we add a new constructor for contexts:

$$(e|e')$$

with new reductions that we guess by symmetry:

$$\langle \iota_1(t) \parallel (e_1|e_2) \rangle \rhd_{\mathrm{R}} \langle t \parallel e_1 \rangle$$
$$\langle \iota_2(u) \parallel (e_1|e_2) \rangle \rhd_{\mathrm{R}} \langle t \parallel e_2 \rangle$$

$$\Gamma \vdash t : A \qquad \Gamma \mid e : A \vdash \star : B \qquad \langle t \| e \rangle : (\Gamma \vdash \star : B)$$

(a) Judgements

$$\frac{}{\Gamma, x : A \vdash x : A} \qquad \frac{}{\Gamma \mid \star : A \vdash \star : A}$$

$$\frac{c : (\Gamma \vdash \star : A)}{\Gamma \vdash c^* : A} \qquad \frac{\Gamma \vdash t : A \qquad \Gamma \mid e : A \vdash \star : B}{\langle t \| e \rangle : (\Gamma \vdash \star : B)}$$

$$\frac{\Gamma \vdash t : A \qquad \Gamma \vdash u : B}{\Gamma \vdash (t, u) : A \wedge B} \qquad \frac{\Gamma \mid e : A \vdash C}{\Gamma \mid \mathsf{fst} \cdot e : A \wedge B \vdash C} \qquad \frac{\Gamma \mid e : B \vdash C}{\Gamma \mid \mathsf{snd} \cdot e : A \wedge B \vdash C}$$

$$\frac{\Gamma \vdash t : A}{\Gamma \vdash \iota_1(t) : A \vee B} \qquad \frac{\Gamma \vdash t : B}{\Gamma \vdash \iota_2(t) : A \vee B} \qquad \dagger$$

$$\frac{\Gamma \vdash t : A}{\Gamma \vdash t : \forall \mathrm{x}\, A}\ \S \qquad \frac{\Gamma \mid e : A(\mathrm{t}) \vdash B}{\Gamma \mid e : \forall \mathrm{x}\, A(\mathrm{x}) \vdash B}$$

$$\frac{\Gamma \vdash t : A(\mathrm{t})}{\Gamma \vdash t : \exists \mathrm{x}\, A(\mathrm{x})} \qquad \ddagger$$

$$\frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x.t : A \to B} \qquad \frac{\Gamma \vdash t : A \qquad \Gamma \mid e : B \vdash \star : C}{\Gamma \mid t \cdot e : A \to B \vdash \star : C}$$

$$\frac{}{\Gamma \mid \star : \bot \vdash \star : A}$$

(b) Rules

$$\dagger : \frac{\Gamma \mid e : C \vdash \star : D \qquad \Gamma \vdash t : A \to C \qquad \Gamma \vdash u : B \to C}{\Gamma \mid (t|u) \cdot e : A \vee B \vdash \star : D}$$

$$\ddagger : \frac{\Gamma \vdash t : \exists \mathrm{y}\, A(\mathrm{y}) \qquad \Gamma \mid e : B \vdash \star : C}{\Gamma \mid t \cdot e : A(\mathrm{x}) \to B \vdash \star : C}\ \S$$

(c) Wrong rules

§: the variable x must not be free in Γ, C

Figure I.5: Typing of machines in **LJ** (wrong)

However, we still miss an ingredient in order to simulate the operation match t with $(u|v)$ in a context e', in other words to simulate the corresponding context $(u|v)\cdot e'$. Indeed, the constructor $(e|e')$ above lets us model the selection of a branch, but now we need to find contexts e_1 and e_2 that push t onto the stack as match t with $(u|v)$ does. In other words we are looking for contexts e_1 and e_2 such that:

$$\langle t \,\|\, e_1 \rangle \rhd_{\mathrm{R}} \langle u \,\|\, t\cdot e' \rangle$$
$$\langle t \,\|\, e_2 \rangle \rhd_{\mathrm{R}} \langle v \,\|\, t\cdot e' \rangle$$

We need a context constructor that is symmetric to the adjoint operation $\cdot^*$.

I.5 Enter $\tilde{\mu}$

We are looking for two contexts e_1 and e_2 that associate to whatever term $\langle\, t \mid$ the commands $\langle\, u \parallel t\cdot e' \,\rangle$ and $\langle\, v \parallel t\cdot e' \,\rangle$ — in an informal notation:

$$|\, e_1 \rangle : x \mapsto \langle u \| x\cdot e' \rangle$$
$$|\, e_2 \rangle : x \mapsto \langle v \| x\cdot e' \rangle$$

We introduce the binder $\tilde{\mu}$ in order to define such contexts. We write $|\, \tilde{\mu}x.c \,\rangle$, with x bound in c by $\tilde{\mu}$, the context that associates to each term $\langle t|$ the command $c[t/x]$:

$$\boxed{\langle t \| \tilde{\mu}x.c \rangle \rhd_{\mathrm{R}} c[t/x]}\,.$$

This suggests that we take:

$$|\, e_1 \rangle = \big|\, \tilde{\mu}x.\langle u \| x\cdot e' \rangle \big\rangle$$
$$|\, e_2 \rangle = \big|\, \tilde{\mu}x.\langle v \| x\cdot e' \rangle \big\rangle$$

A generic motivation for introducing $\langle\, \tilde{\mu}x.c \,\rangle$ is that its presence implies the completeness of the grammar of contexts with respect to the reduction contexts that we can already express as terms with a hole. The contexts we are looking for can indeed already be expressed as terms with a hole:

$$\langle u\,\square \,\|\, e'\rangle^* \text{ and } \langle v\,\square \,\|\, e'\rangle^*.$$

We were only missing a notation to assert their presence in the grammar e. With the binder $\tilde{\mu}$, however, we can represent any term with a hole $E\,\square$ as:

$$\tilde{\mu}x.\langle E[x] \,\|\, \star\rangle$$

(Conversely, the context $\langle\, \tilde{\mu}x.\langle t \,\|\, e\rangle \,\rangle$ corresponds to the following reduction context:

$$(\lambda x.E[t])\,\square$$

if $E\,\square$ is a reduction context that corresponds to e. In particular, defining a term with a hole that corresponds to a context is more tedious since it requires an induction.)

I.5.1 Stacks vs. contexts

The problem is that the reduction rule for $\tilde{\mu}$:

$$\langle t \,\|\, \tilde{\mu}x.c\rangle \rhd_{\mathrm{R}} c[t/x]$$

conflicts with the one we gave so far for c^*:

$$\langle c^* \,\|\, e\rangle \rhd_{\mathrm{R}} c[e/\star]$$

Therefore, if we admit $\tilde{\mu}x.c$ in the grammar of e then the reduction is no longer deterministic. Consider for instance the following command:

$$c \stackrel{\text{def}}{=} \langle t\, u \,\|\, \tilde{\mu}x.\langle v \,\|\, \star\rangle\rangle$$

(it corresponds to the program $\mathsf{let}\, x = t\, u\, \mathsf{in}\, v$). If we want to stick with the call-by-name reductions that we considered so far, then we must allow the following reduction:

$$c \rhd_{\mathrm{R}} \langle v[t\, u/x] \,\|\, \star \rangle$$

and forbid the reduction into the following command:

$$\langle t \,\|\, u \cdot \tilde{\mu} x.\langle v \,\|\, \star \rangle \rangle$$

Indeed, the latter computes t before v, which is not what we want — for the moment. This shows that we must not treat $\tilde{\mu} x.c$ like any other context.

Let us denote a new grammar for contexts that excludes $\tilde{\mu} x.c$ that we call stacks, and which we write π[5] following Krivine [Kri09].

$$\pi ::= \star \mid \mathsf{fst} \cdot \pi \mid \mathsf{snd} \cdot \pi \mid (e|e) \mid u \cdot \pi$$

Contexts e are now either π or $\tilde{\mu} x.c$ (thus, a context is not necessarily a stack).

$$\boxed{e ::= \pi \mid \tilde{\mu} x.c}$$

Notice that we added contexts of the form $(e|e')$ and consider them as stacks.

With this distinction we can now restrict the reduction of c^* to the following rule:

$$\langle c^* \,\|\, \pi \rangle \rhd_{\mathrm{R}} c[\pi/\star]$$

We summarise our development so far with the calculus $\mathbf{L}_{\mathrm{i}}$ (Figure 1.7 on page 64; *i* stands for intuitionistic, or introductory).

[5] for "pile" (stack in French).

	Linear for	In sequent calculus
Context e	Rewriting	Main formula
Stack π	Denotational semantics	Stoup (see IV.6 and III.2.8)

Table I.6: Two notions of linearity

I.5.2 The two notions of linearity

We just uncovered with contexts and stacks a fundamental difference between two notions of linearity (Table I.6 on this page):

- A context e is *syntactically linear*, in the sense that it denotes a term with a hole $\Box$ where $\Box$ occurs exactly once. This notion belongs to rewriting theory and underlies the confluence of the compatible closure of $\triangleright_{\mathrm{R}}$, for instance.
- A stack π denotes a context that is linear according to (a syntactic approximation of) operational linearity. By *operational linearity* we mean that the opposing term is evaluated at most and at least once. This notion of linearity is visible notably in denotational semantics (Girard [Gir87]).

I.5.3 The solution

We are now trying to reproduce the following reductions:

$$\langle \mathsf{match}\ t\ \mathsf{with}\ (u|v) \,\|\, \pi \rangle \triangleright_{\mathrm{R}} \langle t \,\|\, (u|v)\cdot\pi \rangle$$
$$\langle \iota_1(t) \,\|\, (u|v)\cdot\pi \rangle \triangleright_{\mathrm{R}} \langle u \,\|\, t\cdot\pi \rangle$$
$$\langle \iota_2(t) \,\|\, (u|v)\cdot\pi \rangle \triangleright_{\mathrm{R}} \langle v \,\|\, t\cdot\pi \rangle$$

$$
\begin{aligned}
t, u &::= x \mid (t, u) \mid \iota_1(t) \mid \iota_2(t) \mid \lambda x.t \mid c^* \\
e &::= \pi \mid \tilde{\mu} x.c \\
\pi &::= \star \mid \mathsf{fst}\cdot\pi \mid \mathsf{snd}\cdot\pi \mid (e|e) \mid u\cdot\pi \\
c &::= \langle t \parallel e \rangle
\end{aligned}
$$

(a) Terms, contexts, stacks and machines

$$
\begin{aligned}
\langle \lambda x.t \parallel u\cdot\pi \rangle &\rhd_{\mathrm{R}} \langle t[u/x] \parallel \pi \rangle \\
\langle (t, u) \parallel \mathsf{fst}\cdot\pi \rangle &\rhd_{\mathrm{R}} \langle t \parallel \pi \rangle \\
\langle (t, u) \parallel \mathsf{snd}\cdot\pi \rangle &\rhd_{\mathrm{R}} \langle u \parallel \pi \rangle \\
\langle \iota_1(t) \parallel (e_1|e_2) \rangle &\rhd_{\mathrm{R}} \langle t \parallel e_1 \rangle \\
\langle \iota_2(t) \parallel (e_1|e_2) \rangle &\rhd_{\mathrm{R}} \langle t \parallel e_2 \rangle \\
\langle c^* \parallel \pi \rangle &\rhd_{\mathrm{R}} c[\pi/\star] \\
\langle t \parallel \tilde{\mu} x.c \rangle &\rhd_{\mathrm{R}} c[t/x]
\end{aligned}
$$

(b) Reduction rules

$$
\begin{aligned}
\mathsf{fst}(t) &\stackrel{\text{def}}{=} \langle t \parallel \mathsf{fst}\cdot\star \rangle^* \\
\mathsf{snd}(t) &\stackrel{\text{def}}{=} \langle t \parallel \mathsf{snd}\cdot\star \rangle^* \\
\text{match } t \text{ with } (u|v) &\stackrel{\text{def}}{=} \left\langle t \parallel (\tilde{\mu} x.\langle u \parallel x\cdot\star \rangle | \tilde{\mu} x.\langle v \parallel x\cdot\star \rangle) \right\rangle^* \\
t\ u &\stackrel{\text{def}}{=} \langle t \parallel u\cdot\star \rangle^*
\end{aligned}
$$

(c) Definition of destructive operations

Figure I.7: $\mathbf{L}_{\mathsf{i}}$: the calculus

First we define $(u|v)\cdot\pi$ as $(e_1|e_2)$ where e_1 and e_2 are solutions of the following equations:

$$\langle t \,\|\, e_1 \rangle \rhd_{\mathrm{R}} \langle u \,\|\, t\cdot\pi \rangle$$
$$\langle t \,\|\, e_2 \rangle \rhd_{\mathrm{R}} \langle v \,\|\, t\cdot\pi \rangle$$

In other words we take:

$$|e_1\rangle \stackrel{\text{def}}{=} \left|\tilde{\mu}x.\langle u \,\|\, x\cdot\pi\rangle\right\rangle$$
$$|e_2\rangle \stackrel{\text{def}}{=} \left|\tilde{\mu}x.\langle v \,\|\, x\cdot\pi\rangle\right\rangle$$

And finally we take:

$$\boxed{\langle \mathsf{match}\ t\ \mathsf{with}\ (u|v) \,|\, \stackrel{\text{def}}{=} \left\langle t \,\middle\|\, \left(\tilde{\mu}x.\langle u \,\|\, x\cdot\star\rangle|\tilde{\mu}x.\langle v \,\|\, x\cdot\star\rangle\right)\right\rangle^{*}}\ .$$

I.6 Benefits of μ̃

I.6.1 Disappearance of commutative cuts

One goal was to identify the following term:

$$(\mathsf{match}\ x\ \mathsf{with}\ (\lambda y\lambda z.t|\lambda y\lambda z.u))\ z$$

with the following one:

$$\mathsf{match}\ x\ \mathsf{with}\ (\lambda y.t|\lambda y.u)$$

With our new definition, we have:

$$\begin{aligned}
&\langle (\mathsf{match}\ x\ \mathsf{with}\ (\lambda y\lambda z.t|\lambda y\lambda z.u))\ z \,\|\, \pi\rangle \\
&\rhd_{\mathrm{R}} \langle \mathsf{match}\ x\ \mathsf{with}\ (\lambda y\lambda z.t|\lambda y\lambda z.u) \,\|\, z\cdot\pi\rangle \\
&\rhd_{\mathrm{R}} \left\langle x \,\middle\|\, \left(\tilde{\mu}y.\langle \lambda y\lambda z.t \,\|\, y\cdot z\cdot\pi\rangle|\tilde{\mu}y.\langle \lambda y\lambda z.u \,\|\, y\cdot z\cdot\pi\rangle\right)\right\rangle \\
&\to^{*}_{\mathrm{R}} \left\langle x \,\middle\|\, \left(\tilde{\mu}y.\langle t \,\|\, \pi\rangle|\tilde{\mu}y.\langle u \,\|\, \pi\rangle\right)\right\rangle
\end{aligned}$$

$$\begin{aligned}
&\langle \mathsf{match}\ x\ \mathsf{with}\ (\lambda y.t|\lambda y.u) \,\|\, \pi\rangle \\
&\rhd_{\mathrm{R}} \langle x \,\|\, (\tilde{\mu}y.\langle \lambda y.t \,\|\, y\cdot\pi\rangle | \tilde{\mu}y.\langle \lambda y.u \,\|\, y\cdot\pi\rangle)\rangle \\
&\to^{*}_{\mathrm{R}} \langle x \,\|\, (\tilde{\mu}y.\langle t \,\|\, \pi\rangle | \tilde{\mu}y.\langle u \,\|\, \pi\rangle)\rangle
\end{aligned}$$

In other words, the reduction in the calculus $\mathbf{L}_{\mathrm{i}}$ ensures:

$$\boxed{\langle \mathsf{match}\ x\ \mathsf{with}\ (\lambda y\lambda z.t|\lambda y\lambda z.u)\ z \,\|\, \pi\rangle \simeq_{\mathrm{R}} \langle \mathsf{match}\ x\ \mathsf{with}\ (\lambda y.t|\lambda y.u) \,\|\, \pi\rangle}$$

Notice that in order to establish the equivalence we only used reduction: we did not need to introduce extensionality equations.

Commutative cuts are sequences of elimination rules in natural deduction where a local reduction rule move one elimination rule to the leaves of the proof. These reductions essentially implement identifications of the above kind. For instance, the following derivation of **NJ**:

$$\dfrac{\dfrac{A \vee B \qquad \begin{array}{c}[A]\\ \vdots \\ C \to D\end{array} \qquad \begin{array}{c}[B]\\ \vdots \\ C \to D\end{array}}{C \to D} \qquad \begin{array}{c}\vdots \\ C\end{array}}{D}$$

corresponds to $(\mathsf{match}\ x\ \mathsf{with}\ (\lambda y\lambda z.t|\lambda y\lambda z.u))\ z$ and is reduced to the following one:

$$\dfrac{A \vee B \qquad \dfrac{\begin{array}{c}\vdots \\ C\end{array} \qquad \begin{array}{c}[A]\\ \vdots \\ C \to D\end{array}}{D} \qquad \dfrac{\begin{array}{c}\vdots \\ C\end{array} \qquad \begin{array}{c}[B]\\ \vdots \\ C \to D\end{array}}{D}}{D}$$

which corresponds to $\mathsf{match}\ x\ \mathsf{with}\ (\lambda y.t|\lambda y.u)$.

There are many variants, where $C \to D$ is replaced by other connectives. They are tedious to list, but absolutely necessary in the presence of positive connectives such as disjunction and conjunc-

tion. Without them, natural deduction does not have the subformula property [Gir06]. All these rules follow naturally from the reduction rules in the **L** calculi.

I.6.2 A correspondence with sequent calculus

We present in Figure 1.8 on page 69 the typing rules of the calculus $\mathbf{L}_{\mathrm{i}}$ inspired by **LJ**. We have now the correct rules for $\vee$ and $\exists$:

$$\frac{\Gamma \mid e : A \vdash \star : C \qquad \Gamma \mid e' : B \vdash \star : C}{\Gamma \mid (e|e') : A \vee B \vdash \star : C} \qquad \frac{\Gamma \mid e : A \vdash B}{\Gamma \mid e : \exists \mathrm{x}\, A(\mathrm{x}) \vdash B}\ \S$$

The new construct $\tilde{\mu}x$ is typed symmetrically to the adjoint c^*:

$$\frac{c : (\Gamma, x : A \vdash \star : B)}{\Gamma \mid \tilde{\mu}x.c : A \vdash \star : B}$$

This rule is called *activation*; it has an inverse operation that we call *deactivation*. It is derived using a variable and a cut:

$$\frac{\Gamma \mid e : A \vdash \star : B}{\langle x \| e \rangle : (\Gamma, x : A \vdash \star : B)}$$

Like for the adjoint and activation, deactivation corresponds to the identity on the underlying sequent:

$$\frac{\Gamma, A \vdash B}{\Gamma, A \vdash B}$$

With activation and deactivation, the rules on the left are no longer constrained. For instance, there is a derivation:

$$\frac{\Gamma \mid e : A \vdash C}{\Gamma \mid \mathsf{fst} \cdot e : A \wedge B \vdash C}$$

where we define:

$$\boxed{\mathsf{fst}\cdot e \overset{\text{def}}{=} \tilde{\mu}x.\langle\langle x \,\|\, \mathsf{fst}\cdot\star\rangle^* \,\|\, e\rangle}\ .$$

This means that there exist unconstrained evaluation contexts,

$$\mathsf{fst}\cdot e, \mathsf{snd}\cdot e, u\cdot e,$$

that are defined by the following reduction rules that we call ς[6]:

$$\begin{aligned}
\langle t \,\|\, \mathsf{fst}\cdot e\rangle &\rhd_{\mathrm{R}} \langle\langle t \,\|\, \mathsf{fst}\cdot\star\rangle^* \,\|\, e\rangle \\
\langle t \,\|\, \mathsf{snd}\cdot e\rangle &\rhd_{\mathrm{R}} \langle\langle t \,\|\, \mathsf{snd}\cdot\star\rangle^* \,\|\, e\rangle \\
\langle t \,\|\, u\cdot e\rangle &\rhd_{\mathrm{R}} \langle\langle t \,\|\, u\cdot\star\rangle^* \,\|\, e\rangle
\end{aligned}$$

The phenomenon by which certain introduction rules hide cuts is called *focalisation*.

As an off-hand remark, we we will see in Section IV.6 that the **L** calculi have better properties if we explicitly include the ς rules, rather than letting generalised contexts being defined "by hand" as above.

Gentzen eliminated cuts in sequent calculus in order to prove the consistency of arithmetic. We can establish the following correspondence:[7]

$$\text{eliminating cuts} \iff \text{reducing commands}$$

[6] We follow the terminology of Wadler [Wad03].

[7] With the convention that a command $\langle x \,\|\, e\rangle$ or $\langle t \,\|\, \alpha\rangle$ is not a cut, as suggested by Wadler [Wad03]. The cut-elimination procedure is different however from the one considered by Gentzen.

$$\frac{}{\Gamma, x : A \vdash x : A} \qquad \frac{}{\Gamma \mid \star : A \vdash \star : A}$$

$$\frac{c : (\Gamma \vdash \star : A)}{\Gamma \vdash c^* : A} \qquad \frac{c : (\Gamma, x : A \vdash \star : B)}{\Gamma \mid \tilde{\mu}x.c : A \vdash \star : B}$$

$$\frac{\Gamma \vdash t : A \qquad \Gamma \mid e : A \vdash \star : B}{\langle t \| e \rangle : (\Gamma \vdash \star : B)}$$

$$\frac{\Gamma \vdash t : A \qquad \Gamma \vdash u : B}{\Gamma \vdash (t, u) : A \wedge B} \qquad \frac{\Gamma \mid \pi : A \vdash C}{\Gamma \mid \mathsf{fst}\cdot\pi : A \wedge B \vdash C} \qquad \frac{\Gamma \mid \pi : B \vdash C}{\Gamma \mid \mathsf{snd}\cdot\pi : A \wedge B \vdash C}$$

$$\frac{\Gamma \vdash t : A}{\Gamma \vdash \iota_1(t) : A \vee B} \qquad \frac{\Gamma \vdash t : B}{\Gamma \vdash \iota_2(t) : A \vee B} \qquad \frac{\Gamma \mid e : A \vdash \star : C \qquad \Gamma \mid e' : B \vdash \star : C}{\Gamma \mid (e|e') : A \vee B \vdash \star : C}$$

$$\frac{\Gamma \vdash t : A}{\Gamma \vdash t : \forall \mathrm{x}\, A}\ \S \qquad \frac{\Gamma \mid \pi : A(\mathrm{t}) \vdash B}{\Gamma \mid \pi : \forall \mathrm{x}\, A(\mathrm{x}) \vdash B}$$

$$\frac{\Gamma \vdash t : A(\mathrm{t})}{\Gamma \vdash t : \exists \mathrm{x}\, A(\mathrm{x})} \qquad \frac{\Gamma \mid e : A \vdash B}{\Gamma \mid e : \exists \mathrm{x}\, A(\mathrm{x}) \vdash B}\ \S$$

$$\frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x.t : A \to B} \qquad \frac{\Gamma \vdash t : A \qquad \Gamma \mid \pi : B \vdash \star : C}{\Gamma \mid t\cdot\pi : A \to B \vdash \star : C}$$

$$\frac{}{\Gamma \mid \star : \bot \vdash \star : A}$$

§: the variable x must not be free in Γ, C

(a) Rules

Figure I.8: $\mathbf{L}_{\mathbf{i}}$: typing rules

I.6.3 Other improvements

Confluence

Since $\triangleright_{\mathrm{R}}$ is left-linear and has no critical pair, we have that $\to_{\mathrm{R}}$ is confluent (see e.g. Nipkow [Nip91]). We will use this argument for all the **L** calculi in the thesis.

Suitability for orthogonality-based logical relations

Thanks to the presence of explicit contexts and to the detailed decomposition of constructs, the definition of orthogonality-based logical relations is simplified (see M.-M. [Mun09], Brunel [Bru12]). Such logical relations can be used for proving properties like type soundness or (strong) normalisation, among others.

In this context, considering orthogonality with respect to a set of stacks (as in Krivine [Kri09]) is sometimes preferable over considering bi-orthogonality with respect to a set of values (as in Girard [Gir87], Pitts [Pit05]), and conversely. This is a trace of polarisation as we introduce next (see M.-M. [Mun09]).

I.7 Polarisation

We are left with one question that we rephrase. How can we identify the following term:

$$\mathsf{match}\ t\ \mathsf{with}\ (\lambda y.E[\iota_1(y)]|\lambda z.E[\iota_2(z)])$$

with the following one:

$$E[t]$$

where $E\,\Box$ is whatever term with a hole? In other words, we want to enable extensional reasoning.

Let us rephrase again the question by considering the context $e \stackrel{\text{def}}{=} \tilde{\mu}x.\langle E[x] \,\|\, \star\rangle$, which corresponds to $E\,\Box$. The goal is to identify:

$$\left\langle \mathsf{match}\ t\ \mathsf{with}\ (\lambda y.\langle \iota_1(y) \,\|\, e\rangle^* | \lambda z.\langle \iota_2(z) \,\|\, e\rangle^*) \,\middle\|\, \star \right\rangle$$

with:

$$\langle t \parallel e \rangle .$$

Now one has:

$$\begin{aligned}
&\left\langle \mathsf{match}\ t\ \mathsf{with}\ (\lambda y.\langle \iota_1(y) \parallel e\rangle^* | \lambda z.\langle \iota_2(z) \parallel e\rangle^*) \parallel \star \right\rangle \\
&\rhd_{\mathrm{R}} \left\langle t \left\| (\tilde{\mu} y.\underline{\left\langle \lambda y.\langle \iota_1(y) \parallel e\rangle^* \parallel y \cdot \star \right\rangle} | \tilde{\mu} z.\underline{\left\langle \lambda z.\langle \iota_2(z) \parallel e\rangle^* \parallel z \cdot \star \right\rangle}) \right\| \right\rangle \\
&\to^*_{\mathrm{R}} \left\langle t \left\| (\tilde{\mu} y.\underline{\left\langle \langle \iota_1(y) \parallel e\rangle^* \parallel \star \right\rangle} | \tilde{\mu} z.\underline{\left\langle \langle \iota_2(z) \parallel e\rangle^* \parallel \star \right\rangle}) \right\| \right\rangle \\
&\to^*_{\mathrm{R}} \left\langle t \left\| (\tilde{\mu} y.\langle \iota_1(y) \parallel e\rangle | \tilde{\mu} z.\langle \iota_2(z) \parallel e\rangle) \right\| \right\rangle
\end{aligned}$$

Therefore the identification follows from the one between e and:

$$(\tilde{\mu} y.\langle \iota_1(y) \parallel e\rangle | \tilde{\mu} z.\langle \iota_2(z) \parallel e\rangle)$$

The issue with this identification arises when we take $E\,\square \stackrel{\text{def}}{=} (\lambda x.y)\,\square$ and $t = \Omega$ where $\Omega \stackrel{\text{def}}{=} \lambda x.(x\,x)\,\lambda x.(x\,x)$. One has:

$$\langle \Omega \parallel e\rangle \rhd_{\mathrm{R}_p} \langle y \parallel \star\rangle \not\rhd_{\mathrm{R}_p}$$

but:

$$\begin{aligned}
&\left\langle \Omega \left\| (\tilde{\mu} y.\langle \iota_1(y) \parallel e\rangle | \tilde{\mu} z.\langle \iota_2(z) \parallel e\rangle) \right\| \right\rangle \\
&\rhd_{\mathrm{R}} \left\langle \lambda x.(x\,x) \left\| \lambda x.(x\,x) \cdot (\tilde{\mu} y.\langle \iota_1(y) \parallel e\rangle | \tilde{\mu} z.\langle \iota_2(z) \parallel e\rangle) \right\| \right\rangle \\
&\rhd_{\mathrm{R}} \left\langle \Omega \left\| (\tilde{\mu} y.\langle \iota_1(y) \parallel e\rangle | \tilde{\mu} z.\langle \iota_2(z) \parallel e\rangle) \right\| \right\rangle \rhd_{\mathrm{R}_p} \ldots
\end{aligned}$$

The problem comes from the fact that our calculus so far does not take into account the strict evaluation that $\mathsf{match}\ t\ \mathsf{with}\ \ldots$ imposes on t. We see two solutions:

- If we are interested in typed approximations of computation, one can consider a strongly normalising system. For instance, System F [Gir72] has very good properties.
- Or, we can decide that we shall treat differently certain expressions that require a strict evaluation, from ones that require a lazy evaluation. (A completely strict evaluation is not desirable, because then we lose other extensionality axioms such as

$\lambda x.(t\ x) \simeq t.)$

The latter solution corresponds to polarisation.

I.7.1 Polarities

In the **L** calculi, polarisation is reflected at the level of variables, which are split in two as follows:

$$x^+, x^\ominus$$

In turn, the syntactic categories for terms and contexts are split in two as well:

$$t_+, t_\ominus$$
$$e_+, e_\ominus$$

The empty context $\star$ is both positive and negative.

In particular, there are now a positive adjoint operation and a negative one:

$$c^*{}_+, c^*{}_\ominus$$

Then, commands c are either $\langle t_+ \parallel e_+ \rangle$ or $\langle t_\ominus \parallel e_\ominus \rangle$ (the polarity of a command does not appear from the outside). We now describe how reduction takes polarities into account.

Negative polarity A negative term is *called by name*, that is, its evaluation is delayed until it comes in head position, in the terminology of the λ calculus. The head position corresponds to being against a stack. The reduction of the adjoint is therefore restricted to contexts that are stacks:

$$\langle t_\ominus \parallel \tilde{\mu} x^\ominus.c \rangle \rhd_{\mathrm{R}} c[t_\ominus/x^\ominus]$$
$$\langle c^*{}_\ominus \parallel \pi_\ominus \rangle \rhd_{\mathrm{R}} c[\pi_\ominus/\star] \quad \text{if substitution is defined}$$

(The substitution and the reduction are defined if and only if no free occurrence of $\star$ is of the form $\langle t_+ \parallel \star \rangle$.)

Positive polarity The reduction of positive polarity is symmetric to the one of negative polarity. The reduction of $\tilde{\mu}$ is restricted to terms that are values, in other words a positive term is *called by value*. We need to introduce a category of values V_+, and then we take:

$$t_+ ::= V_+ \mid c^*{}_+$$

The adjoint reduction is performed immediately:

$$\begin{aligned} \langle c^*{}_+ \| e_+ \rangle &\rhd_{\mathrm{R}} c[e_+/\star] \quad \text{if substitution is defined} \\ \langle V_+ \| \tilde{\mu} x^+.c \rangle &\rhd_{\mathrm{R}} c[V_+/x^+]\,. \end{aligned}$$

(The substitution and the reduction are defined if and only if no free occurrence of $\star$ is of the form $\langle t_\ominus \| \star \rangle$.)

We can shorten the description of the polarised evaluation order with the convention that any negative term is a value, and any positive context is a stack.

$$\begin{aligned} V &::= V_+ \mid t_\ominus \\ \pi &::= \pi_\ominus \mid e_+ \end{aligned}$$

Then the reduction is entirely defined for both polarities with:

$$\boxed{\begin{aligned} \langle V \| \tilde{\mu} x.c \rangle &\rhd_{\mathrm{R}} c[V/x] \\ \langle c^* \| \pi \rangle &\rhd_{\mathrm{R}} c[\pi/\star] \quad \text{if substitution is defined} \end{aligned}}$$

We can also describe the polarised reduction in terms of the critical pair:

$$c[\tilde{\mu} x.c'/\star] \mathrel{\overset{?}{\lhd}}_{\mathrm{R}} \langle c^* \| \tilde{\mu} x.c' \rangle \mathrel{\overset{?}{\rhd}}_{\mathrm{R}} c'[c^*/x]$$

With the positive polarity, the left-hand reduction is chosen over the right-hand one, while the converse holds with the negative polarity.

Last, in this section we are concerned about extensional reasoning.

Thus, we add the following expansion rules with $\varepsilon \in \{+, \ominus\}$:

$$\boxed{\begin{array}{l} t_\varepsilon \rhd_{\mathrm{E}} \langle t_\varepsilon \parallel \star \rangle^*{}_\varepsilon \\ e_\varepsilon \rhd_{\mathrm{R}} \tilde{\mu} x^\varepsilon . \langle x^\varepsilon \parallel e_\varepsilon \rangle \end{array}}$$

The consequence is that in order to prove an equivalence of terms or contexts, it is sufficient to prove an equivalence of commands.

I.7.2 Extensional branching

We consider values defined as follows:

$$\begin{array}{rcl} V_+ & ::= & x^+ \mid \iota_1(V) \mid \iota_2(V) \\ V & ::= & t_\ominus \mid V_+ \end{array}$$

Positive contexts are defined as follows:

$$e_+ ::= \star \mid \tilde{\mu} x^+ . c \mid \tilde{\mu}(\iota_1(x).c | \iota_2(y).c')$$

Reduction for the branching operation is defined as follows:

$$\begin{array}{l} \langle \iota_1(V) \parallel \tilde{\mu}(\iota_1(x).c | \iota_2(y).c') \rangle \rhd_{\mathrm{R}} c[V/x] \\ \langle \iota_2(V) \parallel \tilde{\mu}(\iota_1(x).c | \iota_2(y).c') \rangle \rhd_{\mathrm{R}} c'[V/y] \end{array}$$

Let t_+, u, and v be terms. We define branching by solving the following equation:

$$\langle \mathsf{match}\ t_+ \ \mathsf{with}\ (\iota_1(x).u | \iota_2(y).v) \parallel e \rangle \rhd_{\mathrm{R}} \left\langle t_+ \middle\| \tilde{\mu}(\iota_1(x).\langle u \parallel e \rangle | \iota_2(y).\langle v \parallel e \rangle) \right\rangle$$

There are two solutions, in other words we have to consider two distinct terms for match … with.

- If u and v are both positive, we define:

$$\mathsf{match}\ t_+ \ \mathsf{with}\ (\iota_1(x).u | \iota_2(y).v)_+ \stackrel{\text{def}}{=} \left\langle t_+ \middle\| \tilde{\mu}(\iota_1(x).\langle u \parallel \star \rangle | \iota_2(y).\langle v \parallel \star \rangle) \right\rangle^*{}_+$$

- If u and v are both negative, we define:

$$\mathsf{match}\ t_+\ \mathsf{with}\ (\iota_1(x).u|\iota_2(y).v)_\ominus \stackrel{\text{def}}{=} \left\langle t_+ \,\middle\|\, \tilde{\mu}(\iota_1(x).\langle u \,\|\, \star\rangle \middle| \iota_2(y).\langle v \,\|\, \star\rangle)\right\rangle^*_\ominus$$

Now let us consider for all e_+ the following expansion:

$$e_+ \rhd_{\mathrm{E}} \tilde{\mu}(\iota_1(x).\langle \iota_1(x) \,\|\, e_+\rangle | \iota_2(y).\langle \iota_2(x) \,\|\, e_+\rangle)$$

(In the thesis, we will use the notation $\rhd_{\mathrm{R}}$ for reductions and $\rhd_{\mathrm{E}}$ for expansions.)

Let $E\,\square$ be a term of any polarity with a positive hole, and let V_+ be a positive value. Obviating polarities than can be omitted, we have:

$$\begin{aligned}
&\langle \mathsf{match}\ V_+\ \mathsf{with}\ (\iota_1(y).E[\iota_1(y)]|\iota_2(z).E[\iota_2(z)]) \,\|\, \star\rangle \\
&\rhd_{\mathrm{R}} \left\langle V_+ \,\middle\|\, \tilde{\mu}\Big(\iota_1(y).\langle E[\iota_1(y)] \,\|\, \star\rangle \Big| \iota_2(z).\underline{\langle E[\iota_2(z)] \,\|\, \star\rangle}\Big)\right\rangle \\
&\leftarrow^*_{\mathrm{R}} \left\langle V_+ \,\middle\|\, \underline{\tilde{\mu}(\iota_1(y).\langle \iota_1(y) \,\|\, \tilde{\mu}x^+.\langle E[x^+] \,\|\, \star\rangle\rangle | \iota_2(z).\langle \iota_1(z) \,\|\, \tilde{\mu}x^+.\langle E[x^+] \,\|\, \star\rangle\rangle)}\right\rangle \\
&\leftarrow_{\mathrm{E}} \left\langle V_+ \,\middle\|\, \tilde{\mu}x^+.\langle E[x^+] \,\|\, \star\rangle\right\rangle \\
&\rhd_{\mathrm{R}} \langle E[V_+] \,\|\, \star\rangle
\end{aligned}$$

Thus, by applying $\mu\star$ on the above equations, we obtain by extensionality:

$$\boxed{\mathsf{match}\ V_+\ \mathsf{with}\ (\iota_1(y).E[\iota_1(y)]|\iota_2(z).E[\iota_2(z)]) \simeq_{\mathrm{RE}} E[V_+]}\,,$$

In other words, we have a standard extensionality axiom which, though it is restricted to values, it imposes no restriction on $E\,\square$.

I.7.3 Composition is not associative

Two terms compose using a let … in … binder. We obtain the binding let x be t in u by solving the following equation:

$$\langle \mathsf{let}\, x\, \mathsf{be}\, t\, \mathsf{in}\, u \| \pi \rangle \rhd_{\mathrm{R}} \langle t \| \tilde{\mu} x.\langle u \| \pi \rangle \rangle$$

There are two possibilities:

- u is positive. Then let x be t in u is defined as the positive term:

$$\mathsf{let}\, x\, \mathsf{be}\, t\, \mathsf{in}\, u_+ \stackrel{\text{def}}{=} \langle t \| \tilde{\mu} x.\langle u \| \star^+ \rangle \rangle^*_+$$

- u is negative. Then let x be t in u is defined as the negative term:

$$\mathsf{let}\, x\, \mathsf{be}\, t\, \mathsf{in}\, u_\ominus \stackrel{\text{def}}{=} \langle t \| \tilde{\mu} x.\langle u \| \star^\ominus \rangle \rangle^*_\ominus$$

In particular, the polarity of let x be t in u is always the same as the polarity of u.

In terms of categorical models, polarisation amounts to relaxing the hypothesis that composition is associative. Indeed, with polarisation there is a difference between the following two ways of composing t_+, $u_\ominus$ and v:

$$T_1 \stackrel{\text{def}}{=} \mathsf{let}\, y^\ominus\, \mathsf{be}\, (\mathsf{let}\, x^+\, \mathsf{be}\, t_+\, \mathsf{in}\, u_\ominus)\, \mathsf{in}\, v$$
$$T_2 \stackrel{\text{def}}{=} \mathsf{let}\, x^+\, \mathsf{be}\, t_+\, \mathsf{in}\, \mathsf{let}\, y^\ominus\, \mathsf{be}\, u_\ominus\, \mathsf{in}\, v$$

One has indeed:

$$\langle T_1 \| \pi \rangle \rhd^*_{\mathrm{R}} \left\langle \langle t_+ \| \tilde{\mu} x^+.\langle u_\ominus \| \star^\ominus \rangle \rangle^*_\ominus \| \tilde{\mu} y^\ominus.\langle v \| \star \rangle \right\rangle$$
$$\langle T_2 \| \pi \rangle \rhd^*_{\mathrm{R}} \left\langle t_+ \| \tilde{\mu} x^+.\langle u_\ominus \| \tilde{\mu} y^\ominus.\langle v \| \star \rangle \rangle \right\rangle$$

If the evaluation of t_+ loops in all contexts, and if $y^\ominus$ does not appear in v, then the top command reduces into $\langle v \| \star \rangle$ while the bottom one loops. Therefore, in general T_1 is different from T_2.

I.8 The sequent calculus LK

The setting we have just seen is intuitionistic. Yet, everything is in place to model proofs of classical logic. By *classical logic*, we mean a system with enough symmetry to reason by contrapositive and where hypotheses can be used arbitrarily many times.

We recall the $\forall, \to, \bot$ fragment of Gentzen's **LK** [Gen35] in Figure I.9. (**K**, as opposed to **J**, means *classical* in Gentzen's naming scheme.)

In **LK**, negation is defined with $\neg N \stackrel{\text{def}}{=} N \to \bot$. Classical means here that any formula N is equivalent to $\neg\neg N$, using the following derivation of the elimination of double negation:

$$\dfrac{\dfrac{\dfrac{}{\Gamma, A \vdash A, \Delta}}{\Gamma \vdash \neg A, A, \Delta}\,(\vdash\neg)}{\Gamma, \neg\neg A \vdash A, \Delta}\,(\neg\vdash)$$

The derivation crucially uses the fact that sequents can have multiple conclusions. In this context, a sequent:

$$A_1, \dots, A_n \vdash B_1, \dots, B_m$$

corresponds to the following proposition:

$$A_1 \wedge \dots \wedge A_n \to B_1 \vee \dots \vee B_m\,.$$

I.8.1 The calculus $\mathbf{L_n}$

Curien and Herbelin's calculus $\bar{\lambda}\mu\tilde{\mu}_T$ (renamed $\mathbf{L_n}$ for internal consistency reasons) is introduced in Figure I.10 on page 79. The letter n evokes the fact that it corresponds to a call-by-name interpretation of the $\forall, \to, \bot$ fragment of **LK**.

Classical logic is obtained in $\mathbf{L_n}$ by adding *co-variables*, which we write $\alpha, \beta \dots$ While variables denote inputs, co-variables denote named outputs. Co-variables replace $\star$ in the syntax of contexts.

$$N, M \ ::= X(\mathrm{t}_1, \ldots, \mathrm{t}_n) \mid N \to M \mid \forall \mathrm{x} N \mid \bot$$

(a) Formulae

$$\frac{}{\Gamma, N \vdash N, \Delta}\ (\mathrm{ax}) \qquad \frac{\Gamma \vdash N, \Delta \qquad \Gamma, N \vdash \Delta}{\Gamma \vdash \Delta}\ (\mathrm{cut})$$

(b) Identity and Structure

$$\frac{\Gamma, N \vdash M, \Delta}{\Gamma \vdash N \to M, \Delta}\ (\vdash \to) \qquad \frac{\Gamma \vdash N, \Delta \qquad \Gamma, M \vdash \Delta}{\Gamma, N \to M \vdash \Delta}\ (\to \vdash)$$

$$\frac{\Gamma \vdash N, \Delta}{\Gamma \vdash \forall \mathrm{x} N, \Delta}\ (\vdash \forall^1)^* \qquad \frac{\Gamma, N[\mathrm{t}/\mathrm{x}] \vdash \Delta}{\Gamma, \forall \mathrm{x} N \vdash \Delta}\ (\forall^1 \vdash)$$

$$\frac{}{\Gamma, \bot \vdash \Delta}\ (\bot \vdash)$$

*: x not free in Γ, Δ.

(c) Logic

Figure I.9: The $\forall, \to, \bot$ fragment of Gentzen's **LK**

The judgement of commands is generalised into the following:

$$c : (x_1 : A_1, \ldots, x_n : A_n \vdash \alpha_1 : B_1, \ldots, \alpha_m : B_m),$$

which echoes the symmetry of the sequents of **LK**. In the next chapters, co-variables will be given a polarity like variables, but not in $\mathbf{L}_\mathrm{n}$. The judgements for expressions and contexts are the following:

$$\Gamma \vdash t : A \mid \Delta \qquad\qquad \Gamma \mid e : A \vdash \Delta$$

with $\Gamma = x_1 : A_1, \ldots, x_n : A_n$ and $\Delta = \alpha_1 : B_1, \ldots, \alpha_m : B_m$. The binder $\langle \mu\alpha.c \mid$ generalises the adjoint operation c^*:

$$\boxed{\langle \mu\alpha.c \| E \rangle \succ c[E/\alpha]}$$

$$t = t_{\ominus} ::= x \mid \mu\alpha.c \mid \lambda x.t$$
$$e = e_{\ominus} ::= \pi \mid \tilde{\mu}x.c$$
$$e_{\ominus} \supseteq \pi ::= \alpha \mid \mathsf{stop} \mid t\cdot\pi$$
$$c ::= \langle t \parallel e \rangle$$

(a) Terms, contexts and commands

$$\langle t \parallel \tilde{\mu}x.c \rangle \rhd_{\mathrm{R}_n} c[t/x]$$
$$\langle \mu\alpha.c \parallel \pi \rangle \rhd_{\mathrm{R}_n} c[\pi/\alpha]$$
$$\langle \lambda x.t \parallel u\cdot\pi \rangle \rhd_{\mathrm{R}_n} \langle t[u/x] \parallel \pi \rangle$$

(b) Reduction rules

$$e \rhd_{\mathrm{E}_n} \tilde{\mu}x.\langle x \parallel e \rangle$$
$$t \rhd_{\mathrm{E}_n} \mu\alpha.\langle t \parallel \alpha \rangle$$
$$t \rhd_{\mathrm{E}_n} \lambda x.\mu\alpha.\langle t \parallel x\cdot\alpha \rangle$$

(c) Expansion rules

Figure I.10: $\mathbf{L}_n$: the calculus

Thus, c^* corresponds to $\mu\star.c$. In other words, we can see $\langle \mu\alpha.c \mid$ as a function of its context:

$$\langle \mu\alpha.c \mid : \alpha \mapsto c$$

Completeness of typing rules with respect to provability in the $\to$, $\forall$, $\bot$ fragment of **LK** requires the rule $(\to \vdash)$ which is derived from the restricted rule $(\to \vdash_f)$ by focalisation (Figure I.11d).

I.8.2 The λ*C* calculus

Let us consider the λC calculus in call by name introduced by Lafont, Reus and Streicher [LRS93], a setting made popular by the works of Krivine [Kri09]. A variant of the Krivine machine with the C operator, described by Streicher and Reus [SR98, p. 21], is displayed in Figure I.12 on page 81. Following Griffin [Gri90], we type C with double negation elimination:

$$C : \neg\neg A \to A\,.$$

$$\Gamma = \vec{x_i} : \vec{N_i} \qquad \Delta = \vec{\alpha_j} : \vec{M_j}$$
$$\Gamma \vdash t : N \mid \Delta \qquad \Gamma \mid e : N \vdash \Delta$$
$$c : (\Gamma \vdash \Delta)$$

(a) Judgements

$$\frac{}{\Gamma, x : N \vdash x : N \mid \Delta}\ (\vdash \mathrm{ax}) \qquad \frac{}{\Gamma \mid \alpha : N \vdash \alpha : N, \Delta}\ (\mathrm{ax} \vdash)$$

$$\frac{c : (\Gamma, x : N \vdash \Delta)}{\Gamma \mid \tilde{\mu} x.c : N \vdash \Delta}\ (\tilde{\mu} \vdash) \qquad \frac{c : (\Gamma \vdash \alpha : N, \Delta)}{\Gamma \vdash \mu \alpha.c : N \mid \Delta}\ (\vdash \mu)$$

$$\frac{\Gamma \vdash t : N \mid \Delta \qquad \Gamma \mid e : N \vdash \Delta}{\langle t \| e \rangle : (\Gamma \vdash \Delta)}\ (\mathrm{cut})$$

(b) Identity and Structure

$$\frac{\Gamma, x : N \vdash t : M \mid \Delta}{\Gamma \vdash \lambda x.t : N \to M \mid \Delta}\ (\vdash \to) \qquad \frac{\Gamma \vdash t : N \mid \Delta \qquad \Gamma \mid \pi : M \vdash \Delta}{\Gamma \mid t \cdot \pi : N \to M \vdash \Delta}\ (\to \vdash_f)$$

$$\frac{\Gamma \vdash t : N \mid \Delta}{\Gamma \vdash t : \forall \mathrm{x} N \mid \Delta}\ (\vdash \forall)^* \qquad \frac{\Gamma \mid e : N[\mathrm{t/x}] \vdash \Delta}{\Gamma \mid e : \forall \mathrm{x} N \vdash \Delta}\ (\forall \vdash)$$

$$\frac{}{\Gamma \mid \mathsf{stop} : \bot \vdash \Delta}\ (\bot \vdash)$$

*: x not free in Γ, Δ

(c) Logic

$$\frac{\Gamma \vdash t : N \mid \Delta \qquad \Gamma \mid e : M \vdash \Delta}{\Gamma \mid \tilde{\mu} x.\langle \mu \alpha.\langle x \| t \cdot \alpha \rangle \| e \rangle : N \to M \vdash \Delta}\ (\to \vdash)$$

(d) Derivability of the rule $(\to \vdash)$

Figure I.11: L_n: typing rules

$$t, u ::= x \mid \lambda x.t \mid t\ u \mid C$$

(a) Quasi-proof terms

$$\frac{}{\Gamma, x : N \vdash x : N} \qquad \frac{}{\Gamma \vdash C : \neg\neg N \to N}$$

$$\frac{\Gamma, x : N \vdash t : M}{\Gamma \vdash \lambda x.t : N \to M} \qquad \frac{\Gamma \vdash t : N \to M \qquad \Gamma \vdash u : N}{\Gamma \vdash t\ u : M}$$

$$\frac{\Gamma \vdash t : N}{\Gamma \vdash t : \forall \mathrm{x}\, N}\ ^{*} \qquad \frac{\Gamma \vdash t : \forall \mathrm{x}\, N}{\Gamma \vdash t : N[\mathrm{u}/\mathrm{x}]}$$

*: x not free in Γ

(b) Second order predicate calculus

Figure I.12: The λC calculus

Krivine's abstract machine introduced in Figure I.13 on the following page is the simplest way to perform weak head reduction of λC terms. One insight is to enrich the set of terms with an operation k_π that appears during reduction. We introduce, following Krivine, the notion of *quasi-proof* term in order to distinguish the original terms from terms with k_π. Let us mention that the notion of quasi-proof term is of technical importance in the construction of models starting from classical realisability [Kri12].

The operations of the λC calculus satisfies the following reductions:

$$\begin{aligned} \langle C \,\|\, t \cdot \pi \rangle &\succ_n \langle t \,\|\, \mathsf{k}_\pi \cdot \mathsf{stop} \rangle \\ \langle \mathsf{k}_\pi \,\|\, t \cdot \pi' \rangle &\succ_n \langle t \,\|\, \pi \rangle \end{aligned}$$

Thanks to the binder μ, we can solve these equations in $\mathbf{L}_{\mathrm{n}}$. Indeed, we can see k_π as a function of its context:

$$\langle \mathsf{k}_\pi \,|\, : x \overset{\lambda}{\mapsto} (\alpha \overset{\mu}{\mapsto} \langle x \| E \rangle)$$

$$c ::= \langle t \| \pi \rangle \qquad t, u ::= \ldots \mid \mathsf{k}_\pi \qquad \pi ::= \mathsf{stop} \mid t \cdot \pi$$

(a) Machines, terms and stacks

$$\begin{array}{rcl} \langle t\ u \| \pi \rangle & \succ_n & \langle t \| u \cdot \pi \rangle \\ \langle \lambda x.t \| u \cdot \pi \rangle & \succ_n & \langle t[u/x] \| \pi \rangle \\ \langle C \| t \cdot \pi \rangle & \succ_n & \langle t \| \mathsf{k}_\pi \cdot \mathsf{stop} \rangle \\ \langle \mathsf{k}_\pi \| t \cdot \pi' \rangle & \succ_n & \langle t \| \pi \rangle \end{array}$$

(b) Reduction rules

$$\begin{array}{rcl} t\ u & \stackrel{\text{def}}{=} & \mu\alpha.\langle t \| u \cdot \alpha \rangle \\ \mathsf{k}_e & \stackrel{\text{def}}{=} & \lambda x.\mu\alpha.\langle x \| e \rangle \\ C & \stackrel{\text{def}}{=} & \lambda x.\mu\alpha.\langle x \| \mathsf{k}_\alpha \cdot \mathsf{stop} \rangle \end{array}$$

(c) Implementation in $\mathbf{L}_\mathbf{n}$

Figure I.13: The Krivine abstract machine

therefore we take:

$$\boxed{\mathsf{k}_\pi \stackrel{\text{def}}{=} \lambda x.\mu\alpha.\langle x \| \pi \rangle}$$

Similarly for C we take:

$$\boxed{C \stackrel{\text{def}}{=} \lambda x.\mu\alpha.\langle x \| \mathsf{k}_\alpha \cdot \mathsf{stop} \rangle}\ .$$

These definitions induce a translation from the λC calculus into the calculus $\mathbf{L}_\mathbf{n}$ that simulates reduction. We identify machines of the λC calculus with commands of the calculus $\mathbf{L}_\mathbf{n}$. As an immediate consequence of having defined the constructs of the λC by solving equations, we have:

Proposition 1.4 (Simulation)**.** *If $c \succ_n c'$ then $c \rhd^+_{\mathrm{R}_n} c'$.*

The purpose of embedding the λC calculus into $\mathbf{L}_\mathbf{n}$ is that it inherits from the latter a compatible equivalence relation which is extensional.

Definition 1.5. For t and u be two quasi-proof terms of λC, we define $t \approx_n u$ as $t \simeq_{\mathrm{RE}_n} u$.

One point of this definition is that it encompasses the observational equivalence defined by Krivine machines:

Proposition 1.6. *Let t and u be two quasi-proof terms of λC. If for all*

stacks π one has $\langle t \| \pi \rangle \ (\prec_n \cup \succ_n)^* \ \langle u \| \pi \rangle$, then one has $t \approx_n u$.

Proof. By extending the category of stacks of λC with some variable α of $\mathbf{L}_{\mathrm{n}}$ it is easy to see that one has $\langle t \| \alpha \rangle \ (\prec_n \cup \succ_n)^* \ \langle u \| \alpha \rangle$. Thus by (the extension of) Proposition 1.4 one has $\langle t \| \alpha \rangle \simeq_{\mathrm{RE}_n} \langle u \| \alpha \rangle$. Thus $t \simeq_{\mathrm{RE}_n} u$ by extensionality in $\mathbf{L}_{\mathrm{n}}$. ∎

The definitions in $\mathbf{L}_{\mathrm{n}}$ are also the middlemen of the systematic reconstruction of types from reduction rules. For instance, we can reconstruct the type of C, first by noticing that k_π has the following rule:

$$\dfrac{\Gamma \mid \pi : A \vdash \Delta}{\Gamma \vdash \mathsf{k}_\pi : A \to B \mid \Delta}$$

Indeed:

$$\dfrac{\dfrac{\dfrac{\Gamma, x : A \mid \pi : A \vdash \alpha : B, \Delta}{\langle x \| \pi \rangle : (\Gamma, x : A \vdash \alpha : B, \Delta)}}{\Gamma, x : A \vdash \mu\alpha.\langle x \| \pi \rangle : B \mid \Delta}}{\Gamma \vdash \mathsf{k}_\pi : A \to B \mid \Delta}$$

Then we can derive:

$$\dfrac{}{\Gamma \vdash C : ((A \to \bot) \to \bot) \to A \mid \Delta}$$

Indeed:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma, x : A \mid \alpha : A \vdash \alpha : A, \Delta}}{\Gamma, x : A \vdash \mathsf{k}_\alpha : A \to \bot \mid \alpha : A, \Delta} \quad \dfrac{}{\Gamma, x : A \mid \mathsf{stop} : \bot \vdash \alpha : B, \Delta}}{\Gamma, x : A \mid \mathsf{k}_\alpha \cdot \mathsf{stop} : (A \to \bot) \to \bot \vdash \alpha : A, \Delta}}{\langle x \| \mathsf{k}_\alpha \cdot \mathsf{stop} \rangle : (\Gamma, x : (A \to \bot) \to \bot \vdash \alpha : A, \Delta)}}{\Gamma, x : (A \to \bot) \to \bot \vdash \mu\alpha.\langle x \| \mathsf{k}_\alpha \cdot \mathsf{stop} \rangle : A \mid \Delta}}{\Gamma \vdash C : ((A \to \bot) \to \bot) \to A \mid \Delta}$$

Notice that the distinction between quasi-proof terms and terms is akin to the one between sequent calculus and natural deduction, in the sense terms that are not quasi-proofs can only be typed in the

sequent calculus (see for instance the typing rule for k_π which does not make sense in natural deduction).

I.9 Historical remarks and contributions

Landin pioneered the use of the λ calculus to model programming languages [Lan65], the use of abstract machines to model reduction [Lan64] and the use of first-class continuations to model control [Lan98]. As Danvy and his collaborators have shown [ABDM03, BD07, Dan08], building on earlier works of Reynolds [Rey72], there are systematic transformations between abstract machines and other techniques ("*structural operational semantics, reduction semantics,* […] *big-step abstract machines, natural semantics, and denotational semantics*"). We will see in Section I.10.2 that **L** calculi can themselves be understood as fine-grained accounts of continuation-based denotational semantics, not only for reductions but also at the level of extensionality.

Girard modernised and made popular the ideas of Gentzen. He introduced polarisation [Gir91] after the discovery of focalisation by Andreoli [And92]. We also owe him the terminology "*adjoint*" inspired from linear algebra to denote a construction τ^* that satisfies $\langle \tau^*(t) \| \pi \rangle \simeq \langle t \| \tau(\pi) \rangle$ in the context of computation [Gir01, Gir06]. Danos, Joinet and Schellinx explained — among other contributions — how polarisation and focalisation are a necessary consequence of extensionality in the setting of classical logic [DJS97].

Griffin was the first to show that Felleisen's variant *C* [FFKD87] of Landin's operator *J* (via Reynold's *escape* [Rey72]) could be typed with the elimination of double negation [Gri90].

Parigot introduced the binder μ with the λμ calculus [Par92]. The symmetric binder $\tilde{\mu}$ was introduced by Curien and Herbelin [CH00], along with the relationship between sequent calculus and abstract machines (in call by value and in call by name) as well as the one between evaluation strategies and the ways of solving the critical pair. Herbelin introduced in an unpublished note [Her08] the terminology "system L" for a syntax for **LK** proofs based on μ and $\tilde{\mu}$. The importance of having an explicit let … in … binding in call by value was recognised earlier, with Moggi's

monad-based models [Mog89]. Ariola and Herbelin [AH08] described the advantages of treating the context explicitly in a λ calculus with control, compared to previous operational semantics of Felleisen and his collaborators [FFKD87, FH92].

Besides Curien and Herbelin, the link between values and focalisation was implied by Ogata [Oga02]. It was explicitly stated by Dyckhoff and Lengrand [DL06, DL07]. Also, the observation that certain rules of the sequent calculus could be explained as pattern-matching is due to Cerrito and Kesner [CK99, CK04]. A unified setting in which values and pattern-matching continuations interact was proposed as the computational explanation of Andreoli's proof search discipline by Zeilberger [Zei08, Zei09]. The interpretation is based on mixing connectives of different "*polarities*" defined by their focalisation properties, one consequence being that *"evaluation order is explicitly reflected at the level of types"*. This interpretation is also related to Levy's decomposition call by value and call by name [Lev99, Lev04, Lev05].

Aside from Girard's polarisation, loss of associativity of composition was noticed in game semantics with Blass's games [Bla92], as described by Abramsky [Abr03] and analysed by Melliès [Mel05]. We will come back on this phenomenon in Chapter II.

The untyped, interactive approach to proof theory was initiated with Girard's ludics [Gir01] and Krivine's classical realisability [Kri09].

The account of this section is inspired from [Mun09] where I introduced, following Krivine's approach, the direct account of focalisation and polarities in a variant of Curien and Herbelin's calculus. Using polarities inspired by Girard and Danos, Joinet and Schellinx, we associate positiveness to strictness and negativeness to laziness. The difference with Zeilberger's polarities is that we do not assume that the polarity of a connective is defined by its focalisation properties. In [Mun09] this allowed us to describe the complex polarisations of modal connectives such as classical logic's ∀ and linear logic's !.[8] The differences and the relationship of our notion of polarities compared to Zeilberger's are further described in Sections II.2.4 and III.3.

The account is also inspired from [CM10] where with Curien we showed

[8] In the context of [Mun09], we may define positive connectives A and ¡ restricted to

how the sequent calculus could have arisen from the desire to type abstract machines for the call-by-value λ calculus. We introduced in particular the idea that the definition of application can be read off its reduction rule. Here we refine this idea by presenting transition rules as equations and **L** calculi as spaces of solutions.

In addition, we contribute in this chapter a summary of most of the above in a coherent presentation. For this purpose we also expose the distinction between syntactic and operational linearity, as well as benefits of $\tilde{\mu}$: making commutative cuts obsolete and simplifying the definition of orthogonality-based logical relations.

I.10 Contributions

I.10.1 Polarisation in categorical models

Polarisation can be negatively described as rejecting, either directly or indirectly, the hypothesis that composition is *a priori* associative. In Chapter II, we introduce a positive and direct description of polarisation. We characterise the polarised evaluation order through a category-like structure where not all composites associate. *Duploid* is the name of the structure, as a reference to Jean-Louis Loday's duplicial algebras [Lod06].

The main result relates duploids to adjunctions. To help understand this relation, let us first recall the correspondence between

values as follows:

$$\frac{\Gamma \vdash V_+ : P}{\Gamma \vdash V_+ : \mathrm{A}X.P}\ (X \notin \mathbf{fv}(\Gamma)) \qquad \frac{¡\Gamma \vdash V_+ : P}{¡\Gamma \vdash V_+ : ¡P}$$

But the respective restrictions on the typing contexts prevent us from deducing the same rules for non-value positive terms using focalisation (introducing a cut). The simplest workaround is to introduce polarity coercions ⇓ from negative to positive, yielding the equations $\forall = \mathrm{A}⇓$ and $! = ¡⇓$. The proto-quantification A describes polymorphism with value restriction (see [Mun09]). The proto-exponential ¡ is reminiscent of Melliès and Tabareau's decomposition of exponentials with ressource modalities in tensor logic [MT10].

direct models of call by value and indirect models *à la* Moggi.

Direct models

In a *direct* denotational model, there should be a close match between the given operations in the model and the constructions in the language. Essentially, type and program constructors should respectively correspond to operations on objects and on morphisms in a category. In particular, it should be possible to reason about an instance of the model within the language.[9] An example of direct models for the simply-typed λ calculus is given by cartesian-closed categories.

In a model such as Moggi's λ_C models [Mog89], or Lafont, Reus and Streicher's model of call by name [LRS93], however, the language is not interpreted directly but through a Kleisli construction for a monad or a co-monad. We have a precise description of the link between direct models and indirect models thanks to Führmann [Füh99]. Categories that model call by value directly are characterised by the presence of a *thunk*, a formal account of the well-known structure used to implement laziness in call-by-value languages [HD97].

The characterisation takes the following form: any direct model arises from the Kleisli construction starting from a λ_C model. However, from the direct model we can only recover a specific λ_C model: it is made of semantic values consisting of all the pure expressions. More precisely, the Kleisli construction is a reflection that conflates any two values equalised by the monad, and turns into a value any *thunkable* expression. An expression is thunkable if it behaves similarly to a value in a sense determined by the monad.

Selinger [Sel01] proves a similar relationship between direct models of the call-by-name $\lambda\mu$ calculus and Lafont, Reus and Streicher's models [LRS93].

[9] Führmann [Füh99], Selinger [Sel99].

Evaluation order	By value	By name	Polarised
Direct model	Thunk	Runnable monad	Duploid
Indirect model	Monad T	Co-monad L	Adjunction $F \dashv G$
Programs	Kleisli maps $P \to TQ$	co-Kleisli maps $LN \to M$	Oblique maps $FP \to N$ $\simeq P \to GN$
Syntactic data	Values	Stacks	Both
Completion into	Thunkable expressions	Linear evaluation contexts	Both

Table I.14: Comparison of the structures underlying various direct-style models of computation.

Adjunction-based models

The chapter deals with the underlying algebraic structure in these models: a monad over a category of values for call by value, a co-monad over a category of stacks for call by name. Duploids generalise the underlying structure to an adjunction between a category of values and a category of stacks. (See Table I.14.)

Relationship with polarities comes from Girard's *polarised translation* of classical logic [Gir91, DJS97, Lau02]. Our duploid construction extends the (skeleton of the) polarised translation to any adjunction. (Essentially, we do not need the assumption that there is an involutive negation operation on formulae.)

We know that there is a practical relevance of decomposing monads, when seen as notions of computation, into adjunctions, thanks to Levy [Lev99, Lev04, Lev05]. Levy's adjunctions subsume models of call by value and call by name. However the model is indirect, and still lacks a corresponding notion of direct model.

The duploid construction

With the duploid construction, we define, starting from an adjunction $F \dashv G : \mathscr{C}_1 \to \mathscr{C}_2$, morphisms $A \to B$ for A and B objects of *either*

category $\mathscr{C}_1$ and $\mathscr{C}_2$.

We first define oblique morphisms $P \to N$, with $P \in |\mathscr{C}_2|$ and $N \in |\mathscr{C}_1|$, equivalently as maps $P \to GN$ or $FP \to N$. Now consider $\bullet$ the composition in $\mathscr{C}_1$ and $\circ$ the composition in $\mathscr{C}_2$. We observe that oblique morphisms compose either in $\mathscr{C}_1$:

$$\dfrac{\dfrac{f : P \to FQ}{f : FP \to FQ}\,(\simeq) \quad \dfrac{g : Q \to N}{g : FQ \to N}\,(\simeq)}{\dfrac{g \bullet f : FP \to N}{g \bullet f : P \to N}\,(\simeq)}\,(\bullet)$$

or in $\mathscr{C}_2$:

$$\dfrac{\dfrac{f : P \to N}{f : P \to GN}\,(\simeq) \quad \dfrac{g : GN \to M}{g : GN \to GM}\,(\simeq)}{\dfrac{g \circ f : P \to GM}{g \circ f : P \to M}\,(\simeq)}\,(\circ)$$

Thus we define a morphism $A \to B$ as an oblique morphism:

$$\boxed{A^+ \to B^\ominus}\,,$$

where:

$$P^+ \stackrel{\text{def}}{=} P \qquad P^\ominus \stackrel{\text{def}}{=} FP$$
$$N^+ \stackrel{\text{def}}{=} GN \qquad N^\ominus \stackrel{\text{def}}{=} N$$

The reason why this does not necessarily define a category is that we can have:

$$f \circ (g \bullet h) \neq (f \circ g) \bullet h$$

when the adjunction is not idempotent, as we will see.

Contribution

In Chapter II, we prove that there exists a reflection:

$$\mathscr{Dupl} \triangleleft \mathscr{Adj}$$

where $\mathcal{Dupl}$ is a category of duploids and *duploid functors*, and where $\mathcal{Adj}$ is the category of adjunctions and pseudo maps of adjunctions. In other words, the duploid construction above extends to a functor $\mathcal{Adj} \to \mathcal{Dupl}$ that admits a full and faithful right adjoint. In particular, any duploid is obtained from an adjunction.

As a consequence of the main result, duploids account for a wide range of computational models, as we will see in various examples from denotational semantics. It suggests that the various biases in denotational semantics: indirect, call-by-value, call-by-name… are ways of hiding the fact that composition is not always associative.

Also, because of the reflection, $\mathcal{Dupl}$ is equivalent to a full subcategory of $\mathcal{Adj}$, which means that the adjunction associated to a duploid has additional properties. We characterise which exactly. We show that there is an equivalence of categories:

$$\mathcal{Dupl} \simeq \mathcal{Adj}_{eq}$$

where $\mathcal{Adj}_{eq}$ is the full subcategory of adjunctions that satisfies the *equalising requirement*: the unit and the co-unit of the adjunction are respectively equalisers and co-equalisers. This requirement means that:

- the category of values is completed with all thunkable expressions;
- the category of stacks is completed with all evaluation contexts that are linear;
- two values and any two stacks that are not distinguished by the model of computation are identified.

In other words, the duploid expresses the point of view from inside the model of computation defined by the adjunction.

The chapter introduces the calculus $\mathbf{L}_{\mathrm{dup}}$ which is an internal language for duploids. With it we explain the operational contents of duploids.

$$
\begin{array}{lcl}
x^{\dagger} & \stackrel{\text{def}}{=} & \lambda k.(k\,x) \\
(\lambda x.t)^{\dagger} & \stackrel{\text{def}}{=} & \lambda k.(k\,\lambda x.t^{\dagger}) \\
(t\,u)^{\dagger} & \stackrel{\text{def}}{=} & \lambda k.(t^{\dagger}\,\lambda x.(u^{\dagger}\,\lambda y.(x\,y\,k)))
\end{array}
$$

(a) A call-by-value translation (Plotkin [Plo75])

$$
\begin{array}{lcl}
x^{\ddagger} & \stackrel{\text{def}}{=} & x \\
(\lambda x.t)^{\ddagger} & \stackrel{\text{def}}{=} & \lambda y.(t^{\ddagger}[\mathsf{fst}(y)/x]\,\mathsf{snd}(y)) \\
(t\,u)^{\ddagger} & \stackrel{\text{def}}{=} & \lambda k.(t^{\ddagger}\,(u^{\ddagger}, k))
\end{array}
$$

(b) A call-by-name translation (Lafont, Reus and Streicher [LRS93])

Figure I.15: Continuation-passing translations into the λ calculus

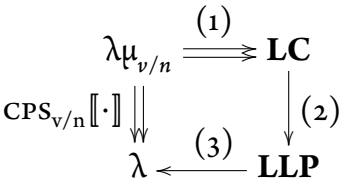


Figure I.16: The polarised decomposition of CPS in three steps

I.10.2 Polarisation in programming languages

Polarisation, and the **L** calculi, allow us to explain the structure of first-class continuations and continuation-passing style (CPS) through a decomposition in three steps. This is the subject of Chapter III.

A polarised decomposition of CPS translations

Thanks to the proof-theoretic study of classical logic [Gir91, DJS95, DJS97, Lau02], we know that the standard call-by-value continuation-passing translation (see Figure I.15 on the current page) corresponds to a positive polarisation of formulae while Lafont, Reus and Streicher's call-by-name translation corresponds to a negative polarisation of formulae. (We consider the variants that take into account the *C* operator.) The symmetry between polarities translates into a categorical duality that we can find exposed in Selinger [Sel01] and Curien and Herbelin [CH00].

The chapter is to begin with an account of how, thanks to notions of polarities, focalisation and sequent calculus, we can decompose

in three steps call-by-name and call-by-value continuation-passing-style translations (Figure 1.16 on the preceding page). In doing so we reformulate existing and well-known translations for simply-typed typed λμ calculus [Lau02].

Step (1) encodes derivations within the λμ calculus as all-positive or all-negative derivations in the polarised classical logics **LC** [Gir91] or $\mathbf{LK}^{\eta}_{p}$ [DJS97]; thereby choosing a way to polarise formulae.

Step (2) is Laurent's translation from **LC** to the so-called polarised linear logic **LLP** [Lau02]. This translation is a particular case of the duploid construction of Chapter II, and therefore step (2) is where the direct style is lost. **LLP** can be thought of as the $\wedge, \vee, \neg$ fragment of **LJ**, which has been made symmetric by introducing an involutive negation operation $\cdot^{\perp}$ on formulae. As a consequence, negation:

$$\frac{\Gamma, P \vdash}{\Gamma \vdash \neg P}\,(\vdash \neg) \qquad\qquad \frac{\Gamma \vdash P}{\Gamma, \neg P \vdash}\,(\neg \vdash)$$

is split into two covariant modalities written ! and ?:

$$\frac{\vdash N, \mathcal{N}}{\vdash\, !N, \mathcal{N}}\,(\text{prom}) \qquad\qquad \frac{\vdash \mathcal{N}, P}{\vdash \mathcal{N}, ?P}\,(\text{der})$$

where $\mathcal{N}$ contains only negative formulae. The modalities reflect changes of polarities; as such they distinguish the formula for the call-by-value arrow $(!(P \multimap ?Q))$ from the formula for the call-by-name one $(!N \multimap M)$.

Step (3) translates the previous logic into the λ calculus with only tail calls, in other words it translates it into a calculus of continuations. This is done as follows:

- Sequents are flattened into single-conclusion ones, i.e. $\vdash \mathcal{N}, P$ is mapped into $\mathcal{N}^{\perp} \vdash P$. This is where the explicit polarities disappear.
- Sequent calculus is cast into natural deduction: in terms of an **L** calculus, a context $\Gamma \mid E : A \vdash \star : B$ is mapped into a λ-term $\Gamma, k :$

Calculus	Ref	Eval. strat.	Style	Red. & Eq.	
$\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^+}$	Fig. III.2	Polarised	Seq. calc.	$\triangleright_{\mathrm{R}_p}, \simeq_{\mathrm{RE}_p}$	
$\lambda_{\mathrm{v}}^{\times}$	Fig. III.3	Indirect	Nat. ded.	$\triangleright_{\mathrm{R}_\lambda}, \simeq_{\mathrm{RE}_\lambda}$	
$\lambda\mu\widehat{\mathrm{tp}}_v$	Fig. III.5	By value	Nat. ded.	$\triangleright_{\mathrm{R}_v}, \simeq_{\mathrm{RE}_v}$	[AHS04]
$\lambda\mu\widehat{\mathrm{tp}}_n$	Fig. III.7	By name	Nat. ded.	$\triangleright_{\mathrm{R}_n}, \simeq_{\mathrm{RE}_n}$	[HG08]
$\mathbf{L}_{\mathrm{exp}}$	Fig. III.11	Indirect	Seq. calc.	$\triangleright_{\mathrm{R}_e}, \simeq_{\mathrm{RE}_e}$	

Table I.17: Summary of the calculi considered in Chapter III

$A \vdash E^* : B$ where k is linear inside E^*. This is where administrative reductions need to be taken care of, as we will see.

In other words, this step essentially erases an interesting structure in order to fit the mould of the λ calculus.

Notice that steps (2) and (3) are the same regardless of the source calculus.

Contribution: a polarised decomposition of delimited CPS translations

Our contribution is two-fold: we underline the difference between steps (1) and (2) thanks to an **L** calculus that takes place in the upper-right-hand corner of Figure I.16, and we extend the decomposition in three steps to Danvy and Filinski's *shift* and *reset* control operators [DF90]. Our approach allows us to rationally reconstruct in direct style four distinct call-by-name variants of *shift* and *reset*, one of which appears to be new.

Historical remarks

The use of first-class continuations to model control features in ALGOL 60 [BBG+63] goes back to Landin [Lan65, Lan98, DM08]. Many people were involved in the recognition the following years of the concept of *continuation*, meaning the functional abstraction of the remainder of a computation — notably van Wijngaarden, Mazurkiewicz, Morris, Wadsworth, Fischer, Reynolds and Abdali (see Reynolds's historical account [Rey93]).

Continuation-passing style has been used to provide both implementations (Sussman and Steele [SS75]) and denotational semantics (Strachey

and Wadsworth [SW74]) to higher-order programming languages.

The qualities of continuation-passing style used in intermediate representations of compilers have long been recognised (Steele [Ste78], Shivers [Shi91]). Such intermediate representations are widely used in contemporary compilers, along with other representations. This is explained by the fact that the structure of continuation-passing style makes the intermediate representation amenable to optimisations and analyses (Demange [Dem12]). Continuation-based representations are also essential in accounting for control in the conception of verified compilers. There, the representation of the context "*inside-out*" is reminiscent of focalisation and abstract machines like the ones we find in this thesis. (Leroy [Ler12])

First-class continuations and continuation-passing style have been used to model both evaluation order and control. In both aspects they also led to discoveries. Danvy and Filinski introduced the delimited control operators *shift* and *reset* [DF90] in order to model in direct style the fact that continuations can return and be composed, as used in functional backtracking with success and failure continuations. The expressiveness of delimited control operators has been demonstrated by Filinski [Fil96, Fil94, Fil99]: they can express in direct style any monadic effect, as long as the laws of the monad can be expressed as pure functional terms. Their use goes beyond programming: for instance, they are used in linguistics to model phenomena of natural languages (Shan [Sha04]), and in logic to derive principles that are ordinarily only admissible (Herbelin [Her10], Ilik [Ili10]).

In a work independent from delimited control operators, Lafont, Reus and Streicher introduced a continuation-passing translation for the call-by-name λ calculus [LRS93]. Its originality, with hindsight, is to return continuations instead of passing them, and to emphasise the value-like nature of stacks. Its quality is to satisfy η equivalence, thus leading to a cartesian-closed category, in contrast with Plotkin's earlier call-by-name translation [Plo75]. This new translation is an input from proof-theory: it is inspired from double-negation translations of Krivine and Girard [Kri90, Gir91].

The point of view of the chapter is that the structures at work in conti-

nuation-passing style can be explained by decomposing the continuation-passing-style translations. The first such decomposition was the one of Danvy and Hatcliff [HD97]: they showed that the call-by-name translation of Plotkin can be explained as the implementation of call by name into call by value using thunks, followed by the call-by-value translation.

I.10.3 Polarisation in proof theory

In Chapter IV, we go back to the proof-theoretic roots of polarisation: to Girard's account [Gir91] of an involutive negation in classical logic, which we seek to explain. However we do not assume the reader to be familiar with Girard's work and take a different angle. Our goal is to investigate the constructive interpretation of an involutive negation, or equivalently of reasoning by contrapositive, assuming that these words take a specific meaning.

Assumption I.7. *By* constructive*, we mean an interpretation that follows the* formulae-as-types *notion of construction, in which formulae are mapped into the types of a programming language.*

This notion of constructiveness is not based on an a priori limitation of proof techniques, such as the refusal of the principle of excluded middle $\forall A\,(A \vee \neg A)$. Thus, Griffin showed that Felleisen's $\mathcal{C}$ [FFKD87] could be typed with the elimination of double negation [Gri90], and therefore it can be used to derive the principle of excluded middle. But the proof provides by no means a decision procedure for any A: the behaviour depends in general on the context in which the principle is invoked. In other words, it should not be seen as a contradiction to intuitionism: the latter assumes that the behaviour of proofs is referentially transparent, while we do not.

Neither should this notion of constructiveness be seen as a restriction to Church-style typed calculi. Just like type systems are an approximation of correct programs, logical systems are seen as an approximation of constructive behaviours. This is the point of view of Krivine's classical realizability [Kri09], for instance. This is why there is no contradiction with our primary interest in untyped calculi.

In this context, the constructive nature is observed in particular for purely positive formulae (Girard [Gir91]). In arithmetic, the notion encompasses Σ_1^0 formulae. Purely positive formulae must satisfy the same criterion as in intuitionistic logic: disjunction property, property of existence… As a consequence, proofs of a formula $\forall \vec{x}(P_0(\vec{x}) \to Q_0(\vec{x}))$ where P_0, Q_0 are purely positive (that is to say Π_2^0 formulae in arithmetic) correspond to algorithms (Murthy [Mur91]).

Such a notion of constructiveness can be understood through a comparison with programming languages: in the presence of side-effecting operations (control operators, state, input/output…), certain types are opaque at runtime.

Assumption 1.8. *By* involutive*, we mean a negation that satisfies a type isomorphism between $\neg\neg A$ and A.*

The reason for asking more than a mere equivalence between A and $\neg\neg A$ is that there are too many choices for the contrapositive of a proposition such as the following:

$$\forall \mathrm{x}, \mathrm{y} \in A, (P(\mathrm{x}) \vee Q(\mathrm{y})) \to (\forall \mathrm{x} \in A, P(\mathrm{x})) \vee (\forall \mathrm{y} \in A, Q(\mathrm{y}))$$

for instance:

$$\begin{array}{rcl} \neg\big((\forall \mathrm{x} \in A, P(\mathrm{x})) \vee (\forall \mathrm{y} \in A, Q(\mathrm{y}))\big) & \to & \neg \forall \mathrm{x}, \mathrm{y} \in A, (P(\mathrm{x}) \vee Q(\mathrm{y})) \\ (\neg \forall \mathrm{x} \in A, P(\mathrm{x})) \vee (\neg \forall \mathrm{y} \in A, Q(\mathrm{y})) & \to & \exists \mathrm{x}, \mathrm{y} \in A, \neg(P(\mathrm{x}) \vee Q(\mathrm{y})) \\ (\exists \mathrm{x} \in A, \neg P(\mathrm{x})) \wedge (\exists \mathrm{y} \in A, \neg Q(\mathrm{y})) & \to & \exists \mathrm{x}, \mathrm{y} \in A, (\neg P(\mathrm{x}) \wedge \neg Q(\mathrm{y})) \end{array}$$

We are no longer overwhelmed with choices once De Morgan laws are type isomorphisms: if there are too many proofs, then we must be able to choose a canonical one, one that preserves meaning.

In the absence of such isomorphisms, we take contrapositive to mean the proposition obtained by pushing negations the closest to the leaves, using De Morgan laws. In the above example this corresponds to the third proposition. Works such as the one of Krivine [Kri09] show from a technical standpoint the importance of

such reasoning by contrapositive; for instance the axiom of countable choice is only realised through its contrapositive. Yet, reasoning by contrapositive is not trivial in the context of the λC calculus used by the author, as the following example shows.

Example 1.9. We set ourselves in the λC calculus from Figure 1.12 on page 81, extended with second order quantification.[10] This allows us to define quantifiers in a standard way: $\forall X \in A, B \stackrel{\text{def}}{=} \forall X(A[X] \to B)$ and $\exists X \in A, B \stackrel{\text{def}}{=} \forall Y(\forall X\,(A[X] \to B \to Y) \to Y)$, and similarly for first-order.

Now let us consider the following formulae, which have the same complexity as the axiom of countable choice and its contrapositive:

$$[C]: \qquad \forall \mathrm{x} \in \mathbb{N}, \exists \mathrm{y} \in E, A_1(\mathrm{x},\mathrm{y}) \to \exists Y \in F, \forall \mathrm{x} \in \mathbb{N}, A_2(\mathrm{x}, Y)$$
$$[\bar{C}]: \quad \forall Y \in F, \exists \mathrm{x} \in \mathbb{N}, \neg A_2(\mathrm{x}, Y) \to \exists \mathrm{x} \in \mathbb{N}, \forall \mathrm{y} \in E, \neg A_1(\mathrm{x},\mathrm{y})$$

We can prove $[C] \leftrightarrow [\bar{C}]$, and if we do so directly, then we encounter the following term of type $[C] \to [\bar{C}]$:

$$\lambda axy.\Big(C\lambda k.\Big(a\,\lambda e.\big(C\,\lambda z.\big(k\,(y\,e\,\lambda np.(z\,\lambda k.(k\,n\,p)))\big)\big)\,\lambda np.\big(x\,n\,\lambda ez.(z\,(p\,e))\big)\Big)\Big)$$

All four De Morgan laws that relate $\forall$ and $\exists$ are used in the proof. Among them, only the following principle is not intuitionistic, and is responsible for the two occurrences of C:

$$\neg\forall \mathrm{x} \in \mathbb{N}, P \to \exists \mathrm{y} \in \mathbb{N}, \neg P\,.$$

As a matter of fact, its proof skeleton is the following:

$$\boxed{\vdash \lambda xy.(C\,\lambda k.(x\,\lambda e.(C\,\lambda l.(k\,(y\,e\,l))))) : \neg\forall \mathrm{x} \in \mathbb{N}, P(\mathrm{x}) \to \exists \mathrm{x} \in \mathbb{N}, \neg P(\mathrm{x})}$$

[10] Formulae are extended with quantification on atoms ($\forall X\,N$), and the following two rules are added:

$$\frac{\Gamma \vdash t : N}{\Gamma \vdash t : \forall X\,N}\ (X \notin \mathbf{fv}(\Gamma)) \qquad \frac{\Gamma \vdash t : \forall X\,N}{\Gamma \vdash t : N[M/X\mathrm{x}_1 \ldots \mathrm{x}_n]}$$

Although it is an elementary classical tautology, the computational role of the term is not immediate, in particular due to the presence of two Cs.

In the above example, illegibility is an idiosyncrasy of the λC calculus that reflects the absence, as we will see, of an involution $\neg\neg N \simeq N$ and therefore of an isomorphism of types $\neg\forall x\, N \simeq \exists x\, \neg N$.

Formulae-as-types for an involutive negation

Girard gives an involutive interpretation of negation by letting it change the polarity of formulae [Gir91]. We show in the chapter how a notion of involutive negation inspired by Girard corresponds to the idea of exposing a high-level interface to captured stacks, as suggested by Felleisen [AH08, Note] and formalised by Clements [Cle06]. In our setting, this translates into the presence of a positive type $\sim A$ of *inspectable stacks* and constants D that provides an access to the components of these stacks. The type $\sim A$ is therefore distinct from the negative type $A \to \bot$ of continuations. The distinction only makes sense in a setting where both polarities are taken into account.

Polarisation explains a technique of Krivine that alleviates the complexity of reasoning in the λC calculus by allowing certain pseudo-types of a positive tinge to the left-hand side of implications [Kri09, Kri08]. Polarisation makes such types first class, because it gives a meaning also for when they are on the right-hand side of implications. In addition, the most important pseudo-type in Krivine's work is $\mathcal{X}^-$, defined as the set $\{\mathsf{k}_\pi \mid \pi \in \mathcal{X}\}$. This amounts to distinguishing captured stacks from continuations, as we do in a direct manner.[11]

In the chapter, control delimiters provide a constructive interpretation for the unit $\bot$, expanding the proof theoretic interpretation of delimited control of Herbelin and others [HG08, AHS09, Her10,

[11] Note also that Krivine's setting is not constructive in our sense: there is an external step involved in turning realisers into algorithms (Krivine [Kri09], see also Miquel [Miq09]). Polarisation removes the external step, since terms that compute witnesses can be identified with realisers of a positive type.

	Fig.	Style	Strategy	$A \simeq \neg\neg A$	Red. & Eq.	
λC	I.12	Nat. ded.	By name	No	$\succ_n, \approx_n$	[HS02]
$\lambda\ell$	IV.1	Nat. ded.	Polarised	Yes	$\succ_p, \approx_p$	
$\mathbf{L}_{\mathrm{n}}$	I.10	Seq. calc.	By name	No	$\triangleright_{\mathrm{R}_n}, \simeq_{\mathrm{RE}_n}$	[CH00]
$\mathbf{L}_{\mathrm{pol},\hat{\mathrm{tp}}^{\ominus}}$	IV.4	Seq. calc.	Polarised	Yes	$\triangleright_{\mathrm{R}_p}, \simeq_{\mathrm{RE}_p}$	

Table I.18: A comparison of calculi appearing in Chapter IV

Ili10] which is meant to generalise the principle of Friedman's A-translation [Fri78]. Thus our result shows how although delimited control, when typed without annotations, does not prove new formulae compared to non-delimited control, it does give better proofs from a constructive standpoint.

Contributions

In Chapter IV, we introduce the extensional and untyped calculi $\lambda\ell$ and $\mathbf{L}_{\mathrm{pol},\hat{\mathrm{tp}}^{\ominus}}$, in which the negation defined as follows is involutive:

$$\neg P \stackrel{\text{def}}{=} P \to \bot$$
$$\neg N \stackrel{\text{def}}{=} {\sim}N$$

The calculus $\lambda\ell$ provides the notion of program, or quasi-proof terms, and the calculus $\mathbf{L}_{\mathrm{pol},\hat{\mathrm{tp}}^{\ominus}}$ decomposes $\lambda\ell$ to allow extensional reasoning. The calculi contain De Groote-Saurin's $\Lambda\mu$ calculus [dG94, Sau05] and (a variant of) the operators $\mathit{shift}_0/\mathit{reset}_0$ [DF90, Sha07, MB12].

Historical remarks

The first non-trivial interpretation of an involutive negation was given by linear logic [Gir87]. Girard extended this interpretation to propositional classical sequent calculus with the logic **LC** [Gir91]. The logic and its double negation translations have been thoroughly studied [Mur92, LRS93, DJS95, DJS97, QTDF96, Lau02, LR03, LQTdF05, Lau05, Lau11]. We notably owe to Murthy [Mur92] an early attempt at explaining **LC** in terms closer to programming languages. Also, Danos, Joinet and Schellinx pro-

posed to reconstruct **LC** though a detailed analysis of cut elimination in **LK** based on a translation into linear logic (the system $\mathbf{LK}^{\eta}_{p}$, [DJS97]).

The involution brings along a duality between the positive and negative fragments of **LC**. It is now well-understood that the positive polarisation corresponds to a strict evaluation strategy whereas the negative polarisation corresponds to a lazy one [Mur92, DJS95, DJS97, CH00, Lau02, Lau11]. Thus, polarisation meets a line of research that sought, since Filinski's Masters thesis [Fil89], to relate continuations in computer science to categorical duality [Fil89, LRS93, Thi97, SR98, Lev99, CH00, Sel01, MT10].

The question of the involutive negation is part of this line of work too, as witnessed by Thielecke in the conclusion of his thesis [Thi97, p. 156]:

> *The duality aspects of CPS may become clearer if addressed in a setting where there actually is a* duality functor*.*

The possibility of understanding the involutive negation in a constructive context has always been discussed. Thus, according to Parigot [Par00]:

> *It is often claimed that computational interpretations of negation in classical logic should be involutive, that is $\neg\neg A = A$ should be realised at the computational level. It is even sometimes claimed that this is the distinguishing feature of classical logic. But the real computational effect of the involutive character is not clear.*

and, according to Laurent, Quatrini and Tortora de Falco [LQTdF05], speaking of the duality they rely on:

> *It is not completely clear whether or not classical negation should be involutive in such a strong sense.*

One criticism holds that identifying negation with duality has no clear meaning in terms of a connective. But, according to Girard [Gir91, p. 9]:

> *Our identification of $\neg\neg A$ with A forces negation to be involutive; but there are two obvious traps that one should avoid about this identification.*

> *Trap #1: the belief that it is enough to declare ¬¬A equal to A;* […]
>
> *Trap #2: the belief that this identification is essential to our approach;* […]

The interpretation of Chapter IV seems a late perspective on this remark.

Yet, negation already appears as a connective in the works of Murthy [Mur92] and Danos, Joinet and Schellinx [DJS97]. With hindsight, these works already implied that polarisation corresponds to an evaluation order that extends call by value and call by name. According to Laurent [Lau05, p. 22], however:

> *As shown by many works* [DJS95, DJS97, Lau02, Lev99] *deterministic classical systems can be classified into two categories: call-by-name and call-by-value systems.*

It is as if the novelty of polarisation as an evaluation strategy was overlooked, as a consequence of equating negation with duality. This most likely explains why, despite the idea being ambitious, the involutive negation was less endorsed than the duality it generalises.

Chapter II

Duploids: Models of a non-associative composition

We characterise the polarised evaluation order through a categorical structure where the hypothesis that composition is associative is relaxed. Duploid is the name of the structure, as a reference to Jean-Louis Loday's duplicial algebras.

The main result is a reflection $\mathcal{Adj} \to \mathcal{Dupl}$ where $\mathcal{Dupl}$ is a category of duploids and duploid functors, and $\mathcal{Adj}$ is the category of adjunctions and pseudo maps of adjunctions. The result suggests that the various biases in denotational semantics: indirect, call-by-value, call-by-name... are ways of hiding the fact that composition is not always associative.

Outline

In Section II.1, we introduce pre-duploids as categories where the associativity of composition is deficient. In Section II.2, we define duploids as pre-duploids with additional structure, and characterise this additional structure. The category $\mathcal{Dupl}$ of duploids is introduced. In Section II.3 we introduce the category $\mathcal{Adj}$ of adjunctions and *pseudo maps of adjunctions* and we prove that there is a reflection:

$$\mathcal{Dupl} \simeq \mathcal{Adj}_{eq} \triangleleft \mathcal{Adj}$$

where $\mathcal{A}dj_{eq}$ is the full subcategory of adjunctions that satisfy an equalising requirement.

II.1 Pre-duploids

We define pre-duploids, which are category-like structures whose objects have a polarity, and which miss associativity of composition when the middle map has polarity $+ \to \ominus$.

Definition II.1. A *pre-duploid* $\mathscr{D}$ is given by:

1. A set $|\mathscr{D}|$ of *objects* together with a *polarity mapping*:
$$\varpi : |\mathscr{D}| \to \{+, \ominus\}\,.$$
2. For all $A, B \in |\mathscr{D}|$, a set of *morphisms* or *hom-set* $\mathscr{D}(A, B)$.
3. For all morphisms $f \in \mathscr{D}(A, B)$ and $g \in \mathscr{D}(B, C)$, a morphism $g \circledcirc f \in \mathscr{D}(A, C)$, also written as follows depending on the polarity of B:
$$g \bullet f \in \mathscr{D}(A, C) \text{ if } \varpi(B) = +\,,$$
$$g \circ f \in \mathscr{D}(A, C) \text{ if } \varpi(B) = \ominus\,.$$
The following associativities must hold for all objects $A, B \in |\mathscr{D}|$; $P, Q \in \varpi^{-1}(\{+\})$ and $N, M \in \varpi^{-1}(\{\ominus\})$:
   - **(••)** For all $A \xrightarrow{f} P \xrightarrow{g} Q \xrightarrow{h} B$, one has $(h \bullet g) \bullet f = h \bullet (g \bullet f)$;
   - **(∘∘)** For all $A \xrightarrow{f} N \xrightarrow{g} M \xrightarrow{h} B$, one has $(h \circ g) \circ f = h \circ (g \circ f)$;
   - **(•∘)** For all $A \xrightarrow{f} N \xrightarrow{g} P \xrightarrow{h} B$, one has $(h \bullet g) \circ f = h \bullet (g \circ f)$.
4. For all $A \in |\mathscr{D}|$, a morphism $\mathrm{id}_A \in \mathscr{D}(A, A)$ neutral for $\circledcirc$.

The mapping ϖ defines a partition of $|\mathscr{D}|$ into the *positive* objects P, Q... in $|\mathscr{P}| \stackrel{\text{def}}{=} \varpi^{-1}(\{+\})$ and the *negative* objects N, M... in

$|\mathscr{N}| \stackrel{\text{def}}{=} \varpi^{-1}(\{\ominus\})$. This partition defines categories $\mathscr{P}$ (whose composition is given by $\bullet$) and $\mathscr{N}$ (whose composition is given by $\circ$) in an obvious way.

II.1.1 Linear and thunkable morphisms

Definition II.2. Let $\mathscr{D}$ be a pre-duploid. A morphism f of $\mathscr{D}$ is *linear* if for all g, h one has:

$$f \circ (g \circ h) = (f \circ g) \circ h\,.$$

A morphism f of $\mathscr{D}$ is *thunkable*[1] if for all g, h one has:

$$h \circ (g \circ f) = (h \circ g) \circ f\,.$$

Notice that by definition any morphism $f : P \to A$ is linear, and any morphism $f : A \to N$ is thunkable. Notice also that these notions are closed under composition and identity. Indeed, if f_1 and f_2 are linear then for all g, h one has:

$$\begin{aligned}
\underline{(f_1 \circ f_2) \circ (g \circ h)} &= f_1 \circ \underline{(f_2 \circ (g \circ h))} && f_1 \text{ linear} \\
&= \underline{f_1 \circ ((f_2 \circ g) \circ h)} && f_2 \text{ linear} \\
&= \underline{(f_1 \circ (f_2 \circ g))} \circ h && f_1 \text{ linear} \\
&= ((f_1 \circ f_2) \circ g) \circ h && f_1 \text{ linear}
\end{aligned}$$

So we may define sub-categories of $\mathscr{D}$ as we do below.

Definition II.3 ($\mathscr{D}_l, \mathscr{D}_t, \mathscr{N}^\bullet, \mathscr{P}^\bullet$)**.** We call $\mathscr{D}_l$ the *category of linear morphisms of* $\mathscr{D}$; i.e the sub-category of $\mathscr{D}$ with all objects and the morphisms that are linear. We call $\mathscr{D}_t$ the *category of thunkable morphisms of* $\mathscr{D}$.

Observe that $\mathscr{N}$ is the full sub-category of $\mathscr{D}_t$ with negative objects, because any $A \to N$ is thunkable. Symmetrically, $\mathscr{P}$ is the full

[1] Terminology adapted from Thielecke [Thi97], Führmann [Füh99].

sub-category of $\mathscr{D}_l$ whose objects are positive, because any $P \to B$ is linear. We define accordingly:

- The category $\mathscr{N}^\bullet$ of negative linear maps as the full subcategory of $\mathscr{D}_l$ whose objects are negative; or equivalently the subcategory of $\mathscr{N}$ whose morphisms are linear.
- The category $\mathscr{P}^\bullet$ of positive thunkable maps as the full subcategory of $\mathscr{D}_t$ whose objects are positive; or equivalently the subcategory of $\mathscr{P}$ whose morphisms are thunkable.

Proposition II.4. *The hom-sets $\mathscr{D}(A,B)$ of a pre-duploid $\mathscr{D}$ extend to a (pro-)functor:*

$$\boxed{\mathscr{D}(-,=) : \mathscr{D}_t{}^{\mathrm{op}} \times \mathscr{D}_l \to \textbf{\textit{Set}}}$$

defined for $f \in \mathscr{D}_t(A,B)$ and $g \in \mathscr{D}_l(C,D)$ with:

$$\begin{aligned} \mathscr{D}(f,g) : \quad \mathscr{D}(B,C) &\to \mathscr{D}(A,D) \\ h &\mapsto g \circ h \circ f \,. \end{aligned}$$

Proof. Restricting to f thunkable and g linear makes the definition unambiguous. Functoriality corresponds to the following two equations:

$$\begin{aligned} \mathscr{D}(\mathrm{id}_A, \mathrm{id}_B)(h) &= h \\ \mathscr{D}(f_2 \circ f_1, g_1 \circ g_2)(h) &= ((g_1 \circ g_2) \circ h) \circ (f_2 \circ f_1) \\ &= ((g_1 \circ g_2) \circ h \circ f_2) \circ f_1 \\ &= (g_1 \circ (g_2 \circ h \circ f_2)) \circ f_1 \\ &= (\mathscr{D}(f_1, g_1) \circ \mathscr{D}(f_2, g_2))(h) \end{aligned}$$

The second equation uses the fact that f_1 and f_2 are thunkable and that g_1 and g_2 are linear. ∎

II.1.2 Examples of pre-duploids

Girard's classical logic

Girard's correlation spaces are a denotational semantics for classical logic. They do not form a category for lack of associativity of the composition [Gir91, LQTdF05]. However, they form a pre-duploid. We will see that duploids are motivated by Girard's translation of classical logic into intuitionistic logic.

Blass games

Blass [Bla92] gives a game semantics for linear logic, which did not satisfy the associativity of composition. Thanks to Abramsky's analysis of this issue [Abr03], we know that associativity fails due to composites of the form $N \to P \to M \to Q$. According to Abramsky, "*none of the other 15 polarisations give rise to a similar problem*". Therefore, Abramsky's formalisation of Blass games yields a pre-duploid. Thanks to Melliès's analysis of this so-called "*Blass problem*" [Mel05], we know that the phenomenon is essentially the same as for Girard's classical logic.

Direct models of call by value

Führmann [Füh99] characterises the Kleisli category of a monad *via* the presence of a structure called *thunk*. In the contexts of models of call by value, the thunk implements laziness.

Recall that a *thunk-force category* is a category $(\mathscr{P}, \bullet, \mathrm{id})$ together with a thunk $(L, \varepsilon, \vartheta)$ as defined next.

Definition II.5. A *thunk* on $\mathscr{P}$ is given by a functor $L : \mathscr{P} \to \mathscr{P}$ together with a natural transformation $\varepsilon : L \dot{\to} 1$ and a transformation $\vartheta : 1 \to L$ such that the transformation $\vartheta_L : L \to L^2$ is natural; satisfying the following equations:

$$\varepsilon \bullet \vartheta = \mathrm{id}$$
$$L\varepsilon \bullet \vartheta_L = \mathrm{id}_L$$
$$\vartheta_L \bullet \vartheta = L\vartheta \bullet \vartheta$$

The transformation ϑ is not necessarily natural, so that for $f \in \mathscr{P}(A,B)$ we may have:

$$Lf \bullet \vartheta_A \neq \vartheta_B \bullet f\,.$$

A thunk is therefore a comonad for which any morphism has a co-extension, with co-multiplication given by ϑ_L. The co-extension of f is ${}^*f \stackrel{\text{def}}{=} Lf \bullet \vartheta_A$.

Proposition II.6. *Let $(\mathscr{P}, \bullet, \mathrm{id}, L, \vartheta, \varepsilon)$ be a thunk-force category. The following defines a pre-duploid.*

$|\mathscr{D}|$ is the following disjoint union for a suitably chosen bijection $\Uparrow$ with domain $|\mathscr{P}|$:

$$|\mathscr{D}| = |\mathscr{P}| \uplus \Uparrow|\mathscr{P}|\,.$$

Let $\omega(P) = +$ and $\omega(\Uparrow P) = \ominus$ (in other words $|\mathscr{N}| = \Uparrow|\mathscr{P}|$). Now we define:

$$\boxed{\begin{array}{r} \mathscr{D}(P,Q) \stackrel{\text{def}}{=} \mathscr{P}(P,Q) \\ \mathscr{D}(P,\Uparrow Q) \stackrel{\text{def}}{=} \mathscr{P}(P,Q) \\ \mathscr{D}(\Uparrow P,Q) \stackrel{\text{def}}{=} \mathscr{P}(LP,Q) \\ \mathscr{D}(\Uparrow P,\Uparrow Q) \stackrel{\text{def}}{=} \mathscr{P}(LP,Q) \end{array}}\,.$$

Composition $\bullet$ and identities id_P are obtained from $\mathscr{P}$. Composition $\circ$ and identities id_N are defined with:

$$\mathrm{id}^{\mathscr{D}}_{\Uparrow P} \stackrel{\text{def}}{=} \varepsilon_P$$

and for all $A \xrightarrow{g} \Uparrow P \xrightarrow{f} B$:

$$f \circ g \stackrel{\text{def}}{=} f \bullet Lg \bullet \vartheta_A\,.$$

Proof. We indeed have $\circ\circ$-associativity for any $A \xrightarrow{h} N \xrightarrow{g} M \xrightarrow{f} B$:

$$\begin{aligned} f \circ (g \circ h) &= f \bullet Lg \bullet LLh \bullet \underline{L\vartheta_N \bullet \vartheta_A} \\ &= f \bullet Lg \bullet \underline{LLh \bullet \vartheta_{LN}} \bullet \vartheta_A \end{aligned}$$

$$= f \bullet Lg \bullet \vartheta_N \bullet Lh \bullet \vartheta_A$$
$$= (f \circ g) \circ h$$

We also have $\bullet\circ$-associativity as a direct consequence of $\bullet\bullet$-associativity. We also have that id is neutral for $\circ$:

$$f \circ \mathrm{id}_{\Uparrow P} = f \bullet L\varepsilon_P \bullet \vartheta_{LP} = f$$
$$\mathrm{id}_{\Uparrow P} \circ f = \underline{\varepsilon_P \bullet Lf} \bullet \vartheta_A = f \bullet \underline{\varepsilon_A \bullet \vartheta_A} = f\,. \qquad \blacksquare$$

In the context of λ_C models, this pre-duploid construction formalises how thunks implement laziness in call by value. In particular, this defines $\mathcal{N}$ as the Kleisli category of the thunk L seen as a comonad. This Kleisli construction is the categorical counterpart of Hatcliff and Danvy's *thunking translation* [HD97] that embeds call by name into call by value.

Proposition 11.7. *A morphism $f : P \to Q$ is thunkable in the sense of thunk-force categories if and only if it is thunkable in the sense of pre-duploids.*

Proof. Let us recall that $f \in \mathcal{P}(P, Q)$ is thunkable in the sense of Führmann [Füh99] if we have:

$$Lf \bullet \vartheta_P = \vartheta_Q \bullet f\,.$$

We have for any $A \xrightarrow{f} P \xrightarrow{g} M \xrightarrow{h} B$:

$$h \circ (g \bullet f) = h \bullet \underline{L(g \bullet f)} \bullet \vartheta_A$$
$$= h \bullet Lg \bullet Lf \bullet \vartheta_A$$
$$(h \circ g) \bullet f = h \bullet Lg \bullet \vartheta_A \bullet f$$

and f is thunkable if and only if the two are equal.

It is clear that the two notions of thunkability coincide. $\blacksquare$

Corollary 11.8. *In the above context, the transformation ϑ is natural if and only if the pre-duploid is a category* (i.e. *statisfies $\bullet\bullet$-associativity).*

Direct models of call by name

The concept dual to Führmann's thunk is the one of *runnable monad*[2]. Runnable monads implement strictness in call by name. We may dualise the result of Führmann: categories with a runnable monad are Kleisli categories of co-monads. The result is specialised by Selinger: control categories (direct models of the call-by-name $\lambda\mu$ calculus) are characterised as the Kleisli category of the double-negation co-monad on the dual of a response category, in the terminology of [Sel01].

Definition 11.9. A *runnable monad* on a category $\mathscr{C}$ is given by a functor $T : \mathscr{C} \to \mathscr{C}$ together with a natural transformation $\eta : 1 \dot{\to} T$ and a transformation $\rho : T \to 1$ such that the transformation $\rho_T : T^2 \to T$ is natural; satisfying the equations $\rho \circ \eta = \mathrm{id}$; $\rho_T \circ T\eta = \mathrm{id}_T$ and $\rho \circ T\rho = \rho \circ \rho_T$.

For any $f \in \mathscr{C}(A, B)$, we may define the *extension* of f as:

$$f^* = \rho_B \circ Tf\,.$$

Let us comment on the computational significance of the structure. Models of the pure lambda-calculus are obtained through the Kleisli construction of the co-monad ! of linear logic [Gir87]. In most models, it is false that the induced runnable monad is trivial, that is to say, enjoys that the transformation ρ is natural. (We shall see later that this condition corresponds to the co-monad being idempotent.) The computational intuition of the runnable monad extending the pure λ calculus is the following: we are given a constant η, together with a term constructor $\cdot^*$. They simulate call by value as follows: when t^* is applied, the argument is considered in head position until a term of the form $\eta\, u$ is reached. Then the evaluation proceeds with $t\, u$.

[2] Terminology is inspired from Erwig and Ren [ER04], who define, for the purpose of studying program transformations, runnable monads as (programmer's) monads enriched with a polymorphic "run" operation $TA \to A$ with equation $\rho_A(\eta_A(x)) = x$.

We may define, given a runnable monad, a pre-duploid with a bijective map $\Downarrow : |\mathscr{N}| \to |\mathscr{P}|$. The construction is obtained by symmetry from the one of thunk-force categories. In particular the positive sub-category is the Kleisli of the runnable monad.

Remark In the context of models coming from linear logic, the Kleisli construction of the runnable monad gives a translation from call by value to linear logic. It coincides with the translation first noticed by Girard [Gir87] (where it is called "*boring*"). The "*boring*" translation was recognised as a monadic model of call by value by Maraist *et al.* and Benton and Wadler [MOTW94, BW95]. The "*boring*" translation, however, only uses the monad part and not *run* maps ρ_N for $N \neq TM$. It seems that only now we recognise that the translation comes from an implementation of strictness in call by name, symmetric to the way we are used to implementing laziness in call by value with thunks.

II.1.3 Syntactic pre-duploid

We give an example which provides the following intuition from programming languages: positive types are evaluated eagerly, while negative types are evaluated lazily. In particular, $f \circ g$ evaluates f before g, whereas $f \bullet g$ evaluates g before f. Thus, we may write unambiguously $f \bullet g \circ h$, which means that g computes first, and subsequently determines if and when f and h are evaluated. But $(f \circ g) \bullet h$ can be distinct from $f \circ (g \bullet h)$, and then parentheses determine whether h or f computes first.

We may observe this phenomenon in the ML and Haskell programming languages. In ML, we may define a lazy application $t \mathbin{@_{\mathrm{n}}} u$ with $t\ (\lambda() \to u)$. Strict application $@_{\mathrm{v}}$ is the usual application. In Haskell, we have for $@_{\mathrm{v}}$ the strict application \$! and we may take the usual application for $@_{\mathrm{n}}$. Lack of associativity corresponds to the distinction between the programs:

$$(\lambda y.h \mathbin{@_{\mathrm{n}}} (g\ y)) \mathbin{@_{\mathrm{v}}} (f\ x)$$

and:

$$h\,@_{\mathrm{n}}\,(g\,@_{\mathrm{v}}\,(f\,x))\,.$$

In both languages, we witness the difference when f is a function that loops and h is a function that immediately returns: the program above loops while the one below immediately returns.

The syntactic pre-duploid formalises the intuition with abstract machines (pairs $\langle\, t \parallel e \,\rangle$ of a term and an evaluation context) whose components t and e are defined abstractly by their transition rules, via the binders μ and $\tilde{\mu}$. Let us consider four sets of variables written $x^+, \alpha^+, x^\ominus, \alpha^\ominus$, and a grammar of the following shape:

$$\begin{aligned}
t_+ &::= V_+ \mid \mu\alpha^+.c \mid \dots \\
V_+ &::= x^+ \mid \dots \\
t_\ominus &::= x^\ominus \mid \mu\alpha^\ominus.c \mid \dots \\
e_+ &::= \alpha^+ \mid \tilde{\mu}x^+.c \mid \dots \\
e_\ominus &::= \pi \mid \tilde{\mu}x^\ominus.c \mid \dots \\
\pi_\ominus &::= \alpha^\ominus \mid \dots \\
c &::= \langle t_+ \parallel e_+ \rangle \mid \langle t_\ominus \parallel e_\ominus \rangle \mid \dots
\end{aligned}$$

We use the notation t for either t_+ or $t_\ominus$ and refer to $+/\ominus$ as its polarity, and so on.

We define the contextual equivalence relation $\simeq_{\mathrm{RE}}$ generated by the following rewrite rules:

$$\begin{array}{llll}
\langle \mu\alpha^\ominus.c \parallel \pi_\ominus \rangle & \rhd_{\mathrm{R}} & c[\pi_\ominus/\alpha^\ominus] & \qquad t_\ominus \rhd_{\mathrm{E}} \mu\alpha^\ominus.\langle t_\ominus \parallel \alpha^\ominus \rangle \\
\langle \mu\alpha^+.c \parallel e_+ \rangle & \rhd_{\mathrm{R}} & c[e_+/\alpha^+] & \qquad t_+ \rhd_{\mathrm{E}} \mu\alpha^+.\langle t_+ \parallel \alpha^+ \rangle \\
\langle V_+ \parallel \tilde{\mu}x^+.c \rangle & \rhd_{\mathrm{R}} & c[V/x^+] & \qquad e_+ \rhd_{\mathrm{E}} \tilde{\mu}x^+.\langle x^+ \parallel e_+ \rangle \\
\langle t_\ominus \parallel \tilde{\mu}x^\ominus.c \rangle & \rhd_{\mathrm{R}} & c[t_\ominus/x^\ominus] & \qquad e_\ominus \rhd_{\mathrm{E}} \tilde{\mu}x^\ominus.\langle x^\ominus \parallel e_\ominus \rangle
\end{array}$$

Let us define, whenever the polarities of t and x are the same, the

following term:

$$\mathsf{let}\, x\, \mathsf{be}\, t\, \mathsf{in}\, u \stackrel{\text{def}}{=} \mu\alpha.\langle t \,\|\, \tilde{\mu}x.\langle u \,\|\, \alpha\rangle\rangle \quad (\alpha \notin \mathbf{fv}(t,u))$$

The polarities of α and of "$\mathsf{let}\, x\, \mathsf{be}\, t\, \mathsf{in}\, u$" are the same as the polarity of u. The definition is designed to satisfy, for any α that has the polarity of u:

$$\langle \mathsf{let}\, x^+\, \mathsf{be}\, V_+\, \mathsf{in}\, u \,\|\, \alpha\rangle \rhd_{\mathrm{R}}^* \langle u[V_+/x^+] \,\|\, \alpha\rangle$$
$$\langle \mathsf{let}\, x^\ominus\, \mathsf{be}\, t_\ominus\, \mathsf{in}\, u \,\|\, \alpha\rangle \rhd_{\mathrm{R}}^* \langle u[t_\ominus/x^\ominus] \,\|\, \alpha\rangle$$

which may be abbreviated as:

$$\langle \mathsf{let}\, x\, \mathsf{be}\, V\, \mathsf{in}\, u \,\|\, \alpha\rangle \rhd_{\mathrm{R}}^* \langle u[V/x] \,\|\, \alpha\rangle$$

with the convention that V is either V_+ or $t_\ominus$. It also satisfies:

$$\begin{aligned} \mathsf{let}\, x\, \mathsf{be}\, t\, \mathsf{in}\, x &\simeq_{\mathrm{RE}} t \\ \mathsf{let}\, y^+\, \mathsf{be}\, (\mathsf{let}\, x\, \mathsf{be}\, t\, \mathsf{in}\, u_+)\, \mathsf{in}\, v &\simeq_{\mathrm{RE}} \mathsf{let}\, x\, \mathsf{be}\, t\, \mathsf{in}\, \mathsf{let}\, y^+\, \mathsf{be}\, u_+\, \mathsf{in}\, v \\ \mathsf{let}\, y\, \mathsf{be}\, (\mathsf{let}\, x^\ominus\, \mathsf{be}\, t_\ominus\, \mathsf{in}\, u)\, \mathsf{in}\, v &\simeq_{\mathrm{RE}} \mathsf{let}\, x^\ominus\, \mathsf{be}\, t_\ominus\, \mathsf{in}\, \mathsf{let}\, y\, \mathsf{be}\, u\, \mathsf{in}\, v \end{aligned} \tag{II.1}$$

Proof. We have:

$$\begin{aligned} \mathsf{let}\, x\, \mathsf{be}\, V\, \mathsf{in}\, t = \mu\alpha.\langle V \,\|\, \underline{\tilde{\mu}x.\langle t \,\|\, \alpha\rangle}\rangle \\ \to_{\mathrm{R}} \underline{\mu\alpha.\langle t[V/x] \,\|\, \alpha\rangle} \\ \lhd_{\mathrm{E}} t[x/y] \quad \text{since } \alpha \notin \mathbf{fv}(t) \\ \mathsf{let}\, x\, \mathsf{be}\, t\, \mathsf{in}\, x = \mu\alpha.\langle t \,\|\, \underline{\tilde{\mu}x.\langle x \,\|\, \alpha\rangle}\rangle \\ \lhd_{\mathrm{E}} \underline{\mu\alpha.\langle t \,\|\, \alpha\rangle} \\ \lhd_{\mathrm{E}} t \quad \text{since } \alpha \notin \mathbf{fv}(t) \end{aligned}$$

and also:

$$\mathsf{let}\ x\ \mathsf{be}\ t\ \mathsf{in}\ \mathsf{let}\ y\ \mathsf{be}\ u\ \mathsf{in}\ v = \mu\alpha.\langle t \,\|\, \tilde{\mu}x.\langle u \,\|\, \tilde{\mu}y.\langle v \,\|\, \alpha\rangle\rangle\rangle \quad (y \notin \mathbf{fv}(v))$$

$$\mathsf{let}\ y\ \mathsf{be}\ (\mathsf{let}\ x\ \mathsf{be}\ t\ \mathsf{in}\ u)\ \mathsf{in}\ v = \mu\alpha.\langle \mu\beta.\langle t \,\|\, \tilde{\mu}x.\langle u \,\|\, \beta\rangle\rangle \,\|\, \tilde{\mu}y.\langle v \,\|\, \alpha\rangle\rangle$$

Therefore when $y = y^+$(is positive), we have:

$$\begin{aligned}
&\mathsf{let}\ y^+\ \mathsf{be}\ (\mathsf{let}\ x\ \mathsf{be}\ t\ \mathsf{in}\ u_+)\ \mathsf{in}\ v \\
&= \mu\alpha.\langle \underline{\mu\beta^+.\langle t \,\|\, \tilde{\mu}x.\langle u_+ \,\|\, \beta^+\rangle\rangle} \,\|\, \tilde{\mu}y^+.\langle v \,\|\, \alpha\rangle\rangle \\
&\to_{\mathrm{R}} \mu\alpha.\langle t \,\|\, \tilde{\mu}x.\langle u_+ \,\|\, \tilde{\mu}y^+.\langle v \,\|\, \alpha\rangle\rangle\rangle \text{ since } x \notin \mathbf{fv}(v) \\
&= \mathsf{let}\ x\ \mathsf{be}\ t\ \mathsf{in}\ \mathsf{let}\ y^+\ \mathsf{be}\ u_+\ \mathsf{in}\ v
\end{aligned}$$

and when $x = x^\ominus$ (is negative), we have:

$$\begin{aligned}
&\mathsf{let}\ y\ \mathsf{be}\ (\mathsf{let}\ x^\ominus\ \mathsf{be}\ t_\ominus\ \mathsf{in}\ u)\ \mathsf{in}\ v \\
&= \mu\alpha.\underline{\langle \mu\beta.\langle t_\ominus \,\|\, \tilde{\mu}x^\ominus.\langle u \,\|\, \beta\rangle\rangle \,\|\, \tilde{\mu}y.\langle v \,\|\, \alpha\rangle\rangle} \\
&\leftarrow_{\mathrm{R}} \mu\alpha.\langle t_\ominus \,\|\, \tilde{\mu}x^\ominus.\langle \mu\beta.\langle x^\ominus \,\|\, \underline{\tilde{\mu}x^\ominus}.\langle u \,\|\, \beta\rangle\rangle \,\|\, \tilde{\mu}y.\langle v \,\|\, \alpha\rangle\rangle\rangle \\
&\quad \text{since } \beta \notin \mathbf{fv}(t) \text{ and } x^\ominus \notin \mathbf{fv}(v) \\
&\to_{\mathrm{R}} \mu\alpha.\langle t_\ominus \,\|\, \tilde{\mu}x^\ominus.\langle \underline{\mu\beta.\langle u \,\|\, \beta\rangle} \,\|\, \tilde{\mu}y.\langle v \,\|\, \alpha\rangle\rangle\rangle \\
&\leftarrow_{\mathrm{E}} \mu\alpha.\langle t_\ominus \,\|\, \tilde{\mu}x^\ominus.\langle u \,\|\, \tilde{\mu}y.\langle v \,\|\, \alpha\rangle\rangle\rangle \text{ since } \beta \notin \mathbf{fv}(u) \\
&= \mathsf{let}\ x^\ominus\ \mathsf{be}\ t_\ominus\ \mathsf{in}\ \mathsf{let}\ y\ \mathsf{be}\ u\ \mathsf{in}\ v
\end{aligned}$$

∎

The reduction semantics of a term t is given by the relation $\triangleright_{\mathrm{R}}$ applied to the command $\langle\ t \,\|\, \alpha\ \rangle$. This corresponds, for illustration purposes, to defining (the skeleton of) a contextual reduction

semantics, in the sense of Wright and Felleisen [WF92], as follows:

$$\begin{aligned} t_\ominus, u_\ominus &::= x^\ominus \mid \mathsf{let}\, x\, \mathsf{be}\, t\, \mathsf{in}\, u_\ominus \mid \dots \\ t_+, u_+ &::= V_+ \mid \mathsf{let}\, x\, \mathsf{be}\, t\, \mathsf{in}\, u_+ \mid \dots \\ t &::= t_+ \mid t_\ominus \\ V_+ &::= x^+ \mid \dots \\ V &::= V_+ \mid t_\ominus \\ E &::= [\,] \mid \mathsf{let}\, x^+\, \mathsf{be}\, E\, \mathsf{in}\, t \mid \dots \end{aligned}$$

$$\mathsf{let}\, x\, \mathsf{be}\, V\, \mathsf{in}\, u \rhd u[V/x] \qquad \frac{t \rhd u}{E[t] \rhd E[u]}$$

Proposition 11.10. *The following defines a pre-duploid.*

Objects: *$|\mathscr{D}|$ and $\mathscr{a}$ are defined with $|\mathscr{P}| = \{+\}$ and $|\mathscr{N}| = \{\ominus\}$ for two objects $+ \neq \ominus$.*

Morphisms: *An element $\hat{x}.t$ of $\mathscr{D}(\varepsilon, \varepsilon')$ is an equivalence class of the relation $\sim$ on pairs (x, t) where x is a variable of polarity ε and t a term of polarity ε', defined with:*

$$\hat{x}.u \sim \hat{y}.v \iff \forall z^\varepsilon \notin \mathbf{fv}(u, v), u[z^\varepsilon/x] \simeq_{\mathrm{RE}} v[z^\varepsilon/y]\,.$$

(In other words, $\hat{x}.t$ is t considered modulo $\simeq_{\mathrm{RE}}$ and modulo renaming of x.)

Identities: *id_ε is $\hat{x}.x$ where x has polarity ε.*

Composition: *The composite of $\hat{x}.t : A \to B$ and $\hat{y}.u : B \to C$ is given with:*

$$\hat{y}.u \circledast \hat{x}.t \stackrel{\text{def}}{=} \hat{x}.(\mathsf{let}\, y\, \mathsf{be}\, t\, \mathsf{in}\, u)\,.$$

(x is assumed to be fresh in u.)

Proof. Identity is neutral for $\circledast$ because for all morphisms $\hat{y}.t : A \to B$ the equations:

$$\hat{y}.t \circledast \mathrm{id}_A = \hat{y}.t$$
$$\mathrm{id}_B \circledast \hat{y}.t = \hat{y}.t$$

correspond respectively to:

$$\mathsf{let}\, y\, \mathsf{be}\, x\, \mathsf{in}\, t \simeq_{\mathrm{RE}} t[x/y]$$
$$\mathsf{let}\, x\, \mathsf{be}\, t\, \mathsf{in}\, x \simeq_{\mathrm{RE}} t$$

and we have $\hat{x}.t[x/y] = \hat{y}.t$.

For all $A \xrightarrow{\hat{x}.t} B \xrightarrow{\hat{y}.u} C \xrightarrow{\hat{z}.v} D$, double composition is written as follows:

$$(\hat{z}.v \circledcirc \hat{y}.u) \circledcirc \hat{x}.t = \hat{x}.(\mathsf{let}\, y\, \mathsf{be}\, t\, \mathsf{in}\, \mathsf{let}\, z\, \mathsf{be}\, u\, \mathsf{in}\, v) \quad (y \notin \mathbf{fv}(v))$$
$$\hat{z}.v \circledcirc (\hat{y}.u \circledcirc \hat{x}.t) = \hat{x}.(\mathsf{let}\, z\, \mathsf{be}\, (\mathsf{let}\, y\, \mathsf{be}\, t\, \mathsf{in}\, u)\, \mathsf{in}\, v)$$

Therefore we have:

$$(\widehat{z^+}.v \bullet \hat{y}.u_+) \circledcirc \hat{x}.t = \widehat{z^+}.v \bullet (\hat{y}.u_+ \circledcirc \hat{x}.t) \quad (C \text{ positive})$$
$$(\hat{z}.v \circledcirc \widehat{y^\ominus}.u) \circ \hat{x}.t_\ominus = \hat{z}.v \circledcirc (\widehat{y^\ominus}.u \circ \hat{x}.t_\ominus) \quad (B \text{ negative})$$

as a consequence of the equations (II.1). ∎

Thus we have defined a pre-duploid. However, it is not possible to rewrite the following term:

$$\mathsf{let}\, y^\ominus\, \mathsf{be}\, (\mathsf{let}\, x^+\, \mathsf{be}\, t_+\, \mathsf{in}\, u_\ominus)\, \mathsf{in}\, v$$

into the following:

$$\mathsf{let}\, x^+\, \mathsf{be}\, t_+\, \mathsf{in}\, \mathsf{let}\, y^\ominus\, \mathsf{be}\, u_\ominus\, \mathsf{in}\, v$$

In other words, without imposing additional equations, we have in general:

$$h \circ (g \bullet f) \neq (h \circ g) \bullet f\,.$$

II.2 Duploids

We now enrich pre-duploids with operators of polarity coercion $\Downarrow, \Uparrow$ called *shifts*.[3]

Definition II.11. A *duploid* is a pre-duploid $\mathscr{D}$ given with mappings:

$$\Downarrow : |\mathscr{N}| \to |\mathscr{P}|$$
$$\Uparrow : |\mathscr{P}| \to |\mathscr{N}|$$

together with, for all $P \in |\mathscr{P}|$ and $N \in |\mathscr{N}|$, morphisms:

$$\begin{array}{rcl} \mathsf{delay}_P & : & P \to \Uparrow P \\ \mathsf{force}_P & : & \Uparrow P \to P \\ \mathsf{wrap}_N & : & N \to \Downarrow N \\ \mathsf{unwrap}_N & : & \Downarrow N \to N \end{array}$$

They must satisfy the following identities:

$$\begin{array}{rcl} \mathsf{force}_P \circ (\mathsf{delay}_P \bullet f) = f \ (\forall f \in \mathscr{D}(A,P)) \\ (f \circ \mathsf{unwrap}_N) \bullet \mathsf{wrap}_N = f \ (\forall f \in \mathscr{D}(N,A)) \\ \mathsf{delay}_P \bullet \mathsf{force}_P = \mathrm{id}_{\Uparrow P} \\ \mathsf{wrap}_N \circ \mathsf{unwrap}_N = \mathrm{id}_{\Downarrow N} \end{array}$$

Now notice the following:

Proposition II.12. *For any N, wrap_N is thunkable. Dually, for any P, force_P is linear.*

Proof. For all g, h, one has:

$$\begin{aligned} \underline{h \circ (g \bullet \mathsf{wrap}_N)} &= (h \circ \underline{(g \bullet \mathsf{wrap}_N) \circ \mathsf{unwrap}_N}) \bullet \mathsf{wrap}_N \\ &= (h \circ (g \bullet \underline{\mathsf{wrap}_N \circ \mathsf{unwrap}_N})) \bullet \mathsf{wrap}_N \\ &= (h \circ g) \bullet \mathsf{wrap}_N \end{aligned}$$

[3] Our notation is reminiscent of Melliès [Mel05].

Hence wrap_N is linear. The other result follows by symmetry. ∎

Thus we have the following equivalent definition of a duploid:

Definition II.13. A *duploid* is a pre-duploid $\mathscr{D}$ given with mappings:

$$\Downarrow : |\mathscr{N}| \to |\mathscr{P}|$$
$$\Uparrow : |\mathscr{P}| \to |\mathscr{N}|$$

together with a family of invertible linear maps $\mathsf{force}_P : \Uparrow P \to P$ and a family of invertible thunkable maps $\mathsf{wrap}_N : N \to \Downarrow N$.

Indeed, notice that if we write $\mathsf{delay}_P = \mathsf{force}_P^{-1}$ and $\mathsf{unwrap}_N = \mathsf{wrap}_N^{-1}$, then we have:

$$\begin{aligned} \mathsf{force}_P \circ (\mathsf{delay}_P \bullet f) &= (\mathsf{force}_P \circ \mathsf{delay}_P) \bullet f && \text{since } \mathsf{force}_P \text{ is linear} \\ &= f \\ (f \circ \mathsf{unwrap}_N) \bullet \mathsf{wrap}_N &= f \circ (\mathsf{unwrap}_N \bullet \mathsf{wrap}_N) && \text{since } \mathsf{wrap}_P \text{ is thunkable} \\ &= f \end{aligned}$$

II.2.1 Syntactic duploid

We concluded in Section II.1 with a language for pre-duploids. We continue where we left, with hopes of providing computational motivations for the shifts.

The syntactic duploid $\mathbf{L}_{\mathrm{dup}}$ is defined in Figure II.1 on the facing page. It extends the syntactic pre-duploid with the following intuitions from programming: $\Uparrow P$ is a type of suspended strict computations, and $\Downarrow N$ is a type of lazy computations encapsulated into a value. Then $\mathsf{delay} \bullet f$ represents the suspended strict computation f and the inverse operation force triggers the evaluation of its argument (hence it is linear in its negative argument). The morphism $\mathsf{wrap} \circ f$ represents f encapsulated into a value (hence it is thunkable) and unwrap removes the encapsulation.

$$t_+ ::= V_+ \mid \mu\alpha^+.c \mid \dots \qquad e_+ ::= \alpha^+ \mid \tilde{\mu}x^+.c \mid \tilde{\mu}\{x^\ominus\}.c \mid \dots$$
$$t_\ominus ::= x^\ominus \mid \mu\alpha^\ominus.c \mid \mu\{\alpha^+\}.c \mid \dots \qquad e_\ominus ::= \pi \mid \tilde{\mu}x^\ominus.c \mid \dots$$
$$V_+ ::= x^+ \mid \{t_\ominus\} \mid \dots \qquad \pi_\ominus ::= \alpha^\ominus \mid \{e_+\} \mid \dots$$
$$V ::= V_+ \mid t_\ominus \qquad \pi ::= \pi_\ominus \mid e_+$$

(a) Terms and values **(b)** Contexts and stacks

$$c ::= \langle t_+ \| e_+ \rangle \mid \langle t_\ominus \| e_\ominus \rangle \mid \dots$$

(c) Commands

$$\langle \mu\alpha.c \| \pi \rangle \triangleright_R c[\pi/\alpha]$$
$$\langle V \| \tilde{\mu}x.c \rangle \triangleright_R c[V/x]$$
$$\langle \{t_\ominus\} \| \tilde{\mu}\{x^\ominus\}.c \rangle \triangleright_R c[t_\ominus/x^\ominus]$$
$$\langle \mu\{\alpha^+\}.c \| \{e_+\} \rangle \triangleright_R c[e_+/\alpha^+]$$

(d) Reductions

$$t \triangleright_E \mu\alpha.\langle t \| \alpha \rangle^+$$
$$e \triangleright_E \tilde{\mu}x.\langle x \| e \rangle^+$$
$$e_+ \triangleright_E \tilde{\mu}\{x^\ominus\}.\langle \{x^\ominus\} \| e_+ \rangle$$
$$t_\ominus \triangleright_E \mu\{\alpha^+\}.\langle t_\ominus \| \{\alpha^+\} \rangle$$

$^+$: for a fresh (co-)variable of the proper polarity.

(e) Expansions

Figure II.1: $\mathbf{L}_{\mathbf{dup}}$, the syntactic duploid

We consider a grammar of abstract machines that extends the one of Section II.1.3 as follows:[4]

$$V_+ ::= \dots \mid \{t_\ominus\} \mid \dots$$
$$t_\ominus ::= \dots \mid \mu\{\alpha^+\}.c \mid \dots$$
$$e_+ ::= \dots \mid \tilde{\mu}\{x^\ominus\}.c \mid \dots$$
$$\pi_\ominus ::= \dots \mid \{e_+\} \mid \dots$$

[4] Thanks to Pierre-Louis Curien for suggesting that the interpretation is after all equivalent to an earlier one.

We extend the rewriting relations $\rhd_{\mathrm{R}}$ and $\rhd_{\mathrm{E}}$ with:

$$\begin{array}{llll} \langle \{t_\ominus\} \| \tilde{\mu}\{x^\ominus\}.c \rangle & \rhd_{\mathrm{R}} & c[t_\ominus/x^\ominus] & e_+ \rhd_{\mathrm{E}} \tilde{\mu}\{x^\ominus\}.\langle \{x^\ominus\} \| e_+ \rangle \\ \langle \mu\{\alpha^+\}.c \| \{e_+\} \rangle & \rhd_{\mathrm{R}} & c[e_+/\alpha^+] & t_\ominus \rhd_{\mathrm{E}} \mu\{\alpha^+\}.\langle t_\ominus \| \{\alpha^+\} \rangle \end{array}$$

$(x^\ominus \in \mathbf{fv}(e_+)$ and $\alpha^+ \notin \mathbf{fv}(t_\ominus))$.

Let us define the following terms:

$$\begin{array}{lll} \mathsf{let}\,\{x^\ominus\}\,\mathsf{be}\,t_+\,\mathsf{in}\,u & \stackrel{\text{def}}{=} & \mu\alpha.\langle t_+ \| \tilde{\mu}\{x^\ominus\}.\langle u \| \alpha \rangle \rangle \quad (\bar{\omega}(\alpha) = \bar{\omega}(u)) \\ \mathsf{delay}(t_+) & \stackrel{\text{def}}{=} & \mu\{\alpha^+\}.\langle t_+ \| \alpha^+ \rangle \\ \mathsf{force}(t_\ominus) & \stackrel{\text{def}}{=} & \mu\alpha^+.\langle t_\ominus \| \{\alpha^+\} \rangle \end{array}$$

The binding $\mathsf{let}\,\{x^\ominus\}\,\mathsf{be}\,t_+\,\mathsf{in}\,u$ must be read as a pattern-matching: if t_+ evaluates to some value V_+, then V_+ is matched against the pattern $\{x^\ominus\}$. If the value is some wrapped negative term, it binds $x^\ominus$ to the unwrapped term and continues with u.

We may prove in particular:

$$\begin{array}{lll} \langle \mathsf{let}\,\{x^\ominus\}\,\mathsf{be}\,\{t_\ominus\}\,\mathsf{in}\,u \| \alpha \rangle \rhd_{\mathrm{R}}^* \langle u[t_\ominus/x^\ominus] \| \alpha \rangle & & \\ \mathsf{let}\,\{x^\ominus\}\,\mathsf{be}\,t_+\,\mathsf{in}\,\{x^\ominus\} \leftarrow_{\mathrm{E}}^* t_+ & & \text{if } x^\ominus \notin \mathbf{fv}(t_+) \\ \langle \mathsf{force}(\mathsf{delay}(t_+)) \| \alpha \rangle \rhd_{\mathrm{R}}^* t_+ & & \\ \mathsf{delay}(\mathsf{force}(t_\ominus)) \simeq_{\mathrm{RE}} t_\ominus & & \end{array}$$

The reduction relation $\rhd_{\mathrm{R}}$ applied to a command $\langle\, t \parallel \alpha \,\rangle$ corresponds to defining, for illustration purposes, a contextual reduction

semantics for the term t as follows:

$$\begin{aligned} t_\ominus, u_\ominus &::= x^\ominus \mid \mathsf{let}\, x\, \mathsf{be}\, t\, \mathsf{in}\, u_\ominus \mid \mathsf{let}\, \{x^\ominus\}\, \mathsf{be}\, t_+\, \mathsf{in}\, u_\ominus \mid \mathsf{delay}(t_+) \mid \dots \\ t_+, u_+ &::= V_+ \mid \mathsf{let}\, x\, \mathsf{be}\, t\, \mathsf{in}\, u_+ \mid \mathsf{let}\, \{x^\ominus\}\, \mathsf{be}\, t_+\, \mathsf{in}\, u_+ \mid \mathsf{force}(t_\ominus) \mid \dots \\ t &::= t_+ \mid t_\ominus \\ V_+ &::= x^+ \mid \{t_\ominus\} \mid \dots \\ V &::= V_+ \mid t_\ominus \\ E &::= [\,] \mid \mathsf{let}\, x^+\, \mathsf{be}\, E\, \mathsf{in}\, t \mid \mathsf{let}\, \{x^\ominus\}\, \mathsf{be}\, E\, \mathsf{in}\, u \mid \mathsf{force}(E) \mid \dots \end{aligned}$$

$$\begin{aligned} \mathsf{let}\, x\, \mathsf{be}\, V\, \mathsf{in}\, u &\rhd u[V/x] \\ \mathsf{let}\, \{x^\ominus\}\, \mathsf{be}\, \{t_\ominus\}\, \mathsf{in}\, u &\rhd u[t_\ominus/x^\ominus] \\ \mathsf{force}(\mathsf{delay}(t_+)) &\rhd t_+ \end{aligned} \qquad \frac{t \rhd u}{E[t] \rhd E[u]}$$

The pattern-matching notation suggests that shifts are in an adjunction $\Downarrow \dashv \Uparrow$. This structure of the shifts is described in Proposition II.2.4.

Proposition II.1.4. *The syntax defines a duploid as follows:*

- *The pre-duploid structure is defined as in Section II.1.3.*
- *We necessarily take* $\Downarrow\ominus \stackrel{\text{def}}{=} +$ *and* $\Uparrow+ \stackrel{\text{def}}{=} \ominus$*, together with:*

$$\begin{aligned} \mathsf{delay} : + \to \ominus &\stackrel{\text{def}}{=} \widehat{x^+}.(\mathsf{delay}\, x^+) \\ \mathsf{force} : \ominus \to + &\stackrel{\text{def}}{=} \widehat{x^\ominus}.(\mathsf{force}\, x^\ominus) \\ \mathsf{wrap} : \ominus \to + &\stackrel{\text{def}}{=} \widehat{x^\ominus}.\{x^\ominus\} \\ \mathsf{unwrap} : + \to \ominus &\stackrel{\text{def}}{=} \widehat{x^+}.(\mathsf{let}\, \{x^\ominus\}\, \mathsf{be}\, x_+\, \mathsf{in}\, x^\ominus) \end{aligned}$$

Proof. We have $\mathsf{force} \circ (\mathsf{delay} \bullet \hat{x}.t_+) = \hat{x}.t_+$ since it is the abstraction in x of

the following:

$$\begin{aligned}
&\mathsf{let}\ \underline{x^{\ominus}\ \mathsf{be}\ (\mathsf{let}\ y^{+}\ \mathsf{be}\ t_{+}\ \mathsf{in}\ \mathsf{delay}(y^{+}))}\ \mathsf{in}\ \mathsf{force}(x^{\ominus})\\
&\simeq_{\mathrm{RE}} \mathsf{force}(\mathsf{let}\ y^{+}\ \mathsf{be}\ t_{+}\ \mathsf{in}\ \mathsf{delay}(y^{+}))\\
&= \mu\alpha^{+}.\Big\langle \underline{\mu\beta^{\ominus}.\big\langle t_{+} \,\big\|\, \tilde{\mu}y^{+}.\langle \mu\{\alpha^{+}\}.\langle y^{+} \| \alpha^{+}\rangle \,\|\, \beta^{\ominus}\rangle\big\rangle} \,\Big\|\, \{\alpha^{+}\}\Big\rangle\\
&\to_{\mathrm{R}} \mu\alpha^{+}.\Big\langle t_{+} \,\Big\|\, \tilde{\mu}y^{+}.\langle \underline{\mu\{\alpha^{+}\}.\langle y^{+} \| \alpha^{+}\rangle} \,\|\, \{\alpha^{+}\}\rangle\Big\rangle\\
&\to_{\mathrm{R}} \mu\alpha^{+}.\big\langle t_{+} \,\big\|\, \underline{\tilde{\mu}y^{+}.\langle y^{+} \| \alpha^{+}\rangle}\big\rangle\\
&\leftarrow_{\mathrm{E}} \underline{\mu\alpha^{+}.\langle t_{+} \| \alpha^{+}\rangle}\\
&\leftarrow_{\mathrm{E}} t_{+}
\end{aligned}$$

We also have $(\widehat{x^{\ominus}}.t \circ \mathsf{unwrap}) \bullet \mathsf{wrap} = \widehat{x^{\ominus}}.t$ since it is the abstraction in $x^{\ominus}$ of the following:

$$\begin{aligned}
&\mathsf{let}\ \underline{x^{+}\ \mathsf{be}\ \{x^{\ominus}\}}\ \mathsf{in}\ \mathsf{let}\ x^{\ominus}\ \mathsf{be}\ (\mathsf{let}\ \{y^{\ominus}\}\ \mathsf{be}\ x^{+}\ \mathsf{in}\ y^{\ominus})\ \mathsf{in}\ t\\
&\triangleright^{*}_{\mathrm{R}} \mathsf{let}\ x^{\ominus}\ \mathsf{be}\ (\mathsf{let}\ \underline{\{y^{\ominus}\}\ \mathsf{be}\ \{x^{\ominus}\}}\ \mathsf{in}\ y^{\ominus})\ \mathsf{in}\ t\\
&\triangleright^{*}_{\mathrm{R}} \mathsf{let}\ x^{\ominus}\ \mathsf{be}\ x^{\ominus}\ \mathsf{in}\ t\\
&\simeq_{\mathrm{RE}} t
\end{aligned}$$

Last, we have regarding $\mathsf{delay} \bullet \mathsf{force}$ and $\mathsf{wrap} \circ \mathsf{unwrap}$:

$$\begin{aligned}
\mathsf{delay} \bullet \mathsf{force} &= \widehat{x^{\ominus}}.x^{\ominus}\\
\mathsf{wrap} \circ \mathsf{unwrap} &= \widehat{x^{+}}.x^{+}
\end{aligned}$$

It is established by abstracting over $x^{\ominus}$ and x^{+} in the following:

$$\begin{aligned}
&\mathsf{let}\ x^{+}\ \mathsf{be}\ \mathsf{force}(x^{\ominus})\ \mathsf{in}\ \mathsf{delay}(x^{+})\\
&= \underline{\mu\alpha^{\ominus}.\big\langle \mathsf{force}(x^{\ominus}) \,\big\|\, \tilde{\mu}x^{+}.\langle \mu\{\alpha^{+}\}.\langle x^{+} \| \alpha^{+}\rangle \,\|\, \alpha^{\ominus}\rangle\big\rangle}\\
&\triangleright_{\mathrm{E}} \mu\{\alpha^{+}\}.\Big\langle \underline{\mu\alpha^{\ominus}.\big\langle \mathsf{force}(x^{\ominus}) \,\big\|\, \tilde{\mu}x^{+}.\langle \mu\{\alpha^{+}\}.\langle t_{+} \| \alpha^{+}\rangle \,\|\, \alpha^{\ominus}\rangle\big\rangle} \,\Big\|\, \{\alpha^{+}\}\Big\rangle
\end{aligned}$$

$$
\begin{aligned}
&\to^*_{\mathrm{R}} \mu\{\alpha^+\}.\langle \mathsf{force}(x^\ominus) \,\|\, \underline{\tilde{\mu}x^+.\langle x^+ \| \alpha^+\rangle}\rangle \\
&\leftarrow_{\mathrm{E}} \mathsf{delay}(\mathsf{force}(x^\ominus)) \\
&\simeq_{\mathrm{RE}} x^\ominus ; \\
&\mathsf{let}\, x^\ominus \,\mathsf{be}\, \mathsf{let}\, \{y^\ominus\} \,\mathsf{be}\, x^+ \,\mathsf{in}\, y^\ominus \,\mathsf{in}\, \{x^\ominus\} \\
&= \mu\alpha^+.\langle \underline{\mu\alpha^\ominus.\langle x^+ \| \tilde{\mu}\{y^\ominus\}.\langle y^\ominus \| \alpha^\ominus\rangle\rangle} \,\|\, \tilde{\mu}x^\ominus.\langle \{x^\ominus\} \| \alpha^+\rangle\rangle \\
&\to^*_{\mathrm{R}} \mu\alpha^+.\langle x^+ \| \underline{\tilde{\mu}\{y^\ominus\}.\langle \{y^\ominus\} \| \alpha^+\rangle}\rangle \\
&\leftarrow^*_{\mathrm{E}} x^+
\end{aligned}
$$

Therefore we have defined a duploid. ∎

Last, one may ask if there is a sound translation from the above syntax to any duploid; in other words if we can reason about any duploid in the above syntax. This is answered positively in a short note of Curien and will likely appear somewhere.

II.2.2 Linear and thunkable morphisms in duploids

In duploids, we have the following useful characterisation of thunkable and linear morphisms.

Proposition II.15. *In a duploid $\mathscr{D}$, let $f \in \mathscr{D}(A, P)$. The following statements are equivalent:*

1. $(\mathsf{wrap}_{\Uparrow P} \circ \mathsf{delay}_P) \bullet f = \mathsf{wrap}_{\Uparrow P} \circ (\mathsf{delay}_P \bullet f)$;
2. $\forall h \in \mathscr{D}(\Uparrow P, B),\ (h \circ \mathsf{delay}_P) \bullet f = h \circ (\mathsf{delay}_P \bullet f)$;
3. *f is thunkable.*

Dually, let $f \in \mathscr{D}(N, B)$. The following statements are equivalent:

1. $f \circ (\mathsf{unwrap}_N \bullet \mathsf{force}_{\Downarrow N}) = (f \circ \mathsf{unwrap}_N) \bullet \mathsf{force}_{\Downarrow N}$;
2. $\forall h \in \mathscr{D}(A, \Downarrow N),\ f \circ (\mathsf{unwrap}_N \bullet h) = (f \circ \mathsf{unwrap}_N) \bullet h$.
3. *f is linear.*

Proof. We establish the equivalence between the first three statements; the argument for the last three is obtained by symmetry.
(1.⇒2.) We have for $h \in \mathscr{D}(\Uparrow P, A)$:

$$\begin{aligned}(h \circ \mathsf{delay}_P) \bullet f &= (h \circ \mathsf{unwrap}_{\Uparrow P}) \bullet \underline{(\mathsf{wrap}_{\Uparrow P} \circ \mathsf{delay}_P) \bullet f} \\ &= (h \circ \underline{\mathsf{unwrap}_{\Uparrow P}) \bullet \mathsf{wrap}_{\Uparrow P}} \circ (\mathsf{delay}_P \bullet f) \quad \text{(assuming 1.)} \\ &= h \circ (\mathsf{delay}_P \bullet f)\end{aligned}$$

(2.⇒3.) We have for $P \xrightarrow{g} N \xrightarrow{h} B$:

$$\begin{aligned}(h \circ \underline{g}) \bullet f &= \underline{(h \circ (g \bullet \mathsf{force}_P) \circ \mathsf{delay}_P) \bullet f} \\ &= h \circ \underline{(g \bullet \mathsf{force}_P) \circ (\mathsf{delay}_P \bullet f)} \quad \text{(assuming 2.)} \\ &= h \circ ((g \bullet \underline{\mathsf{force}_P \circ \mathsf{delay}_P}) \bullet f) \quad \text{(applying 2. again)} \\ &= h \circ (g \bullet f)\end{aligned}$$

(3.⇒1.) is immediate. ∎

Proposition II.16. *For any P, force_P is thunkable if and only if every morphism $f \in \mathscr{D}(A, P)$ is thunkable. Dually, for any N, wrap_N is linear if and only if every morphism $f \in \mathscr{D}(N, B)$ is linear.*

Proof. The argument for the second equivalence is obtained by symmetry from the first one. The implication (⇐) is trivial. Let $f \in \mathscr{D}(A, P)$. We have:

$$\begin{aligned}&(\mathsf{wrap}_{\Uparrow P} \circ \mathsf{delay}_P) \bullet \underline{f} \\ &= \underline{(\mathsf{wrap}_{\Uparrow P} \circ \mathsf{delay}_P) \bullet \mathsf{force}_P} \circ (\mathsf{delay}_P \bullet f) \\ &= \mathsf{wrap}_{\Uparrow P} \circ (\mathsf{delay}_P \bullet \underline{\mathsf{force}_P) \circ (\mathsf{delay}_P} \bullet f) \quad \text{(if } \mathsf{force}_P \text{ is thunkable)} \\ &= \mathsf{wrap}_{\Uparrow P} \circ (\mathsf{delay}_P \bullet f)\end{aligned}$$

Thus we proved (⇒). ∎

Corollary II.17. *A duploid is a category (by which we mean that ∘•-associativity holds) if and only if for all P, force_P is thunkable, or equivalently for all N, wrap_N is linear.*

II.2.3 The duploid construction

We build a duploid given two categories $\mathscr{C}_1$ and $\mathscr{C}_2$ and an adjunction $F \dashv_{(\sharp,\flat)} G : \mathscr{C}_1 \to \mathscr{C}_2$:

$$\mathscr{C}_1(F-,=) \underset{\flat}{\overset{\sharp}{\rightleftarrows}} \mathscr{C}_2(-,G=)\,.$$

Let us introduce the convention that objects of $\mathscr{C}_1$ are negative and written $N, M \ldots$, while the objects of $\mathscr{C}_2$ are positive and written $P, Q \ldots$

Proposition II.18. *Let an adjunction be given as above. The following defines a pre-duploid $\mathscr{D}$:*

$$\begin{aligned} |\mathscr{D}| &\overset{\text{def}}{=} |\mathscr{C}_1| \uplus |\mathscr{C}_2| \\ \mathfrak{a}(|\mathscr{C}_1|) &\overset{\text{def}}{=} \{\ominus\} \\ \mathfrak{a}(|\mathscr{C}_2|) &\overset{\text{def}}{=} \{+\} \end{aligned}$$

Hom-sets are given by:

$$\boxed{\mathscr{D}(A,B) \overset{\text{def}}{=} \mathscr{C}_1(FA^+, B^\ominus)}$$

where:

$$A^+ \overset{\text{def}}{=} \begin{cases} GA & \text{if } A \in |\mathscr{C}_1| \\ A & \text{if } A \in |\mathscr{C}_2| \end{cases}$$

$$A^\ominus \overset{\text{def}}{=} \begin{cases} A & \text{if } A \in |\mathscr{C}_1| \\ FA & \text{if } A \in |\mathscr{C}_2| \end{cases}$$

Identities are given by:

$$\boxed{\begin{aligned} \mathrm{id}_P^{\mathscr{D}} &\overset{\text{def}}{=} \mathrm{id}_{FP}^{\mathscr{C}_1} \in \mathscr{C}_1(FP, FP) \\ \mathrm{id}_N^{\mathscr{D}} &\overset{\text{def}}{=} \mathrm{id}_{GN}^{\mathscr{C}_2{}^\flat} \in \mathscr{C}_1(FGN, N) \end{aligned}}$$

Composition is given by the obvious compositions in $\mathscr{C}_1$ and $\mathscr{C}_2$:

- *If* $f \in \mathscr{D}(A,N) = \mathscr{C}_1(FA^+, N)$ *and* $g \in \mathscr{D}(N,B) = \mathscr{C}_1(FGN, B)$ *then:*
$$\boxed{g \circ^{\mathscr{D}} f \stackrel{\text{def}}{=} (g^\sharp \circ^{\mathscr{C}_2} f^\sharp)^\flat} ;$$
- *If* $f \in \mathscr{D}(A,P) = \mathscr{C}_1(FA^+, FP)$ *and* $g \in \mathscr{D}(P,B) = \mathscr{C}_1(FP, B)$ *then:*
$$\boxed{g \bullet^{\mathscr{D}} f = g \circ^{\mathscr{C}_1} f} ;$$

Proof. That id is neutral for the composition in $\mathscr{D}$ follows immediately from the definition.

Associativity is obtained as follows:

- Let $A \xrightarrow{f} P \xrightarrow{g} Q \xrightarrow{h} B$, that is, $f \in \mathscr{C}_1(FA^+, FP)$; $g \in \mathscr{C}_1(FP, FQ)$ and $h \in \mathscr{C}_1(FQ, B^\ominus)$. Associativity follows from the one of the composition in $\mathscr{C}_1$.
- Let $A \xrightarrow{f} N \xrightarrow{g} M \xrightarrow{h} B$, that is, $f \in \mathscr{C}_1(FA^+, N)$; $g \in \mathscr{C}_1(FGN, M)$ and $h \in \mathscr{C}_1(FGM, B^\ominus)$. Both composites are equal to $(h^\sharp \circ g^\sharp \circ f^\sharp)^\flat$.
- Let $A \xrightarrow{f} N \xrightarrow{g} P \xrightarrow{h} B$, that is, $f \in \mathscr{C}_1(FA^+, N)$; $g \in \mathscr{C}_1(FGN, FP)$ and $h \in \mathscr{C}_1(FP, B^\ominus)$. We have:
$$(h \bullet g) \circ f = ((h \circ g)^\sharp \circ f^\sharp)^\flat$$
$$h \bullet (g \circ f) = (h \circ (g^\sharp \circ f^\sharp)^\flat)$$
Naturality of $\sharp$ corresponds to the following equations for all $a : P \to Q$, $b : FQ \to N$, $c : N \to M$:
$$(c \circ b \circ Fa)^\sharp = Gc \circ b^\sharp \circ a$$
Thus we have in particular $(h \circ g)^\sharp = Gh \circ g^\sharp$ and $(g^\sharp \circ f^\sharp)^\flat = g \circ Ff^\sharp$, and:
$$(h \bullet g) \circ f = (Gh \circ g^\sharp \circ f^\sharp)^\flat$$
$$h \bullet (g \circ f) = (h \circ g \circ Ff^\sharp)$$
The equality of the two expressions on the right-hand-side is again an instance of the naturality of $\sharp$. ∎

Remark 11.19. In particular $\mathscr{P}$ is the Kleisli category $(\mathscr{C}_2)_{GF}$ of the monad GF and $\mathscr{N}$ is the Kleisli category $(\mathscr{C}_1)_{FG}$ of the co-monad FG.

Proposition 11.20. *Let an adjunction as above and let $\mathscr{D}$ be the associated pre-duploid. The following defines a duploid:*

$$\begin{aligned}
\Uparrow P &\stackrel{\text{def}}{=} FP \\
\Downarrow N &\stackrel{\text{def}}{=} GN \\
\mathscr{D}(P, \Uparrow P) \ni \mathsf{delay}_P &\stackrel{\text{def}}{=} \mathrm{id}^{\mathscr{C}_1}_{FP} \in \mathscr{C}_1(FP, FP) \\
\mathscr{D}(\Uparrow P, P) \ni \mathsf{force}_P &\stackrel{\text{def}}{=} (\mathrm{id}_{GFP})^\flat \in \mathscr{C}_1(FGFP, FP) \\
\mathscr{D}(N, \Downarrow N) \ni \mathsf{wrap}_N &\stackrel{\text{def}}{=} \mathrm{id}^{\mathscr{C}_1}_{FGN} \in \mathscr{C}_1(FGN, FGN) \\
\mathscr{D}(\Downarrow N, N) \ni \mathsf{unwrap}_N &\stackrel{\text{def}}{=} (\mathrm{id}_{GN})^\flat \in \mathscr{C}_1(FGN, N)
\end{aligned}$$

Any apparent lack of symmetry comes from the fact that one must choose arbitrarily between $\mathscr{C}_1(FA^+, B^\ominus)$ and $\mathscr{C}_2(A^+, GB^\ominus)$ in the definition of $\mathscr{D}(A, B)$.

Proof. We have the following identities:

- $\mathsf{delay}_P \bullet \mathsf{force}_P = \mathrm{id}^{\mathscr{C}_1}_{FP} \circ \varepsilon_{FP} = \varepsilon_{FP} = \mathrm{id}^{\mathscr{D}}_{\Uparrow P}$;
- $\mathsf{wrap}_N \circ \mathsf{unwrap}_N = (\mathrm{id}^{\mathscr{C}_1}_{FGN}{}^\sharp \circ \varepsilon^\sharp_N)^\flat = \mathrm{id}^{\mathscr{C}_1}_{FGN} = \mathrm{id}^{\mathscr{D}}_{\Downarrow N}$;
- For $f \in \mathscr{D}(N, A)$, one has $(f \circ \mathsf{unwrap}_N) \bullet \mathsf{wrap}_N = (f^\sharp \circ \varepsilon^\sharp_N)^\flat \circ \mathrm{id}^{\mathscr{C}_1}_{FGN} = f$;
- For $f \in \mathscr{D}(A, P)$, one has $\mathsf{force}_P \circ (\mathsf{delay}_P \bullet f) = (\varepsilon^\sharp_{FP} \circ (\mathrm{id}^{\mathscr{C}_1}_{FP} \bullet f)^\sharp)^\flat = f$. ∎

Proposition 11.21. *Let $F \dashv_{(\eta,\varepsilon)} G : \mathscr{C}_1 \to \mathscr{C}_2$ be an adjunction, and consider the associated duploid $\mathscr{D}$. Then $f \in \mathscr{D}(N, A)$ is linear if and only if:*

$$f \circ \varepsilon_{FGN} = f \circ FG\varepsilon_N \ (\text{in } \mathscr{C}_1),$$

and $f \in \mathscr{D}(A, P)$ is thunkable if and only if its transpose $f^\sharp$ satisfies:

$$\eta_{GFP} \circ f^\sharp = GF\eta_P \circ f^\sharp \ (\text{in } \mathscr{C}_2).$$

Proof. According to Proposition II.15, $f \in \mathscr{D}(N, A)$ is linear if and only if $f \circ (\mathsf{unwrap}_N \bullet \mathsf{force}_{\Downarrow N}) = (f \circ \mathsf{unwrap}_N) \bullet \mathsf{force}_{\Downarrow N}$. It is easy to see that the left-hand side corresponds to $f \circ \varepsilon_{FGN}$ while the right-hand side corresponds to $f \circ FG\varepsilon_N$. The reasoning is symmetric for f thunkable. ∎

Proposition II.22. *Let $F \dashv_{(\eta,\varepsilon)} G : \mathscr{C}_1 \to \mathscr{C}_2$. The associated duploid $\mathscr{D}$ is a category if and only if the adjunction is idempotent.*

Proof. Let us recall that by definition $F \dashv G$ is idempotent if and only if the multiplication of the associated monad is an isomorphism, or equivalently the co-multiplication of the associated co-monad is an isomorphism.[5] This is again equivalent to the equation $\varepsilon_{GF} = GF\varepsilon$ (and also to $\eta_{FG} = FG\eta$).

According to Corollary II.17, $\mathscr{D}$ is a category if and only if wrap_N is linear for all N. According to Proposition II.21, this is equivalent to $\varepsilon_{GFN} = GF\varepsilon_N$. ∎

II.2.4 Structure of shifts

As we have seen, the Kleisli category of a co-monad is described by a runnable monad; and the Klesli category of a monad is described by a thunk, which is a co-monad.

We observe a similar phenomenon with duploids. We show that there is an adjunction:

$$\boxed{\Downarrow \dashv \Uparrow : \mathscr{P} \to \mathscr{N}}\ .$$

Notice that the adjunction is reversed, in the sense that the right adjoint $\Uparrow$ is from positives to negatives, quite the converse of the adjunction considered last section.

Actually, we state below a wider adjunction. Remark that we may

[5] Terminology from the nLab.

extend the shifts $\Downarrow, \Uparrow$ to all objects in a straightforward manner:

$$\Downarrow A \stackrel{\text{def}}{=} \begin{cases} \Downarrow N & \text{if } A = N \\ P & \text{if } A = P \end{cases}$$

$$\Uparrow A \stackrel{\text{def}}{=} \begin{cases} N & \text{if } A = N \\ \Uparrow P & \text{if } A = P \end{cases}$$

$$\mathsf{delay}_N \stackrel{\text{def}}{=} \mathrm{id}_N : N \to \Uparrow N$$

$$\mathsf{force}_N \stackrel{\text{def}}{=} \mathrm{id}_N : \Uparrow N \to N$$

$$\mathsf{wrap}_P \stackrel{\text{def}}{=} \mathrm{id}_P : P \to \Downarrow P$$

$$\mathsf{unwrap}_P \stackrel{\text{def}}{=} \mathrm{id}_P : \Downarrow P \to P$$

By "extend", we mean that we have for all f, g:

$$\begin{aligned} (f \circledcirc \mathsf{force}_A) \circ (\mathsf{delay}_A \circledcirc g) &= f \circledcirc g \\ (f \circledcirc \mathsf{unwrap}_A) \bullet (\mathsf{wrap}_A \circledcirc g) &= f \circledcirc g \\ \mathsf{delay}_A \circledcirc \mathsf{force}_A &= \mathrm{id}_A \\ \mathsf{wrap}_A \circledcirc \mathsf{unwrap}_A &= \mathrm{id}_A \end{aligned}$$

Also, extending Proposition 11.12, we have, for all objects A, that unwrap_A and wrap_A are thunkable whereas delay_A and force_A are linear.

Proposition 11.23. *Let $\mathscr{D}$ be a duploid. Then the following:*

$$\begin{aligned} \Uparrow f &\stackrel{\text{def}}{=} \mathsf{delay}_B \circledcirc f \circledcirc \mathsf{force}_A \\ \Downarrow f &\stackrel{\text{def}}{=} \mathsf{wrap}_B \circledcirc f \circledcirc \mathsf{unwrap}_A \end{aligned}$$

define functors $\Uparrow : \mathscr{D}_l \to \mathscr{N}^\bullet$ and $\Downarrow : \mathscr{D}_t \to \mathscr{P}^\bullet$ that take part in adjoint equivalences of categories $I \dashv_{(\mathsf{delay},\mathsf{force})} \Uparrow : \mathscr{D}_l \to \mathscr{N}^\bullet$ and $I \dashv_{(\mathsf{wrap},\mathsf{unwrap})} \Downarrow : \mathscr{D}_t \to \mathscr{P}^\bullet$, where I denotes the inclusion functors.

Proof. We first prove that the statement defines functors $\Uparrow' : \mathscr{D}_l \to \mathscr{D}_l$ and $\Downarrow' : \mathscr{D}_t \to \mathscr{D}_t$ that are naturally isomorphic to the respective identity functors.

In a category $\mathscr{C}$, a functor $F : \mathscr{C} \to \mathscr{C}$ that is naturally isomorphic to the identity functor is given by $|F| : |\mathscr{C}| \to |\mathscr{C}|$ together with families of inverse morphisms $\tau_A : A \to FA$ and $\tau_A^{-1} : FA \to A$ that necessarily define $F(f : A \to B)$ as $\tau_B \circ f \circ \tau_A^{-1}$.

The maps delay and force are such families of inverse linear morphisms and wrap and unwrap are such families of inverse thunkable morphisms, so the statement defines functors and we have $\Uparrow' \simeq 1^{\mathscr{D}_l}$ and $\Downarrow' \simeq 1^{\mathscr{D}_t}$.

Now let us recall that $\mathscr{P}^\bullet$ (respectively $\mathscr{N}^\bullet$) is the full subcategory of $\mathscr{D}_t$ (resp. $\mathscr{D}_l$) whose objects are positive (resp. negative). Therefore, by co-restricting $\Uparrow', \Downarrow'$ we have functors $\Uparrow : \mathscr{D}_l \to \mathscr{N}^\bullet$ and $\Downarrow : \mathscr{D}_t \to \mathscr{P}^\bullet$; as well as $I\Uparrow \simeq 1^{\mathscr{D}_l}$ and $I\Downarrow \simeq 1^{\mathscr{D}_t}$ where I are the inferrable inclusion functors. Since the I are fully faithful, we deduce $\Uparrow I \simeq 1^{\mathscr{N}^\bullet}$ and $\Downarrow I \simeq 1^{\mathscr{P}^\bullet}$ from $I\Uparrow I \simeq I$ and $I\Downarrow I \simeq I$.

The adjunctions follow from $\mathsf{force}_I \circ I\mathsf{delay} = \mathrm{id}_I$ and $\mathsf{unwrap}_I \bullet I\mathsf{wrap} = \mathrm{id}$. ∎

The above is actually a characterisation of the duploid structure. Indeed, a pre-duploid $\mathscr{D}$ with functors $\Uparrow : \mathscr{D}_l \to \mathscr{D}_l$ and $\Downarrow : \mathscr{D}_t \to \mathscr{D}_t$ naturally isomorphic to the identity functor, *such that any object $\Uparrow A$ is negative and any object $\Downarrow A$ is positive,* has a duploid structure defined by the induced families of inverse morphisms.

Proposition II.24. *Let $\mathscr{D}$ be a duploid. We have natural isomorphisms between (pro-)functors $\mathscr{D}_t{}^{\mathrm{op}} \times \mathscr{D}_l \to$ **Set**:*

$$\boxed{\mathscr{D}_t(-, I\Uparrow =) \simeq \mathscr{D}(-,=) \simeq \mathscr{D}_l(I\Downarrow -, =)}$$

where I denotes the inferrable inclusion functors.

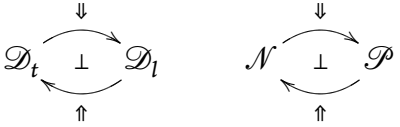
Proof. It follows from Proposition II.23 that we have natural isomorphisms between functors $\mathscr{D}_t{}^{\mathrm{op}} \times \mathscr{D}_l \to$ ***Set***:

$$\mathscr{D}(-, I\Uparrow =) \simeq \mathscr{D}(-,=) \simeq \mathscr{D}(I\Downarrow -, =)\,.$$

Now any morphism $A \to \Uparrow B$ is thunkable, in other words we have $\mathscr{D}(A, \Uparrow B) = \mathscr{D}_t(A, \Uparrow B)$. It follows that the functor $\mathscr{D}(-, I\Uparrow =)$ (with $I : \mathscr{N}^\bullet \to \mathscr{D}_l$) is equal to the functor $\mathscr{D}_t(-, I\Uparrow =)$ (with $I : \mathscr{N}^\bullet \to \mathscr{D}_t$).

For similar reasons, the functor $\mathscr{D}(I\Downarrow -, =)$ (with $I : \mathscr{P}^\bullet \to \mathscr{D}_l$) is equal to the functor $\mathscr{D}_l(I\Downarrow -, =)$ (with $I : \mathscr{P}^\bullet \to \mathscr{D}_t$). Hence the result. ∎

In particular, obviating the inclusion functors, we have the adjunctions:



Let us note that the bijections $\mathscr{D}(A, \Uparrow B) \simeq \mathscr{D}(\Downarrow A, B)$ are given with:

$\mathscr{D}_t(A, \Uparrow B)$		$\mathscr{D}_l(\Downarrow A, B)$
f	$\mapsto$	$(\mathsf{force}_B \circ f) \circledcirc \mathsf{unwrap}_A$
$\mathsf{delay}_B \circledcirc (g \bullet \mathsf{wrap}_A)$	$\mapsfrom$	g

Remark The adjunction $\Downarrow \dashv \Uparrow$ distinguishes our interpretation of polarities from ones based on shifts $\downarrow$ and $\uparrow$ that take part in an adjunction of the form $\uparrow \dashv \downarrow$ (see e.g. Laurent [Lau02], Zeilberger [Zei08]). With the latter, a morphism $A \to P$ usually represents a value and $N \to A$ usually represents a stack (or covalue); in other words it is a notion of polarisation tied to focusing or continuation-passing style. An adjunction $\uparrow \dashv \downarrow$ is the mark of an indirect model for which the interpretation of a language — involving usually at least a CPS translation — requires a development on its own.

Our notion of polarities adds a level of granularity: in our syntax, both a strict expression $A \to P$ and a suspended expression $A \to \Uparrow P$ translate into a morphism $A \to \uparrow P$ in the terminology of Laurent or Zeilberger. But a morphism $A \to \uparrow P$ may be composed in different ways that are determined by the polarities in our sense. In terms of continuations, our polarities distinguish continuations that are meant to be applied from continuations that are meant to be passed.

II.2.5 The category of duploids

Definition II.25. A *functor of pre-duploids* $F : \mathscr{D}_1 \to \mathscr{D}_2$ is given by a mapping on objects $|F| : |\mathscr{D}_1| \to |\mathscr{D}_2|$ that preserves polarities, together with mappings on morphisms $F_{A,B} : \mathscr{D}_1(A, B) \to \mathscr{D}_2(FA, FB)$, satisfying $F(\mathrm{id}_A) = \mathrm{id}_{FA}$ and $F(g \circ f) = Fg \circ Ff$.

A *functor of duploids* $F : \mathscr{D}_1 \to \mathscr{D}_2$ is a functor of pre-duploids such that $F(\mathsf{force}_P)$ is linear for all $P \in |\mathscr{P}_1|$, and $F(\mathsf{wrap}_N)$ is thunkable for all $N \in |\mathscr{N}_1|$.

Proposition II.26 (Characterisation)**.** *Let $\mathscr{D}$ and $\mathscr{D}'$ be two duploids and let $F : \mathscr{D} \to \mathscr{D}'$ be a mapping on objects $|F| : |\mathscr{D}| \to |\mathscr{D}'|$ that preserves polarities, together with mappings on morphisms $F_{A,B} : \mathscr{D}(A, B) \to \mathscr{D}'(FA, FB)$.*

The following three propositions are equivalent:

1. *F is a functor of duploids;*
2. *F is a functor of pre-duploids, and the full sub-pre-duploid $F\mathscr{D}$ of $\mathscr{D}'$ that has objects of the form FA for $A \in |\mathscr{D}|$ has a duploid structure, given by the maps $N \mapsto F{\Downarrow}N$ and $P \mapsto F{\Uparrow}P$ together with the families of morphisms $F\mathsf{delay}$, $F\mathsf{force}$, $F\mathsf{wrap}$ and $F\mathsf{unwrap}$;*
3. *F restricts to functors $F_t : \mathscr{D}_t \to \mathscr{D}'_t$ and $F_l : \mathscr{D}_l \to \mathscr{D}'_l$, such that the transformation $F : \mathscr{D}(-,=) \to \mathscr{D}'(F_t -, F_l =)$ is natural.*

Proof. (1.⇒2.) Suppose that $F : \mathscr{D} \to \mathscr{D}'$ is a functor of duploids. By pre-duploid functoriality, the families of maps $F\mathsf{delay}$ and $F\mathsf{force}$ are inverse of each other and so are $F\mathsf{wrap}$ and $F\mathsf{unwrap}$. Moreover, the following equations:

$$F\mathsf{force}_P \circ (F\mathsf{delay}_P \bullet f) = (F\mathsf{force}_P \circ F\mathsf{delay}_P) \bullet f = f$$
$$(f \circ F\mathsf{unwrap}_N) \bullet F\mathsf{wrap}_N = f \circ (F\mathsf{unwrap}_N \bullet F\mathsf{wrap}_N) = f$$

hold because $F(\mathsf{force}_P)$ is linear and $F(\mathsf{wrap}_N)$ is thunkable. This defines a duploid structure on $F\mathscr{D}$.

(2.⇒3.) Applying Proposition II.15 to the duploid $F\mathscr{D}$, we may characterise thunkability and linearity of morphisms $f : FA \to FB$ in $\mathscr{D}'$ in terms of $F\mathsf{delay}$, $F\mathsf{force}$, $F\mathsf{wrap}$ and $F\mathsf{unwrap}$. Therefore by pre-duploid functoriality, F preserves linearity and thunkability. In other words F restricts to functors $F_t : \mathscr{D}_t \to \mathscr{D}'_t$ and $F_l : \mathscr{D}_l \to \mathscr{D}'_l$. Naturality for $F : \mathscr{D}(-,=) \dot{\to} \mathscr{D}'(F_t -, F_l =)$ finally corresponds to $Fh \circ Fg \circ Ff = F(h \circ g \circ f)$ for any h linear, any g and any f thunkable.

(3.⇒1.) We suppose that F restricts to functors F_t and F_l, such that the transformation $F : \mathscr{D}(-,=) \dot{\to} \mathscr{D}'(F_t -, F_l =)$ is natural. From the data

of the functors F_t and F_l we have $F(\mathrm{id}_A) = \mathrm{id}_{FA}$, $F(\mathsf{force}_P)$ linear and $F(\mathsf{wrap}_N)$ thunkable. From the natural transformation F we have for all h linear, all g and all f thunkable:

$$Fh \circ Fg \circ Ff = F(h \circ g \circ f)\,.$$

In particular for all $g \bullet f$ and all $h \circ g$:

$$F(g \bullet f) = Fg \bullet Ff$$
$$F(h \circ g) = Fh \circ Fg$$

Therefore F is a functor of duploids. ∎

Proposition II.27. *Every functor of duploids $F : \mathscr{D} \to \mathscr{D}'$ preserves the shifts:*

$$F{\Uparrow} \simeq {\Uparrow'}F : \mathscr{D}_l \to \mathscr{N}'^{\bullet}$$
$$F{\Downarrow} \simeq {\Downarrow'}F : \mathscr{D}_t \to \mathscr{P}'^{\bullet}$$

*and also preserves the isomorphisms of Proposition II.24 in the sense that the following diagram, in the category of functors $\mathscr{D}_t{}^{\mathrm{op}} \times \mathscr{D}_l \to$ **Set**, commutes:*

$$\begin{array}{ccccc}
\mathscr{D}_t(-, {\Uparrow}=) & \xleftarrow{\simeq} & \mathscr{D}(-,=) & \xrightarrow{\simeq} & \mathscr{D}_l({\Downarrow}-,=) \\
\downarrow{\scriptstyle F_t} & & \downarrow{\scriptstyle F} & & \downarrow{\scriptstyle F_l} \\
\mathscr{D}'_t(F_t-, F_t{\Uparrow}=) & & & & \mathscr{D}'_l(F{\Downarrow}-, F=) \\
\downarrow{\scriptstyle \simeq} & & & & \downarrow{\scriptstyle \simeq} \\
\mathscr{D}'(F-, {\Uparrow'}F=) & \xleftarrow{\simeq} & \mathscr{D}'(F-,F=) & \xrightarrow{\simeq} & \mathscr{D}'({\Downarrow'}F-, F=)
\end{array}$$

(we leave the inclusion functors implicit.)

Proof. We consider $\mathscr{D}$ and $\mathscr{D}'$ with the isomorphisms of Proposition II.23. According to Proposition II.26, we may post- and pre-compose with restrictions of F to obtain natural isomorphisms:

$${\Uparrow'}F = {\Uparrow'}F_l \simeq F_l \simeq F_l{\Uparrow} = F{\Uparrow}$$
$${\Downarrow'}F = {\Downarrow'}F_t \simeq F_t \simeq F_t{\Downarrow} = F{\Downarrow}$$

given by $F\mathsf{delay} \circ \mathsf{force}'_F$ and $F\mathsf{wrap} \bullet \mathsf{unwrap}'_F$. Now consider the equation in the right-hand square:

$$F(f \circledcirc \mathsf{unwrap}) \bullet F\mathsf{wrap} \circledcirc \mathsf{unwrap} = Ff \circledcirc \mathsf{unwrap}\,.$$

It holds by application of the equation of the duploid $F\mathscr{D}$: $(Ff \circledcirc F\mathsf{unwrap}) \bullet F\mathsf{wrap} = Ff$. The equation in the left-hand square is obtained similarly. ∎

We could have requested that F strictly preserves shifts. It is not hard to see that this coincides with a notion of *strict duploid functor,* that strictly preserves all the data of the duploid.

Definition II.28. $\mathbf{\mathscr{D}upl}$ is the category whose objects are duploids and whose morphisms are duploid functors. The obvious identity in $\mathbf{\mathscr{D}upl}$ is written $1_{\mathscr{D}}$.

II.2.6 Examples of duploids

Duplicial algebras with units

Duploids are named after Jean-Louis Loday's *duplicial algebras* [Lod06]. A duplicial algebra is given by a vector space X together with two associative operations $\bullet, \circ : X \otimes X \to X$ that satisfy the mixed associativity rule:

$$\forall x, y, z \in X,\ (x \bullet y) \circ z = x \bullet (y \circ z)\,.$$

The following example is meant to justify the terminology.

Let a duplicial algebra be given with elements $e^+, e^\ominus \in X$ that are neutral for $\bullet$ and $\circ$ (respectively).

Proposition II.29. *The following defines a duploid:*

Objects $|\mathscr{D}|$ *and* $\mathscr{D}$ *are defined with* $|\mathscr{N}| = \{\ominus\}$ *and* $|\mathscr{P}| = \{+\}$.

Morphisms $\mathscr{D}(A, B) \stackrel{\text{def}}{=} X$ *with* $\mathrm{id}_\varepsilon = e^\varepsilon$.

Composition *Positive composition is given by* $\bullet$*; negative composition by* $\circ$.

Duploid structure *Necessarily we take* $\Downarrow\ominus \stackrel{\text{def}}{=} +$ *and* $\Uparrow + = \ominus$. *We take the following:*

$$\begin{aligned}\mathsf{delay} &\stackrel{\text{def}}{=} e^{+}\\ \mathsf{force} &\stackrel{\text{def}}{=} e^{\ominus}\\ \mathsf{wrap} &\stackrel{\text{def}}{=} e^{+}\\ \mathsf{unwrap} &\stackrel{\text{def}}{=} e^{\ominus}\end{aligned}$$

Proof. The axioms of the duploid are satisfied:

$$\begin{aligned}e^{\ominus} \circ (e^{+} \bullet x) &= x\\ e^{+} \bullet e^{\ominus} &= e^{\ominus}\\ (x \circ e^{\ominus}) \bullet e^{+} &= x\\ e^{+} \circ e^{\ominus} &= e^{+}\end{aligned}$$

∎

The Blass phenomenon in Conway games

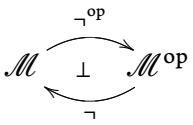
Melliès [Mel05] comes close to building a duploid using the construction of Blass games. His analysis of the Blass problem in Conway games [Mel05, Section 3] can be rephrased in the terminology of Section II.2.3: the Blass phenomenon comes down to the fact that the (pro-)functor $\mathscr{C}_1(F-,=) : \mathscr{C}_2{}^{\mathrm{op}} \times \mathscr{C}_1 \to \boldsymbol{Set}$ in an adjunction $F \dashv G : \mathscr{C}_1 \to \mathscr{C}_2$ does not extend into a functor $\mathscr{P}^{\mathrm{op}} \times \mathscr{N} \to \boldsymbol{Set}$ where $\mathscr{P}$ and $\mathscr{N}$ are the Kleisli categories of the monad GF and the co-monad FG (respectively). This is the essence of Proposition II.22. He then defines a category for an asynchronous variant of Conway games. As he shows, asynchronism is a way to force the double-negation monad to be idempotent, and therefore to recover associativity of composition. He builds this way a game model of linear logic.

Girard's polarisation

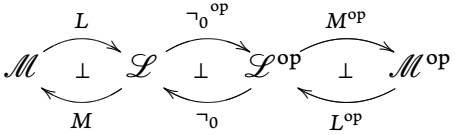
Girard's *polarised translation* of the classical logic **LC** into intuitionistic logic [Gir91], further formulated by Danos, Joinet and Schellinx [DJS97] and Laurent [Lau02], inspired the duploid construction. Girard's translation corresponds to considering in the duploid con-

struction the self-adjunction of the negation functor $\neg = R^-$ in ***Set*** for R arbitrary. But obviously, the duploid obtained from the self-adjunction of negation in any response category (in the terminology of Selinger [Sel01]) gives a denotational semantics of **LC**. The example is rich; for instance Thielecke [Thi97] explains the importance of this self-adjunction in CPS translations. This makes of duploids an element in the understanding of continuation-passing style, anticipating Chapter III.

Melliès and Tabareau [MT10] propose to decompose the self-adjunction of negation in the response category $\mathscr{M}$ through the linear negation $\neg_0$ in a dialogue category $\mathscr{L}$ and the ressource modalities M, L. These modalities can be seen as a uniform way of describing various models of the exponentials of linear logic [Mel09]. In this context, the self-adjunction in a response category:



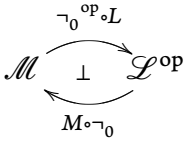
is obtained by composing the following adjunctions:



(we refer the reader to [MT10] for the terminology). They consider models of linear logic obtained through what corresponds to the duploid construction of the adjunction ${\neg_0}^{\mathrm{op}} \dashv \neg_0$, assuming that the adjunction is idempotent. This point of view was conceived as an abstract account of the asynchronous games mentioned above, and asks how properties of linear logic extend to settings where the adjunction is not idempotent (see Melliès [Mel12]).

The adjunction of a negation with itself is peculiar: idempotency is equivalent to commutativity of the double-negation strong monad,

as noticed by Führmann [Füh99][6]. This is why Melliès and Tabareau may emphasise commutativity over idempotency in [Mel05, MT10]. Finally, in the light of our development, it appears that idempotency is more fundamental in this context. We may indeed consider the duploid obtained from the composite adjunction:



with the idea that the corresponding logic is intuitionistic.[7] This will likely give instances where the monad is strong and commutative, but where the adjunction is not idempotent, and thus where the composition is not associative.

Direct models of call by value

We explain how Thunk-force categories give rise to duploids where $\Uparrow$ is bijective. In Section II.1.2, we defined a pre-duploid from a thunk-force category $(\mathscr{P}, \bullet, \mathrm{id}, L, \vartheta, \varepsilon)$ with a bijection $\Uparrow : |\mathscr{P}| \to |\mathscr{N}|$. Let us define $\Downarrow : |\mathscr{N}| \to |\mathscr{P}|$ with:

$$\boxed{\Downarrow\Uparrow P \stackrel{\text{def}}{=} LP}\,.$$

For $N = \Uparrow P$, we define:

$$\begin{aligned}
\mathscr{D}(P, \Uparrow P) \ni \mathsf{delay}_P &\stackrel{\text{def}}{=} \mathrm{id}^{\mathscr{P}}_P \in \mathscr{P}(P, P)\\
\mathscr{D}(\Uparrow P, P) \ni \mathsf{force}_P &\stackrel{\text{def}}{=} \varepsilon_P \in \mathscr{P}(LP, P)\\
\mathscr{D}(N, \Downarrow N) \ni \mathsf{wrap}_N &\stackrel{\text{def}}{=} \mathrm{id}^{\mathscr{P}}_{LP} \in \mathscr{P}(LP, LP)\\
\mathscr{D}(\Downarrow N, N) \ni \mathsf{unwrap}_N &\stackrel{\text{def}}{=} \varepsilon_P \in \mathscr{P}(LP, P)
\end{aligned}$$

[6] Melliès and Tabareau also credit a private communication from Masahito Hasegawa.

[7] It should be compared to the various proposals for a polarised intuitionistic logic by Girard [Gir93, Gir07] and Liang and Miller [LM07].

Proposition II.30. *The above construction yields a duploid.*

Proof. Let us check the equations of the duploid:

$$\begin{aligned}
\mathsf{force}_P \circ (\mathsf{delay}_P \bullet f) &= \underline{\varepsilon_P \bullet Lf} \bullet \vartheta_A = f \\
&= f \bullet \underline{\varepsilon_A \bullet \vartheta_A} \\
&= f \\
(f \circ \mathsf{unwrap}_{\Uparrow P}) \bullet \mathsf{wrap}_{\Uparrow P} &= f \bullet \underline{L\varepsilon_P \bullet \vartheta_{LP}} \\
&= f \\
\mathsf{delay}_P \bullet \mathsf{force}_P &= \varepsilon_P = \mathrm{id}^{\mathscr{D}}_{\Uparrow P} \\
\mathsf{wrap}_{\Uparrow P} \circ \mathsf{unwrap}_{\Uparrow P} &= L\varepsilon_P \bullet \vartheta_{LP} = \mathrm{id}^{\mathscr{D}}_{LP}
\end{aligned}$$

∎

We will see in Section II.3.5 that thunk-force categories are characterised as duploids where $\Uparrow$ is bijective on objects.

Direct models of call by name
Symmetrically, a runnable monad on a category gives a duploid where $\Downarrow$ is bijective. We will see in Section II.3.5 that runnable monads are characterised in this way.

II.3 Structure theorem

This section is devoted to the proof of the main result of the chapter:

Theorem II.31. *There are a reflection and an equivalence as follows:*

$$\mathcal{D}\mathit{upl} \simeq \mathcal{A}\mathit{dj}_{\mathit{eq}} \lhd \mathcal{A}\mathit{dj}\,.$$

The category of adjunctions $\mathcal{A}\mathit{dj}$ is defined below. The full subcategory $\mathcal{A}\mathit{dj}_{\mathit{eq}}$ of $\mathcal{A}\mathit{dj}$ consists in adjunctions that satisfy the *equalising requirement* (see Section II.3.2).

We consider pseudo maps of adjunctions as defined by Jacobs [Jac93]:

Definition II.32. Let $F \dashv_{(\eta,\varepsilon)} G : \mathscr{C}_1 \to \mathscr{C}_2$ and $F' \dashv_{(\eta',\varepsilon')} G' : \mathscr{C}'_1 \to \mathscr{C}'_2$

be two adjunctions. A *pseudo map of adjunctions*:

$$\boxed{(H_1, H_2, \phi, \psi) : (F \dashv_{(\eta,\varepsilon)} G) \to (F' \dashv_{(\eta',\varepsilon')} G')}$$

is given by a pair of functors $H_1 : \mathscr{C}_1 \to \mathscr{C}_1'$ and $H_2 : \mathscr{C}_2 \to \mathscr{C}_2'$ together with natural isomorphisms:

$$\phi : F'H_2 \xrightarrow{\simeq} H_1F$$
$$\psi : G'H_1 \xrightarrow{\simeq} H_2G$$

such that H_1 and H_2 preserve η and ε up to isomorphism:

$$H_2\eta = \psi_F \circ G'\phi \circ \eta'_{H_2}$$
$$H_1\varepsilon = \varepsilon'_{H_1} \circ F'\psi^{-1} \circ \phi_G^{-1}$$

As noted by Jacobs, two pseudo maps (H_1, H_2, ϕ, ψ) and $(H_1', H_2', \phi', \psi')$ compose as:

$$(H_1', H_2', \phi', \psi') \circ (H_1, H_2, \phi, \psi) = (H_1'H_1, H_2'H_2, H_1'\phi \circ \phi'_{H_2}, H_2'\psi \circ \psi'_{H_1})$$

and we write the obvious identity as $1_{F\dashv G} = (1_{\mathscr{C}_1}, 1_{\mathscr{C}_2}, \mathrm{id}_F, \mathrm{id}_G)$. Moreover, the above two equations are equivalent, and ϕ and ψ determine each other [Jac93].

We may reformulate the equations in terms of the natural isomorphism of the adjunction when convenient. Equivalently to preserving η or ε, a pseudo map of adjunctions (H_1, H_2, ϕ, ψ) must preserve one of the natural isomorphisms $\sharp : \mathscr{C}_1(F-, =) \xrightarrow{\simeq} \mathscr{C}_2(-, G=)$ or $\flat = \sharp^{-1}$ of the adjunction, modulo ϕ and ψ:

$$H_2 f^\sharp = \psi_N \circ (H_1 f \circ \phi_P)^{\sharp'} \text{ for all } f \in \mathscr{C}_1(FP, N)$$
$$H_1 g^\flat = (\psi_N^{-1} \circ H_2 g)^{\flat'} \circ \phi_P^{-1} \text{ for all } g \in \mathscr{C}_2(P, GN)$$

Definition 11.33. The category of adjunctions $\mathscr{Adj}$ has adjunctions between locally small categories as objects and pseudo maps of ad-

junctions as morphisms.

Remark Mac Lane [ML71] defines the stricter notion of *map of adjunction* where ϕ and ψ are identities. In comparison with the result above, the reflection in Führmann [Füh99] is based on maps of (Kleisli) adjunctions and *strict* functors of Thunk-force categories; and the structure theorem in Selinger [Sel01] is based on *strict* maps of control categories.

It is obvious in the following development that we could state a reflection between a category $\mathcal{Dupl}$ with strict functors of duploids and a category $\mathcal{Adj}$ with maps of adjunctions. However I found the notion of non-strict functor more elegant.

II.3.1 Every duploid comes from an adjunction

Proposition II.34. *Let $\mathscr{D}$ be a duploid. We define $\uparrow : \mathscr{P}^\bullet \to \mathscr{N}^\bullet$ the restriction of $\Uparrow$ and $\downarrow : \mathscr{N}^\bullet \to \mathscr{P}^\bullet$ the restriction of $\Downarrow$. There is an adjunction:*

$$\boxed{\uparrow \dashv \downarrow}$$

with unit $\mathsf{wrap}_{\Uparrow} \circ \mathsf{delay}$ *and co-unit* $\mathsf{unwrap} \bullet \mathsf{force}_{\Downarrow}$.

Proof. Due to the adjoint equivalences from Proposition II.23, we have the following natural isomorphisms:

$$\mathscr{N}^\bullet(\Uparrow I -, =) \simeq \mathscr{D}_l(I -, I =) : \mathscr{P}^{\bullet\mathrm{op}} \times \mathscr{N}^\bullet \to \boldsymbol{Set}$$
$$\mathscr{P}^\bullet(-, \Downarrow I =) \simeq \mathscr{D}_t(I -, I =) : \mathscr{P}^{\bullet\mathrm{op}} \times \mathscr{N}^\bullet \to \boldsymbol{Set}$$

where I denotes the inferrable inclusion functors. Since we have:

$$\mathscr{D}_l(P, N) = \mathscr{D}(P, N) = \mathscr{D}_t(P, N),$$

we also have:

$$\mathscr{D}_l(I -, I =) = \mathscr{D}_t(I -, I =) : \mathscr{P}^{\bullet\mathrm{op}} \times \mathscr{N}^\bullet \to \boldsymbol{Set}.$$

Thus we have an adjunction:

$$\mathscr{N}^\bullet(\Uparrow I -, =) \simeq \mathscr{P}^\bullet(-, \Downarrow I =) : \mathscr{P}^{\bullet\mathrm{op}} \times \mathscr{N}^\bullet \to \boldsymbol{Set}.$$

The adjunction is given by the bijection:

$$\begin{aligned} \mathscr{N}^\bullet(\uparrow P, N) &\simeq \mathscr{P}^\bullet(P, \downarrow N) \\ f &\mapsto \mathsf{wrap}_N \circ f \circ \mathsf{delay}_P \\ \mathsf{unwrap}_N \bullet g \bullet \mathsf{force}_P &\leftarrow\!\shortmid g \end{aligned}$$

∎

We show that the duploid obtained from this adjunction is isomorphic in $\boldsymbol{\mathcal{Dupl}}$ to $\mathscr{D}$.

Proposition II.35. *Let $\mathscr{D}$ a duploid and $\mathscr{D}'$ the duploid obtained from the adjunction $\uparrow \dashv \downarrow : \mathscr{N}^\bullet \to \mathscr{P}^\bullet$ with the construction of Section II.2.3. There is an isomorphism between the duploids $\mathscr{D}$ and $\mathscr{D}'$.*

Proof. Recall that $\mathscr{D}'$ is defined with:

$$\begin{aligned} |\mathscr{D}'| &= |\mathscr{D}| \\ \mathscr{D}'(A, B) &= \mathscr{N}^\bullet(\uparrow\Downarrow A, \Uparrow B) \end{aligned}$$

(we may identify A^+ with $\Downarrow A$ and $B^\ominus$ with $\Uparrow B$). Furthermore, by definition we have:

$$\begin{aligned} \mathrm{id}_P^{\mathscr{D}'} &= \mathrm{id}_{\Uparrow P}^{\mathscr{D}} \\ \mathrm{id}_N^{\mathscr{D}'} &= \mathsf{unwrap}_N \bullet \mathsf{force}_{\Downarrow N} \\ g \circ^{\mathscr{D}'} f &= (g \circ \mathsf{delay}_{\Downarrow N}) \bullet (\mathsf{wrap}_N \circ f \circ \mathsf{delay}_A) \circledcirc \mathsf{force}_A \\ g \bullet^{\mathscr{D}'} f &= g \circ f \end{aligned}$$

According to Propositions II.23 and II.24, we have natural isomorphisms:

$$\mathscr{D}(-,=) \simeq \mathscr{D}_l(I\Downarrow -,=) \simeq \mathscr{N}^\bullet(\Uparrow I\Downarrow -, \Uparrow =)$$

and thus for all $A, B \in |\mathscr{D}|$ we have a bijection $\mathscr{D}(A,B) \to \mathscr{N}^\bullet(\uparrow\Downarrow A, \Uparrow B) = \mathscr{D}'(A,B)$ given by:

$$F_{A,B} : f \mapsto \mathsf{delay}_B \circledcirc (f \circledcirc \mathsf{unwrap}_A) \bullet \mathsf{force}_{\Downarrow A}$$

It is straightforward to check that this mapping defines a functor of preduploids $F : \mathscr{D} \to \mathscr{D}'$.

Now we have $F(\mathsf{wrap}_N) = \mathrm{id}^{\mathscr{D}}_{\Uparrow\Downarrow N}$ which is the definition of wrap'_N; and $F(\mathsf{force}_P) = \mathsf{unwrap}_{\Uparrow P} \bullet \mathsf{force}_{\Downarrow\Uparrow P}$ which is the definition of force'_P. Therefore $F(\mathsf{wrap}_N)$ is thunkable and $F(\mathsf{force}_P)$ is linear. We conclude that F is a functor of duploids.

Like for functors of categories, it is easy to see, using the characterisation 3. of Proposition II.26, that if F is bijective on objects and morphisms then its inverse (given here with $f \mapsto (\mathsf{force}_B \circ f \circ \mathsf{delay}_{\Downarrow A}) \bullet \mathsf{wrap}_A$) is a functor of duploids. This concludes the proof. ∎

II.3.2 The equalising requirement

Definition II.36. An adjunction $F \dashv_{(\eta,\varepsilon)} G : \mathscr{C}_1 \to \mathscr{C}_2$ satisfies the *equalising requirement* if and only if for all $P \in |\mathscr{C}_2|$, η_P is an equaliser of η_{GFP} and $GF\eta_P$, and for all $N \in |\mathscr{C}_1|$, ε_N is a co-equaliser of ε_{FGN} and $FG\varepsilon_N$.

Definition II.37. Let $\mathscr{A}\mathit{dj}_{\mathit{eq}}$ be the full subcategory of $\mathscr{A}\mathit{dj}$ whose objects are adjunctions that satisfy the equalising requirement.

Let us recall that η_P is an equaliser of η_{GFP} and $GF\eta_P$ if and only if for all $f \in \mathscr{C}_2(P, GFQ)$ such that $\eta_{GFP} \circ f = GF\eta_P \circ f$, there exists a unique $g \in \mathscr{C}_2(P, Q)$ such that $f = \eta_P \circ g$.

Also, ε_N is a co-equaliser of ε_{FGN} and $FG\varepsilon_N$ if and only if for all $f \in \mathscr{C}_1(FGN, M)$ such that $f \circ \varepsilon_{FGN} = f \circ FG\varepsilon_N$, there exists a unique $g \in \mathscr{C}_1(N, M)$ such that $f = g \circ \varepsilon_N$.

We give an equivalent formulation of this condition in terms of the associated duploid:

Proposition II.38. *Let $F \dashv_{(\eta,\varepsilon)} G : \mathscr{C}_1 \to \mathscr{C}_2$ be an adjunction, and consider the associated duploid $\mathscr{D}$. The adjunction satisfies the equalising requirement if and only if for all objects A, P, N the following three conditions hold:*

1. *ε_N is an epimorphism and η_P is a monomorphism; or equivalently G and F are faithful;*
2. *all linear morphisms $f \in \mathscr{D}(N, A)$ are of the form $g \circ \varepsilon_N$ with $g \in \mathscr{C}_1(N, A^{\ominus})$; or equivalently all linear morphisms are in the image of G modulo the adjunction;*

3. *all thunkable morphisms $f \in \mathscr{D}(A,P)$ are (modulo the adjunction) of the form $\eta_P \circ g$ with $g \in \mathscr{C}_2(A^+,P)$; or equivalently all thunkable morphisms are in the image of F;*

Proof. Applying Proposition II.21, it is immediate to see that the conditions (2.) and (3.) are equivalent to the existence part of the equalising requirement; while the condition (1.) is equivalent to the uniqueness part. Equivalence with the second part of each condition is obtained by rephrasing through the natural isomorphism that defines the adjunction. ∎

Proposition II.39. *Let $\mathscr{D}$ be a duploid and consider the adjunction $\uparrow \dashv \downarrow : \mathscr{N}^\bullet \to \mathscr{P}^\bullet$. Then the adjunction satisfies the equalising requirement.*

Proof. We have $\varepsilon = \mathsf{unwrap} \bullet \mathsf{force}_{\Downarrow}$. It has a section in $\mathscr{N}$, namely the natural transformation $\vartheta \stackrel{\text{def}}{=} \mathsf{delay}_{\Downarrow} \bullet \mathsf{wrap} : 1 \dot{\to} \Uparrow\Downarrow$. Let $f \in \mathscr{N}^\bullet(\uparrow\downarrow N, M)$ such that $f \circ \varepsilon_{\uparrow\downarrow N} = f \circ \uparrow\downarrow\varepsilon_N$. By definition this means:

$$f \circ \varepsilon_{\Uparrow\Downarrow N} = f \circ \Uparrow\Downarrow\varepsilon_N \,.$$

Using this assumption we have:

$$\begin{aligned} f &= \underline{f \circ \varepsilon_{\Uparrow\Downarrow N}} \circ \vartheta_{\Uparrow\Downarrow N} \\ &= f \circ \Uparrow\Downarrow\varepsilon_N \circ \underline{\vartheta_{\Uparrow\Downarrow N}} \\ &= f \circ \vartheta_{\Uparrow\Downarrow N} \circ \varepsilon_N \end{aligned}$$

Thus there exists g such that $g \circ \varepsilon_N = f$. Because ε has a section, g is uniquely determined. Thus ε is a co-equaliser of $\varepsilon_{\uparrow\downarrow N}$ and $\uparrow\downarrow\varepsilon_N$. The proof that η is an equaliser of $\eta_{\downarrow\uparrow P}$ and $\downarrow\uparrow\eta_P$ is symmetric. ∎

II.3.3 The functors *i* and *j*

Lemma II.40. *We have a functor $j : \mathcal{Adj} \to \mathcal{Dupl}$ defined as follows.*

Given an adjunction $F \dashv G : \mathscr{C}_1 \to \mathscr{C}_2$, we call $j(F \dashv G)$ the associated duploid defined in Section II.2.3. We keep in particular the convention that $N, M \ldots$ represent objects of $\mathscr{C}_1$ and $P, Q \ldots$ represent objects of $\mathscr{C}_2$.

Now let two adjunctions $F \dashv_{(\sharp,\flat)} G : \mathscr{C}_1 \to \mathscr{C}_2$ *and* $F' \dashv_{(\sharp',\flat')} G' : \mathscr{C}'_1 \to \mathscr{C}'_2$ *and a pseudo map of adjunctions:*

$$(H_1, H_2, \phi, \psi) : (F \dashv_{(\eta,\varepsilon)} G) \to (F' \dashv_{(\eta',\varepsilon')} G').$$

Let us write $\mathscr{D} = j(F \dashv G)$ *and* $\mathscr{D}' = j(F' \dashv G')$*; let us also write:*

$$HN = H_1 N \text{ for } N \in |\mathscr{C}_1|$$
$$HP = H_2 P \text{ for } P \in |\mathscr{C}_2|$$

Let us extend ϕ *and* ψ *into the following families of maps:*

$$\bar{\phi}_A : (HA)^{\ominus} \xrightarrow{\simeq} H_1 A^{\ominus} = \begin{cases} \phi_P & \text{if } A = P \\ \mathrm{id}_N & \text{if } A = N \end{cases}$$
$$\bar{\psi}_A : (HA)^{+} \xrightarrow{\simeq} H_2 A^{+} = \begin{cases} \mathrm{id}_P & \text{if } A = P \\ \psi_N & \text{if } A = N \end{cases}$$

We define a functor of duploids $j(H_1, H_2, \phi, \psi) = H : \mathscr{D} \to \mathscr{D}'$ *by defining* H *on morphisms as follows:*

$$H : \mathscr{C}_1(FA^+, B^{\ominus}) \to \mathscr{C}'_1(F'(HA)^+, (HB)^{\ominus})$$
$$f \mapsto \bar{\phi}_B^{-1} \circ H_1 f \circ \phi_{A^+} \circ F' \bar{\psi}_A$$

Proof. We show that this indeed defines a functor $\mathbf{Adj} \to \mathbf{Dupl}$. We first prove that H is a functor of duploids. We recall that positive identity and composition are defined as the ones of $\mathscr{C}_1$. Therefore it is immediate that $H\mathrm{id}_P^{\mathscr{D}} = \mathrm{id}_{HP}^{\mathscr{D}'}$ and $Hf \bullet^{\mathscr{D}'} Hg = H(f \bullet^{\mathscr{D}} g)$.

We recall that negative identity and composition come from the ones

in $\mathscr{C}_2$ modulo the adjunction. Thus:

$$\begin{aligned} H\mathrm{id}_N^{\mathscr{D}} &= H\varepsilon_N \\ &= H_1\varepsilon_N \circ \phi_{GN} \circ F'\psi_N \\ &= \varepsilon'_{H_1N} \circ \underline{F'\psi_N^{-1} \circ \phi_{GN}^{-1} \circ \phi_{GN}} \circ F'\psi_N \\ &= \varepsilon'_{H_1N} = \mathrm{id}_{HN}^{\mathscr{D}'} \end{aligned}$$

and:

$$H(f \circ^{\mathscr{D}} g) = \bar{\phi}_B^{-1} \circ (H_1 (f^\sharp \circ^{\mathscr{C}_2} g^\sharp)^\flat) \circ \phi_{A^+} \circ F'\bar{\psi}_A$$

where:

$$\begin{aligned} (H_1 (f^\sharp \circ g^\sharp)^\flat) \circ \phi_{A^+} &= (\psi_{B^\ominus}^{-1} \circ \underline{H_2(f^\sharp \circ g^\sharp)})^{\flat'} \circ \underline{\phi_{A^+}^{-1} \circ \phi_{A^+}} \\ &= (\psi_{B^\ominus}^{-1} \circ H_2 f^\sharp \circ H_2 g^\sharp)^{\flat'} \\ &= (\underline{\psi_{B^\ominus}^{-1} \circ \psi_{B^\ominus}} \circ (H_1 f \circ \phi_{GN})^{\sharp'} \circ \psi_N \circ (H_1 g \circ \phi_{A^+})^{\sharp'})^{\flat'} \\ &= ((H_1 f \circ \phi_{GN})^{\sharp'} \circ \psi_N \circ (H_1 g \circ \phi_{A^+})^{\sharp'})^{\flat'} \end{aligned}$$

Therefore by naturality of $\flat'$, we have:

$$H(f \circ^{\mathscr{D}} g) = (G'\bar{\phi}_B^{-1} \circ (H_1 f \circ \phi_{GN})^{\sharp'} \circ \psi_N \circ (H_1 g \circ \phi_{A^+})^{\sharp'} \circ \bar{\psi}_A)^{\flat'}.$$

Hence by naturality of $\sharp'$, we have:

$$\begin{aligned} H(f \circ^{\mathscr{D}} g) &= ((\bar{\phi}_B^{-1} \circ H_1 f \circ \phi_{GN} \circ F'\psi_N)^{\sharp'} \circ^{\mathscr{C}'_2} (H_1 g \circ \phi_{A^+} \circ F'\bar{\psi}_A)^{\sharp'})^{\flat'} \\ &= Hf \circ^{\mathscr{D}'} Hg \end{aligned}$$

Therefore H is a functor of pre-duploids.

Let us now prove that $H\mathsf{force}_P$ is linear and $H\mathsf{wrap}_N$ is thunkable. According to Proposition II.21 we have to show:

$$\begin{aligned} H\mathsf{force}_P \circ \varepsilon'_{F'G'HFP} &= H\mathsf{force}_P \circ F'G'\varepsilon'_{HFP} \\ \eta'_{G'F'HGN} \circ (H\mathsf{wrap}_N)^{\sharp'} &= G'F'\eta'_{HGN} \circ (H\mathsf{wrap}_N)^{\sharp'} \end{aligned}$$

Using the hypothesis of the pseudo-maps $H_1\varepsilon_{FP} = \varepsilon'_{H_1FP} \circ F'\psi_{FP}^{-1} \circ \phi_{GFP}^{-1}$,

we have:

$$\begin{aligned} H\mathsf{force}_P &= \phi_P^{-1} \circ H_1\varepsilon_{FP} \circ \phi_{GFP} \circ F'\psi_{FP} \\ &= \phi_P^{-1} \circ \varepsilon'_{H_1FP} \circ \underline{F'\psi_{FP}^{-1} \circ \phi_{GFP}^{-1} \circ \phi_{GFP}} \circ F'\psi_{FP} \\ &= \phi_P^{-1} \circ \varepsilon'_{H_1FP} \end{aligned}$$

thus the above equations can be written:

$$\phi_P^{-1} \circ \varepsilon'_{H_1FP} \circ \varepsilon'_{F'G'H_1FP} = \phi_P^{-1} \circ \varepsilon'_{H_1FP} \circ F'G'\varepsilon'_{H_1FP}$$
$$\eta'_{G'F'H_2GN} \circ G'F'\psi_N \circ \eta'_{G'H_1N} = G'F'\eta'_{H_2GN} \circ G'F'\psi_N \circ \eta'_{G'H_1N}$$

The one above, is equivalent to the following instance of naturality of ε':

$$\varepsilon'_{H_1FP} \circ \varepsilon'_{F'G'H_1FP} = \varepsilon'_{H_1FP} \circ F'G'\varepsilon'_{H_1FP}$$

therefore $H\mathsf{force}_P$ is linear. The one below holds by naturality of η':

$$\begin{aligned} \eta'_{G'F'H_2GN} \circ G'F'\psi_N \circ \eta'_{G'H_1N} &= \eta'_{G'F'H_2GN} \circ \eta'_{H_2GN} \circ \psi_N \\ &= G'F'\eta'_{H_2GN} \circ \eta'_{H_2GN} \circ \psi_N \\ &= G'F'\eta'_{H_2GN} \circ G'F'\psi_N \circ \eta'_{G'H_1N} \end{aligned}$$

hence $H\mathsf{wrap}_N$ is thunkable. H is therefore a functor of duploids.

It is straightforward to see that $j(1_{\mathscr{C}_1}, 1_{\mathscr{C}_2}, \mathrm{id}_F, \mathrm{id}_G) = 1_{j(F \dashv G)}$ and we also have:

$$\begin{aligned} & j((H'_1, H'_2, \phi', \psi') \circ (H_1, H_2, \phi, \psi))(f) \\ &= \overline{\underline{H'_1\phi \circ \phi'_{H_2B}}}^{-1} \circ H'_1H_1f \circ \underline{(H'_1\phi \circ \phi'_{H_2})_{A^+}} \circ F'\overline{\underline{H_2\psi \circ \psi'_{H_1A}}} \\ &= \bar{\phi}'^{-1}_{HB} \circ H'_1\bar{\phi}_B^{-1} \circ H'_1H_1f \circ H'_1\phi_{A^+} \circ \underline{\phi'_{H_2A^+} \circ F'H_2\bar{\psi}_A} \circ F'\bar{\psi}'_{HA} \\ &= \bar{\phi}'^{-1}_{HB} \circ \underline{H'_1\bar{\phi}_B^{-1} \circ H'_1H_1f \circ H'_1\phi_{A^+} \circ H_1F\bar{\psi}_A} \circ \phi'_{(HA)^+} \circ F'\bar{\psi}'_{HA} \qquad (\phi' \text{ nat.}) \\ &= \bar{\phi}'^{-1}_{HB} \circ H'_1(\bar{\phi}_B^{-1} \circ H_1f \circ \phi_{A^+} \circ F\bar{\psi}_A) \circ \phi'_{(HA)^+} \circ F'\bar{\psi}'_{HA} \\ &= j(H'_1, H'_2, \phi', \psi') \circ j(H_1, H_2, \phi, \psi)(f) \end{aligned}$$

Thus j is a functor. ∎

Lemma II.41. *There is a functor $i : \mathcal{Dupl} \to \mathcal{Adj}$ defined as follows.*

Given a duploid $\mathscr{D}$, the adjunction $i\mathscr{D}$ is $\uparrow \dashv \downarrow : \mathcal{N}^\bullet \to \mathcal{P}^\bullet$ as considered in II.3.1.

Now let $\mathscr{D}$ and $\mathscr{D}'$ two duploids and let us write $i\mathscr{D} = (\uparrow \dashv \downarrow : \mathcal{N}^\bullet \to \mathcal{P}^\bullet)$ and $i\mathscr{D}' = (\uparrow' \dashv \downarrow' : \mathcal{N}'^\bullet \to \mathcal{P}'^\bullet)$. Given a duploid functor $H : \mathscr{D} \to \mathscr{D}'$ we define:

$$iH = (H_1, H_2, \phi, \psi) : (\uparrow \dashv \downarrow) \to (\uparrow' \dashv \downarrow')$$

with H_1 the functor H_l restricted to $\mathcal{N}^\bullet$ and H_2 the functor H_t restrained to $\mathcal{P}^\bullet$; and:

$$\phi_P \overset{\text{def}}{=} H\mathsf{delay}_P \bullet \mathsf{force}'_{HP} : \uparrow' H_2 P \xrightarrow{\simeq} H_1 \uparrow P$$
$$\psi_N \overset{\text{def}}{=} H\mathsf{wrap}_N \circ \mathsf{unwrap}'_{HN} : \downarrow' H_1 N \xrightarrow{\simeq} H_2 \downarrow N$$

Proof. We prove that this indeed defines a functor $\mathcal{Dupl} \to \mathcal{Adj}$. According to Proposition II.26,we have $H_1 : \mathcal{N}^\bullet \to \mathcal{N}'^\bullet$ and $H_2 : \mathcal{P}^\bullet \to \mathcal{P}'^\bullet$. The natural isomorphisms ϕ and ψ are obtained by restriction to $\mathcal{P}^\bullet$ and $\mathcal{N}^\bullet$ of the ones in Proposition II.27.

Now we consider the unit $\eta \overset{\text{def}}{=} \mathsf{wrap}_{\Uparrow} \circ \mathsf{delay}$ of the first adjunction. It is immediate to see that we have $\psi_{\Uparrow} \bullet \Downarrow' \phi \bullet \eta'_{H_2} = H_2 \eta$. This makes iH a pseudo map of adjunctions.

Last, let us consider $iH' \circ iH$ and let us call $\phi^{H'H}$ and $\psi^{H'H}$ the isomorphisms of $i(H'H)$. We prove $iH' \circ iH = i(H'H)$ by showing $\phi^{H'H} = H'_1\phi \circ \phi'_{H_2}$ and $\psi^{H'H} = H_2\psi \circ \psi'_{H_1}$. We have:

$$\begin{aligned} H'_1\phi \circ \phi'_{H_2} &= \underline{H'_1(H\mathsf{delay} \bullet \mathsf{force}'_H)} \circ (H'\mathsf{delay}' \bullet \mathsf{force}''_{H'})_{H_2} \\ &= (H'H\mathsf{delay} \bullet \underline{H'\mathsf{force}'_H}) \circ (H'\mathsf{delay}'_H \bullet \mathsf{force}''_{H'H}) \\ &= H'H\mathsf{delay} \bullet \mathsf{force}''_{H'H} \\ &= \phi^{H'H} \text{ by definition,} \end{aligned}$$

and symmetrically for $\psi^{H'H}$. Together with $i(1_{\mathscr{D}}) = 1_{i\mathscr{D}}$ this makes i a functor $\mathcal{Dupl} \to \mathcal{Adj}$. ∎

II.3.4 Proof of the main result

Lemma II.42. *There is a natural transformation* $\boldsymbol{\eta} : 1^{\mathcal{Adj}} \dot{\to} ij$.

Proof. For any adjunction $F \dashv_{(\eta,\varepsilon)} G : \mathscr{C}_1 \to \mathscr{C}_2$, let us consider the duploid $\mathscr{D} = j(F \dashv G)$. The component $\boldsymbol{\eta}_{F\dashv G} : (F \dashv G) \to i\mathscr{D}$ is given by:

$$H_1^{\boldsymbol{\eta}} N = N$$
$$H_1^{\boldsymbol{\eta}}(f : N \to M) = f \circ^{\mathscr{C}_1} \varepsilon_N \in \mathscr{N}^\bullet(N, M)$$
$$H_2^{\boldsymbol{\eta}} P = P$$
$$H_2^{\boldsymbol{\eta}}(g : P \to Q) = Fg \in \mathscr{P}^\bullet(P, Q)$$

The co-unit ε' of the adjunction $i\mathscr{D}$ is given by:

$$\varepsilon' = \mathsf{unwrap} \bullet \mathsf{force}_{\Downarrow} = \varepsilon \circ^{\mathscr{C}_1} \varepsilon_{FG}$$

Thus H_1 strictly preserves ε:

$$H_1\varepsilon = \varepsilon \circ \varepsilon_{FG} = \varepsilon'_{H_2}$$

We may therefore take the identity isomorphisms $\psi^{\boldsymbol{\eta}} = \mathrm{id}_G^{\mathscr{P}^\bullet} = \mathrm{id}_{FG}^{\mathscr{C}_1}$ and $\phi^{\boldsymbol{\eta}} = \mathrm{id}_F^{\mathscr{N}^\bullet} = \varepsilon_F$. This defines $\boldsymbol{\eta}_{F\dashv G}$ as a (pseudo) map of adjunctions.

Let us prove that $\boldsymbol{\eta}$ is natural. Let us consider a pseudo map of adjunctions:

$$(H_1, H_2, \phi, \psi) : (F \dashv_{(\eta,\varepsilon)} G) \to (F' \dashv_{(\eta',\varepsilon')} G')$$

In the following we consider the duploids $\mathscr{D} = j(F \dashv G)$ and $\mathscr{D}' = j(F' \dashv G')$; the functor of duploid $H' = j(H_1, H_2, \phi, \psi)$ and the pseudo map of adjunctions $iH' = (H'_1, H'_2, \phi', \psi')$. By definition we have for $f \in \mathscr{N}^\bullet(N, M)$ and $g \in \mathscr{P}^\bullet(P, Q)$:

$$H'_1 f = H' f = H_1 f \circ \phi_{GN} \circ F'\psi_N$$
$$H'_2 g = H' g = \phi_Q^{-1} \circ H_1 g \circ \phi_P$$
$$\phi'_P = H' \mathsf{delay}_P \bullet^{\mathscr{D}'} \mathsf{force}'_{H'P} = \phi_P \circ \varepsilon'_{F'H_2P}$$

Now let us prove:

$$\boldsymbol{\eta}_{F' \dashv G'} \circ (H_1, H_2, \phi, \psi) = ij(H_1, H_2, \phi, \psi) \circ \boldsymbol{\eta}_{F \dashv G}$$

On the left hand side we have the following pseudo map:

$$(f \mapsto (H_1 f) \circ \varepsilon'_N,\ g \mapsto F' H_2 g,\ H_1^{\boldsymbol{\eta}} \phi,\ H_2^{\boldsymbol{\eta}} \psi)$$

On the right-hand side we have:

$$(f \mapsto H'_1(f \circ \varepsilon_N),\ g \mapsto H'_2 F g,\ \phi'_{H_2^{\boldsymbol{\eta}}},\ \psi'_{H_1^{\boldsymbol{\eta}}})$$

The first three components coincide:

$$\begin{aligned} H'_1(f \circ \varepsilon_N) &= H_1(f \circ \varepsilon_N) \circ \phi_{GN} \circ F' \psi_N \\ &= H_1 f \circ (H_1 \varepsilon_N \circ \phi_{GN} \circ F' \psi_N) \\ &= H_1 f \circ \varepsilon'_{H_1 N} \\ H'_2 F g &= \phi_Q^{-1} \circ H_1 F g \circ \phi_P = F' H_2 g \\ H_1^{\boldsymbol{\eta}} \phi &= \phi \circ \varepsilon'_{F' H_2} = \phi' = \phi'_{H_2^{\boldsymbol{\eta}}} \end{aligned}$$

Therefore the two sides are equal. This completes the proof. ∎

Lemma II.43. *There is a natural isomorphism* $\boldsymbol{\varepsilon} : ji \xrightarrow{\simeq} 1^{\mathcal{D}upl}$.

Proof. According to Proposition II.35 we have for any duploid $\mathscr{D}$ an isomorphism:

$$\boldsymbol{\varepsilon}_{\mathscr{D}} : ji\mathscr{D} \to \mathscr{D}\,,$$

where:

$$\begin{aligned} \boldsymbol{\varepsilon}_{\mathscr{D}}(A) &= A \\ \boldsymbol{\varepsilon}_{\mathscr{D}}(f) &= (\mathsf{force}_B \circ f \circ \mathsf{delay}_{\Downarrow A}) \bullet \mathsf{wrap}_A \end{aligned}$$

Let $H : \mathscr{D} \to \mathscr{D}'$ be a functor of duploids. By definition we have for any $f \in \mathscr{D}(A, B)$:

$$\begin{aligned} jiH(f) = &(\mathsf{delay}'_{HB} \circledcirc H\mathsf{force}_B) \circ Hf \circ \\ &(H\mathsf{delay}_{\Downarrow A} \bullet \mathsf{force}'_{H \Downarrow A}) \circ \Uparrow'(H\mathsf{wrap}_A \circledcirc \mathsf{unwrap}'_{HA}) \end{aligned}$$

Therefore by linearity of Hf and delay'_{HB} we have:

$$\begin{aligned} jiH(f) &= \Uparrow'(H(\mathsf{force}_B \circ f \circ \mathsf{delay}_{\Downarrow A}) \bullet H\mathsf{wrap}_A \circledast \mathsf{unwrap}'_{HA}) \\ &= \boldsymbol{\varepsilon}^{-1}_{\mathscr{D}'} \circ H \circ \boldsymbol{\varepsilon}_{\mathscr{D}}(f) \end{aligned}$$

The functors are equal on objects, therefore we may conclude:

$$jiH = \boldsymbol{\varepsilon}^{-1}_{\mathscr{D}'} \circ H \circ \boldsymbol{\varepsilon}_{\mathscr{D}}\,. \qquad \blacksquare$$

Lemma II.44. *We have the adjunction $j \dashv_{(\boldsymbol{\eta},\boldsymbol{\varepsilon})} i : \mathcal{Dupl} \to \mathcal{Adj}$.*

Proof. Building on Lemma II.42 and Lemma II.43, it remains us to show:

$$\begin{aligned} i\boldsymbol{\varepsilon} \circ \boldsymbol{\eta}_i &= 1_i \\ \boldsymbol{\varepsilon}_j \circ j\boldsymbol{\eta} &= 1_j \end{aligned}$$

Let $\mathscr{D}$ a duploid. On the one hand we have by definition:

$$\boldsymbol{\eta}_{i\mathscr{D}} = (f \mapsto f \circ (\mathsf{unwrap}_N \bullet \mathsf{force}_{\Downarrow N}),\ \Uparrow,\ \mathrm{id}^{ji\mathscr{D}}_{\Uparrow},\ -)$$

(where the functors are identity on objects). On the other hand, we have:

$$i\boldsymbol{\varepsilon}^{-1}_{\mathscr{D}} = (f \mapsto (f \circ \mathsf{unwrap}_N) \bullet \mathsf{force}_{\Downarrow N},\ g \mapsto \mathsf{delay}_P \bullet g \bullet \mathsf{force}_P,\ \boldsymbol{\varepsilon}^{-1}_{\mathscr{D}}(\mathrm{id}^{\mathscr{D}}_{\Uparrow P}),\ -)$$

(where the functors are identity on objects). The two pseudo maps of adjunctions coincide (in the first component, we use the hypothesis that f is linear). This gives us the first equation.

Now let $F \dashv_{(\eta,\varepsilon)} G : \mathscr{C}_1 \to \mathscr{C}_2$. By definition we have for all $f \in j(F \dashv G)(A, B)$:

$$j\boldsymbol{\eta}_{F\dashv G}(f) = f \circ \varepsilon_{F\,A^+}$$

We also notice that the definition of the duploid $j(F \dashv G)$ Section II.2.3 extends as follows:

$$\begin{aligned} \mathsf{wrap}_A &= \mathrm{id}^{\mathscr{C}_1}_{F(A^+)} \\ \mathsf{force}_A &= \varepsilon_{G(A^\ominus)} \end{aligned}$$

Therefore it follows by definition that we have for all $g \in jij(F \dashv G)(A, B)$:

$$\boldsymbol{\varepsilon}_{j(F\dashv G)}(g) = g \circ F\eta_{A^+}$$

Therefore:

$$\boldsymbol{\varepsilon}_{j(F\dashv G)} \circ j\boldsymbol{\eta}_{F\dashv G} = 1_{j(F\dashv G)}$$

Since both $j\boldsymbol{\eta}_{F\dashv G}$ and $\boldsymbol{\varepsilon}_{j(F\dashv G)}$ are identity on objects, we conclude that the second equation holds. ∎

Lemma II.45. *There is an equivalence of categories:*

$$\boxed{\mathcal{Dupl} \simeq \mathcal{Adj}_{eq}}\,.$$

Proof. According to Proposition II.39, i co-restricts to a functor $\mathcal{Dupl} \to \mathcal{Adj}_{eq}$. There remains to show that the restriction of $\boldsymbol{\eta}$ to $\mathcal{Adj}_{eq}$ is an isomorphism; in other words that $\boldsymbol{\eta}_{F\dashv G}$ is invertible whenever $F \dashv G : \mathscr{C}_1 \to \mathscr{C}_1$ satisfies the equalising requirement.

Consider such an adjunction. It is not hard to see that a pseudo map of adjunctions is invertible if and only if its first two components are. Let us recall that these two components are defined with:

$$\begin{aligned} H_1^{\boldsymbol{\eta}} N &= N \\ H_1^{\boldsymbol{\eta}}(f : N \to M) &= f \circ^{\mathscr{C}_1} \varepsilon_N \in \mathscr{N}^{\bullet}(N, M) \\ H_2^{\boldsymbol{\eta}} P &= P \\ H_2^{\boldsymbol{\eta}}(g : P \to Q) &= Fg \in \mathscr{P}^{\bullet}(P, Q) \end{aligned}$$

Therefore $H_1^{\boldsymbol{\eta}}$ and $H_2^{\boldsymbol{\eta}}$ are invertible because they are bijective on objects, and, by application of Proposition II.38, bijective on morphisms. This concludes the proof. ∎

We may now proceed with the proof of the main result.

Proof of Theorem II.31. According to Lemma II.44 and Lemma II.43, $j : \mathcal{Adj} \to \mathcal{Dupl}$ is a reflection. According to II.45, $\mathcal{Adj}_{eq}$ is the reflective subcategory of $\mathcal{Adj}$ determined by the right adjoint to j. ∎

Intuitively, according to Proposition II.38 and Theorem II.31, the functor j completes the values with all the expressions that are pure, and completes the stacks with all the evaluation contexts that are linear. Moreover j identifies any two values that denote the same expression, and any two stacks that denote the same evaluation context.

II.3.5 Characterisation of Kleisli categories

We have not developed the notion of strict functor of duploids. We leave it as an exercise for the reader to check that our reflection restricts to a reflection between the sub-category $\mathcal{D}\mathit{upl}^{st}$ of $\mathcal{D}\mathit{upl}$ whose morphisms are strict functors of duploids, and the sub-category $\mathcal{A}\mathit{dj}^{st}$ of $\mathcal{A}\mathit{dj}$ whose morphisms are maps of adjunctions in the sense of Mac Lane. The full sub-category $\mathcal{A}\mathit{dj}^{st}_{eq}$ of $\mathcal{A}\mathit{dj}_{eq}$ determined by the right adjoint to the reflection consists of adjunctions satisfying the equalising requirement.

As an immediate corollary, thunk-force categories are essentially duploids where $\Uparrow$ is bijective on objects:

Corollary II.46. *Führmann's category $\mathcal{T}\mathit{f}$ [Füh99] of thunk-force categories is equivalent to the full sub-category $\mathcal{D}\mathit{upl}^{st}_{+}$ of $\mathcal{D}\mathit{upl}^{st}$ whose objects are duploids with $\Uparrow$ bijective on objects.*

Proof. Notice that i and j preserves the fact that F and $\Uparrow$ are bijective on objects. Therefore the equivalence $\mathcal{D}\mathit{upl}^{st} \simeq \mathcal{A}\mathit{dj}^{st}_{eq}$ restricts to an equivalence between $\mathcal{D}\mathit{upl}^{st}_{+}$ and the full subcategory of $\mathcal{A}\mathit{dj}^{st}_{eq}$ whose objects are adjunctions $F \dashv G$ with F bijective on objects. According to Führmann, the latter is equivalent to $\mathcal{T}\mathit{f}$ [Füh99, Theorem 9]. ∎

By duality, categories with a runnable monad are essentially duploids where $\Downarrow$ is bijective on objects.

Chapter III

Decomposing delimited CPS translations

We show how delimited CPS translations decompose in three steps:

1. Call by value and call by name are implemented in a single polarised calculus with delimited control by choosing the appropriate polarity coercions;
2. A variant of the duploid construction relates the polarised calculus to an indirect model inspired by linear logic;
3. Erasing the interesting structure take us back into a calculus of continuations. Polarities disappear while administrative redexes appear.

Our approach allow us to rationally reconstruct four distinct call-by-name variants of Danvy and Filinski's *shift* and *reset*, one of which appears to be new. Although the decomposition also holds for non-delimited CPS translations, the full decomposition did not appear in its entirety: the direct, polarised, intermediate calculus was missing.

III.1 Delimited control operators

Delimited control operators model in direct style the fact that captured contexts have a finite extent. They do so by including a first-class context delimiter $< \cdot >$, as introduced by Felleisen [Fel88]. The

literature distinguishes four ways of capturing a delimited context, depending on how the delimiter is inserted in the result:

$$\begin{aligned}
<E[\mathcal{F}_0(\lambda x.t)]> &\triangleright\ \ t[\lambda y.\ E[y]\ /x] \\
<E[\mathcal{F}(\lambda x.t)]> &\triangleright\ <t[\lambda y.\ E[y]\ /x]> \\
<E[\mathcal{S}_0(\lambda x.t)]> &\triangleright\ \ t[\lambda y.<E[y]>/x] \\
<E[\mathcal{S}(\lambda x.t)]> &\triangleright\ <t[\lambda y.<E[y]>/x]>
\end{aligned}$$

$\mathcal{F}$ is the operator *control* of Felleisen *et al.* [FWFD88, Fel88]; $\mathcal{S}$ is Danvy and Filinski's *shift* [DF90]; and $\mathcal{F}_0$ and $\mathcal{S}_0$ are respectively *control*$_0$ and *shift*$_0$ in the terminology of Shan [Sha07]. In the context of Danvy and Filinski's shift, the delimiter $<\cdot>$ is called *reset*. With all variants, the delimiter is erased once a value is reached:

$$<V> \triangleright V\,.$$

In this chapter we are interested in Danvy and Filinski's shift $\mathcal{S}$ and reset $<\cdot>$. The first reason is that there are many examples of uses (see Biernacki *et al.* [BDS06]). The second reason is that it was long thought to be the only one to have a defining CPS translation [Sha07]. Thus, there is a consistent body of work studying their equational theory [KH03], their proof theoretic meaning [AHS04, AHS09, Ili10] as well as alternatives to the call-by-value evaluation order [HG08, BB09, KT10].

Danvy and Filinski's CPS translation is well understood from the point of view of monads, and the type system that Danvy and Filinski provide on top of their calculus can be described by "*annotated*" monads (Wadler [Wad94]). As far as the call-by-value shift and reset calculus is concerned, our polarised analysis parallels the monadic analysis. Monads, however, are a limiting approach to CPS, as they do not account for the Lafont-Reus-Streicher translation, and even less so for polarisation, as we saw in Section II.3.5 on page 152.

Our approach is based on the proof-theoretic decomposition of shift and reset of Herbelin *et al.* [AHS04, Her05, AHS09] with the $\lambda\mu\tilde{\mu}\hat{p}_v$

calculus and its $\bar{\lambda}\mu\tilde{\mu}$ variant. As they show, shift and reset decompose into the $\mu\alpha$ context binder of Parigot together with a continuation variable written $\hat{\mathfrak{tp}}$ and a context binder $\mu\hat{\mathfrak{tp}}$ with an unconventional scope. With our polarised approach we can decompose the $\lambda\mu\hat{\mathfrak{tp}}_v$ calculus as well as its call-by-name variant $\lambda\mu\hat{\mathfrak{tp}}_n$ introduced by Herbelin and Ghilezan [HG08]. We can also rationally reconstruct three other call-by-name variants, one of which appears to be new.

While the call-by-name λμ calculus can be deduced from the call-by-value one by duality [Sel01, CH00], this is no longer true with delimited control. Polarisation is therefore crucial in our analysis.

A linear analysis of the CPS target

Our approach is also different for its analysis of the target of Danvy and Filinski's CPS translation. Delimited CPS translations relax the hypothesis that calls to continuations are made in tail position. As a consequence, contrarily to non-delimited CPS translations, the order of evaluation of the target calculus matters, and the target of Danvy and Filinski's calculus is in call by value.

Unsurprisingly, the literature expresses this call-by-value order by resorting to a second CPS translation when necessary. However, in this last translation it is important that continuations are used linearly. (See Kameyama and Hasegawa [KH03]; more generally on linearity in CPS translations when continuations are not first-class see Danvy and Lawall [DL92, Dan00].) We noticed that the second translation could be replaced with the so-called "*boring*" translation into linear logic [Gir87] which is based on the encoding $!(P \multimap Q)$ of implication. This translation indeed yields a model of call by value (Maraist et al. [MOTW94]), and more precisely, corresponds to Moggi's monadic model [Mog89] where the strong monad is commutative (Benton and Wadler [BW95]).

However, instead of carrying out a second translation, we directly translate into a call-by-value calculus whose equational theory is informed by the fact that it arises from a commutative strong monad. In other words, the target of our translation is a call-by-value λ calculus extended with the following equation that accounts for commut-

ativity:

$$\mathsf{let}\, x\, \mathsf{be}\, t\, \mathsf{in}\, (\mathsf{let}\, y\, \mathsf{be}\, u\, \mathsf{in}\, v) \simeq \mathsf{let}\, y\, \mathsf{be}\, u\, \mathsf{in}\, (\mathsf{let}\, x\, \mathsf{be}\, t\, \mathsf{in}\, v)\,. \qquad \text{(III.1)}$$

We call the calculus $\lambda_{\mathrm{v}}^{\times}$.

Our realization in choosing the "*boring*" translation of call by value is that it allows us to link up with the polarised study of double-negation translations. This approach is based on the decomposition of the type of continuations into the exponential modality ! of **LLP** and the involutive negation:

$$\neg P = !P^{\perp}\,.$$

Delimited CPS translations replace negation by implication, which can be understood in the context of CPS translations as an annotated negation modality:

$$\neg_Q P \stackrel{\text{def}}{=} P \to Q$$

Now, this annotated negation modality can be seen through the "*boring*" translation as an annotated exponential:

$$\boxed{\neg_Q P = !_Q P^{\perp} \stackrel{\text{def}}{=} !(P^{\perp} \parr Q)}\,.$$

(We call the connective $!_P$ an exponential in the sense that it defines a functor that enjoys $!_P(A \,\&\, B) \simeq !_P A \otimes !_P B$ in linear logic, since $\parr$ distributes over &.)

We see as an additional argument in favour of a call-by-value target the fact that **LLP** lends itself to analogies with call by value, rather than call by name, λ calculus. Indeed, if the target of the CPS translation had been in call by name, as in Sabry's variant of the calculus [Sab96] or as in Chapter IV, then we would have been unable to provide such a decomposition inspired from linear logic.

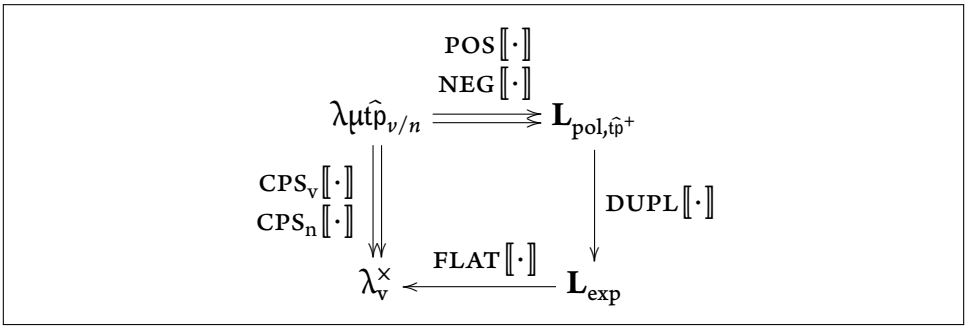


Figure III.1: The polarised decomposition of delimited CPS

Technical contribution

We decompose the delimited CPS translations in three steps (Figure III.1 on this page). See also Table I.17 on page 93 for a summary of the systems considered in this chapter.

In Section III.2 we introduce the polarised calculus for delimited control $\mathbf{L}_{\mathrm{pol},\hat{\mathrm{tp}}^+}$ and we describe its translation $\mathrm{CPS}[\![\cdot]\!]$ into $\lambda^{\times}_{\mathrm{v}}$. We reconstruct five calculi, including four in call by name. Then for two of them: the calculi $\lambda\mu\hat{\mathrm{tp}}_v$ and $\lambda\mu\hat{\mathrm{tp}}_n$, we decompose the CPS translations through $\mathbf{L}_{\mathrm{pol},\hat{\mathrm{tp}}^+}$. Finally we introduce $\mathbf{LK}_{\mathrm{delim}}$, a variant of Girard's **LC** [Gir91] and Danos, Joinet and Schellinx's $\mathbf{LK}^{\eta}_{p}$ [DJS97] that has type annotations. This system allows us to reconstruct the type systems of $\lambda\mu\hat{\mathrm{tp}}_v$ and $\lambda\mu\hat{\mathrm{tp}}_n$, which shows that the polarised decomposition is also informative for annotated type systems.

In Section III.3, we introduce the calculus $\mathbf{L}_{\mathrm{exp}}$ and we show how the $\mathrm{CPS}[\![\cdot]\!]$ decomposes through $\mathbf{L}_{\mathrm{exp}}$: it is equal to a variant of the duploid construction $\mathrm{DUPL}[\![\cdot]\!]$, followed by the translation $\mathrm{FLAT}[\![\cdot]\!]$. $\mathrm{FLAT}[\![\cdot]\!]$ removes polarities and introduces administrative redexes. Then we introduce $\mathbf{L}\bar{\mathbf{J}}$, a variant of **LLP** which is obtained by replacing the exponentials ! and ? by the annotated modalities $?_P$ and $!_P$.[1]

[1] In terms of proofs nets, these modalities are described by adding auxiliary doors to the ! and ? cells together with immediate reduction and expansion laws. Such proofs nets were used extensively in the development of our decomposition, but as far as formalisation was concerned we found easier to

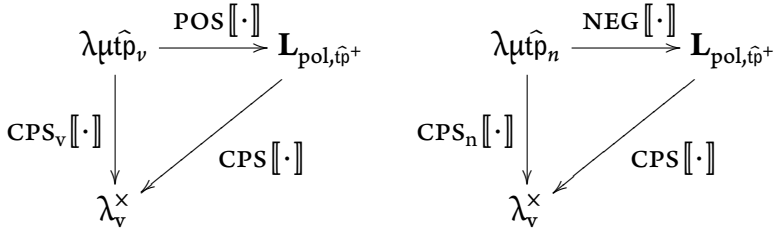
All the translations preserve equivalences; the translations $\textsc{cps}[\![\cdot]\!]$ and $\textsc{dupl}[\![\cdot]\!]$ also simulate the reduction.

It is important to note that:

- We provide the simple type systems $\mathbf{L}\bar{\mathbf{J}}$ and $\mathbf{LK}_{\text{delim}}$ for their illustrative purpose regarding the untyped translations. We prove that typing is preserved by the translations. Other than that, the chapter focuses on untyped calculi and their untyped translations.
- Additive connectives are omitted in the development. This is because they are not useful in the decomposition of the CPS translations we consider.

III.2 A polarised calculus for delimited control

We introduce in Figure III.2 on the facing page the calculus $\mathbf{L}_{\text{pol},\hat{\text{tp}}^+}$. The goal is to show that the CPS translations of the call-by-value $\lambda\mu\hat{\text{tp}}_v$ calculus of Ariola, Herbelin and Sabry and the call-by-name $\lambda\mu\hat{\text{tp}}_n$ calculus of Herbelin and Ghilezan factor in the following manner:



where the top arrow consists in canonical implementations of abstraction and application in $\mathbf{L}_{\text{pol},\hat{\text{tp}}^+}$ (modulo administrative reductions for the second diagram).

Proposition III.1. *The reduction $\to_{\text{R}_p}$ is confluent.*

use the techniques of this thesis.

pos. variables			$x, y, z \ldots$
neg. variables			$\alpha, \beta, \gamma \ldots$
variables	κ	::=	$x \mid \alpha$
values	V	::=	$V_+ \mid t_\ominus$
pos. values	V_+	::=	$x \mid (V, V') \mid \{t_\ominus\}$
pos. terms	t_+	::=	$x \mid (V, V') \mid \{t_\ominus\} \mid \mu\alpha.c \mid \mu\hat{\mathrm{tp}}.c$
neg. terms	$t_\ominus$	::=	$\alpha \mid \mu(\kappa, \kappa').c \mid \mu\{\alpha\}.c \mid \mu x.c \mid \hat{\mathrm{tp}}$
commands	c	::=	$\langle t_+ \| t_\ominus \rangle \quad \left(\overset{\text{not.}}{=} \langle t_\ominus \| t_+ \rangle\right)$

(a) Syntax

$(\mu\alpha)$	$\langle \mu\alpha.c \| t_\ominus \rangle$	$\triangleright_{\mathrm{R}_p}$	$c[t_\ominus/\alpha]$
(μx)	$\langle V_+ \| \mu x.c \rangle$	$\triangleright_{\mathrm{R}_p}$	$c[V_+/x]$
$(\mu(\kappa, \kappa'))$	$\langle (V, V') \| \mu(\kappa, \kappa').c \rangle^\dagger$	$\triangleright_{\mathrm{R}_p}$	$c[V/\kappa, V'/\kappa']$
$(\mu\{\alpha\})$	$\langle \{t_\ominus\} \| \mu\{\alpha\}.c \rangle$	$\triangleright_{\mathrm{R}_p}$	$c[t_\ominus/\alpha]$
$(\mu\hat{\mathrm{tp}})$	$\langle \mu\hat{\mathrm{tp}}.\langle V_+ \| \hat{\mathrm{tp}} \rangle \| t_\ominus \rangle^\ddagger$	$\triangleright_{\mathrm{R}_p}$	$\langle V_+ \| t_\ominus \rangle$

†: when the polarities of values match the polarities of variables
‡: even when $\hat{\mathrm{tp}}$ appears freely in V_+

(b) Reductions

(η_+)	t_+	$\triangleright_{\mathrm{E}_p}$	$\mu\alpha.\langle t_+ \| \alpha \rangle$
$(\eta_\ominus)$	${t_\ominus}^\dagger$	$\triangleright_{\mathrm{E}_p}$	$\mu q.\langle q \| t_\ominus \rangle$
$(\eta_{\hat{\mathrm{tp}}})$	$c^\ddagger$	$\triangleright_{\mathrm{E}_p}$	$\langle \mu\hat{\mathrm{tp}}.c \| \hat{\mathrm{tp}} \rangle$
$(\mathrm{c}_{\hat{\mathrm{tp}}})$	$\langle W \| \mu q.\langle \mu\hat{\mathrm{tp}}.c \| t_\ominus \rangle \rangle^\dagger$	$\triangleright_{\mathrm{E}_p}$	$\langle \mu\hat{\mathrm{tp}}.\langle W \| \mu q.c \rangle \| t_\ominus \rangle$

†: where q is x, $\{\alpha\}$ or (κ, κ') and where W is V_+ or $\mu\hat{\mathrm{tp}}.c'$
‡: even when $\hat{\mathrm{tp}}$ appears freely in c

(c) Expansions

Figure III.2: $\mathbf{L}_{\mathbf{pol},\hat{\mathrm{tp}}^+}$: the calculus

Proof. The reduction $\triangleright_{\mathrm{R}_p}$ is left-linear and has no critical pairs. ∎

We now explain the specific features of the calculus $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^+}$.

III.2.1 Involutive negation

In the calculus $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^+}$, instead of six categories:

$$t_+, e_\ominus, t_\ominus, e_+, V, \pi,$$

we give only three:

$$t_+, \quad t_\ominus, \quad V.$$

We can do so by identifying $e_\ominus$ with t_+, e_+ with $t_\ominus$, and π with V. This means that the distinction between expressions and contexts is now informal, through the notation $\langle\, t \mid$ (for expressions) and $\mid t \,\rangle$ (for contexts). We have $\langle t \,\|\, u\rangle = \langle u \,\|\, t\rangle$ thanks to the informal notation:

$$\langle t_\ominus \,\|\, t_+\rangle \overset{\text{not.}}{=} \langle t_+ \,\|\, t_\ominus\rangle$$

Positive variables written x, y replace positive variables x^+ and negative co-variables $\alpha^\ominus$. Negative variables written α, β replace negative variables $x^\ominus$ and positive co-variables α^+. With κ we denote either of x or α. One consequence of identifying co-variables with variables is that now both can be bound in multiple locations by a single binder. Thus $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^+}$ is an extension of the classical calculus $\mathbf{L}_{\mathrm{n}}$.

In terms of types, this identification corresponds to introducing an operation $\cdot^\perp$ on formulae, defined such that we have $A^{\perp\perp} = A$, and that sequents of the following form:

$$\Gamma \vdash \Delta$$

are replaced by the following:

$$\vdash \Gamma^\perp, \Delta$$

In this context, indeed, the premises of the cut rule are interchange-

able:

$$\frac{\vdash t : A \mid \Gamma \qquad \vdash u : A^{\perp} \mid \Delta}{\langle t \parallel u \rangle : (\vdash \Gamma, \Delta)}$$

This operation corresponds in logic to a strictly involutive negation, in a manner inspired by Girard [Gir87, Gir91].

We will devote Chapter IV to explaining such an involutive negation with the idea of providing an explicit access to contexts as values with accessors, rather than as continuations. For now, although we could introduce a grammatical distinction between terms and contexts and an explicit type $\neg$ for mediating the two sides of a sequent, this would not make our decompositions more informative. This is because, as we will see, the CPS translation of $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^{+}}$ forces the identification of $\langle t_{+} \parallel t_{\ominus} \rangle$ with $\langle t_{\ominus} \parallel t_{+} \rangle$, and thus negation to be strictly involutive.

III.2.2 Abstraction and application

In $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^{+}}$, the pair of two values V, W is given with (V, W); pairing also denote the stacking operation, as we will see in this section. The symmetric construction is written $\mu(\kappa, \kappa').c$ and denotes the context for pattern-matching on pairs. We also explain how it is used to encode abstraction.

We chose to restrict any non-value to be of the form $\mu\alpha.c$ or $\mu\widehat{\mathrm{tp}}.c$, thus pairs are made of values. In order to constitute the pair of t and u in a left-to-right evaluation order, we proceed as follows:

$$(t, u) \stackrel{\text{def}}{=} \mu\alpha.\left\langle t \,\middle\|\, \mu x.\langle u \| \mu y.\langle (x, y) \| \alpha \rangle \right\rangle$$

We can similarly define the right-to-left pairing. (This restriction corresponds to Danos, Joinet and Schellinx's η restriction [DJS97] which extends Flanagan *et al.*'s A-normal form [FSDF93] and Hatcliff and Danvy's monadic normal form [HD94]. We further discuss this restriction in Section IV.6.1.)

Call-by-value abstraction and application

Following Curien and Herbelin [CH00], we retrieve the call-by-value abstraction λ^v and application $@_v$ by solving the following equations in the calculus $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^+}$:

$$\langle t_+ \,@_v\, u_+ \| t'_\ominus \rangle \rhd^*_{\mathrm{R}_p} \langle t_+ \| u_+ \odot t'_\ominus \rangle$$
$$\langle \lambda^v x.t_+ \| u_+ \odot t'_\ominus \rangle \rhd^*_{\mathrm{R}_p} \left\langle u_+ \middle\| \mu x.\langle t_+ \| t'_\ominus \rangle \right\rangle$$

In other words we define for t_+, u_+ positive terms and $u_\ominus$ a negative term:

$$\langle \lambda^v x.t_+ | \stackrel{\text{def}}{=} \left\langle \{\mu(x,\alpha).\langle t_+ \| \alpha \rangle\} \middle| \right.$$
$$| t_+ \odot u_\ominus \rangle \stackrel{\text{def}}{=} \left| \mu\{\alpha\}.\left\langle t_+ \middle\| \mu x.\langle \alpha \| (x, u_\ominus) \rangle \right\rangle \right\rangle$$
$$\langle t_+ \,@_v\, u_+ | \stackrel{\text{def}}{=} \left\langle \mu\alpha.\langle t_+ \| u_+ \odot \alpha \rangle \right| \right.$$

Proposition III.2. *We have:*

$$\langle (\lambda^v x.t_+) \,@_v\, V_+ \| u_\ominus \rangle \rhd^*_{\mathrm{R}_p} \langle t_+[V_+/x] \| u_\ominus \rangle$$
$$\lambda^v x.(V_+ \,@_v\, x) \simeq_{\mathrm{RE}_p} V_+$$

Proof. Indeed we have:

$$\begin{aligned}
&\langle (\lambda^v x.t_+) \,@_v\, V_+ \| u_\ominus \rangle \\
&\rhd_{\mathrm{R}_p} \langle \lambda^v x.t_+ \| V_+ \odot u_\ominus \rangle \\
&\rhd_{\mathrm{R}_p} \left\langle V_+ \middle\| \mu x.\langle t_+ \| u_\ominus \rangle \right\rangle \\
&\rhd_{\mathrm{R}_p} \langle t_+[V_+/x] \| u_\ominus \rangle
\end{aligned}$$

and:

$$
\begin{aligned}
&\lambda^v x.(V_+ \mathbin{@_v} x)\\
&= \Big\langle \mu(x,\alpha).\underline{\Big\langle \mu\alpha.\Big\langle V_+ \Big\| \mu\{\beta\}.\underline{\langle x \| \mu y.\langle \beta \| (y,\alpha)\rangle\rangle}\Big\rangle \Big\| \alpha\Big\rangle}\Big\rangle\\
&\to^*_{\mathrm{R}_p} \underline{\big\langle \mu(x,\alpha).\langle V_+ \| \mu\{\beta\}.\langle \beta \| (x,\alpha)\rangle\rangle\big\rangle}\\
&\rhd_{\mathrm{E}_p} \mu\gamma.\Big\langle \big\langle \mu(x,\alpha).\langle \underline{V_+} \| \mu\{\beta\}.\langle \beta \| (x,\alpha)\rangle\rangle\big\rangle \Big\| \gamma\Big\rangle\\
&\leftarrow_{\mathrm{R}_p} \mu\gamma.\Big\langle V_+ \Big\| \underline{\mu y.\Big\langle \big\langle \mu(x,\alpha).\langle y \| \mu\{\beta\}.\langle \beta \| (x,\alpha)\rangle\rangle\big\rangle \Big\| \gamma\Big\rangle}\Big\rangle\\
&\to_{\mathrm{E}_p} \mu\gamma.\Big\langle V_+ \Big\| \mu\{\beta\}.\Big\langle \{\beta\} \Big\| \mu y.\Big\langle \Big\{\mu(x,\alpha).\underline{\langle y \| \mu\{\beta\}.\langle \beta \| (x,\alpha)\rangle\rangle}\Big\} \Big\| \gamma\Big\rangle\Big\rangle\Big\rangle\\
&\to^*_{\mathrm{R}_p} \underline{\mu\gamma.\Big\langle V_+ \Big\| \mu\{\beta\}.\Big\langle \underline{\Big\{\mu(x,\alpha).\langle \beta \| (x,\alpha)\rangle\Big\}} \| \gamma\Big\rangle\Big\rangle}\\
&\leftarrow^*_{\mathrm{E}_p} V_+
\end{aligned}
$$

∎

Call-by-name abstraction and application

Also, as we saw already, the call-by-name application λ^n and abstraction $@_n$ can be derived by solving the rules of Krivine's abstract machine. We adopt the notation $\langle\, t_\ominus \parallel t_+ \,\rangle$ instead of $\langle\, t_+ \parallel t_\ominus \,\rangle$ to emphasize that negative terms are now seen as expressions.

$$
\begin{aligned}
\langle t_\ominus \mathbin{@_n} u_\ominus \| V_+\rangle &\rhd^*_{\mathrm{R}_p} \langle t_\ominus \| u_\ominus \cdot V_+\rangle\\
\langle \lambda^n\alpha.t_\ominus \| u_\ominus \cdot V_+\rangle &\rhd^*_{\mathrm{R}_p} \langle t_\ominus[u_\ominus/\alpha] \| V_+\rangle
\end{aligned}
$$

Thus, given $t_\ominus, u_\ominus$ negative terms and V_+ a positive value of $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^+}$, we define:

$$
\begin{aligned}
\langle \lambda^n\alpha.t_\ominus | &\stackrel{\text{def}}{=} \big\langle \mu(\alpha,x).\langle t_\ominus \| x\rangle\big|\\
|t_\ominus \cdot V_+\rangle &\stackrel{\text{def}}{=} |(t_\ominus, V_+)\rangle\\
\langle t_\ominus \mathbin{@_n} u_\ominus | &\stackrel{\text{def}}{=} \big\langle \mu x.\langle t_\ominus \| u_\ominus \cdot x\rangle\big|
\end{aligned}
$$

Following our conventions, negative variables are written α, β and

therefore the call-by-name abstraction is written $\lambda\alpha$. This notation at odds with the tradition is a consequence of the focus of this chapter on the distinction between positive and negative variables rather than the one between variables and co-variables.

Generic abstration and application

Notice that both definitions of abstraction and application stem from the more general definition:

$$\langle \lambda\kappa.t \,|\, \stackrel{\text{def}}{=} \left\langle \mu(\kappa,\kappa').\langle t \,\|\, \kappa'\rangle \right| \qquad \langle t\ u \,|\, \stackrel{\text{def}}{=} \left\langle \mu\kappa'.\langle u \,\|\, \mu\kappa.\langle t \,\|\, (\kappa,\kappa')\rangle\rangle \right|$$

by appropriately coercing the polarities.

III.2.3 Top-level context

The negative term $\hat{\mathrm{tp}}$ term is used to model the top-level context (when seen as a positive context). It can be bound in a special way by the binder $\mu\hat{\mathrm{tp}}.c$. Together they are used to implement control delimiters. The operator $\mu\hat{\mathrm{tp}}$ differs from a binder $\mu\alpha$ because $\hat{\mathrm{tp}}$ is not a standard variable. Therefore it is not subject to standard renaming conventions nor subject to a capture-avoiding substitution. Also $\langle \mu\hat{\mathrm{tp}}.c \,\|\, t_\ominus \rangle$ is not always a redex. Reduction is possible only when c is of the form $\langle V_+ \,\|\, \hat{\mathrm{tp}}\rangle$:

$$(\mu\hat{\mathrm{tp}}) \qquad\qquad \mu\hat{\mathrm{tp}}.\langle V_+ \,\|\, \hat{\mathrm{tp}}\rangle \rhd_{\mathrm{R}_p} V_+$$

Because the binding is not standard, this rule is applied even if $\hat{\mathrm{tp}}$ appears in V_+.

The binder $\mu\hat{\mathrm{tp}}.c$ is used to define abstract machines as pairs $c[\sigma]$ of a command c and a list $\sigma = (t^1_\ominus, \ldots, t^n_\ominus)$ of negative terms, inductively as follows:

$$\boxed{\begin{array}{c} c[\,] \stackrel{\text{def}}{=} c \\ c[t^1_\ominus, \ldots, t^n_\ominus] \stackrel{\text{def}}{=} \langle \mu\hat{\mathrm{tp}}.c \,\|\, t^1_\ominus\rangle[t^2_\ominus, \ldots, t^n_\ominus] \end{array}}$$

The operator $\mu\hat{\mathsf{tp}}$ therefore corresponds to the operation that lets the list grow. Note that the definition does not yield uniqueness of the decomposition of a command c under the form $c'[\sigma]$.

The list σ is accessed when the variable $\hat{\mathsf{tp}}$ comes in head position, that is to say, is encountered against a value V_+. Indeed the following expresses the $(\mu\hat{\mathsf{tp}})$ rule:

$$\langle V_+ \| \hat{\mathsf{tp}} \rangle [t_\ominus, \sigma] \to_{\mathrm{R}_p} \langle V_+ \| t_\ominus \rangle [\sigma]$$

Also, the following expansions express the rules $(\eta^{\hat{\mathsf{tp}}})$ and $(c^{\hat{\mathsf{tp}}})$:

$$c[\sigma, \sigma'] \to_{\mathrm{E}_p} c[\sigma, \hat{\mathsf{tp}}, \sigma']$$
$$c[\sigma, \mu q.(c'[t_\ominus]), \sigma'] \to_{\mathrm{E}_p} c[\sigma, \mu q.c', t_\ominus, \sigma']$$

We obtain Danvy and Filinski's shift and reset by solving the following equations:

$$\langle S \| \lambda x.t_+ \odot t_\ominus \rangle [u_\ominus, \sigma] \rhd_{\mathrm{R}_p} \langle t_+[\mathsf{k}_{t_\ominus}/x] \| u_\ominus \rangle [\sigma]$$
$$\langle \mathsf{k}_{t_\ominus} \| V_+ \odot u_\ominus \rangle [\sigma] \rhd_{\mathrm{R}_p} \langle V_+ \| t_\ominus \rangle [u_\ominus, \sigma]$$
$$\langle <t_+> \| t_\ominus \rangle [\sigma] \rhd_{\mathrm{R}_p} \langle t_+ \| \hat{\mathsf{tp}} \rangle [t_\ominus, \sigma]$$

In other words we define:

$$\langle \mathsf{k}_{t_\ominus} | \stackrel{\text{def}}{=} \langle \lambda^v x.\mu\hat{\mathsf{tp}}.\langle x \| t_\ominus \rangle |$$
$$\langle S | \stackrel{\text{def}}{=} \langle \lambda x.\mu\alpha.\langle x \| \mathsf{k}_{t_\ominus} \odot \hat{\mathsf{tp}} \rangle |$$
$$\langle <t_+> | \stackrel{\text{def}}{=} \langle \mu\hat{\mathsf{tp}}.\langle t_+ \| \hat{\mathsf{tp}} \rangle |$$

We have by definition:

$$\langle <V> \| t_\ominus \rangle \rhd_{\mathrm{R}_p} \langle V \| t_\ominus \rangle$$

Also, let $| t_\ominus(\alpha) \rangle$ be any negative term that has an adjoint positive

term $\langle E(x) |$ in the following sense:

$$\langle E(x) \| \alpha \rangle \rhd^*_{\mathrm{R}_p} \langle x \| t_\ominus(\alpha) \rangle$$

$\mathcal{S}$ and $<\cdot>$ implement Danvy and Filinski's shift and reset in the following sense:

Proposition III.3. *One has:*

$$\langle <E(\mathcal{S}\ @_v\ \lambda x.t_+)> \| u_\ominus \rangle \to^*_{\mathrm{R}_p} \langle <t_+[\mathsf{k}_{t_\ominus(\hat{\mathsf{tp}})}/x]> \| u_\ominus \rangle$$
$$\mathsf{k}_{t_\ominus(\hat{\mathsf{tp}})} \simeq_{\mathrm{RE}_p} \lambda y.<E(y)>$$

Proof. Indeed, we have:

$$\begin{aligned}
&\langle <E(\mathcal{S}\ @_v\ \lambda x.t_+)> \| u_\ominus \rangle \\
&= \underline{\langle E(\mathcal{S}\ @_v\ \lambda x.t_+) \| \hat{\mathsf{tp}} \rangle}[u_\ominus] \\
&\to^*_{\mathrm{R}_p} \underline{\langle \mathcal{S}\ @_v\ \lambda x.t_+ \| t_\ominus(\hat{\mathsf{tp}}) \rangle}[u_\ominus] \\
&\to_{\mathrm{R}_p} \underline{\langle \mathcal{S} \| \lambda x.t_+ \odot t_\ominus(\hat{\mathsf{tp}}) \rangle}[u_\ominus] \\
&\to_{\mathrm{R}_p} \underline{\langle \lambda x.t_+ \| \mathsf{k}_{t_\ominus(\hat{\mathsf{tp}})} \odot \hat{\mathsf{tp}} \rangle}[u_\ominus] \\
&\to_{\mathrm{R}_p} \langle t_+[\mathsf{k}_{t_\ominus(\hat{\mathsf{tp}})}/x] \| \hat{\mathsf{tp}} \rangle[u_\ominus] \\
&= \langle <t_+[\mathsf{k}_{t_\ominus(\hat{\mathsf{tp}})}/x]> \| u_\ominus \rangle
\end{aligned}$$

and:

$$\begin{aligned}
&\langle \mathsf{k}_{t_\ominus(\hat{\mathsf{tp}})} \| V \odot u_\ominus \rangle \\
&\rhd_{\mathrm{R}_p} \langle V \| t_\ominus(\hat{\mathsf{tp}}) \rangle[u_\ominus] \\
&\lhd^*_{\mathrm{R}_p} \langle E(V) \| \hat{\mathsf{tp}} \rangle[u_\ominus] \\
&= \langle <E(V)> \| u_\ominus \rangle \\
&\lhd^*_{\mathrm{R}_p} \langle \lambda y.<E(y)> \| V \odot u_\ominus \rangle
\end{aligned}$$

Thus by extensionality (Proposition III.2) we have indeed:

$$\mathsf{k}_{t_{\ominus}(\hat{\mathfrak{tp}})} \simeq_{\mathrm{RE}_p} \lambda y.\langle E(y)\rangle \qquad \blacksquare$$

III.2.4 The CPS translation

The introduce in Figure III.3 on the next page the calculus $\lambda_{\mathrm{v}}^{\times}$. It serves as the target of all the CPS translations of the chapter. The CPS translation of $\mathbf{L}_{\mathrm{pol},\hat{\mathfrak{tp}}^+}$ into $\lambda_{\mathrm{v}}^{\times}$ is defined in Figure III.4 on page 169.

We write the abstraction for a pair as follows:

$$\lambda(x,y).t \stackrel{\text{def}}{=} \lambda z.\mathsf{let}\ (x,y)\ \mathsf{be}\ z\ \mathsf{in}\ t\,.$$

Also, some decompositions are going to be stated modulo the linear reductions $\rhd_{\mathrm{admin}}$ that we call administrative. As a consequence of the following lemma, the reduction $\rhd_{\mathrm{admin}_\lambda}$ is redundant from the point of view of equivalences.

Lemma III.4. *In $\lambda_v^{\times}$, for any M, N we have:*

$$(\lambda q.M)\ N \simeq_{\mathrm{RE}_p} \mathsf{let}\ q\ \mathsf{be}\ N\ \mathsf{in}\ M$$

Proof. Indeed we have:

$$\begin{aligned}
&(\lambda q.M)\ \underline{N}\\
&\to_{\mathrm{E}_\lambda} (\lambda q.M)\ (\underline{\mathsf{let}\ q\ \mathsf{be}\ N\ \mathsf{in}\ q})\\
&\rhd_{\mathrm{E}_\lambda} \mathsf{let}\ q\ \mathsf{be}\ N\ \mathsf{in}\ (\lambda q.M)\ q\\
&\to_{\mathrm{R}_\lambda} \mathsf{let}\ q\ \mathsf{be}\ N\ \mathsf{in}\ M
\end{aligned} \qquad \blacksquare$$

Corollary III.5. *If $M \rhd_{\mathrm{admin}_\lambda} N$, then $M \simeq_{\mathrm{RE}_\lambda} N$.*

We now state that the translation is substitutive:

Lemma III.6. *For p a command or a term, x and α variables, V a*

$$V, V' ::= x \mid \lambda x.M \mid (V, V')$$
$$M, N ::= V \mid M\,N \mid \mathsf{let}\ x\ \mathsf{be}\ M\ \mathsf{in}\ N \mid \mathsf{let}\ (x, y)\ \mathsf{be}\ M\ \mathsf{in}\ N$$

(a) Syntax

$$\begin{aligned}
(\lambda x.M)\ V &\rhd_{\mathrm{R}_\lambda} M[V/x] \\
\mathsf{let}\ x\ \mathsf{be}\ V\ \mathsf{in}\ M &\rhd_{\mathrm{R}_\lambda} M[V/x] \\
\mathsf{let}\ (x, y)\ \mathsf{be}\ (V, V')\ \mathsf{in}\ M &\rhd_{\mathrm{R}_\lambda} M[V/x, V'/y]
\end{aligned}$$

(b) Reductions

$$E\,\Box ::= \mathsf{let}\ q\ \mathsf{be}\ \Box\ \mathsf{in}\ M \mid \mathsf{let}\ q\ \mathsf{be}\ M\ \mathsf{in}\ \Box \mid \Box\,M \mid M\,\Box$$
where q stands for x and (x, y).

(c) Linear evaluation contexts

$$\begin{aligned}
V &\rhd_{\mathrm{E}_\lambda} \lambda x.(V\ x) \\
M &\rhd_{\mathrm{E}_\lambda} \mathsf{let}\ q\ \mathsf{be}\ M\ \mathsf{in}\ q \\
E[\mathsf{let}\ q\ \mathsf{be}\ M\ \mathsf{in}\ u] &\rhd_{\mathrm{E}_\lambda} \mathsf{let}\ q\ \mathsf{be}\ M\ \mathsf{in}\ E[u] \\
\lambda x.\mathsf{let}\ q\ \mathsf{be}\ V\ \mathsf{in}\ M &\rhd_{\mathrm{E}_\lambda} \mathsf{let}\ q\ \mathsf{be}\ V\ \mathsf{in}\ \lambda x.M
\end{aligned}$$

(d) Expansions

$$\begin{aligned}
(\lambda x.N)\ M &\rhd_{\mathrm{admin}_\lambda} \mathsf{let}\ x\ \mathsf{be}\ M\ \mathsf{in}\ N \\
\mathsf{let}\ x\ \mathsf{be}\ M\ \mathsf{in}\ E[x] &\rhd_{\mathrm{admin}_\lambda} E[M]
\end{aligned}$$

(e) Administrative reduction

Figure III.3: The call-by-value calculus with pairs $\lambda_{\mathrm{v}}^{\times}$.

$$\textsc{val}[\![x]\!] \stackrel{\text{def}}{=} x$$
$$\textsc{val}[\![(V, V')]\!] \stackrel{\text{def}}{=} (\textsc{val}[\![V]\!], \textsc{val}[\![V']\!])$$
$$\textsc{val}[\![\{t_\ominus\}]\!] \stackrel{\text{def}}{=} \textsc{cps}[\![t_\ominus]\!]$$
$$\textsc{val}[\![t_\ominus]\!] \stackrel{\text{def}}{=} \textsc{cps}[\![t_\ominus]\!]$$

(a) Translation $\textsc{val}[\![\cdot]\!] : V \mapsto V$

$$\textsc{cps}[\![V_+]\!](M) \stackrel{\text{def}}{=} M\ \textsc{val}[\![V_+]\!]$$
$$\textsc{cps}[\![\mu\alpha.c]\!](M) \stackrel{\text{def}}{=} \mathsf{let}\ \alpha\ \mathsf{be}\ M\ \mathsf{in}\ \textsc{cps}[\![c]\!]$$
$$\textsc{cps}[\![\mu\widehat{\mathsf{tp}}.c]\!](M) \stackrel{\text{def}}{=} M\ \textsc{cps}[\![c]\!]$$

(b) Translation $\textsc{cps}[\![\cdot]\!](M) : t_+ \mapsto M$

$$\textsc{cps}[\![\alpha]\!] \stackrel{\text{def}}{=} \alpha$$
$$\textsc{cps}[\![\widehat{\mathsf{tp}}]\!] \stackrel{\text{def}}{=} \lambda x.x$$
$$\textsc{cps}[\![\mu x.c]\!] \stackrel{\text{def}}{=} \lambda x.\textsc{cps}[\![c]\!]$$
$$\textsc{cps}[\![\mu(\kappa, \kappa').c]\!] \stackrel{\text{def}}{=} \lambda(\kappa, \kappa').\textsc{cps}[\![c]\!]$$
$$\textsc{cps}[\![\mu\{\alpha\}.c]\!] \stackrel{\text{def}}{=} \lambda\alpha.\textsc{cps}[\![c]\!]$$

(c) Translation $\textsc{cps}[\![\cdot]\!] : t_\ominus \mapsto V$

$$\textsc{cps}[\![\langle t_+ \,\|\, u_\ominus\rangle]\!] \stackrel{\text{def}}{=} \textsc{cps}[\![t_+]\!](\textsc{cps}[\![u_\ominus]\!])$$

(d) Translation $\textsc{cps}[\![\cdot]\!] : c \mapsto M$

Figure III.4: The translation $\textsc{cps}[\![\cdot]\!]$ from $\mathbf{L}_{\mathbf{pol},\widehat{\mathsf{tp}}^+}$ to $\boldsymbol{\lambda}^{\times}_{\mathbf{v}}$.

value, V_+ a positive value and $t_\ominus$ a negative term, one has:

$$\begin{aligned}\textsc{val}[\![V[V_+/x]]\!] &= \textsc{val}[\![V]\!][\textsc{val}[\![V_+]\!]/x]\\ \textsc{cps}[\![p[V_+/x]]\!] &= \textsc{cps}[\![p]\!][\textsc{val}[\![V_+]\!]/x]\\ \textsc{cps}[\![p[t_\ominus/\alpha]]\!] &= \textsc{cps}[\![p]\!][\textsc{cps}[\![t_\ominus]\!]/\alpha]\end{aligned}$$

(In the case of $p = t_+$, equality is meant point-wise.)

Proof. Follows from a straightforward induction over V and p. ∎

As a consequence we have:

Proposition III.7. *The translation $\textsc{cps}[\![\cdot]\!] : \boldsymbol{L}_{pol,\hat{\mathrm{tp}}^\ominus} \to \lambda_\nu^\times$ is a simulation: if $c \to_{\mathrm{R}_p} c'$ then $\textsc{cps}[\![c]\!] \to^+_{\mathrm{R}_\lambda} \textsc{cps}[\![c']\!]$.*

Proof. The result holds for $\triangleright_{\mathrm{R}_p}$ by case analysis. In particular we have:

$$\begin{aligned}\textsc{cps}[\![\langle \mu\hat{\mathrm{tp}}.\langle V_+ \| \hat{\mathrm{tp}}\rangle \| t_\ominus\rangle]\!] &= \textsc{cps}[\![t_\ominus]\!]\ (\underline{\lambda x.x\ \textsc{val}[\![V_+]\!]})\\ &\to_{\mathrm{R}_\lambda} \textsc{cps}[\![t_\ominus]\!]\ \textsc{val}[\![V_+]\!]\\ &= \textsc{cps}[\![\langle V_+ \| t_\ominus\rangle]\!]\end{aligned}$$

We conclude by induction on the definition of $\to_{\mathrm{R}_p}$. ∎

Proposition III.8. *The translation $\textsc{cps}[\![\cdot]\!] : \boldsymbol{L}_{pol,\hat{\mathrm{tp}}^+} \to \lambda_\nu^\times$ preserves equivalences:*

- *if $t \simeq_{\mathrm{RE}_p} t'$ then $\textsc{cps}[\![t]\!](M) \simeq_{\mathrm{RE}_\lambda} \textsc{cps}[\![t']\!](M)$ for all terms M.*
- *if $c \simeq_{\mathrm{RE}_p} c'$ then $\textsc{cps}[\![c]\!] \simeq_{\mathrm{RE}_\lambda} \textsc{cps}[\![c']\!]$.*

Proof. By Proposition III.7 we are left with proving that if $p \triangleright_{\mathrm{E}_p} p$ then $\textsc{cps}[\![t]\!] \simeq_{\mathrm{RE}_\lambda} \textsc{cps}[\![u]\!]$. *Case of the rule ($c_{\hat{\mathrm{tp}}}$).* We have to show:

$$\textsc{cps}[\![\langle W \| \mu q.\langle \mu\hat{\mathrm{tp}}.c \| t_\ominus\rangle\rangle]\!] \simeq_{\mathrm{RE}_\lambda} \textsc{cps}[\![\langle \mu\hat{\mathrm{tp}}.\langle W \| \mu q.c\rangle \| t_\ominus\rangle]\!]$$

where q is x, $\{\alpha\}$ or (κ, κ') and where W is V_+ or $\mu\hat{\mathrm{tp}}.c'$. We have:

$$\begin{aligned}\textsc{cps}[\![\langle W \| \mu q.\langle \mu\hat{\mathrm{tp}}.c \| t_\ominus\rangle\rangle]\!] &= \textsc{cps}[\![W]\!]\big(\lambda q.(\textsc{cps}[\![t_\ominus]\!]\ \textsc{cps}[\![c]\!])\big)\\ \textsc{cps}[\![\langle \mu\hat{\mathrm{tp}}.\langle W \| \mu q.c\rangle \| t_\ominus\rangle]\!] &= \textsc{cps}[\![t_\ominus]\!]\ \textsc{cps}[\![W]\!](\lambda q.\textsc{cps}[\![c]\!])\end{aligned}$$

q is x, α or (κ, κ') in the target.

We have either $\textsc{cps}[\![W]\!](M) = M\ \textsc{val}[\![V]\!]$ if $W = V$, or $\textsc{cps}[\![\mu\hat{\mathsf{tp}}.c]\!](M) = M\ \textsc{cps}[\![c]\!]$ if $W = \mu\hat{\mathsf{tp}}.c$. Let us deal with both cases at once by taking N such that $\textsc{cps}[\![W]\!](M) = M\,N$. We have:

$$\begin{aligned}
&\textsc{cps}[\![\langle W \,\|\, \mu q.\langle \mu\hat{\mathsf{tp}}.c \,\|\, t_\ominus\rangle\rangle]\!] \\
&= \lambda q.(\textsc{cps}[\![t_\ominus]\!]\ \textsc{cps}[\![c]\!])\ N \\
&\simeq_{\mathrm{RE}_\lambda} \underline{\mathsf{let}\ q\ \mathsf{be}\ N\ \mathsf{in}\ \textsc{cps}[\![t_\ominus]\!]\ \textsc{cps}[\![c]\!]} \\
&\lhd_{\mathrm{E}_\lambda} \textsc{cps}[\![t_\ominus]\!]\ (\underline{\mathsf{let}\ q\ \mathsf{be}\ N\ \mathsf{in}\ \textsc{cps}[\![c]\!]}) \\
&\simeq_{\mathrm{RE}_\lambda} \textsc{cps}[\![t_\ominus]\!]\ (\lambda q.\textsc{cps}[\![c]\!]\ N) \\
&= \textsc{cps}[\![\langle \mu\hat{\mathsf{tp}}.\langle W \,\|\, \mu q.c\rangle \,\|\, t_\ominus\rangle]\!]
\end{aligned}$$

The other cases are straightforward. ∎

III.2.5 The $\lambda\mu\hat{\mathsf{tp}}_\nu$ calculus

The goal is to decompose the CPS translation of Ariola, Herbelin and Sabry's $\lambda\mu\hat{\mathsf{tp}}_\nu$ calculus [AHS04]. The calculus is defined in Figure III.5 on the next page. Its CPS translation is defined in Figure III.6 on page 173 along the lines of Herbelin and Ghilezan [HG08].[2]

Ariola, Herbelin and Sabry define S and $<\cdot>$ as follows:

$$\begin{aligned}
S\,M &\stackrel{\text{def}}{=} \mu\alpha.[\hat{\mathsf{tp}}](M\,\lambda x.\mu\hat{\mathsf{tp}}.[\alpha]\,x) \\
<M> &\stackrel{\text{def}}{=} \mu\hat{\mathsf{tp}}.[\hat{\mathsf{tp}}]\,M
\end{aligned}$$

They show that the calculus $\lambda\mu\hat{\mathsf{tp}}_\nu$ is in an equational correspondence with Kameyama and Hasegawa's complete axiomatisation [KH03] of Danvy and Filinski's shift and reset.

[2] We differ from [HG08] essentially in the treatment of administrative redexes. Herbelin and Ghilezan define $\textsc{cps}_{\mathrm{v}}[\![t\,u]\!](M) = \textsc{cps}_{\mathrm{v}}[\![t]\!](\lambda x.\textsc{cps}_{\mathrm{v}}[\![u]\!](\lambda y.x\,(y, M)))$ and $\textsc{cps}_{\mathrm{v}}[\![\mu\alpha.c]\!](M) = \textsc{cps}_{\mathrm{v}}[\![c]\!][M/\alpha]$. But then we lose the fact that M is used linearly. In particular this can lead to duplication of code. Linearity in M is necessary if we want that the second part of the decomposition goes well (Section III.3).

$$V ::= x \mid \lambda x.t \qquad e ::= \alpha \mid \hat{\mathsf{tp}}$$
$$t ::= V \mid t\ t \mid \mu\hat{\mathsf{tp}}.c \mid \mu\alpha.c \qquad c ::= [e]\ t$$

(a) Values, terms, co-variables, commands.

$$\begin{array}{lll}
(\lambda x.t)\ V & \triangleright_{\mathrm{R}_\nu} & t[V/x] \\
(\mu\alpha.c)\ t & \triangleright_{\mathrm{R}_\nu} & \mu\beta.c[[\beta]\ (\Box\ t)/\alpha] \quad \beta \text{ fresh} \\
V\ (\mu\alpha.c) & \triangleright_{\mathrm{R}_\nu} & \mu\beta.c[[\beta]\ (V\ \Box)/\alpha] \quad \beta \text{ fresh} \\
[e]\ \mu\alpha.c & \triangleright_{\mathrm{R}_\nu} & c[e/\alpha] \\
\mu\hat{\mathsf{tp}}.[\hat{\mathsf{tp}}]\ V & \triangleright_{\mathrm{R}_\nu} & V \quad \text{even if } \hat{\mathsf{tp}} \text{ occurs in } V
\end{array}$$

The *structural substitution* $[[\beta]\ E\ \Box/\alpha]$ replaces all occurrences of the form $[\alpha]\ t$ by $[\beta]\ E[t]$.

(b) Reduction

$$\begin{array}{rcl}
c & \triangleright_{\mathrm{E}_\nu} & [\hat{\mathsf{tp}}]\ \mu\hat{\mathsf{tp}}.c \\
t & \triangleright_{\mathrm{E}_\nu} & \mu\alpha.[\alpha]\ t \\
V & \triangleright_{\mathrm{E}_\nu} & \lambda x.(V\ x) \\
(\lambda x.\mu\hat{\mathsf{tp}}.[e]\ t)\ (\mu\hat{\mathsf{tp}}.c) & \triangleright_{\mathrm{E}_\nu} & \mu\hat{\mathsf{tp}}.[e]\ ((\lambda x.t)\ (\mu\hat{\mathsf{tp}}.c)) \\
(\lambda x.\mu\alpha.[e]\ t)\ u & \triangleright_{\mathrm{E}_\nu} & \mu\alpha.[e]\ ((\lambda x.t)\ u) \\
E_\nu[t] & \triangleright_{\mathrm{E}_\nu} & (\lambda x.E_\nu[x])\ t
\end{array}$$

where $E_\nu\ \Box ::= \Box \mid E_\nu[\Box\ t] \mid E_\nu[V\ \Box]$.

(c) Expansions

Figure III.5: The $\lambda\mu\hat{\mathsf{tp}}_\nu$ calculus

$$\begin{aligned}
\mathrm{VAL}_{\mathrm{v}}[\![x]\!] &\stackrel{\text{def}}{=} x \\
\mathrm{VAL}_{\mathrm{v}}[\![\lambda x.t]\!] &\stackrel{\text{def}}{=} \lambda(x,k).\mathrm{CPS}_{\mathrm{v}}[\![t]\!](k)
\end{aligned}$$

(a) $\mathrm{VAL}_{\mathrm{v}}[\![\cdot]\!] : V \mapsto V$

$$\begin{aligned}
\mathrm{CPS}_{\mathrm{v}}[\![[\alpha]\,t]\!] &\stackrel{\text{def}}{=} \mathrm{CPS}_{\mathrm{v}}[\![t]\!](\alpha) \\
\mathrm{CPS}_{\mathrm{v}}[\![[\hat{\mathrm{tp}}]\,t]\!] &\stackrel{\text{def}}{=} \mathrm{CPS}_{\mathrm{v}}[\![t]\!](\lambda x.x)
\end{aligned}$$

(b) $\mathrm{CPS}_{\mathrm{v}}[\![\cdot]\!] : c \mapsto M$

$$\begin{aligned}
\mathrm{CPS}_{\mathrm{v}}[\![V]\!](M) &\stackrel{\text{def}}{=} M\ \mathrm{VAL}_{\mathrm{v}}[\![V]\!] \\
\mathrm{CPS}_{\mathrm{v}}[\![t\,u]\!](M) &\stackrel{\text{def}}{=} \text{let } k \text{ be } M \text{ in } \mathrm{CPS}_{\mathrm{v}}[\![t]\!]\big(\lambda x.\mathrm{CPS}_{\mathrm{v}}[\![u]\!](\lambda y.x\,(y,k))\big) \\
\mathrm{CPS}_{\mathrm{v}}[\![\mu\alpha.c]\!](M) &\stackrel{\text{def}}{=} \text{let } \alpha \text{ be } M \text{ in } \mathrm{CPS}_{\mathrm{v}}[\![c]\!] \\
\mathrm{CPS}_{\mathrm{v}}[\![\mu\hat{\mathrm{tp}}.c]\!](M) &\stackrel{\text{def}}{=} M\ \mathrm{CPS}_{\mathrm{v}}[\![c]\!]
\end{aligned}$$

(c) $\mathrm{CPS}_{\mathrm{v}}[\![\cdot]\!](M) : t \mapsto M$

Figure III.6: The translation $\mathrm{CPS}_{\mathrm{v}}[\![\cdot]\!]$ of the $\lambda\mu\hat{\mathrm{tp}}_{\nu}$ calculus

We defined in Section III.2.2 the call-by-value abstraction and application λ^{ν} and $@_{\nu}$. This induces a translation $\mathrm{POS}[\![\cdot]\!]$ from terms and commands of $\lambda\mu\hat{\mathrm{tp}}_{\nu}$ to terms and commands of $\mathbf{L}_{\mathrm{pol},\hat{\mathrm{tp}}^{+}}$:

$$\boxed{\mathrm{POS}[\![\cdot]\!] : \lambda\mu\hat{\mathrm{tp}}_{\nu} \to \mathbf{L}_{\mathrm{pol},\hat{\mathrm{tp}}^{+}}}$$

(it takes $[e]$ to $\mid e \rangle$, x to x, $\mu\alpha$ to $\mu\alpha$, $\mu\hat{\mathrm{tp}}$ to $\mu\hat{\mathrm{tp}}$, λx to $\lambda^{\nu}x$ and application to $@_{\nu}$). We can notice that this translation preserves our definitions of $\mathcal{S}$ and $<\cdot>$, with $\lambda x.\mu\hat{\mathrm{tp}}.[\alpha]\,x$ playing the role of k_{α} in $\lambda\mu\hat{\mathrm{tp}}_{\nu}$. We also have:

Proposition III.9. *The translation* $\mathrm{POS}[\![\cdot]\!]$ *preserves equivalences: if* $p \simeq_{\mathrm{RE}_{\nu}} p'$ *then* $\mathrm{POS}[\![p]\!] \simeq_{\mathrm{RE}_{p}} \mathrm{POS}[\![p']\!]$ *where* p *is a term or a command.*

Proof. It follows from the base cases which are of the form $p \rhd_{\mathrm{RE}_{\nu}} p'$.
Case $V \rhd_{\mathrm{E}_{\nu}} \lambda x.(V\,x)$. In Proposition III.2 we proved $\lambda^{\nu}x.(V_{+}\,@_{\nu}\,x) \simeq_{\mathrm{RE}_{p}}$

V_+.
Case $(\lambda x.\mu\hat{\mathrm{tp}}.[e]\,t)\,(\mu\hat{\mathrm{tp}}.c) \rhd_{\mathrm{E}_v} \mu\hat{\mathrm{tp}}.[e]\,((\lambda x.t)\,(\mu\hat{\mathrm{tp}}.c))$. This amounts to showing:

$$\left\langle \mu\hat{\mathrm{tp}}.c \middle\| \mu x.\left\langle \mu\hat{\mathrm{tp}}.\langle \mathrm{POS}[\![t]\!] \| e\rangle \middle\| \alpha\right\rangle\right\rangle \simeq_{\mathrm{RE}_p} \left\langle \mu\hat{\mathrm{tp}}.\left\langle \mu\hat{\mathrm{tp}}.c \middle\| \mu x.\langle \mathrm{POS}[\![t]\!] \| e\rangle\right\rangle \middle\| \alpha\right\rangle.$$

This is an instance of the rule $(\mathrm{c}_{\hat{\mathrm{tp}}})$.
Case $(\lambda x.\mu\alpha.[e]\,t)\,u \rhd_{\mathrm{E}_v} \mu\alpha.[e]\,((\lambda x.t)\,u)$. This amounts to showing:

$$\left\langle \mathrm{POS}[\![u]\!] \middle\| \mu x.\left\langle \mu\alpha.\langle \mathrm{POS}[\![t]\!] \| e\rangle \middle\| \alpha\right\rangle\right\rangle \simeq_{\mathrm{RE}_p} \left\langle \mu\alpha.\left\langle \mathrm{POS}[\![u]\!] \middle\| \mu x.\langle \mathrm{POS}[\![t]\!] \| e\rangle\right\rangle \middle\| \alpha\right\rangle$$

which is immediate.
All the other cases are straightforward. ∎

Proposition III.10. *The translation* $\mathrm{CPS}_{\mathrm{v}}[\![\cdot]\!]$ *decomposes as follows: for* p *a command or a term of* $\lambda\mu\hat{\mathrm{tp}}_v$*, we have* $\mathrm{CPS} \circ \mathrm{POS}[\![p]\!] = \mathrm{CPS}_{\mathrm{v}}[\![p]\!]$.

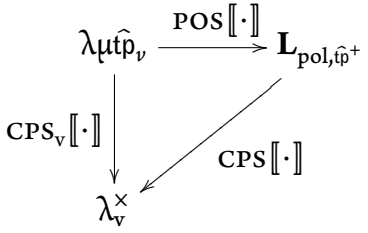
Proof. By a straightforward induction on p. In particular we have indeed:

$$\begin{aligned}
&\mathrm{VAL}[\![\mathrm{POS}[\![\lambda x.t]\!]]\!] \\
&= \mathrm{VAL}[\![\{\mu(x,\alpha).\langle \mathrm{POS}[\![t]\!] \| \alpha\rangle\}]\!] \\
&= \lambda(x,\alpha).\mathrm{CPS} \circ \mathrm{POS}[\![t]\!](\alpha)
\end{aligned}$$

and:

$$\begin{aligned}
&\mathrm{CPS}[\![\mathrm{POS}[\![t\,u]\!]]\!](M) \\
&= \mathrm{CPS}[\![\mu\alpha.\left\langle \mathrm{POS}[\![t]\!] \middle\| \mu\{\beta\}.\langle \mathrm{POS}[\![u]\!] \| \mu x.\langle \beta \| (x,\alpha)\rangle\right\rangle\right\rangle]\!](M) \\
&= \text{let } \alpha \text{ be } M \text{ in } \mathrm{CPS} \circ \mathrm{POS}[\![t]\!]\big(\lambda\beta.\mathrm{CPS} \circ \mathrm{POS}[\![u]\!](\lambda x.\beta\,(x,\alpha))\big)
\end{aligned}$$ ∎

This proves that the first diagram of our decomposition commutes.



We have therefore reconstructed the following result:

Corollary III.11. *The translation* $\mathrm{CPS_v}[\![\cdot]\!] : \lambda\mu\hat{\mathrm{tp}}_v \to \lambda_v^{\times}$ *preserves equivalences:*

- *If* $t \simeq_{\mathrm{RE}_v} t'$ *then* $\mathrm{CPS_v}[\![t]\!](M) \simeq_{\mathrm{RE}_\lambda} \mathrm{CPS_v}[\![t']\!](M)$ *for all terms* M.
- *If* $c \simeq_{\mathrm{RE}_v} c'$ *then* $\mathrm{CPS_v}[\![c]\!] \simeq_{\mathrm{RE}_\lambda} \mathrm{CPS_v}[\![c']\!]$.

We come close to having a simulation result as well, but do not, as is standard with the structural substitution $[[\beta]\, E\, \Box/\alpha]$ used in the calculus $\lambda\mu\hat{\mathrm{tp}}_v$.

III.2.6 The design space of call-by-name delimited control

Implementing delimited control operators in call by name, based on the calculus $\mathbf{L}_{\mathrm{pol},\hat{\mathrm{tp}}^+}$, amounts to finding ways of letting the $\mu\hat{\mathrm{tp}}$ and $\hat{\mathrm{tp}}$ operators interact with the terms of the call-by-name calculus. The challenge resides in the fact that $\mu\hat{\mathrm{tp}}$ has a positive polarity, while we would like to use it to bind a positive value representing a stack. We have to imagine ways of integrating $\mu\hat{\mathrm{tp}}$ and $\hat{\mathrm{tp}}$ together with call-by-name abstractions and applications. We count four distinct ways.

1) Simulating call by name in call by value

We can re-define abstraction and application as positive terms, by simulating call by name into call by value using thunks (see Hatcliff and Danvy [HD97]). This leads to Biernacka and Biernacki's call-by-name variant of shift and reset [BB09]. As is well known, due to matters orthogonal to delimited control, this model is not extensional.

2) Casting stacks into negative terms

This solution encodes call by name with λ^n and $@_n$. The top-level context $\hat{\mathrm{tp}}$ is bound to a positive value V_+ representing a stack by casting it into a negative term, as follows:

$$\langle \mu\hat{\mathrm{tp}}.c \,\|\, \mu\{\alpha\}.\langle V_+ \,\|\, \alpha\rangle\rangle$$

This suggests that we define a negative term $\mu\hat{\mathrm{tp}}^{\ominus}.c$ and a positive term $\hat{\mathrm{tp}}^{\ominus}$ by solving the following equations:

$$\langle \mu\hat{\mathrm{tp}}^{\ominus}.c \,\|\, V_+ \rangle \rhd^*_{\mathrm{R}_p} c[\mu\{\alpha\}.\langle V_+ \,\|\, \alpha\rangle]$$
$$\langle t_\ominus \,\|\, \hat{\mathrm{tp}}^{\ominus}\rangle[\mu\{\alpha\}.\langle V_+ \,\|\, \alpha\rangle] \rhd^*_{\mathrm{R}_p} \langle t_\ominus \,\|\, V_+\rangle$$

In other words, given $t_\ominus$ a negative term and c a command of $\mathbf{L}_{\mathrm{pol},\hat{\mathrm{tp}}^+}$, we define:

$$\langle \mu\hat{\mathrm{tp}}^{\ominus}.c| \stackrel{\text{def}}{=} \left\langle \mu x.\langle \mu\hat{\mathrm{tp}}.c \,\middle\|\, \mu\{\alpha\}.\langle \alpha \,\|\, x\rangle\right\rangle\middle|$$
$$|\hat{\mathrm{tp}}^{\ominus}\rangle \stackrel{\text{def}}{=} \left|\mu\alpha.\langle\{\alpha\} \,\|\, \hat{\mathrm{tp}}\rangle\right\rangle$$

With these definitions, the control delimiter is vacuous:

$$\begin{aligned}
<t_\ominus>_\ominus &\stackrel{\text{def}}{=} \mu\hat{\mathrm{tp}}^{\ominus}.\langle t_\ominus \,\|\, \hat{\mathrm{tp}}^{\ominus}\rangle \\
\langle <t_\ominus>_\ominus \,\|\, V_+\rangle &\rhd_{\mathrm{R}_p} \left\langle \mu\hat{\mathrm{tp}}.\underline{\langle t_\ominus \,\|\, \mu\alpha.\langle\{\alpha\} \,\|\, \hat{\mathrm{tp}}\rangle\rangle} \,\middle\|\, \mu\{\alpha\}.\langle \alpha \,\|\, V_+\rangle\right\rangle \\
&\to_{\mathrm{R}_p} \left\langle \mu\hat{\mathrm{tp}}.\langle\{t_\ominus\} \,\|\, \hat{\mathrm{tp}}\rangle \,\middle\|\, \mu\{\alpha\}.\langle \alpha \,\|\, x\rangle\right\rangle \\
&\rhd_{\mathrm{R}_p} \left\langle \{t_\ominus\} \,\middle\|\, \mu\{\alpha\}.\langle \alpha \,\|\, x\rangle\right\rangle \\
&\rhd_{\mathrm{R}_p} t_\ominus
\end{aligned}$$

As we will see in the next section, this gives the continuation-passing-style translation of Herbelin and Ghilezan's $\lambda\mu\tilde{\mu}\hat{\mathrm{tp}}_n$ calculus [HG08].

3) Un-casting stacks unto negative terms

Alternatively, again in the context of λ^n and $@_n$, we can bind $\hat{\mathrm{tp}}$ to a positive value V_+ representing a stack by un-casting it as a negative term, as follows:

$$\left\langle \mu\{\alpha\}.\langle \mu\hat{\mathrm{tp}}.c \,\|\, \alpha\rangle \,\middle\|\, V_+\right\rangle$$

This makes sense only if we assume that it is of the form $\{t_\ominus\}$. One reason for such a stack $\{t_\ominus\}$ to come up is if we consider, together with λ^n and $@_n$, positive constants n, … of some base type. These

constants are cast in a negative setting under the following form:

$$\bar{n} \stackrel{\text{def}}{=} \mu\{\alpha\}.\langle n \| \alpha \rangle$$

Thus, the contexts of $\bar{n}$ are indeed of the form $\{t_\ominus\}$.

This suggests that we define operators $\hat{\mathrm{tp}}^{\ominus'}$ and $\mu\hat{\mathrm{tp}}^{\ominus'}$ by solving the following equations:

$$\langle \bar{n} \| \hat{\mathrm{tp}}^{\ominus'} \rangle [t_\ominus] \rhd^*_{\mathrm{R}_p} \langle n \| t_\ominus \rangle$$
$$\langle \mu\hat{\mathrm{tp}}^{\ominus'}.c \| \{t_\ominus\} \rangle \rhd^*_{\mathrm{R}_p} c[t_\ominus]$$

In other words we define:

$$|\hat{\mathrm{tp}}^{\ominus'}\rangle \stackrel{\text{def}}{=} |\{\hat{\mathrm{tp}}\}\rangle$$
$$\langle \mu\hat{\mathrm{tp}}^{\ominus'}.c| \stackrel{\text{def}}{=} \langle \mu\{\alpha\}.\langle \mu\hat{\mathrm{tp}}.c \| \alpha \rangle|$$

This defines a control delimiter as follows:

$$<t_\ominus>_{\ominus'} \stackrel{\text{def}}{=} \mu\hat{\mathrm{tp}}^{\ominus'}.\langle t_\ominus \| \hat{\mathrm{tp}}^{\ominus'} \rangle$$

This delimiter is not vacuous since we have:

$$\langle <\bar{n}>_{\ominus'} \| V_+ \rangle \rhd^*_{\mathrm{R}_p} \langle \bar{n} \| V_+ \rangle$$

but $<t_\ominus>_{\ominus'}$ for $t_\ominus \neq \bar{n}$ first computes $t_\ominus$. However, $<\cdot>_{\ominus'}$ is defined only on terms of an appropriate type, since the following term is stuck:

$$\langle <t_\ominus>_{\ominus'} \| t_\ominus \cdot V_+ \rangle$$

The constructors λ^n and $@_n$ together with the operators $\hat{\mathrm{tp}}^{\ominus'}$ and $\mu\hat{\mathrm{tp}}^{\ominus'}$ correspond to a system described by Kameyama and Tanaka [KT10, Section 8] where they already note that "*a reset-term is restricted to be of basic types*".

4) Dualising

Lastly, since $\hat{\mathrm{tp}}$ is negative and $\mu\hat{\mathrm{tp}}$ is positive, we can consider the former as an expression and the latter as a context. This means that $\mu\hat{\mathrm{tp}}$ binds expressions in call by name. Thus, $\hat{\mathrm{tp}}$ is a linear *read* operation:

$$\langle \hat{\mathrm{tp}} \| V_+ \rangle [t_\ominus] \rhd^*_{\mathrm{R}_p} \langle t_\ominus \| V_+ \rangle$$

The $\mu\hat{\mathrm{tp}}$ context implements a *store* operation:

$$\langle \mathsf{store}\, t_\ominus \,\mathsf{in}\, u_\ominus \| V_+ \rangle \rhd^*_{\mathrm{R}_p} \langle u_\ominus \| V_+ \rangle [t_\ominus]$$

Indeed, we can solve the above equation and obtain:

$$\langle \mathsf{store}\, t_\ominus \,\mathsf{in}\, u_\ominus | \stackrel{\text{def}}{=} \left\langle \mu\alpha.\langle t_\ominus \| \mu\hat{\mathrm{tp}}.\langle u_\ominus \| \alpha \rangle \rangle \right|$$

The call-by-name λ calculus with *read* and *store* operations appears to be new. It corresponds to the dual of *shift* and *reset*. (There is a formal duality between state and exceptions described by Dumas, Duval, Fousse and Reynaud [DDFR12], but no formal comparison has been established yet.)

III.2.7 The λμ$\hat{\mathrm{tp}}_n$ calculus

In this section we analyse in more details the CPS translation of Herbelin and Ghilezan's λμ$\hat{\mathrm{tp}}_n$ calculus [HG08]. We define a variant that satisfies the same[3] translation in Figure III.7 on the next page. The goal is to decompose the translation through the calculus $\mathbf{L}_{\mathrm{pol},\hat{\mathrm{tp}}^+}$, using the call-by-name abstraction and application λ^n and $@_n$ from Section III.2.2, as well as the operators $\hat{\mathrm{tp}}^\ominus$ and $\mu\hat{\mathrm{tp}}^\ominus$ from the previous section.

[3] We differ from [HG08] in that we applied an η expansion in the definition of $\mathrm{CPS}_\mathrm{n}[\![\mu\hat{\mathrm{tp}}^\ominus.c]\!]$. This leaves the denotational semantics of Herbelin and Ghilezan unchanged when the target calculus is in call by name, but it is important for meaningfulness when the target is in call by value.

$$t ::= \alpha \mid \lambda\alpha.t \mid t\ t \mid \mu\hat{\mathrm{tp}}^{\ominus}.c \mid \mu x.c$$
$$c ::= [x]\ t \mid [\hat{\mathrm{tp}}^{\ominus}]\ t$$

(a) Terms and commands

$$t \ \triangleright_{\mathrm{E}_n}\ \mu x.[x]\ t$$
$$t \ \triangleright_{\mathrm{E}_n}\ \lambda\alpha.(t\ \alpha)$$

(b) Expansions

$$(\lambda\alpha.t)\ u \ \triangleright_{\mathrm{R}_n}\ t[u/\alpha]$$
$$(\mu x.c)\ t \ \triangleright_{\mathrm{R}_n}\ \mu y.c[[y]\ (\Box\, t)/x] \quad y \text{ fresh}$$
$$[y]\ \mu x.c \ \triangleright_{\mathrm{R}_n}\ c[y/x]$$
$$\mu\hat{\mathrm{tp}}^{\ominus}.[\hat{\mathrm{tp}}^{\ominus}]\ t \ \triangleright_{\mathrm{R}_n}\ t \quad \text{even if } \hat{\mathrm{tp}}^{\ominus} \text{ occurs in } t$$

(c) Reductions

Figure III.7: The $\lambda\mu\hat{\mathrm{tp}}_n$ calculus

$$\mathrm{CPS_n}[\![\alpha]\!] \stackrel{\text{def}}{=} \alpha$$
$$\mathrm{CPS_n}[\![\lambda\alpha.t]\!] \stackrel{\text{def}}{=} \lambda(\alpha, x).\mathrm{CPS_n}[\![t]\!]\,x$$
$$\mathrm{CPS_n}[\![t\ u]\!] \stackrel{\text{def}}{=} \lambda x.\mathrm{CPS_n}[\![t]\!]\,(\mathrm{CPS_n}[\![u]\!], x)$$
$$\mathrm{CPS_n}[\![\mu x.c]\!] \stackrel{\text{def}}{=} \lambda x.\mathrm{CPS_n}[\![c]\!]$$
$$\mathrm{CPS_n}[\![\mu\hat{\mathrm{tp}}^{\ominus}.c]\!] \stackrel{\text{def}}{=} \lambda x.(\mathrm{CPS_n}[\![c]\!]\,x)$$

(a) $\mathrm{CPS_n}[\![\cdot]\!] : t \mapsto V$

$$\mathrm{CPS_n}[\![[x]\ t]\!] \stackrel{\text{def}}{=} \mathrm{CPS_n}[\![t]\!]\,x$$
$$\mathrm{CPS_n}[\![[\hat{\mathrm{tp}}^{\ominus}]\ t]\!] \stackrel{\text{def}}{=} \mathrm{CPS_n}[\![t]\!]$$

(b) $\mathrm{CPS_n}[\![\cdot]\!] : c \mapsto t$

Figure III.8: The translation $\mathrm{CPS_n}[\![\cdot]\!]$ of the $\lambda\mu\hat{\mathrm{tp}}_n$ calculus

The CPS translation is defined in Figure III.8 on the preceding page and is based on Lafont, Reus and Streicher's model which we have already seen. Of course, by changing the target of a translation we change the denotational semantics. In [HG08], the target of the translation is in call by name while here it is replaced by $\lambda_{\mathrm{v}}^{\times}$. Thus the calculus is slightly different.[4] We should note however that the $\lambda\mu\widehat{\mathrm{tp}}_n$ calculus is not defined by its denotational semantics (in other words by the target of the translation) but is defined as the result of a particular way of solving critical pairs in a non-deterministic $\lambda\mu\widehat{\mathrm{tp}}$ calculus. Now, the variant validates the same rules of reductions. In other words, we can associate different calculi to a particular way of solving critical pairs in Herbelin and Ghilezan's methodology.

Recall that we defined the call-by-name application and abstraction λ^n and $@_n$ in Section III.2.2, as well as the operators $\widehat{\mathrm{tp}}^{\ominus}$ and $\mu\widehat{\mathrm{tp}}^{\ominus}$ in the previous section. There is a straightforward translation $\mathrm{NEG}[\![\cdot]\!]$ from terms and commands of $\lambda\mu\widehat{\mathrm{tp}}_n$ to terms and commands of $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^+}$, that takes it takes λ to λ^n, application to $@_n$, $\mu\widehat{\mathrm{tp}}^{\ominus}$to $\mu\widehat{\mathrm{tp}}^{\ominus}$, α to α, μx to μx and $[e]\,t$ to $\langle t \,\|\, e\rangle$.

$$\mathrm{NEG}[\![\cdot]\!] : \lambda\mu\widehat{\mathrm{tp}}_n \ \to \mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^+}$$

We have:

Proposition III.12. *The translation* $\mathrm{NEG}[\![\cdot]\!]$ *preserves equivalences: if* $p \simeq_{\mathrm{RE}_n} p'$ *then* $\mathrm{NEG}[\![p]\!] \simeq_{\mathrm{RE}_p} \mathrm{NEG}[\![p']\!]$ *where* p *is a term or a command.*

Proof. Immediate from what we have seen; in particular from the previous section we have $\mu\widehat{\mathrm{tp}}^{\ominus}.\langle t_{\ominus} \,\|\, \widehat{\mathrm{tp}}^{\ominus}\rangle \simeq_{\mathrm{RE}_p} t_{\ominus}$. ∎

Proposition III.13. *The translation* $\mathrm{CPS_n}[\![\cdot]\!]$ *decomposes as follows: for any* p *a command or a term of* $\lambda\mu\widehat{\mathrm{tp}}_n$*, we have* $\mathrm{CPS} \circ \mathrm{NEG}[\![p]\!] \to^*_{\mathrm{admin}_\lambda} \mathrm{CPS_n}[\![p]\!]$.

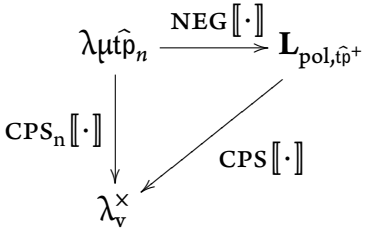
[4] The translation no longer validates the additional equation $[\widehat{\mathrm{tp}}^{\ominus}]\,\mu\widehat{\mathrm{tp}}^{\ominus}.c \simeq c$ given in [HG08].

Proof. By a straightforward induction on p. In particular we have:

$$
\begin{aligned}
&\textsc{cps} \circ \textsc{neg}\,[\![\lambda\alpha.t]\!] \\
&= \textsc{cps}\,[\![\mu(\alpha, x).\langle \textsc{neg}\,[\![t]\!] \,\|\, x\rangle]\!] \\
&= \lambda(\alpha, x).(\textsc{cps} \circ \textsc{neg}\,[\![t]\!]\; x) \\
&\textsc{cps} \circ \textsc{neg}\,[\![t\; u]\!] \\
&= \textsc{cps}\,[\![\mu x.\langle \textsc{neg}\,[\![t]\!] \,\|\, \textsc{neg}\,[\![u]\!] \cdot x\rangle]\!] \\
&= \lambda x.(\textsc{cps} \circ \textsc{neg}\,[\![t]\!]\; (\textsc{cps} \circ \textsc{neg}\,[\![u]\!], x)) \\
&\textsc{cps} \circ \textsc{neg}\,[\![[\widehat{\mathrm{tp}}^{\ominus}]\, t]\!] \\
&= \textsc{cps}\,[\![\langle \textsc{neg}\,[\![t]\!] \,\|\, \mu\alpha.\langle\{\alpha\} \,\|\, \widehat{\mathrm{tp}}\rangle\rangle]\!] \\
&= \text{let } \alpha \text{ be } \textsc{neg}\,[\![t]\!] \text{ in } (\lambda x.x)\; \alpha \\
&\rhd^{*}_{\mathrm{admin}_\lambda} \textsc{cps} \circ \textsc{neg}\,[\![t]\!] \\
&\textsc{cps} \circ \textsc{neg}\,[\![\mu\widehat{\mathrm{tp}}^{\ominus}.c]\!] \\
&= \textsc{cps}\,[\![\mu x.\langle \mu\widehat{\mathrm{tp}}.\textsc{neg}\,[\![c]\!] \,\|\, \mu\{\alpha\}.\langle \alpha \,\|\, x\rangle\rangle]\!] \\
&= \lambda x.((\lambda k.k\; x)\; \textsc{cps} \circ \textsc{neg}\,[\![c]\!]) \\
&\to^{*}_{\mathrm{admin}_\lambda} \lambda x.\textsc{cps} \circ \textsc{neg}\,[\![c]\!]\; x
\end{aligned}
$$

∎

This proves that the diagram representing the second decomposition commutes:



The result could be enhanced into a strict decomposition by replacing $\rhd_{\mathrm{admin}_\lambda}$ by a treatment of administrative reductions similar to Section IV.4.2. However, this would require to adapt the translations $\textsc{cps}_{\mathrm{v}}[\![\cdot]\!]$ and $\textsc{cps}_{\mathrm{n}}[\![\cdot]\!]$, whereas for our exposition we chose to match closely Herbelin and Ghilezan.

$$P ::= X \mid P_* \otimes P_* \mid \Downarrow N_P \qquad \Gamma, \Delta \text{ of the form } \alpha : P_Q, x : N, \ldots$$
$$N ::= X^\perp \mid N_* \parr N_* \mid \Uparrow P_P \qquad T \text{ of the form } \hat{\mathfrak{tp}} : P$$
$$P_* ::= P \mid N_P \qquad c : (\vdash \Gamma, T) \qquad \vdash V : P_*; \Gamma$$
$$N_* ::= N \mid P_P \qquad \vdash t_\ominus : N \mid \Gamma, T \quad \vdash t_+ : P_Q \mid \Gamma, T$$

(a) Formulae **(b)** Judgements

$$(X)^\perp \stackrel{\text{def}}{=} X^\perp \quad (P_* \otimes Q_*)^\perp \stackrel{\text{def}}{=} {P_*}^\perp \parr {Q_*}^\perp \quad (\Downarrow N_P)^\perp \stackrel{\text{def}}{=} \Uparrow {N^\perp}_P$$
$$X^{\perp\perp} \stackrel{\text{def}}{=} X \quad (N_* \parr M_*)^\perp \stackrel{\text{def}}{=} {N_*}^\perp \otimes {M_*}^\perp \quad \left(\Uparrow P_Q\right)^\perp \stackrel{\text{def}}{=} \Downarrow {P^\perp}_Q$$

where ${A_P}^\perp$ is $(A^\perp)_P$.

(c) Implicit negation

Figure III.9: $\mathbf{LK}_{\text{delim}}$, a system of simple types for $\mathbf{L}_{\text{pol},\hat{\mathfrak{tp}}^+}$ (types)

As a consequence of the decomposition, we have a reconstruction of the following result:

Proposition III.14. *The translation* $\text{CPS}_n[\![\cdot]\!]$ *preserves equivalences: if* $p \simeq_{\text{RE}_n} p'$ *then* $\text{CPS}_n[\![p]\!] \simeq_{\text{RE}_\lambda} \text{CPS}_n[\![p']\!]$ *where* p *is a term or a command of* $\lambda\mu\tilde{\mu}\hat{\mathfrak{tp}}_n$.

Because of the structural substitution again, and like Herbelin and Ghilezan, we come close to having a simulation result as well, but do not.

III.2.8 $\mathbf{LK}_{\text{delim}}$, a type system with annotations

We propose in Figure III.10 on the next page a type system for $\mathbf{L}_{\text{pol},\hat{\mathfrak{tp}}^+}$ which we call $\mathbf{LK}_{\text{delim}}$. It is a variant of Girard's **LC** and Danos-Joinet-Schellinx's $\mathbf{LK}^\eta_p$ [DJS97] with annotations suitable for typing delimited continuations. It adds an intermediate judgement $\vdash V : P_*; \Gamma$ where V is a value, possibly negative, and P_* is of the form P or N_P.

$$\frac{}{\vdash x : P; x : P^{\perp}}\ (\mathrm{ax}_+) \qquad \frac{}{\vdash \alpha : N \mid \alpha : N^{\perp}_P, \hat{\mathfrak{tp}} : P}\ (\mathrm{ax}_\ominus)$$

$$\frac{c : (\vdash x : N, \Gamma, T)}{\vdash \mu x.c : N \mid \Gamma, T}\ (\mu_\ominus) \qquad \frac{c : (\vdash \alpha : P_Q, \Gamma, T)}{\vdash \mu\alpha.c : P_Q \mid \Gamma, T}\ (\mu_+)$$

$$\frac{}{\vdash \hat{\mathfrak{tp}} : N \mid \hat{\mathfrak{tp}} : N^{\perp}}\ (\hat{\mathfrak{tp}}) \qquad \frac{c : (\vdash \Gamma, \hat{\mathfrak{tp}} : P)}{\vdash \mu\hat{\mathfrak{tp}}.c : P_Q \mid \Gamma, \hat{\mathfrak{tp}} : Q}\ (\mu\hat{\mathfrak{tp}})$$

$$\frac{\vdash t_+ : P_Q \mid \Gamma, \hat{\mathfrak{tp}} : R \qquad \vdash t_\ominus : P^{\perp}_Q; \Delta}{\langle t_+ \| t_\ominus \rangle : (\vdash \Gamma, \Delta, \hat{\mathfrak{tp}} : R)}\ (\mathrm{cut})$$

(a) Identity

$$\frac{\vdash V_+ : P; \Gamma}{\vdash V_+ : P_Q \mid \Gamma, \hat{\mathfrak{tp}} : Q}\ (\mathrm{der}) \qquad \frac{\vdash t_\ominus : N \mid \Gamma, \hat{\mathfrak{tp}} : P}{\vdash t_\ominus : N_P; \Gamma}\ (\mathrm{prom})$$

$$\frac{c : (\vdash \Gamma, T)}{c : (\vdash x : N, \Gamma, T)}\ (\mathrm{w}_+) \qquad \frac{c : (\vdash x : N, y : N, \Gamma, T)}{c[x/y] : (\vdash x : N, \Gamma, T)}\ (\mathrm{c}_+)$$

$$\frac{c : (\vdash \Gamma, T)}{c : (\vdash \alpha : P_Q, \Gamma, T)}\ (\mathrm{w}_\ominus) \qquad \frac{c : (\vdash \alpha : P_Q, \beta : P_Q, \Gamma, T)}{c[\alpha/\beta] : (\vdash \alpha : P_Q, \Gamma, T)}\ (\mathrm{c}_\ominus)$$

In addition there are 12 rules similar to $(w_\pm)$ and $(c_\pm)$ involving t_+, $t_\ominus$ and V instead of c which we do not mention.

(b) Structure

$$\frac{\vdash V : P_*; \Gamma \qquad \vdash V' : Q_*; \Delta}{\vdash (V, V') : P_* \otimes Q_*; \Gamma, \Delta}\ (\otimes) \qquad \frac{\vdash t_\ominus : N_P; \Gamma}{\vdash \{t_\ominus\} : \Downarrow N_P; \Gamma}\ (\Downarrow)$$

$$\frac{c : (\vdash \kappa : N_*, \kappa' : M_*, \Gamma, T)}{\vdash \mu(\kappa, \kappa').c : N_* \parr M_* \mid \Gamma, T}\ (\parr) \qquad \frac{c : (\vdash \alpha : P_Q, \Gamma, T)}{\vdash \mu\{\alpha\}.c : \Uparrow P_Q \mid \Gamma, T}\ (\Uparrow)$$

(c) Logic

Figure III.10: $\mathbf{LK}_{\mathbf{delim}}$, a system of simple types for $\mathbf{L}_{\mathbf{pol},\hat{\mathfrak{tp}}^+}$ (rules)

Remark We used the notation of Girard's stoup [Gir91] for a linearity constraint that corresponds more closely to Danos et al's η restriction [DJS97]. Tradition of having negatives in the stoup goes back to Girard [Gir93]. An alternative to our concise treatment would be to instantiate each logical rule for all possible combination of polarities of the premises, as in **LC** or **LU** [Gir91, Gir93]. (Witness the number of rules in these systems!)

CPS translation on types

We assume that $\lambda_v^\times$ is given a system of simple types with connectives $\to, \times$. Our justification for the system $\mathbf{LK}_{\text{delim}}$ is that it is mapped through the CPS translation into the simple types of $\lambda_v^\times$. Positive and annotated negative types translate as follows:

$$\begin{array}{lcl} \textsc{cps}[\![X]\!] & \stackrel{\text{def}}{=} & X \\ \textsc{cps}[\![P_* \otimes Q_*]\!] & \stackrel{\text{def}}{=} & \textsc{cps}[\![P_*]\!] \times \textsc{cps}[\![Q_*]\!] \\ \textsc{cps}[\![\Downarrow N_P]\!] & \stackrel{\text{def}}{=} & \textsc{cps}[\![N^\perp]\!] \to \textsc{cps}[\![P]\!] \\ \textsc{cps}[\![N_P]\!] & \stackrel{\text{def}}{=} & \textsc{cps}[\![N^\perp]\!] \to \textsc{cps}[\![P]\!] \end{array}$$

Typing is preserved as follows:

- If $c : (\vdash \Gamma, \widehat{\mathfrak{tp}} : P)$ then $\textsc{cps}[\![\Gamma^\perp]\!] \vdash \textsc{cps}[\![c]\!] : \textsc{cps}[\![P]\!]$
- If $\vdash V : P_*; \Gamma$ then $\textsc{cps}[\![\Gamma^\perp]\!] \vdash \textsc{val}[\![V]\!] : \textsc{cps}[\![P_*]\!]$
- If $\vdash t_\ominus : N \mid \Gamma, \widehat{\mathfrak{tp}} : P$ then $\textsc{cps}[\![\Gamma^\perp]\!] \vdash \textsc{cps}[\![t_\ominus]\!] : \textsc{cps}[\![N_P]\!]$
- If $\vdash t_+ : P_Q \mid \Gamma, \widehat{\mathfrak{tp}} : R$ then $\textsc{cps}[\![\Gamma^\perp]\!], k : \textsc{cps}[\![{P^\perp}_Q]\!] \vdash \textsc{cps}[\![t_+]\!](k) : \textsc{cps}[\![R]\!]$

where:

$$\textsc{cps}[\![\Gamma^\perp]\!] = \left(\overrightarrow{x : \textsc{cps}[\![N^\perp]\!]}, \overrightarrow{\alpha : \textsc{cps}[\![{P^\perp}_Q]\!]}\right)$$

whenever $\Gamma = (\overrightarrow{x : N}, \overrightarrow{\alpha : P_Q})$. (With $\vec{X}$ we denote a sequence of objects of the form X.)

The decomposition of type systems with annotations

Herbelin and Ghilezan give in [HG08] type systems with annotations for $\lambda\mu\tilde{\mu}\widehat{\mathfrak{tp}}_v$ and for $\lambda\mu\tilde{\mu}\widehat{\mathfrak{tp}}_n$. For the case of the $\lambda\mu\tilde{\mu}\widehat{\mathfrak{tp}}_v$ calculus, sequents

are of the form:

$$\overrightarrow{x : A} \vdash t : B_R \mid \overrightarrow{\alpha : C_S}; \widehat{\mathsf{tp}} : T$$

where the connective for implication has two annotations: $A_R \to B_S$. Now, the positive coding of implication induces in $\mathbf{L}_{\mathrm{pol},\widehat{\mathsf{tp}}^+}$ the following definition:

$$P_R \to Q_S \stackrel{\text{def}}{=} \Downarrow(P^\perp \parr Q_S)_R$$

and the following sequent:

$$\overrightarrow{x : P} \vdash t : Q_R \mid \overrightarrow{\alpha : Q'_S}; \widehat{\mathsf{tp}} : T$$

By a two-sided sequent $\Gamma \vdash \Delta$, we mean the sequent $\vdash \Gamma^\perp, \Delta$. Thus, thanks to this definition we recover Herbelin and Ghilezan's type system for $\lambda\mu\widehat{\mathsf{tp}}_\nu$ with $\mathbf{LK}_{\mathrm{delim}}$.

For the case of the $\lambda\mu\widehat{\mathsf{tp}}_n$ calculus, sequents are of the form:

$$\overrightarrow{\alpha : A_\Sigma} \vdash t : B \mid \overrightarrow{x : C}; \widehat{\mathsf{tp}} : \Xi$$

where:

$$\Sigma, \Xi ::= \perp \mid A \cdot \Sigma$$

and the connective for implication has one annotation:

$$A_\Sigma \to B$$

The implementation of $\lambda\mu\widehat{\mathsf{tp}}_n$ in $\mathbf{L}_{\mathrm{pol},\widehat{\mathsf{tp}}^+}$ yields, for N, M, L denoting negative formulae, the following definition:

$$N_\Sigma \to M \stackrel{\text{def}}{=} {N^\perp}_\Sigma \parr M$$
$$\perp \stackrel{\text{def}}{=} X_0$$
$$L \cdot \Sigma \stackrel{\text{def}}{=} \Downarrow L_\Sigma$$

where X_0 is some distinguished positive atom, and the following

sequent:

$$\overrightarrow{\alpha : N_\Sigma} \vdash t : M \mid \overrightarrow{x : L}; \widehat{\mathrm{tp}} : \Xi$$

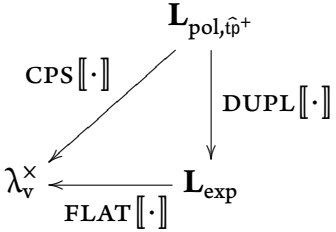
Thus we recover the annotated type system of $\lambda\mu\widehat{\mathrm{tp}}_n$ with $\mathbf{LK}_{\mathrm{delim}}$.

Related work Kiselyov and Shan [KS07] give a type system and a type checking algorithm for Danvy and Filinski's shift and reset calculus, whose goal is to be more liberal than the Danvy and Filinski's type system. Our goal on the contrary is merely to decompose existing type systems.

Kiselyov and Shan use connectives $\cdot \downarrow T$ and $\cdot \uparrow T$, which correspond modulo duality to annotated negation modalities $(\neg_P Q)$ similar to the exponentials $!_P$ and $?_P$ of next section. They note that typing a term amounts to transforming it into continuation-passing style. The type system $\mathbf{LK}_{\mathrm{delim}}$ is in direct style, but still anticipates to some degree the CPS translation through the rules (der) and (prom), and therefore does not do better in this respect. But "*investigating the delimited case*" of the duality between call by name and call by value was left for "*future work*" by the authors.

III.3 The indirect calculus $\mathbf{L}_{\mathbf{exp}}$

The goal is to decompose the translation $\mathrm{CPS}[\![\cdot]\!]$ by introducing the intermediate calculus $\mathbf{L}_{\mathrm{exp}}$ (Figure III.11 on the facing page).



The calculus $\mathbf{L}_{\mathrm{exp}}$ is in indirect style, in the sense that the translation $\mathrm{DUPL}[\![\cdot]\!] : \mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^+} \to \mathbf{L}_{\mathrm{exp}}$ is an adaptation to delimited continuations of the duploid construction of Chapter II. The indirect

Positive variables are $x, y \ldots$
$\star$ is a distinguished negative variable

(values)	$V ::= x \mid (V, V) \mid \mu x^{\diamond}.c$
(positive terms)	$t_+ ::= V \mid \mu\star.c$
(negative terms)	$t_\ominus ::= \star \mid \mu x.c \mid \mu(x,x).c \mid t_+^{\diamond}$
(commands)	$c ::= \langle t_+ \| t_\ominus \rangle \quad \left(\overset{\text{not.}}{=} \langle t_\ominus \| t_+ \rangle\right)$

(a) Syntax

$$
\begin{array}{ll}
\langle V \| \mu x.c \rangle & \rhd_{\mathrm{R}_e} \; c[V/x] \\
\langle (V, V') \| \mu(x,y).c \rangle & \rhd_{\mathrm{R}_e} \; c[V, V'/x, y] \\
\langle V_+ \| t_+^{\diamond} \rangle^{\dagger} & \rhd_{\mathrm{R}_e} \; \langle t_+ \| \mu x.\langle V_+ \| x^{\diamond} \rangle \rangle \\
\langle \mu x^{\diamond}.c \| V^{\diamond} \rangle & \rhd_{\mathrm{R}_e} \; c[V/x] \\
\langle \mu\star.\langle t_+ \| \star \rangle \| t_\ominus \rangle & \rhd_{\mathrm{R}_e} \; \langle t_+ \| t_\ominus \rangle
\end{array}
$$

†: when t_+ is not a value, *i.e.* $t_+ = \mu\star.c$.

(b) Reductions

$$
\begin{array}{ll}
t_\ominus & \rhd_{\mathrm{E}_e} \; \mu q.\langle q \| t_\ominus \rangle \\
V & \rhd_{\mathrm{E}_e} \; \mu x^{\diamond}.\langle V \| x^{\diamond} \rangle \\
c & \rhd_{\mathrm{E}_e} \; \langle \mu\star.c \| \star \rangle \\
\langle u_+ \| \mu q'.\langle t_+ \| \mu q.c \rangle \rangle & \rhd_{\mathrm{E}_e} \; \langle t_+ \| \mu q.\langle u_+ \| \mu q'.c \rangle \rangle \\
\langle \mu\star.\langle t_+ \| \mu q.c \rangle \| u_\ominus \rangle & \rhd_{\mathrm{E}_e} \; \langle t_+ \| \mu q.\langle \mu\star.c \| u_\ominus \rangle \rangle \\
\langle \mu x^{\diamond}.\langle V \| \mu q.c \rangle \| u_\ominus \rangle & \rhd_{\mathrm{E}_e} \; \langle V \| \mu q.\langle \mu x^{\diamond}.c \| u_\ominus \rangle \rangle
\end{array}
$$

where q denotes x or (x, y).

(c) Expansions

Figure III.11: $\mathbf{L}_{\mathbf{exp}}$: the calculus

nature is expressed through the linearity of the distinguished negative variable $\star$, which shows that $\mu\star$ is unable to implement control operators. $\mathbf{L}_{\mathrm{exp}}$ is polarised in the sense that there are both positive and negative terms. Like in the calculus $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^+}$, we have $\langle t \| u \rangle = \langle u \| t \rangle$ thanks to the informal notation:

$$\langle t_\ominus \| u_+ \rangle \overset{\text{not.}}{=} \langle u_+ \| t_\ominus \rangle$$

and therefore negation has to be strictly involutive. Last, $\mathbf{L}_{\mathrm{exp}}$ is in sequent calculus style, as we will see through its type system with annotations.

The translation $\textsc{flat}[\![\cdot]\!]$ flattens the calculus $\mathbf{L}_{\mathrm{exp}}$ into the $\lambda_{\mathrm{v}}^{\times}$ calculus. By flattening we mean that the translation consists in removing the interesting structure of sequent calculus, polarisation and the involutive negation. Flattening has two consequences. First, we have to reduce administrative redexes in the process in order to make up for the loss of the sequent calculus structure. Second, terms are translated regardless of their informal status as expressions or contexts. The latter means that even if we wanted to introduce an explicit distinction between expressions and contexts in the calculi $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^+}$ and $\mathbf{L}_{\mathrm{exp}}$, as we did in Chapter IV, then the continuation-based denotational semantics would still make the identification. In particular it induces the presence of a strictly involutive negation. In retrospect, this circumstantiates the novelty of Laurent's decomposition of $\lambda\mu$ calculi in proof nets [Lau02], compared to ordinary calculi of continuations.

We have:

Proposition III.15. *The reduction $\to_{\mathrm{R}_e}$ is confluent.*

Proof. The reduction $\triangleright_{\mathrm{R}_e}$ is left-linear and has no critical pair. ∎

III.3.1 Translating $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^+}$ into $\mathbf{L}_{\mathrm{exp}}$

We translate $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^+}$ into $\mathbf{L}_{\mathrm{exp}}$. This is an application to delimited continuations of the duploid construction.

κ denotes x or α in $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^+}$, and is mapped to a *positive* variable of $\mathbf{L}_{\mathrm{exp}}$ that we write identically.

$$
\begin{aligned}
\textsc{dupl}_{\mathrm{V}}[\![x]\!] &\stackrel{\text{def}}{=} x \\
\textsc{dupl}_{\mathrm{V}}[\![(V, V')]\!] &\stackrel{\text{def}}{=} (\textsc{dupl}_{\mathrm{V}}[\![V]\!], \textsc{dupl}_{\mathrm{V}}[\![V']\!]) \\
\textsc{dupl}_{\mathrm{V}}[\![\{t_\ominus\}]\!] &\stackrel{\text{def}}{=} \textsc{dupl}[\![t_\ominus]\!] \\
\textsc{dupl}_{\mathrm{V}}[\![t_\ominus]\!] &\stackrel{\text{def}}{=} \textsc{dupl}[\![t_\ominus]\!]
\end{aligned}
$$

(a) Translation $\textsc{dupl}_{\mathrm{V}}[\![\cdot]\!] : V \mapsto V$

$$
\begin{aligned}
\textsc{dupl}[\![V_+]\!] &\stackrel{\text{def}}{=} \textsc{dupl}_{\mathrm{V}}[\![V_+]\!]^\diamond \\
\textsc{dupl}[\![\mu\alpha.c]\!] &\stackrel{\text{def}}{=} \mu\alpha.\textsc{dupl}[\![c]\!] \\
\textsc{dupl}[\![\mu\widehat{\mathrm{tp}}.c]\!] &\stackrel{\text{def}}{=} (\mu\star.\textsc{dupl}[\![c]\!])^\diamond
\end{aligned}
$$

(b) Translation $\textsc{dupl}[\![\cdot]\!] : t_+ \mapsto t_\ominus$

$$
\begin{aligned}
\textsc{dupl}[\![\alpha]\!] &\stackrel{\text{def}}{=} \alpha \\
\textsc{dupl}[\![\widehat{\mathrm{tp}}]\!] &\stackrel{\text{def}}{=} \mu x^\diamond.\langle x \| \star \rangle \\
\textsc{dupl}[\![\mu x.c]\!] &\stackrel{\text{def}}{=} \mu x^\diamond.\textsc{dupl}[\![c]\!] \\
\textsc{dupl}[\![\mu(\kappa, \kappa').c]\!] &\stackrel{\text{def}}{=} \mu(\kappa, \kappa')^\diamond.\textsc{dupl}[\![c]\!] \\
&\stackrel{\text{def}}{=} \mu x^\diamond.\langle x \| \mu(\kappa, \kappa').\textsc{dupl}[\![c]\!] \rangle \\
\textsc{dupl}[\![\mu\{\alpha\}.c]\!] &\stackrel{\text{def}}{=} \mu\alpha^\diamond.\textsc{dupl}[\![c]\!]
\end{aligned}
$$

(c) Translation $\textsc{dupl}[\![\cdot]\!] : t_\ominus \mapsto V$

$$
\textsc{dupl}[\![\langle t_+ \| t_\ominus \rangle]\!] \stackrel{\text{def}}{=} \langle \textsc{dupl}[\![t_+]\!] \| \textsc{dupl}[\![t_\ominus]\!] \rangle
$$

(d) Translation $\textsc{dupl}[\![\cdot]\!] : c \mapsto c$

Figure III.12: The translation $\textsc{dupl}[\![\cdot]\!]$ from $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^+}$ to $\mathbf{L}_{\mathrm{exp}}$

Proposition III.16. *The translation* $\textsc{dupl}[\![\cdot]\!]$ *is a simulation: if* $c \to_{\mathrm{R}_p} c'$ *then* $\textsc{dupl}[\![c]\!] \to^{+}_{\mathrm{R}_e} \textsc{dupl}[\![c']\!]$.

Proof. By induction on the definition of $\to_{\mathrm{R}_p}$. We treat the base cases.
(Case $\langle \mu\hat{\mathrm{tp}}.\langle V_+ \| \hat{\mathrm{tp}}\rangle \| t_\ominus\rangle \rhd_{\mathrm{R}_p} \langle V_+ \| t_\ominus\rangle$*)* We have:

$$
\begin{aligned}
&\textsc{dupl}[\![\langle \mu\hat{\mathrm{tp}}.\langle V_+ \| \hat{\mathrm{tp}}\rangle \| t_\ominus\rangle]\!] \\
&= \underline{\left\langle (\mu\star.\langle \textsc{dupl}_{\mathrm{V}}[\![V_+]\!]^\diamond \| \mu x^\diamond.\langle x \| \star\rangle\rangle)^\diamond \middle\| \textsc{dupl}[\![t_\ominus]\!]\right\rangle} \\
&\rhd_{\mathrm{R}_p} \left\langle \mu\star.\underline{\langle \textsc{dupl}_{\mathrm{V}}[\![V_+]\!]^\diamond \| \mu x^\diamond.\langle x \| \star\rangle\rangle} \middle\| \mu y.\langle y^\diamond \| \textsc{dupl}[\![t_\ominus]\!]\rangle\right\rangle \\
&\to_{\mathrm{R}_p} \underline{\left\langle \mu\star.\langle \textsc{dupl}_{\mathrm{V}}[\![V_+]\!] \| \star\rangle \middle\| \mu y.\langle y^\diamond \| \textsc{dupl}[\![t_\ominus]\!]\rangle\right\rangle} \\
&\rhd_{\mathrm{R}_p} \underline{\left\langle \textsc{dupl}_{\mathrm{V}}[\![V_+]\!] \middle\| \mu y.\langle y^\diamond \| \textsc{dupl}[\![t_\ominus]\!]\rangle\right\rangle} \\
&\rhd_{\mathrm{R}_p} \langle \textsc{dupl}_{\mathrm{V}}[\![V_+]\!]^\diamond \| \textsc{dupl}[\![t_\ominus]\!]\rangle \\
&= \textsc{dupl}[\![\langle V_+ \| t_\ominus\rangle]\!]
\end{aligned}
$$

The other base cases are straightforward. ∎

Proposition III.17. *The translation* $\textsc{dupl}[\![\cdot]\!]$ *preserves equivalences: if* $p \simeq_{\mathrm{RE}_p} p'$ *for* p, p' *terms of commands of* $\boldsymbol{L}_{pol,\hat{\mathrm{tp}}^+}$ *then* $\textsc{dupl}[\![p]\!] \simeq_{\mathrm{RE}_\lambda} \textsc{dupl}[\![p']\!]$.

Proof. By induction on the definition of $\simeq_{\mathrm{RE}_p}$. There remains to show the base cases $p \rhd_{\mathrm{E}_p} p'$.
(Case $c \rhd_{\mathrm{E}_p} \langle \mu\hat{\mathrm{tp}}.c \| \hat{\mathrm{tp}}\rangle$*)* We have:

$$
\begin{aligned}
&\textsc{dupl}[\![\langle \mu\hat{\mathrm{tp}}.c \| \hat{\mathrm{tp}}\rangle]\!] \\
&= \underline{\langle (\mu\star.\textsc{dupl}[\![c]\!])^\diamond \| \mu x^\diamond.\langle x \| \star\rangle\rangle} \\
&\rhd_{\mathrm{R}_p} \left\langle \mu\star.\textsc{dupl}[\![c]\!] \middle\| \mu y.\underline{\langle y^\diamond \| \mu x^\diamond.\langle x \| \star\rangle\rangle}\right\rangle \\
&\rhd_{\mathrm{R}_p} \langle \mu\star.\textsc{dupl}[\![c]\!] \| \mu y.\langle y \| \star\rangle\rangle \\
&\lhd^{*}_{\mathrm{E}_p} \textsc{dupl}[\![c]\!]
\end{aligned}
$$

(Case $\langle W \| \mu q.\langle \mu\hat{\mathrm{tp}}.c \| t_\ominus\rangle\rangle \rhd_{\mathrm{E}_p} \langle \mu\hat{\mathrm{tp}}.\langle W \| \mu q.c\rangle \| t_\ominus\rangle$ *where* q *is* x, $\{\alpha\}$ *or* (κ, κ') *and where* W *is* V_+ *or* $\mu\hat{\mathrm{tp}}.c'$*)* We have $\textsc{dupl}[\![W]\!]$ of the form $t^\diamond_+$. Notice

that we have in general.

$$\langle t_+^\diamond \| \mu q^\diamond.c \rangle \simeq_{\mathrm{RE}_e} \langle t_+ \| \mu q.c \rangle$$

Therefore we have:

$$\begin{aligned}
&\textsc{dupl}[\![\langle W \| \mu q.\langle \mu\widehat{\mathrm{tp}}.c \| t_\ominus \rangle \rangle]\!] \\
&= \underline{\langle t_+^\diamond \| \mu q^\diamond.\langle (\mu\star.\textsc{dupl}[\![c]\!])^\diamond \| \textsc{dupl}[\![t_\ominus]\!] \rangle \rangle} \\
&\simeq_{\mathrm{RE}_e} \langle t_+ \| \mu q.\underline{\langle (\mu\star.\textsc{dupl}[\![c]\!])^\diamond \| \textsc{dupl}[\![t_\ominus]\!] \rangle} \rangle \\
&\to_{\mathrm{R}_e} \underline{\langle t_+ \| \mu q.\langle \mu\star.\textsc{dupl}[\![c]\!] \| \mu x.\langle x^\diamond \| \textsc{dupl}[\![t_\ominus]\!] \rangle \rangle \rangle} \\
&\lhd_{\mathrm{E}_e} \underline{\langle \mu\star.\langle t_+ \| \mu q.\textsc{dupl}[\![c]\!] \rangle \| \mu x.\langle x^\diamond \| \textsc{dupl}[\![t_\ominus]\!] \rangle \rangle} \\
&\lhd_{\mathrm{R}_e} \langle (\mu\star.\underline{\langle t_+ \| \mu q.\textsc{dupl}[\![c]\!] \rangle})^\diamond \| \textsc{dupl}[\![t_\ominus]\!] \rangle \\
&\simeq_{\mathrm{RE}_e} \langle (\mu\star.\langle t_+^\diamond \| \mu q^\diamond.\textsc{dupl}[\![c]\!] \rangle)^\diamond \| \textsc{dupl}[\![t_\ominus]\!] \rangle \\
&= \textsc{dupl}[\![\langle \mu\widehat{\mathrm{tp}}.\langle W \| \mu q.c \rangle \| t_\ominus \rangle]\!]
\end{aligned}$$

The other cases are straightforward. ∎

III.3.2 Flattening $\mathbf{L}_{\mathbf{exp}}$ into $\lambda_{\mathrm{v}}^{\times}$

The translation $\textsc{flat}[\![\cdot]\!]$ of $\mathbf{L}_{\mathrm{exp}}$ into $\lambda_{\mathrm{v}}^{\times}$ is defined in Figure III.13 on the next page. This last step of translating sequent calculus into natural deduction accounts for the appearance of administrative reductions. Indeed, administrative substitutions are performed in the translation the command $\langle t_+ \| u_\ominus \rangle$: the negative term $t_\ominus$ is sent onto $\textsc{flat}[\![u_\ominus]\!](k)$, and then k is substituted with the term $\textsc{flat}[\![t_+]\!]$. It matters that the substitution of k in $\textsc{flat}[\![t_\ominus]\!](k)$ does not duplicate code since it is performed during the translation: this is achieved by ensuring that $\textsc{flat}[\![t_\ominus]\!]$ is linear in its argument for the syntactic notion of linearity.

Proposition III.18. *The translation* $\textsc{flat}[\![\cdot]\!]$ *preserves equivalences:*

- *If* $t_+ \simeq_{\mathrm{RE}_e} u_+$ *then* $\textsc{flat}[\![t_+]\!] \simeq_{\mathrm{RE}_\lambda} \textsc{flat}[\![u_+]\!]$.
- *If* $t_\ominus \simeq_{\mathrm{RE}_e} u_\ominus$ *then* $\textsc{flat}[\![t_\ominus]\!](M) \simeq_{\mathrm{RE}_\lambda} \textsc{flat}[\![u_\ominus]\!](M)$ *for any* M.

$$\begin{aligned}
\textsc{flat}[\![x]\!] &\stackrel{\text{def}}{=} x\\
\textsc{flat}[\![V, V']\!] &\stackrel{\text{def}}{=} (\textsc{flat}[\![V]\!], \textsc{flat}[\![V']\!])\\
\textsc{flat}[\![\mu x^{\diamond}.c]\!] &\stackrel{\text{def}}{=} \lambda x.\textsc{flat}[\![c]\!]\\
\textsc{flat}[\![\mu\star.c]\!] &\stackrel{\text{def}}{=} \textsc{flat}[\![c]\!]
\end{aligned}$$

(a) $\textsc{flat}[\![\cdot]\!] : t_+ \mapsto M$

$$\begin{aligned}
\textsc{flat}[\![\star]\!](M) &\stackrel{\text{def}}{=} M\\
\textsc{flat}[\![\mu x.c]\!](M) &\stackrel{\text{def}}{=} \mathsf{let}\ x\ \mathsf{be}\ M\ \mathsf{in}\ \textsc{flat}[\![c]\!]\\
\textsc{flat}[\![\mu(x,y).c]\!](M) &\stackrel{\text{def}}{=} \mathsf{let}\ (x,y)\ \mathsf{be}\ M\ \mathsf{in}\ \textsc{flat}[\![c]\!]\\
\textsc{flat}[\![t_+^{\diamond}]\!](M) &\stackrel{\text{def}}{=} M\ \textsc{flat}[\![t_+]\!]
\end{aligned}$$

(b) $\textsc{flat}[\![\cdot]\!](M) : t_\ominus \mapsto M$

$$\textsc{flat}[\![\langle t_+ \| u_\ominus \rangle]\!] \stackrel{\text{def}}{=} \textsc{flat}[\![u_\ominus]\!](\textsc{flat}[\![t_+]\!])$$

(c) $\textsc{flat}[\![\cdot]\!] : c \mapsto M$

Figure III.13: The translation $\textsc{flat}[\![\cdot]\!]$ from $\mathbf{L}_{\mathbf{exp}}$ to $\boldsymbol{\lambda}_{\mathbf{v}}^{\times}$

- *If* $c \simeq_{\mathrm{RE}_e} c'$ *then* $\textsc{flat}[\![c]\!] \simeq_{\mathrm{RE}_\lambda} \textsc{flat}[\![c']\!]$.

Proof. By induction on the definition of $\simeq_{\mathrm{RE}_e}$. We treat the non-trivial base cases.
(Case $\langle u_+ \| \mu q'.\langle t_+ \| \mu q.c\rangle\rangle \rhd_{\mathrm{E}_e} \langle t_+ \| \mu q.\langle u_+ \| \mu q'.c\rangle\rangle$*)* This corresponds to the equation:

$$\begin{aligned}
&\mathsf{let}\ q'\ \mathsf{be}\ \textsc{flat}[\![u_+]\!]\ \mathsf{in}\ \mathsf{let}\ q\ \mathsf{be}\ \textsc{flat}[\![t_+]\!]\ \mathsf{in}\ \textsc{flat}[\![c]\!]\\
&\rhd_{\mathrm{E}_\lambda} \mathsf{let}\ q\ \mathsf{be}\ \textsc{flat}[\![t_+]\!]\ \mathsf{in}\ \mathsf{let}\ q'\ \mathsf{be}\ \textsc{flat}[\![u_+]\!]\ \mathsf{in}\ \textsc{flat}[\![c]\!]
\end{aligned}$$

(Case $\langle \mu\star.\langle t_+ \| \mu q.c\rangle \| u_\ominus\rangle \rhd_{\mathrm{E}_e} \langle t_+ \| \mu q.\langle \mu\star.c \| u_\ominus\rangle\rangle$*)* This corresponds to the equation:

$$\begin{aligned}
&\textsc{flat}[\![u_\ominus]\!](\mathsf{let}\ q\ \mathsf{be}\ \textsc{flat}[\![t_+]\!]\ \mathsf{in}\ \textsc{flat}[\![c]\!])\\
&\mathcal{R}\ \mathsf{let}\ q\ \mathsf{be}\ \textsc{flat}[\![t_+]\!]\ \mathsf{in}\ \textsc{flat}[\![u_\ominus]\!](\textsc{flat}[\![c]\!])
\end{aligned}$$

where $\mathcal{R}$ is $=$ if $u_\ominus = \star$, and $\mathcal{R}$ is $\triangleright_{\mathrm{E}_\lambda}$ otherwise. Indeed, in all the other cases $\textsc{flat}[\![u_\ominus]\!](\square)$ is a linear evaluation context.
(Case $\langle \mu x^\diamond.\langle V \| \mu q.c\rangle \| u_\ominus\rangle \triangleright_{\mathrm{E}_e} \langle V \| \mu q.\langle \mu x^\diamond.c \| u_\ominus\rangle\rangle$*)* This corresponds to the equation:

$$\begin{aligned}&\textsc{flat}[\![u_\ominus]\!](\lambda x.\mathsf{let}\ q\ \mathsf{be}\ \textsc{flat}[\![V]\!]\ \mathsf{in}\ \textsc{flat}[\![c]\!])\\ &\simeq_{\mathrm{RE}_\lambda}\ \mathsf{let}\ q\ \mathsf{be}\ \textsc{flat}[\![V]\!]\ \mathsf{in}\ \textsc{flat}[\![u_\ominus]\!](\lambda x.\textsc{flat}[\![c]\!])\end{aligned}$$

We are reduced to the previous case by noticing that we have:

$$\begin{aligned}&\lambda x.\mathsf{let}\ q\ \mathsf{be}\ \textsc{flat}[\![V]\!]\ \mathsf{in}\ \textsc{flat}[\![c]\!]\\ &\triangleright_{\mathrm{E}_\lambda}\ \mathsf{let}\ q\ \mathsf{be}\ \textsc{flat}[\![V]\!]\ \mathsf{in}\ \lambda x.\textsc{flat}[\![c]\!]\end{aligned}$$

because $\textsc{flat}[\![V]\!]$ is a value.

The remaining cases are immediate. ∎

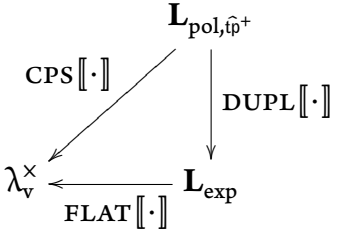
III.3.3 Decomposition

Proposition III.19. *The CPS translation of* $\mathbf{L}_{pol,\widehat{\mathfrak{tp}}^+}$ *decomposes as follows:*

$$\begin{aligned}\textsc{cps}[\![\cdot]\!] &= \textsc{flat}\circ\textsc{dupl}[\![\cdot]\!]\\ \textsc{val}[\![\cdot]\!] &= \textsc{flat}\circ\textsc{dupl}_{\mathrm{V}}[\![\cdot]\!]\end{aligned}$$

Proof. The result is proved in each value, term or command p by a straightforward induction on p. ∎

This proves the third and last decomposition of this chapter:



III.3.4 L$\bar{\text{J}}$ and annotated exponentials

The system of simple types $\mathbf{L}\bar{\mathbf{J}}$ for $\mathbf{L}_{\text{exp}}$ is introduced in Figure III.14 on the facing page. $\mathbf{L}\bar{\mathbf{J}}$ is essentially Laurent's **LLP** [Lau02] with annotations in the exponentials. As its name implies, it is also a symmetrised version of the intuitionistic sequent calculus.

Duploid construction in types

Types and sequents of $\mathbf{L}_{\text{pol},\hat{\text{tp}}^+}$ are translated trough $\textsc{dupl}[\![\cdot]\!]$ as follows:

$$\begin{aligned}
\textsc{dupl}[\![X]\!] &\stackrel{\text{def}}{=} X\\
\textsc{dupl}[\![(P_* \otimes Q_*)]\!] &\stackrel{\text{def}}{=} \textsc{dupl}[\![P_*]\!] \otimes \textsc{dupl}[\![Q_*]\!]\\
\textsc{dupl}[\![\Downarrow N_P]\!] &\stackrel{\text{def}}{=} !_{\textsc{dupl}[\![P]\!]}\textsc{dupl}[\![N]\!]\\
\textsc{dupl}[\![N_P]\!] &\stackrel{\text{def}}{=} !_{\textsc{dupl}[\![P]\!]}\textsc{dupl}[\![N]\!]
\end{aligned}$$

(and by duality such that $\textsc{dupl}[\![P^\perp]\!] = \textsc{dupl}[\![P]\!]^\perp$.)

Contexts are translated as follows:

$$\begin{aligned}
\textsc{dupl}[\![(\overrightarrow{x : N}, \overrightarrow{\alpha : P_Q})]\!] &\stackrel{\text{def}}{=} (\overrightarrow{x : \textsc{dupl}[\![N]\!]}, \overrightarrow{\alpha : ?_{\textsc{dupl}[\![Q]\!]}\textsc{dupl}[\![P]\!]})\\
\textsc{dupl}[\![\hat{\text{tp}} : P]\!] &\stackrel{\text{def}}{=} \star : \textsc{dupl}[\![P]\!]
\end{aligned}$$

For instance, we have:

$$\textsc{dupl} \circ \textsc{pos}[\![P_R \to Q_S]\!] = \textsc{dupl}[\![\Downarrow(P^\perp \parr Q_S)_R]\!] = !_{R'}(P'^\perp \parr ?_{S'}Q')$$

with the notation $P' = \textsc{dupl} \circ \textsc{pos}[\![P]\!]$. Thus the translation $\textsc{pos}[\![\cdot]\!]$ decomposes the encoding "$!(P^\perp \parr ?Q)$" described by Laurent [Lau02].

Also, we have:

$$\textsc{dupl} \circ \textsc{neg}[\![N_\Sigma \to M]\!] = \textsc{dupl}[\![N^\perp{}_\Sigma \parr M]\!] = ?_{\Sigma'}N'^\perp \parr M'$$

with the notation $N' = \textsc{dupl} \circ \textsc{neg}[\![N]\!]$. Thus the translation $\textsc{neg}[\![\cdot]\!]$ decomposes the encoding "$?N^\perp \parr M$" described by Laurent.

$$P ::= X \quad | P \otimes P \,|\, !_P N$$
$$N ::= X^\perp \,|\, N \parr N \,|\, ?_P P$$

(a) Formulae

$$(X)^\perp \stackrel{\text{def}}{=} X^\perp \qquad (P \otimes Q)^\perp \stackrel{\text{def}}{=} P^\perp \parr Q^\perp \qquad (!_P N)^\perp \stackrel{\text{def}}{=} ?_P N^\perp$$
$$(X^\perp)^\perp \stackrel{\text{def}}{=} X \qquad (N \parr M)^\perp \stackrel{\text{def}}{=} N^\perp \otimes M^\perp \qquad \left(?_Q P\right)^\perp \stackrel{\text{def}}{=} !_Q P^\perp$$

(b) Implicit negation

$$\vdash t_\ominus : N \,|\, \Gamma, \Pi \quad \vdash t_+ : P \,|\, \Gamma \quad c : (\vdash \Gamma, \Pi)$$

Γ, Δ is of the form $x_1 : N_1, \dots, x_n : N_n$ and Π is of the form $\star : P$.

(c) Judgements

$$\frac{}{\vdash x : P \,|\, x : P^\perp}\,(\mathrm{ax}_+) \qquad \frac{}{\vdash \star : N \,|\, \star : N^\perp}\,(\mathrm{ax}_\ominus)$$

$$\frac{c : (\vdash x : N, \Gamma)}{\vdash \mu x.c : N \,|\, \Gamma}\,(\mu_\ominus) \qquad \frac{c : (\vdash \Gamma, \star : P)}{\vdash \mu\star.c : P \,|\, \Gamma}\,(\mu_+)$$

$$\frac{\vdash t_+ : P \,|\, \Gamma \qquad \vdash t_\ominus : P^\perp \,|\, \Delta, \Pi}{\langle t_+ \| t_\ominus \rangle : (\vdash \Gamma, \Delta, \Pi)}\,(\mathrm{cut})$$

(d) Identity

$$\frac{c : (\vdash x : N, \Gamma, \star : P)}{\vdash \mu x^\diamond.c : !_P N \,|\, \Gamma}\,(\mathrm{prom}) \qquad \frac{\vdash V : P \,|\, \Gamma}{\vdash V^\diamond : ?_Q P \,|\, \Gamma, \star : Q}\,(\mathrm{der})$$

$$\frac{c : (\vdash \Gamma, \Pi)}{c : (\vdash x : N, \Gamma, \Pi)}\,(\mathrm{w}) \qquad \frac{c : (\vdash x : N, y : N, \Gamma, \Pi)}{c[x/y] : (\vdash x : N, \Gamma, \Pi)}\,(\mathrm{c})$$

Plus rules similar to (w) and (c) for t_+ and $t_\ominus$ left unmentioned.

(e) Structure

$$\frac{\vdash V : P \,|\, \Gamma \qquad \vdash V' : Q \,|\, \Delta}{\vdash (V, V') : P \otimes Q \,|\, \Gamma, \Delta}\,(\otimes) \qquad \frac{c : (\vdash x : N, y : M, \Gamma, \Pi)}{\vdash \mu(x, y).c : N \parr M \,|\, \Gamma, \Pi}\,(\parr)$$

(f) Logic

Figure III.14: LJ̄, a system of simple types for $\mathbf{L}_{\mathbf{exp}}$

Typing is preserved as follows:

$$\begin{array}{lll}
\text{If} & c : (\vdash_{\mathbf{LK}_{\text{delim}}} \Gamma, T) & \text{then} \quad [\![c]\!] : (\vdash_{\mathbf{L\bar{J}}} [\![\Gamma]\!], [\![T]\!]) \\
\text{If} & \vdash_{\mathbf{LK}_{\text{delim}}} t_{\ominus} : N \mid \Gamma, \widehat{\mathrm{tp}} : P & \text{then} \quad \vdash_{\mathbf{L\bar{J}}} [\![t_{\ominus}]\!] : !_{[\![P]\!]}[\![N]\!] \mid [\![\Gamma]\!] \\
\text{If} & \vdash_{\mathbf{LK}_{\text{delim}}} V : P_* \mid \Gamma & \text{then} \quad \vdash_{\mathbf{L\bar{J}}} \textsc{dupl}_{\mathrm{V}}[\![V]\!] : [\![P_*]\!] \mid [\![\Gamma]\!] \\
\text{If} & \vdash_{\mathbf{LK}_{\text{delim}}} t_+ : P_Q \mid \Gamma, T & \text{then} \quad \vdash_{\mathbf{L\bar{J}}} [\![t_+]\!] : ?_{[\![Q]\!]}[\![P]\!] \mid [\![\Gamma]\!], [\![T]\!]
\end{array}$$

with the notation $[\![x]\!] = \textsc{dupl}[\![x]\!]$.

The (extension of the) duploid construction best appears when we restore the left side of the sequents. The following sequent:

$$\vec{P}, \vec{N_R} \vdash_{\mathbf{LK}_{\text{delim}}} \vec{Q_S}, \vec{N}, T$$

is translated into the following one:

$$\overrightarrow{[\![P]\!]}, \overrightarrow{!_{[\![R]\!]}[\![N]\!]} \vdash_{\mathbf{L\bar{J}}} \overrightarrow{?_{[\![S]\!]}[\![Q]\!]}, \overrightarrow{[\![N]\!]}, [\![T]\!]$$

Flattening in types

Then the types are mapped into the standard simple type system of $\lambda_{\mathrm{v}}^{\times}$ with pairs and arrows, also known as the $\times, \to$ fragment of intuitionistic logic. Formulae translate as follows:

$$\begin{array}{lll}
\textsc{flat}[\![X]\!] & \stackrel{\text{def}}{=} & X \\
\textsc{flat}[\![P \otimes Q]\!] & \stackrel{\text{def}}{=} & \textsc{flat}[\![P]\!] \times \textsc{flat}[\![Q]\!] \\
\textsc{flat}[\![!_p N]\!] & \stackrel{\text{def}}{=} & \textsc{flat}[\![N^{\perp}]\!] \to \textsc{flat}[\![P]\!]
\end{array}$$

Typing is preserved as follows:

$$\begin{array}{lll}
\text{If} & c : (\vdash_{\mathbf{L\bar{J}}} \Gamma, \star : P) & \text{then} \quad [\![\Gamma^{\perp}]\!] \vdash [\![c]\!] : [\![P]\!] \\
\text{If} & \vdash_{\mathbf{L\bar{J}}} t : N \mid \Gamma, \star : P & \text{then} \quad [\![\Gamma^{\perp}]\!], k : [\![N^{\perp}]\!] \vdash [\![t]\!](k) : [\![P]\!] \\
\text{If} & \vdash_{\mathbf{L\bar{J}}} t : P \mid \Gamma & \text{then} \quad [\![\Gamma^{\perp}]\!] \vdash [\![t]\!] : [\![P]\!]
\end{array}$$

where we write $[\![x]\!] = \text{FLAT}[\![x]\!]$ and $[\![\Gamma^{\perp}]\!] = (\overrightarrow{x : \text{FLAT}[\![N^{\perp}]\!]})$ whenever $\Gamma = (\overrightarrow{x : N})$.

Related work Recently, Zeilberger proposed [Zei10] — among other contributions — a "*simple account of delimited continuations through a natural generalisation of classical polarised logic*". It is "*in continuation-passing style*", thus located at the level of **LLP**.

Zeilberger's work is based on the hypothesis that delimited continuations affect the structure of the polarity-changing negation. Our polarised analysis of delimited CPS translations differs in this aspect, since the decomposition shows above that the type of delimited continuations is instead related to annotated exponentials of $\mathbf{L\bar{J}}$. In particular, in our approach, the hypothesis that the symmetry between positive and negative connectives has to be "*broken*" does not seem necessary. Indeed, $\mathbf{L\bar{J}}$ contrasts indeed with Zeilberger's logic by being symmetric.

While we expect that a translation of Danvy and Filinski's calculus can be given in the positive part of Zeilberger's system, the author makes no claims of connection with delimited control calculi of the literature.

III.3.5 Reversing translation into linear logic

In this section we describe a translation of $\mathbf{L\bar{J}}$ into linear logic [Gir87] (**LL**) that explains the annotated exponential $!_p$. We restrict ourselves to the typed case, since we apply η expansion in the translation. There is no standard acceptation of what would an untyped syntax of **LL** be.

Also, we are voluntarily brief: the reader can refer to Benton and Wadler [BW95] who establish a correspondence between the call-by-value λ calculus obtained through the boring translation and Moggi's monadic model for a commutative strong monad. This result establishes our target calculus on firmer grounds, since $\lambda_v^{\times}$ is essentially a call-by-value λ calculus extended with equations reflecting the commutativity of the monad.

In **LL**, it is possible to take the following definitions:

$$!_P A \stackrel{\text{def}}{=} !(A \parr P) \qquad\qquad ?_P A \stackrel{\text{def}}{=} ?(A \otimes P^\perp)$$

with the following derived rules:

$$\frac{\vdash ?\Gamma, A, P}{\vdash ?\Gamma, !_P A}\,(\text{prom}_P) \qquad\qquad \frac{\vdash \Gamma, A}{\vdash \Gamma, ?_P A, P}\,(\text{der}_P)$$

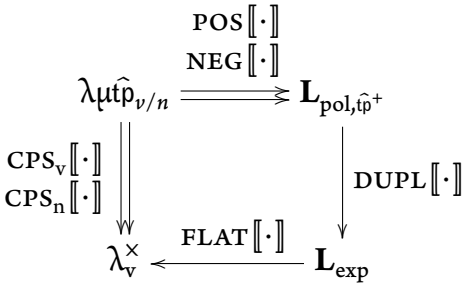
We describe an extension of Laurent and Regnier's *reversing* translation ρ [LR03] of **LLP** into **LL**. The goal with ρ is to confine structural rules to the places where they conform to the constraints of **LL**: contraction, weakening and the context of promotion are restricted to formulae of the form $?A$.

At the level of terms, the translation η-expands positive terms. η-expansion allows us to replace contractions and weakenings on a formulae N by the same rules on formulae of the form $X^\perp$ or $?_Q P$. Indeed, for instance, if z is not linear in $c : (\vdash z : X^\perp \parr Y^\perp, \Gamma)$, then it is possible to get rid of structural rules on $X^\perp \parr Y^\perp$ by replacing c by $\langle\, \mu(x,y).c[(x,y)/z] \parallel z \,\rangle$, which is equivalent and only uses structural rules on $X^\perp$ and $Y^\perp$. The negative context of promotion is decomposed similarly so that it only contains formulae of the form $X^\perp$ or $?_Q P$. Dereliction and this promotion are replaced by their derivation above.

We are done when we eventually get rid of structural rules on atoms X and $X^\perp$ by replacing these atoms by $!X$ and $?X^\perp$ (the usual exponential, not the annotated one). Changing the atoms is the only operation that we need to carry out at the level of types.

III.4 Conclusion

We have proved that the call-by-value and the call-by-name delimited CPS translations decompose in three steps:



The first step implements a user language into a calculus of machines in which the description of reduction and extensional equivalences is convenient. The second step makes side-effects explicit through the duploid construction, which generalises the Kleisli construction for monads and for co-monads. The last step seems to have little value: it obfuscates the CPS translation by erasing interesting structures.

The decomposition pictures the calculi **L** as middlemen between languages on the one hand and physical or mathematical models on the other hand.

$$\text{Language} \longrightarrow \mathbf{L} \longrightarrow \begin{array}{c}\text{Denotation}\\ \text{Implementation}\end{array}$$

The calculi **L** are syntaxes that delegate the implementation of user-oriented features, such as presenting the system in a functional style, to an additional layer of abstraction. They focus instead on an interactive representation of computation. For this reason, they are more succinct syntaxes than idealised programming languages, and closer to the physical or mathematical model.

Chapter IV

On the constructive interpretation of an involutive negation

We are interested in the constructive interpretation of reasoning by contrapositive in classical logic, or, in other words, of an involutive negation.

We first review the impossibility of an involutive negation in the λC calculus, and see that it is related to an inappropriate order of evaluation: its call-by-name evaluation strategy forces the calculus to identify captured stacks with continuations.

Then we introduce the extensional, untyped and polarised calculi $\lambda\ell$ (for natural deduction) and $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^{\ominus}}$ (for sequent calculus). They refine the λC calculus by distinguishing the positive type of captured stacks from the negative type of continuations. Also, a delimitation of control operators gives a constructive interpretation for the unit $\bot$.

Captured stacks are given accessors that allow us to understand isomorphisms such as $A \simeq \neg\neg A$ and $\neg\forall x\, A \simeq \exists x\, \neg A$. Thus, the calculi $\lambda\ell$ and $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^{\ominus}}$ realise a *formulae-as-types* correspondence between an involutive negation and a notion of high-level access to the stack similar to Felleisen-Clements's.

Outline

Section IV.1 We review the lack of involution of negation in calculi λC and $\mathbf{L}_{\mathrm{n}}$. We show that this deficiency comes from too strong a constraint imposed on the evaluation order.

Section IV.2 We recall the direct link between the proofs of a formula and the ones of its formal contrapositive through categorical duality. We explain why it is not sufficient to provide a constructive interpretation of reasoning by contrapositive.

Section IV.3 We introduce the $\lambda\ell$ calculus. To do so we introduce the notion of polarity in this context and the one of inspectable stack. We also introduce a natural deduction for $\lambda\ell$ where the type of inspectable stacks refine the type of negations.

Sections IV.4 and IV.5 We introduce the calculus $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^{\ominus}}$ which extends the calculus $\mathbf{L}_{\mathrm{n}}$ in the same way as $\lambda\ell$ refines λC (see Table I.18 on page 99 for a summary of the calculi).

We give a CPS translation that simulates reduction and preserves equivalence. We also show that we have a type isomorphism $\neg\neg A \simeq A$ in the calculi $\lambda\ell$ and $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^{\ominus}}$.

Section IV.6 We revisit Girard's "*stoup*" [Gir91] and Danos-Joinet-Schellinx's η restriction [DJS97] through a study of commutation properties in $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^{\ominus}}$.

IV.1 A review of the problem of involution

In this section, we review the absence of involution in the call-by-name λC calculus (Figure I.12 on page 81) and we show how it comes from an inappropriate order of evaluation. (The argument can be adapted to all similar settings, and by duality to call-by-value settings.)

We reduce the problem to the coexistence of two distinct left introduction rules for $\neg\neg$ in sequent calculus: one is focused, the other one is not.

IV.1.1 Rules of negation

To begin with, we consider the rules of negation in **LK**:

$$\frac{\Gamma \mid e : A \vdash \Delta}{\Gamma \vdash [e] : \neg A \mid \Delta}\,(\vdash \neg) \qquad \frac{\Gamma \vdash t : A \mid \Delta}{\Gamma \mid [t] : \neg A \vdash \Delta}\,(\neg \vdash)$$

By taking the definition "$\neg N = N \to \bot$" in the calculus $\mathbf{L}_{\mathrm{n}}$ (Figure I.10 on page 79), we have the following constructors:

$$[t_\ominus] \stackrel{\text{def}}{=} t_\ominus \cdot \mathsf{stop}$$
$$[e_\ominus] \stackrel{\text{def}}{=} \mathsf{k}_{e_\ominus}$$

Thus the following rule:

$$\langle [e_\ominus] \parallel [t_\ominus] \rangle \rhd^*_{\mathrm{R}_n} \langle t_\ominus \parallel e_\ominus \rangle$$

Indeed this corresponds to the reduction $\langle \mathsf{k}_{e_\ominus} \parallel t_\ominus \cdot \mathsf{stop} \rangle \rhd^*_{\mathrm{R}_n} \langle t_\ominus \parallel e_\ominus \rangle$.

IV.1.2 Implementing call by value

The essential observation is that in the λC calculus, terms of type $\neg\neg N$ support call-by-value evaluation. This is observed by considering $\Gamma, x : N \vdash t : A$ and $\Gamma \vdash u : \neg\neg N$. There are two ways of obtaining a proof of $\Gamma \vdash A$:

$$M_n \stackrel{\text{def}}{=} (\lambda x.t)\,(C\,u) \qquad \Gamma \vdash M_n : A$$
$$M_v \stackrel{\text{def}}{=} C\,\lambda k.(u\,\lambda x.(k\,t)) \qquad \Gamma \vdash M_v : A$$

The two terms are distinguished through their execution:

$$\boxed{\begin{array}{l}\langle M_n \| \pi\rangle \rhd^*_{\mathrm{R}_n} \langle t[C\, u/x] \| \pi\rangle \\ \langle M_v \| \pi\rangle \rhd^*_{\mathrm{R}_n} \langle u \| \lambda x.(\mathsf{k}_\pi\, t)\cdot \mathsf{stop}\rangle\end{array}}$$

As we see, M_v evaluates u before t (call by value), while M_n evaluates t before u (call by name).

IV.1.3 The issue with involution in $\mathbf{L_n}$

We show that the terms M_v and M_n correspond to two distinct ways of introducing $\neg\neg$ on the left in the calculus $\mathbf{L}_{\mathrm{n}}$:

- M_v to a naive one which interferes with the evaluation order;
- M_n to a focused one which does not interfere.

Naive introduction

The first way to introduce $\neg\neg$ on the left is obvious:

$$\dfrac{\dfrac{\Gamma \mid e_\ominus : N \vdash \Delta}{\Gamma \vdash [e_\ominus] : \neg N \mid \Delta}\,(\vdash\neg)}{\Gamma \mid [[e_\ominus]] : \neg\neg N \vdash \Delta}\,(\neg\vdash)$$

Let e_v denote the following context:

$$e_v \overset{\text{def}}{=} [[e_\ominus]] = (\lambda x.(\mathsf{k}_{e^\ominus}\, t_\ominus))\cdot \mathsf{stop}\,.$$

A term in the context e_v is evaluated immediately because e_v is a stack for any $e_\ominus$.

Evaluating M_v corresponds to evaluating $u_\ominus$ in the context e_v for $e_\ominus = \tilde{\mu}x.\langle t_\ominus \| \alpha\rangle$:

$$\boxed{\langle M_v \| \alpha\rangle \simeq_{\mathrm{R}_n} \langle u_\ominus \| e_v\rangle}\,.$$

Proof. Indeed:

$$\langle M_v \| \alpha\rangle \rhd^*_{\mathrm{R}_n} \langle u \| \lambda x.(\mathsf{k}_\alpha\, t_\ominus)\cdot \mathsf{stop}\rangle\,.$$

Also:

$$\begin{aligned}\lambda x.\underline{(\mathsf{k}_\alpha\, t_\ominus)} &= \lambda x.\mu\beta.\underline{\langle \mathsf{k}_\alpha \parallel t_\ominus \cdot \beta\rangle}\\ &\to^*_{\mathrm{R}_n} \lambda x.\mu\beta.\langle t_\ominus \parallel \alpha\rangle\\ &\leftarrow^*_{\mathrm{R}_n} \lambda x.\mu\beta.\langle x \parallel e_\ominus\rangle = \mathsf{k}_{e_\ominus}\,.\end{aligned}$$ ∎

Focused introduction

The second way to introduce $\neg\neg$ on the left is a focused variant of the naive one. This means that we are careful to apply the rule $(\vdash \neg)$ only on stacks, at the price of introducing a cut, as follows:

$$\dfrac{\dfrac{\dfrac{}{\Gamma \mid [[\beta^\ominus]] : \neg\neg N \vdash \beta^\ominus : N, \Delta}}{\Gamma, x^\ominus : \neg\neg N \vdash \mu\beta^\ominus.\langle x^\ominus \parallel [[\beta^\ominus]]\rangle : N \mid \Delta} \qquad \Gamma \mid e_\ominus : N \vdash \Delta}{\Gamma \mid \tilde{\mu}x^\ominus.\big\langle \mu\beta^\ominus.\langle x^\ominus \parallel [[\beta^\ominus]]\rangle \big\| e_\ominus\big\rangle : \neg\neg N \vdash \Delta}$$

Let e_n denote the context thus obtained:

$$e_n \stackrel{\text{def}}{=} \tilde{\mu}x^\ominus.\big\langle \mu\beta^\ominus.\langle x^\ominus \parallel [[\beta^\ominus]]\rangle \big\| e_\ominus\big\rangle$$

The context e_n uses its counter-term if and only if $e_\ominus$ does.

Evaluating the term M_n amount to evaluating $u_\ominus$ in the context e_n for $e_\ominus = \tilde{\mu}x^\ominus.\langle t_\ominus \parallel \alpha^\ominus\rangle$:

$$\boxed{\langle M_n \parallel \alpha^\ominus\rangle \simeq_{\mathrm{R}_n} \langle u_\ominus \parallel e_n\rangle}\,.$$

Proof. Indeed we have:

$$\begin{aligned}\langle M_n \parallel \alpha\rangle &\to^*_{\mathrm{R}_n} \langle t_\ominus[\underline{(C)u_\ominus}/x] \parallel \alpha\rangle\\ &= \big\langle t_\ominus[\mu\beta.\langle \underline{C} \parallel u_\ominus \cdot \beta\rangle/x] \big\| \alpha\big\rangle\\ &\to^*_{\mathrm{R}_n} \big\langle t_\ominus[\underline{\mu\beta.\langle u_\ominus \parallel \mathsf{k}_\beta \cdot \mathsf{stop}\rangle}/x] \big\| \alpha\big\rangle\\ &\leftarrow_{\mathrm{R}_n} \big\langle \mu\beta.\langle \underline{u_\ominus} \parallel \mathsf{k}_\beta \cdot \mathsf{stop}\rangle \big\| \tilde{\mu}x.\langle t_\ominus \parallel \alpha\rangle\big\rangle\end{aligned}$$

$$\leftarrow_{\mathrm{R}_n} \left\langle u_\ominus \middle\| \tilde{\mu}x.\langle \mu\beta.\langle x \| \underline{\mathsf{k}_\beta \cdot \mathsf{stop}} \rangle \middle\| \underline{\tilde{\mu}x.\langle t_\ominus \| \alpha \rangle} \right\rangle$$
$$= \left\langle u_\ominus \middle\| \tilde{\mu}x.\langle \mu\beta.\langle x \| [[\beta]] \rangle \middle\| e_\ominus \right\rangle = \langle u_\ominus \| e_n \rangle . \qquad \blacksquare$$

IV.1.4 Lifting the ambiguity of evaluation order

We recall below that in the more general context of the models of negated objects, the isomorphism $\neg\neg N \simeq N$ is equivalent to equating (the homologues of) M_n and M_v, even in other (linear) settings where $\neg\neg N \simeq N$ does not mean that all proofs are identified. We conclude that the impossibility of an involutive $\neg$ is characterised by the necessity to distinguish M_n and M_v as soon as the system is affine on the left (*e.g.* x may be free in t and u might not use its argument).

Let us recall the notion of runnable monad from Chapter II, which is going to describe the implementation of call by value with M_v in the denotational semantics of the λC calculus.

Definition IV.1. A runnable monad on a category $\mathscr{C}$ is a monad (T, η, μ) on $\mathscr{C}$ where $\mu = \rho_T$ for a *run* transformation $\rho : T \to 1$ that satisfies $\rho \circ \eta = \mathrm{id}$ and $\rho \circ T\rho = \rho \circ \rho_T$.

The transformation ρ can be natural; in which case $\eta \circ \rho = \rho_T \circ T\eta = \mathrm{id}$ yields $T \simeq 1$; or not. In the latter case the inequality $\rho_M \circ Tf \neq f \circ \rho_N : TN \to M$ means that the model distinguishes two distinct ways to compose $g : L \to TN$ with $f : N \to M$.

Proposition IV.2. *Let $\mathscr{R}$ be a category with a self-adjoint functor $\neg : \mathscr{R}^{\mathrm{op}} \to \mathscr{R}$:*

$$\neg \dashv_{(\eta,\varepsilon)} \neg : \mathscr{R}^{\mathrm{op}} \to \mathscr{R}$$

The full sub-category $\mathscr{C}$ of negated objects $\neg A$ has the runnable monad $(\neg\neg, \eta_\neg, \rho)$ with $\rho_{\neg A} = \neg\varepsilon_A$.

Proof. The monad $(\neg\neg, \eta_\neg, \neg\varepsilon_{\neg\neg})$ on $\mathscr{C}$ comes from the monad $(\neg\neg, \eta, \neg\varepsilon_\neg)$ on $\mathscr{R}$ and the equations on ρ are an easy consequence of the adjunction. $\blacksquare$

The result is related to the one of Führmann [Füh99] through duality: $\mathscr{C}$ above is essentially the Kleisli category of the co-monad $\neg\neg$ on the category $\mathscr{R}^{\mathrm{op}}$.[1] The above construction can be instantiated with Streicher and Reus's category of negated domains [SR98] which is a model of the λC calculus. We easily check that C is given by $\rho_N : \neg\neg N \to N$. *A fortiori*, $f \circ \rho$ and $\rho \circ \neg\neg f$ respectively correspond to M_n and M_v through the identification of f to $x \mapsto t$.

Lastly, Melliès and Tabareau [MT10] instantiate $\mathscr{R}$ above with a dialogue category (in the terminology of the authors). They get with $\mathscr{C}$ a model of linear logic (with $\neg\neg N \simeq N$), by identifying $f \circ \rho$ and $\rho \circ \neg\neg f$ ("depolarisation"). In other words, they obtain an involutive negation by forcing call by value to coincide with call by name, and they can do so because their setting is linear.

The method used with the $\lambda\ell$ calculus to get an involutive negation is, on the contrary, to acknowledge the coexistence of the call-by-name and the call-by-value evaluation schemes. Then, as we will see, we obtain a new connective $\sim$ by adding a focalisation reduction to the negation, so that there is no more ambiguity on the order of evaluation at the type $\neg\neg A$.

IV.2 On duality

Here we answer whether the duality between call by name and call by value gives an interpretation of reasoning by contrapositive.

Recall that regarding the constructive interpretations of propositional classical logic, it is well-understood that interpretations in call by name and the ones in call by value are related through duality, as much in models (Streicher and Reus [SR98], Selinger [Sel01]) as in term syntaxes (Curien and Herbelin [CH00], Wadler [Wad03]). Duality goes past the functional interpretation of Brouwer, Heyting and Kolmogorov. Thus, the following implication has no intuitionistic

[1] I am grateful to Paul-André Melliès for mentioning this argument early in my research.

interpretation:

$$\forall x, y \in A, P(x) \vee Q(y) \to (\forall x \in A, P(x)) \vee (\forall y \in A, Q(y)) \quad \text{(IV.1)}$$

Indeed, that would require the possibility of determining which side of the conclusion is satisfied, only from bits of information "P or Q" for each pair (x, y) in the hypothesis. This is not possible in a systematic way, since sets can be infinite.

But intuitionism provides an interpretation for the formal contrapositive of formula (IV.1):

$$(\exists x \in A, \neg P(x)) \wedge (\exists y \in A, \neg Q(y)) \to \exists x, y \in A, \neg P(x) \wedge \neg Q(y)$$

According to the intuitionistic interpretation:

> the proof gets a witness of "$\exists x$", one of "$\exists y$", and combines them into a witness of "$\exists x, y$".

Duality associates to the call-by-value interpretation of the formula above an interpretation in call by name for (IV.1). Duality formally exchanges strict data structures with lazy ones (Filinski [Fil89]), inputs with outputs (as noticed by Girard through linear logic [Gir87]), and also programs with their evaluation contexts (Curien and Herbelin [CH00]).

We obtain in this way a constructive interpretation for the classical proposition (IV.1):

> the proof receives a refutation (or counter-example) for "$\forall x$", one for "$\forall y$", and combines them into a refutation of "$\forall x, y$".

The role of laziness is to delay the realisation of the conclusion "$\vee$" until the simultaneous utilisation of both branches of the conclusion, that is to say until the refutations are indeed received. (Such a classical disjunction in call by name has been studied by Pym and Ritter [PR01] and Selinger [Sel01].)

… : Main additions to Figure I.12 on page 81 — C is removed.

$$\varepsilon ::= + \mid \ominus$$

$$t, u ::= x^{\varepsilon} \mid \lambda x.t \mid (t_{\ominus}\, u)_{\varepsilon} \mid \boxed{\mathsf{let}\ x^{+}\ \mathsf{be}\ t_{+}\ \mathsf{in}\ u \mid}$$

$$\boxed{(t, u) \mid \mathsf{let}\ (x, y)\ \mathsf{be}\ t_{+}\ \mathsf{in}\ u \mid \mathsf{send} \mid \ell \mid D_{\to} \mid D_{\bot} \mid D_{\forall}}$$

t_{ε} : term t such that $\bar{\omega}(t) = \varepsilon \in \{+, \ominus\}$

(a) Quasi-proof terms

t	$\bar{\omega}(t)$
let … in u	$\bar{\omega}(u)$
x^{ε}, $(t\, u)_{\varepsilon}$	ε
$\lambda x.t$, send, ℓ, $D_{\to}$, $D_{\bot}$, $D_{\forall}$	$\ominus$
(t, u)	$+$

(b) Polarity $\bar{\omega}(t) \in \{+, \ominus\}$

Figure IV.1: The λℓ calculus

Notice that this shows that the intuitionistic interpretation of disjunction as a strong sum, besides being not always possible for classical tautologies, is not always useful either. Indeed, the interpretation obtained by duality makes no use of an hypothetical decision procedure for "$\forall \mathrm{x}, \mathrm{y} \in A, (P(\mathrm{x}) \vee Q(\mathrm{y}))$".

We believe that duality can provide useful intuitions. However, the explanation through duality is not sufficient. The reason is that duality is external to a calculus, whereas contraposition is an internal operation.

IV.3 The λℓ calculus

In this section we introduce the λℓ calculus. Sections IV.3.1 through IV.3.4 explain the operations and the types of the calculus. Section

… : Main additions to Figure I.12 on page 81.

$$A, B \begin{cases} N, M ::= A \to B \mid \forall \mathrm{x}\, N \mid \bot \\ P, Q ::= \boxed{X(\mathrm{t}_1, \ldots, \mathrm{t}_n) \mid A \otimes B \mid \exists \mathrm{x}\, P \mid {\sim}A} \end{cases}$$

(a) Formulae

$$\neg A \stackrel{\text{def}}{=} \begin{cases} {\sim}N & \text{if } A = N \\ P \to \bot & \text{if } A = P \end{cases}$$

(b) Negation

$$\Gamma \vdash t_+ : P \qquad \Gamma \vdash t_\ominus : N \qquad \text{where:} \quad \boxed{x : A ::= x^+ : P \mid x^\ominus : N} \qquad \Gamma = x_1 : A_1, \ldots, x_n : A_n$$

(c) Judgements

$$\boxed{\ell : (A \to \bot) \to {\sim}A \qquad \mathsf{send} : {\sim}A \to (A \to \bot) \qquad D_\bot : {\sim}\bot \to A \to A \qquad D_\to : {\sim}(A \to B) \to A \otimes {\sim}B \qquad D_\forall : {\sim}\forall x\, N \to \exists x\, {\sim}N}$$

$$\frac{}{\Gamma, x : A \vdash x : A}\ (\mathrm{ax}) \qquad \boxed{\frac{\Gamma \vdash t_+ : P \qquad \Gamma, x^+ : P \vdash u : B}{\Gamma \vdash \mathsf{let}\, x^+ \,\mathsf{be}\, t_+ \,\mathsf{in}\, u : B}\ (\mathrm{let})}$$

$$\frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x.t : A \to B}\ (\to_i) \qquad \frac{\Gamma \vdash t : A \to B \qquad \Gamma \vdash u : A}{\Gamma \vdash (t_\ominus\, u)_\varepsilon : B}\ (\to_e, \varepsilon = \bar{\omega}(B))$$

$$\boxed{\frac{\Gamma \vdash t : A \qquad \Gamma \vdash u : B}{\Gamma \vdash (t, u) : A \otimes B}\ (\otimes_i)} \qquad \boxed{\frac{\Gamma \vdash t_+ : A \otimes B \qquad \Gamma, x : A, y : B \vdash u : C}{\Gamma \vdash \mathsf{let}\, (x, y) \,\mathsf{be}\, t_+ \,\mathsf{in}\, u : C}\ (\otimes_e)}$$

$$\frac{\Gamma \vdash t_\ominus : N}{\Gamma \vdash t_\ominus : \forall \mathrm{x}\, N}\ (\forall_i)^* \qquad \frac{\Gamma \vdash t_\ominus : \forall \mathrm{x}\, N}{\Gamma \vdash t_\ominus : N[\mathrm{t}/\mathrm{x}]}\ (\forall_e)$$

$$\boxed{\frac{\Gamma \vdash t_+ : P[\mathrm{t}/\mathrm{x}]}{\Gamma \vdash t_+ : \exists x\, P}\ (\exists_i)} \qquad \boxed{\frac{\Gamma \vdash t_+ : \exists \mathrm{x}\, P \qquad \Gamma, x^+ : P \vdash u : A}{\Gamma \vdash \mathsf{let}\, x^+ \,\mathsf{be}\, t_+ \,\mathsf{in}\, u : A}\ (\exists_e)^*}$$

*: $\mathrm{x} \notin \mathbf{fv}(\Gamma, A)$.

(d) Rules

$$\begin{array}{lcll} D_{\forall\to} & \stackrel{\text{def}}{=} & \lambda x^+.\mathsf{let}\, y^+ \,\mathsf{be}\, D_\forall\, x^+ \,\mathsf{in}\, D_\to\, y^+ & : {\sim}\forall \mathrm{x}(A \to B) \to \exists \mathrm{x}\, (A \otimes {\sim}B) \\ D_\neg & \stackrel{\text{def}}{=} & \lambda x^+.\mathsf{let}\, (y^\varepsilon, z^+) \,\mathsf{be}\, D_\to\, x^+ \,\mathsf{in}\, (D_\bot\, z^+\, y^\varepsilon)_\varepsilon & : {\sim}(A \to \bot) \to A \\ \mathcal{A}_\varepsilon & \stackrel{\text{def}}{=} & \lambda x^\ominus.\mathsf{let}\, (y^\varepsilon, z^+) \,\mathsf{be}\, D_\to\, \ell\_.x^\ominus \,\mathsf{in}\, y^\varepsilon & : \bot \to A \\ E & \stackrel{\text{def}}{=} & \lambda x.\ell y^\ominus.(y^\ominus\, x)_\ominus & : A \to {\sim}(A \to \bot) \\ \mathcal{T} & \stackrel{\text{def}}{=} & \lambda x^\ominus.(D_\neg\, \ell y^\ominus.(x^\ominus\, (\ell y^\ominus)_+)) & : ({\sim}A \to \bot) \to A \\ C & \stackrel{\text{def}}{=} & \lambda x^\ominus.\mathsf{let}\, (y, z^+) \,\mathsf{be}\, D_\to\, (\ell x^\ominus)_+ \,\mathsf{in}\, y & : ((A \to \bot) \to \bot) \to A \end{array}$$

(e) Derived terms (notation $\ell x.t_\ominus \stackrel{\text{def}}{=} (\ell\, \lambda x.t_\ominus)_+$)

Figure IV.2: Polarised first-order predicate calculus

[…]: Main additions to Figure I.12 on page 81.

$\alpha^+, \beta^\ominus$ … : stack constants

$$t ::= \ldots \mid \boxed{[\pi] \mid \mathsf{j}_{\pi_+}}$$

$$t \supseteq \boxed{V, W} ::= t_\ominus \mid x^+ \mid (V, W) \mid [\pi]$$

$$\pi ::= \alpha^\varepsilon \mid V \cdot \pi \mid \boxed{\tilde{\mu} x^+.c \mid \tilde{\mu}(x,y).c}$$

(a) Terms, values and stacks

$$c ::= \langle t_\ominus \| \pi_\ominus \rangle \mid \boxed{\langle t_+ \| \pi_+ \rangle}$$

$$m ::= \boxed{c[\sigma]}$$

$$\sigma ::= \cdot \mid \pi_\ominus, \sigma$$

(b) Commands and machines

t	$\bar{\omega}(t)$
j_{π_+}	$\ominus$
$[\pi]$	$+$

(c) Polarity $\bar{\omega}(t) \in \{+, \ominus\}$

π	$\bar{\omega}(\pi)$
$\alpha^\ominus, V \cdot \pi$	$\ominus$
$\alpha^+, \tilde{\mu} x^+.c, \tilde{\mu}(x,y).c$	$+$

(d) Polarity $\bar{\omega}(\pi) \in \{+, \ominus\}$

$$\begin{array}{rcl}
\langle (t_\ominus\, u)_\varepsilon \| \pi_\varepsilon \rangle[\sigma] & \succ_p & \begin{cases} \langle t_\ominus \| V \cdot \pi_\varepsilon \rangle[\sigma] & \text{if } u = V \\ \langle u \| \tilde{\mu} x^+.\langle t_\ominus \| x^+ \cdot \pi_\varepsilon \rangle \rangle[\sigma] & \text{otherwise} \end{cases} \\
\langle \lambda x.t \| V \cdot \pi \rangle[\sigma] & \succ_p & \langle t[V/x] \| \pi \rangle[\sigma] \\
\hline
\langle \mathsf{let}\ q\ \mathsf{be}\ t_+\ \mathsf{in}\ u \| \pi \rangle[\sigma] & \succ_p & \langle t_+ \| \tilde{\mu} q.\langle u \| \pi \rangle \rangle[\sigma] \\
\langle V_+ \| \tilde{\mu} x^+.c \rangle[\sigma] & \succ_p & c[V_+/x^+][\sigma] \\
\hline
\langle (t_+, u) \| \pi \rangle[\sigma] & \succ_p & \langle t_+ \| \tilde{\mu} x^+.\langle (x^+, u) \| \pi \rangle \rangle[\sigma] \quad \text{if } t_+ \neq V \\
\langle (V, t_+) \| \pi \rangle[\sigma] & \succ_p & \langle t_+ \| \tilde{\mu} x^+.\langle (V, x^+) \| \pi \rangle \rangle[\sigma] \quad \text{if } t_+ \neq V \\
\langle (V, W) \| \tilde{\mu}(x,y).c \rangle[\sigma] & \succ_p & c[V/x, W/y][\sigma] \\
\hline
\langle \ell \| t_\ominus \cdot \pi_+ \rangle[\pi_\ominus, \sigma] & \succ_p & \langle t_\ominus \| \mathsf{j}_{\pi_+} \cdot \pi_\ominus \rangle[\sigma] \\
\langle \mathsf{j}_{\pi_+} \| \pi \rangle[\sigma] & \succ_p & \langle [\pi] \| \pi_+ \rangle[\sigma] \\
\langle \mathsf{send} \| [\pi_\varepsilon] \cdot t_\varepsilon \cdot \pi'_\ominus \rangle[\sigma] & \succ_p & \langle t_\varepsilon \| \pi_\varepsilon \rangle[\pi'_\ominus, \sigma] \\
\langle D_\to \| [V \cdot \pi] \cdot \pi_+ \rangle[\sigma] & \succ_p & \langle (V, [\pi]) \| \pi_+ \rangle[\sigma] \\
\langle D_\bot \| [\pi_\ominus] \cdot t_\varepsilon \cdot \pi'_\varepsilon \rangle[\sigma] & \succ_p & \langle t_\varepsilon \| \pi'_\varepsilon \rangle[\pi_\ominus, \sigma] \\
\langle D_\forall \| V \cdot \pi' \rangle[\sigma] & \succ_p & \langle V \| \pi' \rangle[\sigma]
\end{array}$$

(e) Rules of reduction

Figure IV.3: Abstract machine for the λℓ calculus

IV.3.5 shows how the operator C can be simulated with ℓ.

The role of the $\lambda\ell$ calculus is to show how the λC calculus can be extended so as to correspond to a natural deduction with an involutive negation. This is why two constraints guided the design of the calculus:

1. Negation must be there as a connective;
2. There must be a clear distinction between on the one hand quasi-proof terms, in other words programs, and on the other hand terms that appear during the evaluation in a machine.

Quasi-proof terms of the $\lambda\ell$ calculus are defined in Figure IV.1 on page 209. This calculus extends the λC calculus with a control operator ℓ that refines the C operator. Both terms and stacks have *polarities* determined by the function $\bar{\omega}$; let us write t_+, $t_\ominus$, π_+ or $\pi_\ominus$ to refer to a term or a stack of a given polarity.

A polarised predicate calculus is given in Figure IV.2 on page 210. New types $\sim$ and $\bot$ refine the interpretation of negation. The latter is interpreted depending on the polarity of A:

$$\neg A \stackrel{\text{def}}{=} \begin{cases} \sim N & \text{if } A = N \\ P \to \bot & \text{if } A = P \end{cases}$$

Figure IV.3 defines the evaluation of terms with a machine that extends Krivine's. Initial stacks are infinitely many constants $\alpha, \beta \ldots$ that have a fixed polarity.

The two above constraints prevail at times over simplicity. For instance, to ensure that we can statically determine a polarity for each term, application is annotated with the expected polarity of the result:

$$(t\,u)_+ \text{ or } (t\,u)_\ominus.$$

We omit this annotation when it can be deduced from context. (This issue finds a formal solution through sequent calculus in Section IV.4, at the price of relaxing constraint #2.)

IV.3.1 Negating a positive: polarised arrows

The negation of a positive formula P is given with $P \to \bot$. Let us first explain the polarised arrow.

The application is in call by value, by which we mean that in $t\ u$, the argument u is evaluated first, when it is not already a value:

$$\boxed{\langle t_\ominus\ u_+ \| \pi\rangle \succ_p \langle u_+ \| \tilde{\mu}x^+.\langle t_\ominus \| x^+\cdot\pi\rangle\rangle} \text{ if } u_+ \text{ is not a value}$$
$$\langle t_\ominus\ V \| \pi\rangle \succ_p \langle t_\ominus \| V\cdot\pi\rangle$$

In particular, stacks captured by control operators are guaranteed to be of the form $V\cdot\pi$ where V is a value.

Example IV.3. In this context, polarisation means that given three terms:

$$\Gamma \vdash t_+ : P$$
$$\Gamma, x^+ : P \vdash u_\ominus : N$$
$$\Gamma, y^\ominus : N \vdash v : A$$

there are two ways of composing them:

$$(\lambda y^\ominus.v)\ (\lambda x^+.u_\ominus\ t_+)_\ominus$$
$$(\lambda x^+.(\lambda y^\ominus.v)\ u_\ominus)\ t_+$$

which correspond to the following distinct behaviours:

$$\langle (\lambda y^\ominus.v)\ (\lambda x^+.u_\ominus\ t_+)_\ominus \| \pi\rangle \succ_p^* \langle v[(\lambda x^+.u_\ominus\ t_+)_\ominus/y^\ominus] \| \pi\rangle$$
$$\langle (\lambda x^+.(\lambda y^\ominus.v)\ u_\ominus)\ t_+ \| \pi\rangle \succ_p^* \langle t_+ \| \tilde{\mu}x^+.\langle (\lambda y^\ominus.v)\ u_\ominus \| \pi\rangle\rangle$$

Thus, for lack of associativity of composition, the λℓ calculus escapes from the following argument of category theory, which historically opposed the existence of non-boolean categorical models of classical logic: as it is well-known, a cartesian-closed category never has a dualising object $\bot$, that is to say which satisfies a natural isomorphism $\bot^{\bot^A} \simeq A$, unless it is a boolean algebra.

This follows more generally from the difficulty of interpreting negation already in intuitionistic logic: in a bi-cartesian-closed category, there is at most one morphism from any object to the initial object [LS86, p.67].

IV.3.2 Falsity: delimited control

We assume that execution happens in a machine of the following form:

$$\boxed{\langle t \| \pi \rangle [\pi_\ominus^1, \dots, \pi_\ominus^n]}$$

where $\sigma = \pi_\ominus^1, \dots, \pi_\ominus^n$ is a list of negative stacks.

As a consequence, the notation $c \succ_p c'$ is an abbreviation which denotes:

$$\forall \sigma,\ c[\sigma] \succ_p c'[\sigma]$$

In typed settings we may expect these stacks to be of type $\bot$. The list σ interacts with control operators: we shall see that the send operator lets it grow whereas the operator ℓ lets it shrink.

We can think of σ as a list of exception handlers, with terms of type $\bot$ being handled by raising an exception. (This interpretation is inspired from the decomposition of delimited control operators of Herbelin *et al.* [AHS04, HG08])

The goal is to reconcile the stack stop of the λC calculus with equations of extensionality that we will introduce in Section IV.4. The natural equation in this context, indeed, which the $\lambda\ell$ calculus rejects:

$$\forall \pi\ (\mathsf{stop} \simeq \pi)$$

identifies all the terms of of the λC calculus, as we will explain.

Recall that the stack stop appears during the reduction of the λC calculus as a context of type $\bot$ (Figure I.12 on page 81):

$$\langle C \| t \cdot \pi \rangle \succ_n \langle t \| \mathsf{k}_\pi \cdot \mathsf{stop} \rangle$$

As we shall see below, the operator ℓ replaces stop with the head of

the list σ:

$$\langle \ell \| t \cdot \pi \rangle [\pi'_\ominus, \sigma] \succ_p \langle t \| \mathrm{j}_\pi \cdot \pi'_\ominus \rangle [\sigma]$$

IV.3.3 Negating a negative: inspectable stacks

In order to interpret the negation of a negative formula N, we introduce the positive type $\sim N$ of inspectable stacks.

An *inspectable stack* is:

> a value that denotes a captured stack and that exports accessors to its components.

An inspectable stack is denoted with $V_+ = [\pi]$ (Figure IV.3a). The constants $D_\to$, $D_\forall$ and $D_\bot$ let us access the components of an inspectable stack:

$$\boxed{\begin{array}{ll} D_\to & : \sim(A \to B) \to A \otimes \sim B \\ D_\forall & : \sim(\forall x\, A) \to \exists x\, \sim A \\ D_\bot & : \sim\bot \to A \to A \end{array}}$$

$$\begin{array}{rcl} \langle D_\to \| [V \cdot \pi_1] \cdot \pi_2 \rangle & \succ_p & \langle (V, [\pi_1]) \| \pi_2 \rangle \\ \langle D_\forall \| [\pi_1] \cdot \pi_2 \rangle & \succ_p & \langle [\pi_1] \| \pi_2 \rangle \\ \langle D_\bot \| [\pi_\ominus] \cdot t \cdot \pi' \rangle & \succ_p & \langle t \| \pi' \rangle [\pi_\ominus] \end{array}$$

It is therefore important to distinguish the type $\sim N$ from the type $N \to \bot$ of continuations, in the sense of functions that do not return.

Also, a captured stack can be positive, which gives a value of the positive type $\sim P$. However we provide no accessor.

Remark That captured contexts are more primitive than continuations is obvious in programming languages, with the examples of the operation *getcontext* in the language C or the operation *thisContext* in the language Smalltalk, where the contents of captured contexts can be accessed.

Clements gives a theoretical status to the idea of high-level access to the components of the contexts [Cle06]. This allows for instance the port-

able, high-level, implementation of a debugger. Now this idea appears again through a theoretical question about type isomorphisms.

Example IV.4. We can combine $D_\forall$ and $D_\to$ to get a proof of:

$$\neg\forall \mathrm{x}(A \to N) \to \exists \mathrm{x}\,(A \otimes \neg N)$$

Take $B = N$ in the following:

$$\begin{aligned}
&D_{\forall\to} \stackrel{\text{def}}{=} \lambda x^+.\mathsf{let}\, y^+ \,\mathsf{be}\, D_\forall\, x^+ \,\mathsf{in}\, D_\to\, y^+\\
&D_{\forall\to} : \sim\forall \mathrm{x}(A \to B) \to \exists \mathrm{x}\,(A \otimes \sim B)\\
&\langle D_{\forall\to} \| [V\cdot\pi]\cdot\pi_+\rangle[\sigma] \succ^*_p \langle (V, [\pi]) \| \pi_+\rangle[\sigma]
\end{aligned}$$

Proof. In the following derivation we omit contexts Γ.

$$\dfrac{\dfrac{\overline{D_\forall} \qquad [x^+ : \sim\forall \mathrm{x}(A \to B)]}{D_\forall\, x^+ : \exists \mathrm{x} \sim(A \to B)}\,(\to_e) \qquad \dfrac{\dfrac{\overline{D_\to} \qquad [y^+ : \sim(A \to B)]}{D_\to\, y^+ : A \otimes \sim B}\,(\to_e)}{D_\to\, y^+ : \exists \mathrm{x}\,(A \otimes \sim B)}\,(\exists_i)}{\mathsf{let}\, y^+ \,\mathsf{be}\, D_\forall\, x^+ \,\mathsf{in}\, D_\to\, y^+ : \exists \mathrm{x}\,(A \otimes \sim B)}\,(\exists_e)$$
$$\dfrac{}{D_{\forall\to} : \sim\forall \mathrm{x}(A \to B) \to \exists \mathrm{x}\,(A \otimes \sim B)}\,(\to_i)$$

We have the reduction:

$$\begin{aligned}
&\langle D_{\forall\to} \| [V\cdot\pi]\cdot\pi_+\rangle[\sigma]\\
&\succ_p \langle \mathsf{let}\, y^+ \,\mathsf{be}\, D_\forall\, [V\cdot\pi] \,\mathsf{in}\, D_\to\, y^+ \| \pi_+\rangle[\sigma]\\
&\succ_p \langle D_\forall\, [V\cdot\pi] \| \tilde{\mu} y^+.\langle D_\to\, y^+ \| \pi_+\rangle\rangle[\sigma]\\
&\succ_p \langle D_\forall \| [V\cdot\pi]\cdot\tilde{\mu} y^+.\langle D_\to\, y^+ \| \pi_+\rangle\rangle[\sigma]\\
&\succ_p \langle [V\cdot\pi] \| \tilde{\mu} y^+.\langle D_\to\, y^+ \| \pi_+\rangle\rangle[\sigma]\\
&\succ_p \langle D_\to\, [V\cdot\pi] \| \pi_+\rangle[\sigma]\\
&\succ_p \langle D_\to \| [V\cdot\pi]\cdot\pi_+\rangle[\sigma]\\
&\succ_p \langle (V, [\pi]) \| \pi_+\rangle[\sigma]
\end{aligned}$$

∎

The immediateness of the operation is in sharp contrast with its homologue of the λC calculus given in Example I.9. The essential

difference is that $D_{\forall\to}$ accesses the components of a stack which is already captured, whereas the term of the λC calculus contains two control operators.[2]

Example IV.5. We can combine $D_\to$ and $D_\bot$ to obtain a proof in particular of $\neg\neg P \to P$ (take $A = P$):

$$
\begin{aligned}
&D_\neg \stackrel{\text{def}}{=} \lambda x^+.\mathsf{let}\,(y^\varepsilon, z^+)\,\mathsf{be}\,D_\to\, x^+\,\mathsf{in}\,(D_\bot\, z^+\, y^\varepsilon)_\varepsilon\\
&D_\neg : \sim(A\to\bot)\to A\\
&\langle D_\neg \,\|\, [V_\varepsilon\cdot\pi_\ominus]\cdot\pi'_\varepsilon\rangle[\sigma] \succ_p^* \langle V_\varepsilon \,\|\, \pi'_\varepsilon\rangle[\pi_\ominus, \sigma]
\end{aligned}
$$

The term $D_\neg$ keeps $\pi_\ominus$ at the head of the list σ. A variant erases $\pi_\ominus$:

$$
\begin{aligned}
&D'_\neg \stackrel{\text{def}}{=} \lambda x^+.\mathsf{let}\,(y^\varepsilon, z^+)\,\mathsf{be}\,D_\to\, x^+\,\mathsf{in}\,y^\varepsilon\\
&D'_\neg : \sim(A\to\bot)\to A \;\mathsf{avec}\;\varepsilon = \bar\omega(A)\\
&\langle D'_\neg \,\|\, [V_\varepsilon\cdot\pi_\ominus]\cdot\pi'_\varepsilon\rangle[\sigma] \succ_p^* \langle V_\varepsilon \,\|\, \pi'_\varepsilon\rangle[\sigma]
\end{aligned}
$$

Proof. In the following derivation, we omit contexts Γ.

$$
\dfrac{\dfrac{\dfrac{[x^+ : \sim(A\to\bot)]}{D_\to\, x^+ : A\otimes\sim\bot} \qquad \dfrac{\dfrac{[z^+ : \sim\bot]}{D_\bot\, z^+ : A\to A} \qquad [y^\varepsilon : A]}{(D_\bot\, z^+\, y^\varepsilon)_\varepsilon : A}\,(\to_e)}{\mathsf{let}\,(y^\varepsilon, z^+)\,\mathsf{be}\,D_\to\, x^+\,\mathsf{in}\,(D_\bot\, z^+\, y^\varepsilon)_\varepsilon : A}\,(\otimes_e)}{D_\neg : \sim(A\to\bot)\to A}\,(\to_i)
$$

(where $\varepsilon = \bar\omega(A)$).

We have the reduction:

$$
\begin{aligned}
&\langle D_\neg \,\|\, [V_\varepsilon\cdot\pi_\ominus]\cdot\pi'_\varepsilon\rangle[\sigma]\\
&\succ_p \langle \mathsf{let}\,(y^\varepsilon, z^+)\,\mathsf{be}\,D_\to\,[V_\varepsilon\cdot\pi_\ominus]\,\mathsf{in}\,(D_\bot\, z^+\, y^\varepsilon)_\varepsilon \,\|\, \pi'_\varepsilon\rangle[\sigma]\\
&\succ_p \langle D_\to\,[V_\varepsilon\cdot\pi_\ominus] \,\|\, \tilde\mu(y^\varepsilon, z^+).\langle (D_\bot\, z^+\, y^\varepsilon)_\varepsilon \,\|\, \pi'_\varepsilon\rangle\rangle[\sigma]
\end{aligned}
$$

[2] The new term would be simpler still if we took the trouble of introducing a notion of subtyping. The behaviour of $D_\forall$ indeed corresponds to a subtyping rule $\sim(\forall x\, A) <: \exists x\, \sim A$.

$$\begin{aligned}
&\succ_p \langle D_{\to} \| [V_\varepsilon \cdot \pi_\ominus] \cdot \tilde{\mu}(y^\varepsilon, z^+).\langle (D_\perp z^+ y^\varepsilon)_\varepsilon \| \pi'_\varepsilon \rangle \rangle [\sigma] \\
&\succ_p \langle (V_\varepsilon, [\pi_\ominus]) \| \tilde{\mu}(y^\varepsilon, z^+).\langle (D_\perp z^+ y^\varepsilon)_\varepsilon \| \pi'_\varepsilon \rangle \rangle [\sigma] \\
&\succ_p \langle (D_\perp [\pi_\ominus] V_\varepsilon)_\varepsilon \| \pi'_\varepsilon \rangle [\sigma] \\
&\succ_p \langle D_\perp [\pi_\ominus] \| V_\varepsilon \cdot \pi'_\varepsilon \rangle [\sigma] \\
&\succ_p \langle D_\perp \| [\pi_\ominus] \cdot V_\varepsilon \cdot \pi'_\varepsilon \rangle [\sigma] \\
&\succ_p \langle V_\varepsilon \| \pi'_\varepsilon \rangle [\pi_\ominus, \sigma] \qquad \blacksquare
\end{aligned}$$

The isomorphism $\neg\neg P \simeq P$ will discriminate between $D_\neg$ and $D'_\neg$: indeed, the invertible morphism is going to be $D_\neg$ and not $D'_\neg$.

IV.3.4 Capturing and installing stacks

Given a captured stack $[\pi]$, the stack is re-installed as the context of another term t by the constant `send`:

$$\langle \mathsf{send} \| [\pi] \cdot t \cdot \pi'_\ominus \rangle [\sigma] \succ_p \langle t \| \pi \rangle [\pi'_\ominus, \sigma]$$

In other words, the constant `send` converts a captured stack into a continuation:

$$\boxed{\mathsf{send} : {\sim}A \to A \to \perp}$$

The stack $\pi'_\ominus$, supposedly of type $\perp$ according to the type of `send`, is added on top of σ.

The operator responsible for the apparition of inspectable stacks is ℓ:

$$\boxed{\ell : (A \to \perp) \to {\sim}A}$$

The notation $\ell x.t_\ominus$ stands for $(\ell \lambda x.t_\ominus)_+$. It evaluates $t_\ominus$ until x comes in head position, that is to say in front of a stack π. When this happens, the operator ℓ captures π and supplies the inspectable stack $[\pi]$ to the context where ℓ was applied. Last, the operator ℓ falls back to the head of the list σ in case $t_\ominus$ returns without using x.[3]

[3] This comes as an explanation to the fact that Murthy's computational interpretation of **LC** must evaluate "*under the λ-abstraction*" [Mur92]. Murthy was,

This operation is formally described by introducing the j operator. The operator ℓ saves in j the context π_+ in which ℓ is applied, and installs the head $\pi_\ominus$ as the new context:

$$\langle \ell \| t_\ominus \cdot \pi_+ \rangle [\pi_\ominus, \sigma] \succ_p \langle t_\ominus \| \mathsf{j}_{\pi_+} \cdot \pi_\ominus \rangle [\sigma]$$

Once the operator j_{π_+} comes in head position, it captures the stack and restores the context π_+:

$$\langle \mathsf{j}_{\pi_+} \| \pi \rangle [\sigma] \succ_p \langle [\pi] \| \pi_+ \rangle [\sigma]$$

Example IV.6. Using ℓ, we derive an operator $\mathcal{A}$ (*abort*) which interprets the rule *Ex Falso Quodlibet:*

$$\begin{array}{l} \mathcal{A}_\varepsilon \stackrel{\text{def}}{=} \lambda x^\ominus.\mathsf{let}\ (y^\varepsilon, z^+)\ \mathsf{be}\ D_\to\ \ell\_.x^\ominus\ \mathsf{in}\ y^\varepsilon \\ \mathcal{A}_\varepsilon : \bot \to A\ (\text{where } \varepsilon = \bar{\omega}(A)) \\ \langle \mathcal{A}_\varepsilon \| t_\ominus \cdot \pi_\varepsilon \rangle [\pi'_\ominus, \sigma] \succ_p^* \langle t_\ominus \| \pi'_\ominus \rangle [\sigma] \end{array}$$

Proof. There is the derivation:

$$\dfrac{\dfrac{\overline{D_\to} \qquad \dfrac{\overline{\ell} \qquad \dfrac{x^\ominus : \bot}{\lambda\_.x^\ominus : (A \to B) \to \bot}\,(\to_i)}{\ell\,\lambda\_.x^\ominus : \sim(A \to B)}\,(\to_e)}{D_\to\ \ell\_.x^\ominus : A \otimes \sim B}\,(\to_e) \qquad [y^\varepsilon : A]}{\mathsf{let}\ (y^\varepsilon, z^+)\ \mathsf{be}\ D_\to\ \ell\_.x^\ominus\ \mathsf{in}\ y^\varepsilon : A}\,(\otimes_e)}{\mathcal{A}_\varepsilon : \bot \to A}\,(\to_i)$$

There also is the reduction:

$$\begin{array}{l} \langle \mathcal{A}_\varepsilon \| t_\ominus \cdot \pi_\varepsilon \rangle [\pi'_\ominus, \sigma] \\ \succ_p \langle \mathsf{let}\ (y^\varepsilon, z^+)\ \mathsf{be}\ D_\otimes\ \ell\_.t_\ominus\ \mathsf{in}\ y^\varepsilon \| \pi_\varepsilon \rangle [\pi'_\ominus, \sigma] \\ \succ_p \langle D_\otimes\ \ell\_.t_\ominus \| \tilde{\mu}(y^\varepsilon, z^+).\langle y^\varepsilon \| \pi_\varepsilon \rangle \rangle [\pi'_\ominus, \sigma] \\ \succ_p \left\langle \ell\_.t_\ominus \middle\| \tilde{\mu}x^+.\langle D_\otimes \| x^+ \cdot \tilde{\mu}(y^\varepsilon, z).\langle y^\varepsilon \| \pi_\varepsilon \rangle \rangle \right\rangle [\pi'_\ominus, \sigma] \end{array}$$

in advance of his time, in fact describing the behaviour of a variant of ℓ.

$$\succ_p \left\langle \ell \middle\| \lambda\_.t_\ominus \cdot \tilde{\mu}x^+.\langle D_\otimes \| x^+ \cdot \tilde{\mu}(y^\varepsilon, z).\langle y^\varepsilon \| \pi_\varepsilon \rangle \rangle \right\rangle [\pi'_\ominus, \sigma]$$
$$\succ_p \left\langle \lambda\_.t_\ominus \middle\| \mathsf{j}_{\tilde{\mu}x^+.\langle D_\otimes \| x^+ \cdot \tilde{\mu}(y^\varepsilon, z).\langle y^\varepsilon \| \pi_\varepsilon \rangle \rangle} \cdot \pi'_\ominus \right\rangle [\sigma]$$
$$\succ_p \langle t_\ominus \| \pi'_\ominus \rangle [\sigma] \qquad \blacksquare$$

Example IV.7. By combining $\lambda xy.yx : A \to ((A \to \bot) \to \bot)$ with ℓ we obtain a proof of $P \to \neg\neg P$ (take $A = P$):

$$E : A \to {\sim}(A \to \bot)$$
$$E \stackrel{\text{def}}{=} \lambda x.\ell y^\ominus.(y^\ominus x)_\ominus$$
$$\langle E \| V \cdot \pi_+ \rangle [\pi_\ominus, \sigma] \succ_p^* \langle [V \cdot \pi_\ominus] \| \pi_+ \rangle [\sigma]$$

Proof. We have the reduction:

$$\langle E \| V \cdot \pi_+ \rangle [\pi_\ominus, \sigma]$$
$$\succ_p \langle \ell y^\ominus.(y^\ominus V)_\ominus \| \pi_+ \rangle [\pi_\ominus, \sigma]$$
$$\succ_p \langle \ell \| \lambda y^\ominus.(y^\ominus V)_\ominus \cdot \pi_+ \rangle [\pi_\ominus, \sigma]$$
$$\succ_p \langle \lambda y^\ominus.(y^\ominus V)_\ominus \| \mathsf{j}_{\pi_+} \cdot \pi_\ominus \rangle [\sigma]$$
$$\succ_p \langle (\mathsf{j}_{\pi_+} V)_\ominus \| \pi_\ominus \rangle [\sigma]$$
$$\succ_p \langle \mathsf{j}_{\pi_+} \| V \cdot \pi_\ominus \rangle [\sigma]$$
$$\succ_p \langle [V \cdot \pi_\ominus] \| \pi_+ \rangle [\sigma] \qquad \blacksquare$$

Example IV.8. We obtain the elimination of double negation $\neg\neg N \to N$ as follows (take $A = N$):

$$\mathcal{T} : ({\sim}A \to \bot) \to A$$
$$\mathcal{T} \stackrel{\text{def}}{=} \lambda x^\ominus.(D_\neg\, \ell y^\ominus.(x^\ominus (\ell y^\ominus)_+))$$
$$\langle \mathcal{T} \| t_\ominus \cdot \pi \rangle [\pi'_\ominus, \pi''_\ominus, \sigma] \succ_p^* \langle t_\ominus \| [\pi] \cdot \pi'_\ominus \rangle [\pi''_\ominus, \sigma]$$

Proof. We have the derivation:

$$\dfrac{\overline{D_\neg} \qquad \dfrac{\overline{\ell} \qquad \dfrac{\dfrac{[x^\ominus : {\sim}A \to \bot] \qquad \dfrac{\overline{\ell} \qquad [y^\ominus : A \to \bot]}{(\ell\, y^\ominus)_+ : {\sim}A}\,(\to_e)}{x^\ominus\,(\ell\, y^\ominus)_+ : \bot}\,(\to_e)}{\lambda y^\ominus.(x^\ominus\,(\ell\, y^\ominus)_+) : (A \to \bot) \to \bot}\,(\to_i)}{\ell y^\ominus.(x^\ominus\,(\ell\, y^\ominus)_+) : {\sim}(A \to \bot)}\,(\to_e)}{D_\neg\, \ell y^\ominus.(x^\ominus\,(\ell\, y^\ominus)_+) : A}\,(\to_e)$$
$$\mathcal{T} : ({\sim}A \to \bot) \to A\,(\to_i)$$

We also have the reduction:

$$\begin{aligned}
&\langle \mathcal{T} \parallel t_\ominus \cdot \pi \rangle [\pi'_\ominus, \pi''_\ominus, \sigma] \\
&\succ_p \langle D_\neg\, \ell y^\ominus.(t_\ominus\,(\ell\, y^\ominus)_+) \parallel \pi \rangle [\pi'_\ominus, \pi''_\ominus, \sigma] \\
&\succ_p \left\langle \ell y^\ominus.(t_\ominus\,(\ell\, y^\ominus)_+) \,\middle\|\, \tilde{\mu} x^+.\langle D_\neg \parallel x^+ \cdot \pi \rangle \right\rangle [\pi'_\ominus, \pi''_\ominus, \sigma] \\
&\succ_p \left\langle \ell \,\middle\|\, \lambda y^\ominus.(t_\ominus\,(\ell\, y^\ominus)_+) \cdot \tilde{\mu} x^+.\langle D_\neg \parallel x^+ \cdot \pi \rangle \right\rangle [\pi'_\ominus, \pi''_\ominus, \sigma] \\
&\succ_p \left\langle \lambda y^\ominus.(t_\ominus\,(\ell\, y^\ominus)_+) \,\middle\|\, \mathrm{j}_{\tilde{\mu} x^+.\langle D_\neg \parallel x^+ \cdot \pi \rangle} \cdot \pi'_\ominus \right\rangle [\pi''_\ominus, \sigma] \\
&\succ_p \left\langle t_\ominus\,(\ell\, \mathrm{j}_{\tilde{\mu} x^+.\langle D_\neg \parallel x^+ \cdot \pi \rangle})_+ \,\middle\|\, \pi'_\ominus \right\rangle [\pi''_\ominus, \sigma] \\
&\succ_p \left\langle (\ell\, \mathrm{j}_{\tilde{\mu} x^+.\langle D_\neg \parallel x^+ \cdot \pi \rangle})_+ \,\middle\|\, \tilde{\mu} x^+.\langle t_\ominus \parallel x^+ \cdot \pi'_\ominus \rangle \right\rangle [\pi''_\ominus, \sigma] \\
&\succ_p \left\langle \ell \,\middle\|\, \mathrm{j}_{\tilde{\mu} x^+.\langle D_\neg \parallel x^+ \cdot \pi \rangle} \cdot \tilde{\mu} x^+.\langle t_\ominus \parallel x^+ \cdot \pi'_\ominus \rangle \right\rangle [\pi''_\ominus, \sigma] \\
&\succ_p \left\langle \mathrm{j}_{\tilde{\mu} x^+.\langle D_\neg \parallel x^+ \cdot \pi \rangle} \,\middle\|\, \mathrm{j}_{\tilde{\mu} x^+.\langle t_\ominus \parallel x^+ \cdot \pi'_\ominus \rangle} \cdot \pi''_\ominus \right\rangle [\sigma] \\
&\succ_p \left\langle [\mathrm{j}_{\tilde{\mu} x^+.\langle t_\ominus \parallel x^+ \cdot \pi'_\ominus \rangle} \cdot \pi''_\ominus] \,\middle\|\, \tilde{\mu} x^+.\langle D_\neg \parallel x^+ \cdot \pi \rangle \right\rangle [\sigma] \\
&\succ_p \left\langle D_\neg \,\middle\|\, [\mathrm{j}_{\tilde{\mu} x^+.\langle t_\ominus \parallel x^+ \cdot \pi'_\ominus \rangle} \cdot \pi''_\ominus] \cdot \pi \right\rangle [\sigma] \\
&\succ_p \left\langle \mathrm{j}_{\tilde{\mu} x^+.\langle t_\ominus \parallel x^+ \cdot \pi'_\ominus \rangle} \,\middle\|\, \pi \right\rangle [\pi''_\ominus, \sigma] \\
&\succ_p \left\langle [\pi] \,\middle\|\, \tilde{\mu} x^+.\langle t_\ominus \parallel x^+ \cdot \pi'_\ominus \rangle \right\rangle [\pi''_\ominus, \sigma] \\
&\succ_p \langle t_\ominus \parallel [\pi] \cdot \pi'_\ominus \rangle [\pi''_\ominus, \sigma]
\end{aligned}$$

∎

IV.3.5 Refining C

The following example shows that the operators ℓ and j refine the operators C and k. Assume that we have chosen a distinguished negative stack constant stop.

Example IV.9. Let us take:

$$C \stackrel{\text{def}}{=} \lambda x^{\ominus}.D'_{\neg}(\ell\, x^{\ominus})_{+}$$
$$\mathsf{k}_{\pi_\varepsilon} \stackrel{\text{def}}{=} \mathsf{j}_{\tilde{\mu}x^{+}.\langle D'_{\neg}\, x^{+} \| \pi_\varepsilon \rangle}$$

We have:

$$C : ((N \to \bot) \to \bot) \to N$$
$$\langle C \| t_{\ominus} \cdot \pi_{\ominus} \rangle [\mathsf{stop}, \sigma] \succ_p^* \langle t_{\ominus} \| \mathsf{k}_{\pi_{\ominus}} \cdot \mathsf{stop} \rangle [\sigma]$$
$$\langle \mathsf{k}_{\pi_\varepsilon} \| t_\varepsilon \cdot \pi'_{\ominus} \rangle \succ_p^* \langle t_\varepsilon \| \pi_\varepsilon \rangle$$

Proof. We have the following derivation:

$$\dfrac{\dfrac{x^{\ominus} : (A \to \bot) \to \bot}{(\ell\, x^{\ominus})_{+} : {\sim}(A \to \bot)}}{D'_{\neg}(\ell\, x^{\ominus})_{+} : A}$$

We have the reductions:

$$\begin{aligned}
&\langle C \| t_{\ominus} \cdot \pi_{\ominus} \rangle [\mathsf{stop}, \sigma] \\
&\succ_p \langle D'_{\neg}(\ell\, t_{\ominus})_{+} \| \pi_{\ominus} \rangle [\mathsf{stop}, \sigma] \\
&\succ_p \langle (\ell\, t_{\ominus})_{+} \| \tilde{\mu}x^{+}.\langle D'_{\neg}\, x^{+} \| \pi_{\ominus} \rangle \rangle [\mathsf{stop}, \sigma] \\
&\succ_p \langle (\ell\, t_{\ominus})_{+} \| \tilde{\mu}x^{+}.\langle D'_{\neg}\, x^{+} \| \pi_{\ominus} \rangle \rangle [\mathsf{stop}, \sigma] \\
&\succ_p \langle \ell \| t_{\ominus} \cdot \tilde{\mu}x^{+}.\langle D'_{\neg}\, x^{+} \| \pi_{\ominus} \rangle \rangle [\mathsf{stop}, \sigma] \\
&\succ_p \langle t_{\ominus} \| \mathsf{j}_{\tilde{\mu}x^{+}.\langle D'_{\neg}\, x^{+} \| \pi_{\ominus} \rangle} \cdot \mathsf{stop} \rangle [\sigma] \\
&= \langle t_{\ominus} \| \mathsf{k}_{\pi_{\ominus}} \cdot \mathsf{stop} \rangle [\sigma] \\
&\langle \mathsf{k}_{\pi_{\ominus}} \| t_{\ominus} \cdot \pi'_{\ominus} \rangle [\sigma] \\
&\succ_p \langle [t_{\ominus} \cdot \pi'_{\ominus}] \| \tilde{\mu}x^{+}.\langle D'_{\neg}\, x^{+} \| \pi_{\ominus} \rangle \rangle [\sigma]
\end{aligned}$$

$$\succ_p \langle D'_{\neg}\,[t_{\ominus}\cdot\pi'_{\ominus}] \,\|\, \pi_{\ominus}\rangle[\sigma]$$
$$\succ_p \langle D'_{\neg} \,\|\, [t_{\ominus}\cdot\pi'_{\ominus}]\cdot\pi_{\ominus}\rangle[\sigma]$$
$$\succ_p \langle t_{\ominus} \,\|\, \pi_{\ominus}\rangle[\sigma] \qquad \blacksquare$$

As a consequence, given an initial list $\sigma = \mathsf{stop}, \dots, \mathsf{stop}$ which is long enough, the reduction in the $\lambda\ell$ calculus simulates the reduction in the $\lambda\mathcal{C}$ calculus in the following sense:

$$c \succ_n^+ c' \Rightarrow \exists n \in \mathbb{N}\; c\left[\mathsf{stop}^n\right] \succ_p^+ c'$$

IV.4 The calculus $\mathbf{L}_{\mathbf{pol},\hat{\mathbf{tp}}^{\ominus}}$

In this section we introduce the calculus $\mathbf{L}_{\mathrm{pol},\hat{\mathrm{tp}}^{\ominus}}$. In IV.4.1 we explain its operations, and we show how they implement the $\lambda\ell$ calculus. Then, in Section IV.4.2 on page 230 we introduce a CPS translation and we prove that not all terms are identified. Finally, in IV.4.3 we explain how $\mathbf{L}_{\mathrm{pol},\hat{\mathrm{tp}}^{\ominus}}$ relates to other calculi with control operators.

We present the calculus $\mathbf{L}_{\mathrm{pol},\hat{\mathrm{tp}}^{\ominus}}$ in Figure IV.4 on page 225. It enriches $\mathbf{L}_{\mathrm{n}}$ with:

- The positive conjunction $\otimes$, with the corresponding focusing rules;
- The positive negation $\sim$, and the corresponding focusing rule;
- Alternate rules for $\bot$ for which we introduce the operators $\hat{\mathrm{tp}}$ and $\mu\hat{\mathrm{tp}}$.

The calculus $\mathbf{L}_{\mathrm{pol},\hat{\mathrm{tp}}^{\ominus}}$ differs from the calculus $\mathbf{L}_{\mathrm{pol},\hat{\mathrm{tp}}^{+}}$ from Chapter III in the fact that the term $\mu\hat{\mathrm{tp}}.c$ is negative in $\mathbf{L}_{\mathrm{pol},\hat{\mathrm{tp}}^{\ominus}}$ while it is positive in $\mathbf{L}_{\mathrm{pol},\hat{\mathrm{tp}}^{+}}$.

For conciseness we chose to leave aside the shifts of the calculus $\mathbf{L}_{\mathrm{dup}}$ from II, although there is no difficulty in adding them. The calculus still has weak shifts, as we will see in Section IV.6.

IV.4.1 Interpreting λℓ

Positive bindings

The binder $\tilde{\mu}$ allows us to derive the bindings $\mathsf{let}\ x\ \mathsf{be}\ t\ \mathsf{in}\ u$. The term $\mathsf{let}\ x^+\ \mathsf{be}\ t_+\ \mathsf{in}\ u$ of the λℓ calculus is obtained through the definition:

$$\boxed{\mathsf{let}\ x^+\ \mathsf{be}\ t_+\ \mathsf{in}\ u_\varepsilon \stackrel{\text{def}}{=} \mu\alpha^\varepsilon.\langle t_+ \,\|\, \tilde{\mu}x^+.\langle u_\varepsilon \,\|\, \alpha^\varepsilon\rangle\rangle}\,,$$

for any terms t_+ and u of $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^\ominus}$. The binder $\tilde{\mu}$ also decomposes pairs of values. Thus we define similarly:

$$\boxed{\mathsf{let}\ (x,y)\ \mathsf{be}\ t_+\ \mathsf{in}\ u_\varepsilon \stackrel{\text{def}}{=} \mu\alpha^\varepsilon.\langle t_+ \,\|\, \tilde{\mu}(x,y).\langle u_\varepsilon \,\|\, \alpha^\varepsilon\rangle\rangle}\,.$$

Last, application is defined with:

$$\boxed{(t_\ominus\, u)_\varepsilon \stackrel{\text{def}}{=} \mu\alpha^\varepsilon.\langle t_\ominus \,\|\, u\cdot\alpha^\varepsilon\rangle}\,.$$

Evaluation in call by value is enforced by the focusing rule $(\varsigma_{\to_1})$. Indeed, it implies that if u_+ is not a value, then we have:

$$\begin{aligned}\langle (t_\ominus\, u_+)_\varepsilon \,\|\, \pi_\varepsilon\rangle &= \langle \mu\alpha^\varepsilon.\langle t_\ominus \,\|\, u_+\cdot\alpha^\varepsilon\rangle \,\|\, \pi_\varepsilon\rangle \\ &\rhd_{\mathrm{R}_p} \langle \mu\alpha^\varepsilon.\langle t_\ominus \,\|\, u_+\cdot\alpha^\varepsilon\rangle \,\|\, \pi_\varepsilon\rangle \\ &\rhd_{\mathrm{R}_p} \langle t_\ominus \,\|\, u_+\cdot\pi_\varepsilon\rangle \\ &\rhd_{\mathrm{R}_p} \langle u_+ \,\|\, \tilde{\mu}x^+.\langle t_\ominus \,\|\, x^+\cdot\pi_\varepsilon\rangle\rangle\end{aligned}$$

This coincides with the reduction rule of λℓ.

The operator $\mu\widehat{\mathrm{tp}}$

As in the calculus $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^+}$, the operator $\mu\widehat{\mathrm{tp}}$ implements the list of stacks, which is defined inductively as follows:

$$\boxed{\begin{aligned} c[\,] &\stackrel{\text{def}}{=} c \\ c[\pi_\ominus^1, \ldots, \pi_\ominus^n] &\stackrel{\text{def}}{=} \langle \mu\widehat{\mathrm{tp}}.c \,\|\, \pi_\ominus^1\rangle[\pi_\ominus^2, \ldots, \pi_\ominus^n] \end{aligned}}$$

$\boxed{\ldots}$: Main additions to Figure I.10.

$$t\begin{cases} t_{\ominus} ::= x^{\ominus} \mid \lambda x.t \mid \mu\alpha^{\ominus}.c \mid \boxed{\mu\hat{\mathrm{tp}}.c} \\ t_{+} ::= \boxed{x^{+} \mid (t,t) \mid [e] \mid \mu\alpha^{+}.c} \end{cases} \qquad t \supseteq V\begin{cases} t_{\ominus} \\ V_{+} ::= \boxed{x^{+} \mid (V,V) \mid [\pi]} \end{cases}$$

$$e\begin{cases} e_{\ominus} ::= \alpha^{\ominus} \mid t\cdot e \mid \tilde{\mu}x^{\ominus}.c \mid \boxed{\hat{\mathrm{tp}}} \\ e_{+} ::= \boxed{\alpha^{+} \mid \tilde{\mu}x^{+}.c \mid \tilde{\mu}(x,y).c \mid \tilde{\mu}[\alpha].c} \end{cases} \qquad e \supseteq \pi\begin{cases} \pi_{\ominus} ::= \alpha^{\ominus} \mid V\cdot\pi \\ e_{+} \end{cases}$$

$$c ::= \boxed{\langle t_{+} \| e_{+} \rangle} \mid \langle t_{\ominus} \| e_{\ominus} \rangle$$

(a) Terms, values, contexts, stacks, and commands

$$\begin{array}{lll}
(R_{\tilde{\mu}}) & \langle V \| \tilde{\mu}x.c \rangle & \rhd_{\mathrm{R}_p} c[V/x] \\
(R_{\mu}) & \langle \mu\alpha.c \| \pi \rangle & \rhd_{\mathrm{R}_p} c[\pi/\alpha] \\
(R_{\to}) & \langle \lambda x.t \| V\cdot\pi \rangle^{\dagger} & \rhd_{\mathrm{R}_p} \langle t[V/x] \| \pi \rangle \\
(R_{\otimes}) & \langle (V,W) \| \tilde{\mu}(x,y).c \rangle^{\dagger} & \rhd_{\mathrm{R}_p} c[V/x, W/y] \\
(R_{\sim}) & \langle [\pi_{\varepsilon}] \| \tilde{\mu}[\alpha^{\varepsilon}].c \rangle & \rhd_{\mathrm{R}_p} c[\pi_{\varepsilon}/\alpha^{\varepsilon}] \\
(R_{\hat{\mathrm{tp}}}) & \mu\hat{\mathrm{tp}}.\langle t_{\ominus} \| \hat{\mathrm{tp}} \rangle^{\ddagger} & \rhd_{\mathrm{R}_p} t_{\ominus}
\end{array}$$

†When polarities match pairwise. ‡Even if $\hat{\mathrm{tp}}$ occurs in $t_{\ominus}$

$$\begin{array}{lll}
(\varsigma_{\otimes_1}) & \langle (t_{+},u) \| e_{+} \rangle^{\dagger} & \rhd_{\mathrm{R}_p} \langle t_{+} \| \tilde{\mu}x.\langle (x,u) \| e_{+} \rangle \rangle \\
(\varsigma_{\otimes_2}) & \langle (V,t_{+}) \| e_{+} \rangle^{\dagger} & \rhd_{\mathrm{R}_p} \langle t_{+} \| \tilde{\mu}y.\langle (V,y) \| e_{+} \rangle \rangle \\
(\varsigma_{\sim}) & \langle [e_{\ominus}] \| e_{+} \rangle^{\ddagger} & \rhd_{\mathrm{R}_p} \langle \mu\beta^{\ominus}.\langle [\beta^{\ominus}] \| e_{+} \rangle \| e_{\ominus} \rangle \\
(\varsigma_{\to_1}) & \langle u_{\ominus} \| t_{+}\cdot e \rangle^{\dagger} & \rhd_{\mathrm{R}_p} \langle t_{+} \| \tilde{\mu}x.\langle u_{\ominus} \| x\cdot e \rangle \rangle \\
(\varsigma_{\to_2}) & \langle u_{\ominus} \| V\cdot e_{\ominus} \rangle^{\ddagger} & \rhd_{\mathrm{R}_p} \langle \mu\alpha.\langle u_{\ominus} \| V\cdot\alpha \rangle \| e_{\ominus} \rangle
\end{array}$$

†When t_{+} is not a value. ‡When $e_{\ominus}$ is not a stack.

(b) Reduction rules

$$\begin{array}{llll}
(E_{\tilde{\mu}}) & e & \rhd_{\mathrm{E}_p} \tilde{\mu}x.\langle x \| e \rangle & (E_{\otimes}) \ \pi_{\ominus} \rhd_{\mathrm{E}_p} \tilde{\mu}(x,y).\langle (x,y) \| \pi_{\ominus} \rangle \\
(E_{\mu}) & t & \rhd_{\mathrm{E}_p} \mu\alpha.\langle t \| \alpha \rangle & (E_{\sim}) \ e_{+} \rhd_{\mathrm{E}_p} \tilde{\mu}[\alpha].\langle [\alpha] \| e_{+} \rangle \\
(E_{\to}) & t_{\ominus} & \rhd_{\mathrm{E}_p} \lambda x.\mu\alpha.\langle t_{\ominus} \| x\cdot\alpha \rangle & (E_{\hat{\mathrm{tp}}}) \ c \rhd_{\mathrm{E}_p} \langle \mu\hat{\mathrm{tp}}.c \| \hat{\mathrm{tp}} \rangle
\end{array}$$

(c) Expansion rules (new variables are fresh)

Figure IV.4: $\mathbf{L}_{\mathrm{pol},\hat{\mathrm{tp}}^{\ominus}}$: the calculus

[…]: Main additions to Figure I.11.

$$A, B \begin{cases} N, M ::= A \to B \mid \forall \mathrm{x}\, N \mid \bot \\ P, Q ::= \boxed{X(\mathrm{t}_1, \dots, \mathrm{t}_n) \mid A \otimes B \mid \exists x\, P \mid {\sim}A} \end{cases}$$

(a) Formulae

$$\boxed{f : A ::= f^{+} : P \mid f^{\ominus} : N}$$

for $f \in \{x, \alpha, t, e\}$.

$$\Gamma = \vec{x_i} : \vec{A_i} \qquad \Delta = \vec{\alpha_j} : \vec{B_j}$$

$$\Gamma \vdash t : A \mid \Delta \qquad \Gamma \mid e : A \vdash \Delta$$

$$c : (\Gamma \vdash \Delta)$$

(b) Judgements

$$\frac{}{\Gamma, x : A \vdash x : A \mid \Delta}\ (\vdash \mathrm{ax}) \qquad \frac{}{\Gamma \mid \alpha : A \vdash \alpha : A, \Delta}\ (\mathrm{ax} \vdash)$$

$$\frac{c : (\Gamma, x : A \vdash \Delta)}{\Gamma, \tilde{\mu} x.c : A \vdash \Delta}\ (\tilde{\mu} \vdash) \qquad \frac{c : (\Gamma \vdash \alpha : A, \Delta)}{\Gamma \vdash \mu \alpha.c : A \mid \Delta}\ (\vdash \mu)$$

$$\frac{\Gamma \vdash t : A \mid \Delta \qquad \Gamma \mid e : A \vdash \Delta}{\langle t \parallel e \rangle : (\Gamma \vdash \Delta)}\ (\mathrm{cut})$$

(c) Group Identity and Structure

$$\frac{\Gamma, x : A \vdash t : B \mid \Delta}{\Gamma \vdash \lambda x.t : A \to B \mid \Delta}\ (\vdash \to) \qquad \frac{\Gamma \vdash t : A \mid \Delta \qquad \Gamma \mid e : B \vdash \Delta}{\Gamma \mid t \cdot e : A \to B \vdash \Delta}\ (\to \vdash)$$

$$\boxed{\frac{\Gamma \vdash t : A \mid \Delta \qquad \Gamma \vdash u : B \mid \Delta}{\Gamma \vdash (t, u) : A \otimes B \mid \Delta}\ (\vdash \otimes)} \qquad \boxed{\frac{c : (\Gamma, x : A, y : B \vdash \Delta)}{\Gamma \mid \tilde{\mu}(x, y).c : A \otimes B \vdash \Delta}\ (\otimes \vdash)}$$

$$\boxed{\frac{\Gamma \mid e : A \vdash \Delta}{\Gamma \vdash [e] : {\sim}A \mid \Delta}\ (\vdash {\sim})} \qquad \boxed{\frac{c : (\Gamma \vdash \alpha : A, \Delta)}{\Gamma \mid \mu[\alpha].c : {\sim}A \vdash \Delta}\ ({\sim} \vdash)}$$

$$\frac{\Gamma \vdash t_{\ominus} : N \mid \Delta}{\Gamma \vdash t_{\ominus} : \forall x\, N \mid \Delta}\ (\vdash \forall^1)^{*} \qquad \frac{\Gamma \mid e_{\ominus} : N[\mathrm{t}/\mathrm{x}] \vdash \Delta}{\Gamma \mid e_{\ominus} : \forall \mathrm{x}\, N \vdash \Delta}\ (\forall^1 \vdash)$$

$$\boxed{\frac{\Gamma \vdash t_{+} : P[\mathrm{t}/\mathrm{x}] \mid \Delta}{\Gamma \vdash t_{+} : \exists x\, P \mid \Delta}\ (\exists_i)} \qquad \boxed{\frac{\Gamma \mid e_{+} : P \vdash \Delta}{\Gamma \mid e_{+} : \exists x\, P \vdash \Delta}\ (\exists_e)^{*}}$$

$$\boxed{\frac{c : (\Gamma \vdash \Delta)}{\Gamma \vdash \mu \hat{\mathrm{tp}}.c : \bot \mid \Delta}\ (\vdash \bot)} \qquad \frac{}{\Gamma \mid \hat{\mathrm{tp}} : \bot \vdash \Delta}\ (\bot \vdash)$$

*: $\mathrm{x} \notin \mathbf{fv}(\Gamma, \Delta)$.

(d) Group Logic.

Figure IV.5: $\mathbf{L}_{\mathbf{pol}, \hat{\mathrm{tp}}^{\ominus}}$: typing rules

The difference is that in $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^{+}}$ the elements of σ are negative terms, that is to say positive contexts for $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^{\ominus}}$, while in $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^{\ominus}}$ they are negative stacks $\pi_{\ominus}$.

Similarly to the calculus $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^{+}}$, reduction is possible only when c is of the form $\langle t_{\ominus} \| \widehat{\mathrm{tp}} \rangle$:

$$\mu\widehat{\mathrm{tp}}.\langle t_{\ominus} \| \widehat{\mathrm{tp}} \rangle \rhd_{\mathrm{R}_p} t_{\ominus}$$

This rule holds even if $\widehat{\mathrm{tp}}$ appears in $t_{\ominus}$. In other words, the context $\widehat{\mathrm{tp}}$ corresponds to the operation that extracts the head of the list:

$$\langle t_{\ominus} \| \widehat{\mathrm{tp}} \rangle [\pi^1_{\ominus}, \pi^2_{\ominus}, \dots, \pi^n_{\ominus}] \to_{\mathrm{R}_p} \langle t_{\ominus} \| \pi^1_{\ominus} \rangle [\pi^2_{\ominus}, \dots, \pi^n_{\ominus}]$$

Thus there exists a term ℓ and, for all positive context e_+ a term j_{e_+}, that have the following reduction rules:

$$\langle \ell \| t_{\ominus} \cdot e_+ \rangle [\pi_{\ominus}, \sigma] \to^{+}_{\mathrm{R}_p} \langle t_{\ominus} \| \mathrm{j}_{e_+} \cdot \pi_{\ominus} \rangle [\sigma]$$
$$\langle \mathrm{j}_{e_+} \| \pi \rangle [\sigma] \to^{+}_{\mathrm{R}_p} \langle [\pi] \| e_+ \rangle [\sigma]$$

Indeed, we once again solve the desired reductions:

$$\boxed{\begin{array}{l} \ell \stackrel{\mathrm{def}}{=} \lambda x^{\ominus}.\mu\alpha^{+}.\langle x^{\ominus} \| \mathrm{j}_{\alpha^+} \cdot \widehat{\mathrm{tp}} \rangle \\ \mathrm{j}_{e_+} \stackrel{\mathrm{def}}{=} \mu\alpha.\langle [\alpha] \| e_+ \rangle \end{array}}$$

Accessing stacks

The calculus $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^{\ominus}}$ introduces the binder $\tilde{\mu}[\alpha].c$ which is responsible for accessing the stacks. We extend it to a λ abstraction as follows:

$$\lambda[\alpha].t \stackrel{\mathrm{def}}{=} \lambda x.\mu\beta.\langle x \| \tilde{\mu}[\alpha].\langle t \| \beta \rangle \rangle$$

Therefore we can define the following terms, by solving the rules of reduction of the $\lambda\ell$ calculus:

$$\begin{aligned}
&\mathsf{send} \overset{\text{def}}{=} \lambda[\alpha^{\varepsilon}].\lambda x^{\varepsilon}.\mu\widehat{\mathrm{tp}}.\langle x^{\varepsilon} \| \alpha^{\varepsilon}\rangle \\
&D_{\to} = \lambda[x{\cdot}\gamma].(x, [\gamma]) \\
&\qquad \overset{\text{def}}{=} \lambda[\alpha^{\ominus}].\mu\beta^{+}.\langle \lambda x.\mu\gamma.\langle (x, [\gamma]) \| \beta^{+}\rangle \| \alpha^{\ominus}\rangle \\
&D_{\bot} \overset{\text{def}}{=} \lambda[\alpha^{\ominus}].\lambda x.\mu\beta.\langle \mu\widehat{\mathrm{tp}}.\langle x \| \beta\rangle \| \alpha^{\ominus}\rangle \\
&D_{\forall} \overset{\text{def}}{=} \lambda[\alpha].[\alpha]
\end{aligned}$$

Translating $\lambda\ell$ in $\mathbf{L}_{\mathbf{pol},\widehat{\mathrm{tp}}^{\ominus}}$

In addition to the above definitions, stack constants of the $\lambda\ell$ calculus are interpreted by free co-variables of $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^{\ominus}}$.[4] The definitions, summarised in Figure IV.6 on the facing page, induce a translation of $\lambda\ell$ into $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^{\ominus}}$ defined by induction. We identify elements of $\lambda\ell$ to their image in $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^{\ominus}}$.

Remark IV.10. Quasi-proof terms are mapped to terms with no free co-variables.

From what we have seen, we easily conclude:

Proposition IV.11 (Simulation)**.** *If $m \succ_p m'$ for two machines m and m' of the calculus $\lambda\ell$, then $m \to_{\mathrm{R}_p}^{+} m'$ in the calculus $\boldsymbol{L}_{pol,\widehat{\mathrm{tp}}^{\ominus}}$.*

Definition IV.12. In $\lambda\ell$, we define the compatible equivalence relation $\approx_p$ between quasi-proof terms with:

$$t \approx_p u \overset{\text{def}}{\iff} t \simeq_{\mathrm{RE}_p} u$$

The equivalence is compatible with observational equivalence in the following sense:

Proposition IV.13. *Let $n \in \mathbb{N}$. If for all π and all σ of length $\geq n$ one*

[4] Interpreting such stack constants as open variables goes back to Hofmann and Streicher [HS02].

$$
\begin{array}{lcl}
(t_{\ominus}\, u)_{\varepsilon} & \stackrel{\text{def}}{=} & \mu\alpha^{\varepsilon}.\langle t_{\ominus} \| u \cdot \alpha^{\varepsilon}\rangle \\
\text{let } x^{+} \text{ be } t_{+} \text{ in } u_{\varepsilon} & \stackrel{\text{def}}{=} & \mu\alpha^{\varepsilon}.\langle t_{+} \| \tilde{\mu} x^{+}.\langle u_{\varepsilon} \| \alpha^{\varepsilon}\rangle\rangle \\
\text{let } (x,y) \text{ be } t_{+} \text{ in } u_{\varepsilon} & \stackrel{\text{def}}{=} & \mu\alpha^{\varepsilon}.\langle t_{+} \| \tilde{\mu}(x,y).\langle u_{\varepsilon} \| \alpha^{\varepsilon}\rangle\rangle \\
\text{send} & \stackrel{\text{def}}{=} & \lambda[\alpha^{\varepsilon}].\lambda x^{\varepsilon}.\mu\widehat{\mathrm{tp}}.\langle x^{\varepsilon} \| \alpha^{\varepsilon}\rangle \\
\ell & \stackrel{\text{def}}{=} & \lambda x^{\ominus}.\mu\alpha^{+}.\langle x^{\ominus} \| \mathsf{j}_{\alpha^{+}} \cdot \widehat{\mathrm{tp}}\rangle \\
D_{\to} & \stackrel{\text{def}}{=} & \lambda[\alpha^{\ominus}].\mu\beta^{+}.\langle \lambda x.\mu\gamma.\langle (x,[\gamma]) \| \beta^{+}\rangle \| \alpha^{\ominus}\rangle \\
D_{\bot} & \stackrel{\text{def}}{=} & \lambda[\alpha^{\ominus}].\lambda x.\mu\beta.\langle \mu\widehat{\mathrm{tp}}.\langle x \| \beta\rangle \| \alpha^{\ominus}\rangle \\
D_{\forall} & \stackrel{\text{def}}{=} & \lambda[\alpha].[\alpha]
\end{array}
$$

where $\lambda[\alpha].t \stackrel{\text{def}}{=} \lambda x.\mu\beta.\langle x \| \tilde{\mu}[\alpha].\langle t \| \beta\rangle\rangle$

(a) Quasi-proof terms

$$
\begin{array}{lcl}
\mathsf{j}_{e_{+}} & \stackrel{\text{def}}{=} & \mu\alpha.\langle [\alpha] \| e_{+}\rangle \\
\alpha^{\varepsilon} & \stackrel{\text{def}}{=} & \alpha^{\varepsilon}\ (\text{free}) \\
c[\,] & \stackrel{\text{def}}{=} & c \\
c[\pi^{1}_{\ominus}, \ldots, \pi^{n}_{\ominus}] & \stackrel{\text{def}}{=} & \langle \mu\widehat{\mathrm{tp}}.c \| \pi^{1}_{\ominus}\rangle [\pi^{2}_{\ominus}, \ldots, \pi^{n}_{\ominus}]
\end{array}
$$

(b) Machines

Figure IV.6: Definition of the constructs of λℓ in $\mathbf{L}_{\mathbf{pol},\widehat{\mathrm{tp}}^{\ominus}}$

has $\langle t \| \pi\rangle[\sigma]\ (\prec_{p} \cup \succ_{p})^{*}\ \langle u \| \pi\rangle[\sigma]$ *in the calculus λℓ, then one also has* $t \approx_{p} u$.

Proof. Indeed we have in particular in $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^{\ominus}}$, thanks to Proposition IV.11:

$$\langle t \| \alpha_{0}\rangle[\alpha_{1}, \ldots, \alpha_{n}] \simeq_{\mathrm{RE}_{p}} \langle u \| \alpha_{0}\rangle[\alpha_{1}, \ldots, \alpha_{n}]$$

Now if one has $\langle \mu\widehat{\mathrm{tp}}.c \| \alpha\rangle \simeq_{\mathrm{RE}_{p}} \langle \mu\widehat{\mathrm{tp}}.c' \| \alpha\rangle$ for some commands c and c', one

also has $c \simeq_{\mathrm{RE}_p} c'$. Indeed one has by extensionality:

$$\begin{aligned} c &\rhd^*_{\mathrm{E}_p} \langle \mu\alpha.\langle \mu\widehat{\mathrm{tp}}.c \parallel \alpha\rangle \parallel \widehat{\mathrm{tp}}\rangle \\ &\simeq_{\mathrm{RE}_p} \langle \mu\alpha.\langle \mu\widehat{\mathrm{tp}}.c' \parallel \alpha\rangle \parallel \widehat{\mathrm{tp}}\rangle \\ &\lhd^*_{\mathrm{E}_p} c' \end{aligned}$$

Thus we can conclude by induction $\langle t \parallel \alpha_0 \rangle \simeq_{\mathrm{RE}_p} \langle u \parallel \alpha_0 \rangle$, and therefore $t \simeq_{\mathrm{RE}_p} u$. ∎

Last, we have:

Proposition IV.14. *The reduction $\to_{\mathrm{R}_p}$ is confluent.*

Proof. The reduction $\rhd_{\mathrm{R}_p}$ is left-linear and has no critical pairs. ∎

IV.4.2 Coherence

We define a CPS translation for the calculus $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^{\ominus}}$ that simulates the reduction and preserves equivalences. We deduce that $\simeq_{\mathrm{RE}_p}$ and $\approx_p$ do not identify all the terms.

Target of the CPS translation

The target of the CPS translation is the call-by-name λ calculus with pairs considered by Herbelin and Ghilezan [HG08].

$$M, N ::= x \mid \lambda x.M \mid M\,N \mid (M, N) \mid \mathsf{let}\ x\ \mathsf{be}\ M\ \mathsf{in}\ N \mid \mathsf{let}\ (x, y)\ \mathsf{be}\ M\ \mathsf{in}\ N$$

The calculus comes with the following reductions and expansions:

$$
\begin{array}{ll}
(\lambda x.M)\, N & \triangleright_{\mathrm{R}_\lambda} \; M[N/x] \\
\mathsf{let}\, x\, \mathsf{be}\, M\, \mathsf{in}\, N & \triangleright_{\mathrm{R}_\lambda} \; N[M/x] \\
\mathsf{let}\, (x,y)\, \mathsf{be}\, (M,M')\, \mathsf{in}\, N & \triangleright_{\mathrm{R}_\lambda} \; N[M/x, M'/y] \\
F[\mathsf{let}\, (x,y)\, \mathsf{be}\, M\, \mathsf{in}\, N] & \triangleright_{\mathrm{R}_\lambda} \; \mathsf{let}\, (x,y)\, \mathsf{be}\, M\, \mathsf{in}\, F[N] \\
\text{où } F[\,] ::= [\,]\, M \mid \mathsf{let}\, x\, \mathsf{be}\, [\,]\, \mathsf{in}\, M \mid \mathsf{let}\, (x,y)\, \mathsf{be}\, [\,]\, \mathsf{in}\, M & \\
M & \triangleright_{\mathrm{E}_\lambda} \; \lambda x.M\, x \\
M & \triangleright_{\mathrm{E}_\lambda} \; \mathsf{let}\, (x,y)\, \mathsf{be}\, M\, \mathsf{in}\, (x,y)
\end{array}
$$

The above calculus is less natural a target than the $\lambda_{\mathrm{v}}^{\times}$ calculus used in the previous chapter, in the sense that on the one hand we rely on strict evaluation for pairs via the "$\mathsf{let}\, (x,y)\, \mathsf{in}\ \ldots$" binder in order to obtain a simulation result, but on the other hand, our isomorphisms are going to require the above η expansions[5], which makes it a call-by-name calculus.

This calculus does not identify all terms.

Proposition IV.15. *We have $x \neq_{\mathrm{RE}_\lambda} y$ for any pair of distinct variables x and y.*

Proof. We consider the more standard λ calculus with surjective pairs which is the λ calculus extended with pairs (M,N), projections $\mathsf{fst}(M)$ and $\mathsf{snd}(M)$ and rules $\mathsf{fst}(M,N) \triangleright_{\lambda_{\mathrm{SP}}} M$, $\mathsf{snd}(M,N) \triangleright_{\lambda_{\mathrm{SP}}} N$ and $M \triangleright_{\lambda_{\mathrm{SP}}} (\mathsf{fst}(M), \mathsf{snd}(M))$. It is easy to see that the following definition:

$$\mathsf{let}\, (x,y)\, \mathsf{be}\, M\, \mathsf{in}\, N \stackrel{\mathrm{def}}{=} (\lambda z.(\lambda x.\lambda y.N)\, \mathsf{fst}(z)\, \mathsf{snd}(z))\, M$$

[5] The goal is to have the equation $c \triangleright_{\mathrm{E}_n} \langle\, \mu\widehat{\mathrm{tp}}.c \parallel \widehat{\mathrm{tp}} \,\rangle$ for negative $\widehat{\mathrm{tp}}$, the absence of which was noticed in Section III.2.7 when the target language was the $\lambda_{\mathrm{v}}^{\times}$ calculus. For this purpose, the η expansion $M \triangleright_{\mathrm{E}_\lambda} \lambda x.M\, x$ is used at places where M is linear. This leaves us hope of giving up the call-by-name λ calculus in favour of a refined $\lambda_{\mathrm{v}}^{\times}$ calculus where this linearity is taken into account. This would provide a unification with Chapter III, and also allow us to extend the $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^{\ominus}}$ calculus with additive connectives.

induces a translation from Herbelin-Ghilezan's λ calculus with pairs into the λ calculus with surjective pairing that preserves equivalences. We conclude $x \neq_{\mathrm{RE}_\lambda} y$, since we have $x \neq_{\lambda_{\mathrm{SP}}} y$ (see for instance Støvring [Stø06]). ∎

The translation

We define in Figure IV.7 on the next page the CPS translation of the calculus $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^{\ominus}}$ into Herbelin-Ghilezan's λ calculus with pairs. We assume that $x^+, x^\ominus, \alpha^+, \alpha^\ominus, k \ldots$ represent distinct subsets of variables of the λ calculus with pairs.

A value or a stack translates into a λ-term. A negative term or a positive context translates into a linear function from λ-terms into λ-terms (notation φ, ψ). By linear we mean the naive (syntactic) notion based on counting occurrences, which simply ensures that the translation hides no duplication or erasure of the source terms.

A positive term or a negative context translates into a linear functional from the previous functions φ, ψ into λ-terms. Then the command translates into the application of the linear functional to the linear function. In this aspect, the CPS translation is similar to the definition of distributions in mathematics, with the analogy between t_+ or $e_\ominus$ and distributions, and between $t_\ominus$ or e_+ and test functions.

This formulation with functionals is a reformulation of *one-pass CPS transforms* (see Danvy and Filinski [DF90]).[6]

Lemma IV.16. *Let (f, κ) be a pair of a value and a variable or of a stack and a co-variable, with f and κ of the same polarity. We write $[\![f]\!] = \mathrm{VAL}_V[\![f]\!]$ or $[\![f]\!] = \mathrm{VAL}_\pi[\![f]\!]$ depending on the case. For any command c, any term or context g, any context e, any value V, any stack*

[6] Few such translations would require functionals of higher order than ours, an exception being Danvy and Nielsen's [DN05].

$$\begin{array}{rcl}\mathrm{CPS}[\![\langle t_+ \| e_+ \rangle]\!] & \stackrel{\text{def}}{=} & \mathrm{CPS}_+[\![t_+]\!](\mathrm{CPS}_\ominus[\![e_+]\!]) \\ \mathrm{CPS}[\![\langle t_\ominus \| e_\ominus \rangle]\!] & \stackrel{\text{def}}{=} & \mathrm{CPS}_+[\![e_\ominus]\!](\mathrm{CPS}_\ominus[\![t_\ominus]\!])\end{array}$$

(a) $\mathrm{CPS}[\![\cdot]\!] : c \to M$

$$\begin{array}{rcl}\mathrm{CPS}_+[\![V_+]\!](\varphi) & \stackrel{\text{def}}{=} & \varphi(\mathrm{VAL}_{\mathrm{V}}[\![V_+]\!]) \\ \mathrm{CPS}_+[\![\mu\alpha^+.c]\!](\varphi) & \stackrel{\text{def}}{=} & \mathsf{let}\ \alpha^+\ \mathsf{be}\ \lambda k.\varphi(k)\ \mathsf{in}\ \mathrm{CPS}[\![c]\!] \\ \mathrm{CPS}_+[\![(t_+,u)]\!](\varphi)^\dagger & \stackrel{\text{def}}{=} & \mathrm{CPS}_+[\![t_+]\!](\psi)\ \text{where}\ \psi : M \mapsto \mathsf{let}\ x^+\ \mathsf{be}\ M\ \mathsf{in}\ \mathrm{CPS}_+[\![(x^+,u)]\!](\varphi) \\ \mathrm{CPS}_+[\![(V,t_+)]\!](\varphi)^\dagger & \stackrel{\text{def}}{=} & \mathrm{CPS}_+[\![t_+]\!](\psi)\ \text{where}\ \psi : M \mapsto \mathsf{let}\ x^+\ \mathsf{be}\ M\ \mathsf{in}\ \mathrm{CPS}_+[\![(V,x^+)]\!](\varphi) \\ \mathrm{CPS}_+[\![[e_\ominus]]\!](\varphi)^\ddagger & \stackrel{\text{def}}{=} & \mathrm{CPS}_+[\![e_\ominus]\!](\psi)\ \text{where}\ \psi : M \mapsto \mathsf{let}\ \alpha^\ominus\ \mathsf{be}\ M\ \mathsf{in}\ \mathrm{CPS}_+[\![[\alpha^\ominus]]\!](\varphi)\end{array}$$

$$\begin{array}{rcl}\mathrm{CPS}_+[\![\pi^\ominus]\!](\varphi) & \stackrel{\text{def}}{=} & \varphi(\mathrm{VAL}_\pi[\![\pi^\ominus]\!]) \\ \mathrm{CPS}_+[\![\tilde{\mu}x^\ominus.c]\!](\varphi) & \stackrel{\text{def}}{=} & \mathsf{let}\ x^\ominus\ \mathsf{be}\ \lambda k.\varphi(k)\ \mathsf{in}\ \mathrm{CPS}[\![c]\!] \\ \mathrm{CPS}_+[\![t_+ \cdot e]\!](\varphi)^\dagger & \stackrel{\text{def}}{=} & \mathrm{CPS}_+[\![t_+]\!](\psi)\ \text{où}\ \psi : M \mapsto \mathsf{let}\ x^+\ \mathsf{be}\ M\ \mathsf{in}\ \mathrm{CPS}_+[\![x^+ \cdot e]\!](\varphi) \\ \mathrm{CPS}_+[\![V \cdot e_\ominus]\!](\varphi)^\ddagger & \stackrel{\text{def}}{=} & \mathrm{CPS}_+[\![e^\ominus]\!](\psi)\ \text{où}\ \psi : M \mapsto \mathsf{let}\ \alpha^\ominus\ \mathsf{be}\ M\ \mathsf{in}\ \mathrm{CPS}_+[\![V \cdot \alpha^\ominus]\!](\varphi) \\ \mathrm{CPS}_+[\![\widehat{\mathrm{tp}}]\!](\varphi) & \stackrel{\text{def}}{=} & \lambda k.\varphi(k)\end{array}$$

†If t_+ is not a value
‡If $e_\ominus$ is not a stack.

(b) $\mathrm{CPS}_+[\![\cdot]\!](\varphi) : t_+ \cup e_\ominus \to M$

$$\begin{array}{rcl}\mathrm{CPS}_\ominus[\![x^\ominus]\!](M) & \stackrel{\text{def}}{=} & x^\ominus\, M \\ \mathrm{CPS}_\ominus[\![\mu\alpha^\ominus.c]\!](M) & \stackrel{\text{def}}{=} & \mathsf{let}\ \alpha^\ominus\ \mathsf{be}\ M\ \mathsf{in}\ \mathrm{CPS}[\![c]\!] \\ \mathrm{CPS}_\ominus[\![\lambda x.t]\!](M) & \stackrel{\text{def}}{=} & \mathsf{let}\ (x,k)\ \mathsf{be}\ M\ \mathsf{in}\ \mathrm{CPS}_\ominus[\![t]\!](k) \\ \mathrm{CPS}_\ominus[\![\mu\widehat{\mathrm{tp}}.c]\!](M) & \stackrel{\text{def}}{=} & \mathrm{CPS}[\![c]\!]\, M \\ \mathrm{CPS}_\ominus[\![t_+]\!](M) & \stackrel{\text{def}}{=} & \mathrm{CPS}_+[\![t_+]\!](\varphi)\ \text{où}\ \varphi : N \mapsto M\, N\end{array}$$

$$\begin{array}{rcl}\mathrm{CPS}_\ominus[\![\alpha^+]\!](M) & \stackrel{\text{def}}{=} & \alpha^+\, M \\ \mathrm{CPS}_\ominus[\![\tilde{\mu}x^+.c]\!](M) & \stackrel{\text{def}}{=} & \mathsf{let}\ x^+\ \mathsf{be}\ M\ \mathsf{in}\ \mathrm{CPS}[\![c]\!] \\ \mathrm{CPS}_\ominus[\![\tilde{\mu}(x,y).c]\!](M) & \stackrel{\text{def}}{=} & \mathsf{let}\ (x,y)\ \mathsf{be}\ M\ \mathsf{in}\ \mathrm{CPS}[\![c]\!] \\ \mathrm{CPS}_\ominus[\![\tilde{\mu}[\alpha].c]\!](M) & \stackrel{\text{def}}{=} & \mathsf{let}\ \alpha\ \mathsf{be}\ M\ \mathsf{in}\ \mathrm{CPS}[\![c]\!] \\ \mathrm{CPS}_\ominus[\![e_\ominus]\!](M) & \stackrel{\text{def}}{=} & \mathrm{CPS}_+[\![e_\ominus]\!](\varphi)\ \text{où}\ \varphi : N \mapsto M\, N\end{array}$$

(c) $\mathrm{CPS}_\ominus[\![\cdot]\!](M) : t \cup e \to M$

$$\begin{array}{rcl}\mathrm{VAL}_{\mathrm{V}}[\![x^+]\!] & \stackrel{\text{def}}{=} & x^+ \\ \mathrm{VAL}_{\mathrm{V}}[\![(V,V')]\!] & \stackrel{\text{def}}{=} & (\mathrm{VAL}_{\mathrm{V}}[\![V]\!], \mathrm{VAL}_{\mathrm{V}}[\![V']\!]) \\ \mathrm{VAL}_{\mathrm{V}}[\![[\pi]]\!] & \stackrel{\text{def}}{=} & \mathrm{VAL}_\pi[\![\pi]\!] \\ \mathrm{VAL}_{\mathrm{V}}[\![t_\ominus]\!] & \stackrel{\text{def}}{=} & \lambda k.\mathrm{CPS}_\ominus[\![t_\ominus]\!](k)\end{array}$$

(d) $\mathrm{VAL}_{\mathrm{V}}[\![\cdot]\!] : V \to M$

$$\begin{array}{rcl}\mathrm{VAL}_\pi[\![\alpha^\ominus]\!] & \stackrel{\text{def}}{=} & \alpha^\ominus \\ \mathrm{VAL}_\pi[\![V \cdot \pi]\!] & \stackrel{\text{def}}{=} & (\mathrm{VAL}_{\mathrm{V}}[\![V]\!], \mathrm{VAL}_\pi[\![\pi]\!]) \\ \mathrm{VAL}_\pi[\![e_+]\!] & \stackrel{\text{def}}{=} & \lambda k.\mathrm{CPS}_\ominus[\![e_+]\!](k)\end{array}$$

(e) $\mathrm{VAL}_\pi[\![\cdot]\!] : \pi \to M$

Figure IV.7: Translation of $\mathbf{L}_{\mathbf{pol},\widehat{\mathrm{tp}}^{\ominus}}$ into the λ calculus with pairs

π, and any λ-term u, one has:

$$\begin{aligned}
\textsc{cps}[\![c]\!][[\![f]\!]/\kappa] &\to^*_{\mathrm{R}_\lambda} \textsc{cps}[\![c[f/\kappa]]\!] \\
\textsc{cps}_+[\![g]\!](\varphi)[[\![f]\!]/\kappa] &\to^*_{\mathrm{R}_\lambda} \textsc{cps}_+[\![g[f/\kappa]]\!](\psi) \\
&\qquad o\grave{u}\ \psi : M \mapsto \varphi(M)[[\![f]\!]/\kappa] \\
\textsc{cps}_\ominus[\![g]\!](M)[[\![f]\!]/\kappa] &\to^*_{\mathrm{R}_\lambda} \textsc{cps}_\ominus[\![g[f/\kappa]]\!](M[[\![f]\!]/\kappa]) \\
\textsc{val}_V[\![V]\!][[\![f]\!]/\kappa] &\to^*_{\mathrm{R}_\lambda} \textsc{val}_V[\![V[f/\kappa]]\!] \\
\textsc{val}_\pi[\![\pi]\!][[\![f]\!]/\kappa] &\to^*_{\mathrm{R}_\lambda} \textsc{val}_\pi[\![\pi[f/\kappa]]\!]
\end{aligned}$$

Proof. By induction on the definition of the translation. *Case* $\textsc{val}_V[\![x^+]\!]$. If $x^+ \neq \kappa$ one has $\textsc{val}_V[\![x^+]\!][[\![f]\!]/\kappa] = x^+[f/\kappa]$; otherwise one has:

$$\textsc{val}_V[\![x^+]\!][[\![f]\!]/x^+] = x^+[[\![f]\!]/x^+] = [\![f]\!]$$

One concludes with $f = x^+[f/x^+]$ using the hypothesis that f and x^+ have the same polarity. *Case* $\textsc{val}_\pi[\![\alpha^\ominus]\!]$. Same reasoning. *Case* $\textsc{cps}_\ominus[\![x^\ominus]\!](M)$. By hypothesis, f is a term of the same polarity as $x^\ominus$. Thus we have:

$$[\![f]\!] = \textsc{val}_V[\![f]\!] = \lambda k.\textsc{cps}_\ominus[\![f]\!](k)$$

Hence:

$$\begin{aligned}
\textsc{cps}_\ominus[\![x^\ominus]\!](M)[[\![f]\!]/x^\ominus] = (\underline{\lambda k.}\textsc{cps}_\ominus[\![f]\!](k))\, M[[\![f]\!]/x^\ominus] \\
\rhd_{\mathrm{R}_\lambda} \textsc{cps}_\ominus[\![f]\!](M[[\![f]\!]/x^\ominus])
\end{aligned}$$

Case $\textsc{cps}_\ominus[\![\alpha^+]\!](M)$. Same reasoning. The remaining cases are easily deduced from induction hypothesis. ∎

Lemma IV.17.

1. *For any c, c' such that $c \rhd_{\mathrm{R}_p} c'$, one has:*

$$\textsc{cps}[\![c]\!] \to^+_{\mathrm{R}_\lambda} \textsc{cps}[\![c']\!]\,;$$

2. *For any $t_\ominus$, one has:*

$$\textsc{cps}_\ominus[\![\mu\hat{\mathrm{tp}}.\langle t_\ominus \| \hat{\mathrm{tp}}\rangle]\!](M) \rhd_{\mathrm{R}_\lambda} \textsc{cps}_\ominus[\![t_\ominus]\!](M)\,;$$

3. *For any terms or contexts f, f' such that $f \rhd_{\mathrm{E}_p} f'$, one has:*

$$\mathrm{CPS}_\ominus[\![f]\!](M) \simeq_{\mathrm{RE}_\lambda} \mathrm{CPS}_\ominus[\![f']\!](M)\,;$$

4. *For any positive terms or negative contexts f, f' such that $f \rhd_{\mathrm{E}_p} f'$, one has:*

$$\mathrm{CPS}_+[\![f]\!](\varphi) \simeq_{\mathrm{RE}_\lambda} \mathrm{CPS}_+[\![f']\!](\varphi)\,;$$

5. *For any command c, one has:*

$$\mathrm{CPS}[\![c]\!] \rhd_{\mathrm{E}_\lambda} \mathrm{CPS}[\![\langle \mu\widehat{\mathrm{tp}}.c \,\|\, \widehat{\mathrm{tp}}\rangle]\!]\,.$$

Proof. The proof relies on Lemma IV.16. (1.) *Case* $\langle V \,\|\, \tilde{\mu}x.c\rangle \rhd_{\mathrm{R}_p} c[V/x]$. One has:

$$\begin{aligned}\mathrm{CPS}[\![\langle V \,\|\, \tilde{\mu}x.c\rangle]\!] &= \mathsf{let}\ x\ \mathsf{be}\ \mathrm{VAL_V}[\![V]\!]\ \mathsf{in}\ \mathrm{CPS}[\![c]\!] \\ &\rhd_{\mathrm{R}_\lambda} \mathrm{CPS}[\![c]\!][\mathrm{VAL_V}[\![V]\!]/x] \\ &\to^*_{\mathrm{R}_\lambda} \mathrm{CPS}[\![c[V/x]]\!]\end{aligned}$$

Case $\langle \mu\alpha.c \,\|\, \pi\rangle \rhd_{\mathrm{R}_p} c[\pi/\alpha]$ *and* $\langle [\pi] \,\|\, \tilde{\mu}[\alpha].c\rangle \rhd_{\mathrm{R}_p} c[\pi/\alpha]$. Same reasoning.

Case $\langle \lambda x.t \,\|\, V\cdot\pi\rangle \rhd_{\mathrm{R}_p} \langle t[V/x] \,\|\, \pi\rangle$ *with V of the same polarity as x and t of the same polarity as π.* One has:

$$\begin{aligned}\mathrm{CPS}[\![\langle \lambda x.t \,\|\, V\cdot\pi\rangle]\!] &= \mathsf{let}\ (x,k)\ \mathsf{be}\ (\mathrm{VAL_V}[\![V]\!], \mathrm{VAL}_\pi[\![\pi]\!])\ \mathsf{in}\ \mathrm{CPS}_\ominus[\![t]\!](k) \\ &\rhd_{\mathrm{R}_\lambda} \mathrm{CPS}_\ominus[\![t]\!](k)[\mathrm{VAL_V}[\![V]\!]/x, \mathrm{VAL}_\pi[\![\pi]\!]/k] \\ &\to^*_{\mathrm{R}_\lambda} \mathrm{CPS}_\ominus[\![t[V/x]]\!](k)[\mathrm{VAL}_\pi[\![\pi]\!]/k] \\ &= \mathrm{CPS}_\ominus[\![t[V/x]]\!](\mathrm{VAL}_\pi[\![\pi]\!])\end{aligned}$$

We conclude depending on the polarity of $t[V/x]$:

$$\begin{aligned}\mathrm{CPS}_\ominus[\![t_\ominus]\!](\mathrm{VAL}_\pi[\![\pi]\!]) &= \mathrm{CPS}[\![\langle t_\ominus \,\|\, \pi\rangle]\!] \\ \mathrm{CPS}_\ominus[\![t_+]\!](\mathrm{VAL}_\pi[\![\pi]\!]) &= \mathrm{CPS}_+[\![t_+]\!](M \mapsto \lambda k.\mathrm{CPS}_\ominus[\![\pi]\!](k)\,M) \\ &\to_{\mathrm{R}_\lambda} \mathrm{CPS}_+[\![t_+]\!](\mathrm{CPS}_\ominus[\![\pi]\!](M)) \\ &= \mathrm{CPS}[\![\langle t_+ \,\|\, \pi\rangle]\!]\end{aligned}$$

Case $\langle (V,W) \,\|\, \tilde{\mu}(x,y).c\rangle \triangleright_{\mathrm{R}_p} c[V/x, W/y]$ *with* V *of the same polarity as* x *and* W *of the same polarity as* y. Same reasoning.
Case $(\varsigma_{\to_1})$: $\langle\, u_\ominus \parallel t_+ \cdot e \,\rangle \triangleright_{\mathrm{R}_p} \langle\, t_+ \parallel \tilde{\mu}x.\langle u_\ominus \| x\cdot e\rangle \,\rangle$ *with* t_+ *not a value.* By definition, one has $\mathrm{CPS}[\![\langle\, u_\ominus \parallel t_+\cdot e \,\rangle]\!] = \mathrm{CPS}_+[\![t_+]\!](\varphi)$ where $\varphi : M \mapsto \mathsf{let}\, x^+ \,\mathsf{be}\, M \,\mathsf{in}\, \mathrm{CPS}[\![\langle u_\ominus \| x\cdot e\rangle]\!]$. Therefore one has $\varphi = \mathrm{CPS}_\ominus[\![\tilde{\mu}x.\langle u_\ominus \| x^+\cdot e\rangle]\!]$ and furthermore $\mathrm{CPS}[\![\langle u_\ominus \| t_+\cdot e\rangle]\!] = \mathrm{CPS}[\![\langle t_+ \| \tilde{\mu}x.\langle u_\ominus \| x\cdot e\rangle\rangle]\!]$.
Other cases (ς). Same reasoning.

(2.) One has:

$$\mathrm{CPS}_\ominus[\![\mu\hat{\mathfrak{p}}.\langle t_\ominus \| \hat{\mathfrak{p}}\rangle]\!](M) = (\lambda k.\mathrm{CPS}_\ominus[\![t_\ominus]\!](k))\, M \triangleright_{\mathrm{R}_\lambda} \mathrm{CPS}_\ominus[\![t_\ominus]\!](M)$$

(3.) *Case* $t_\ominus \triangleright_{\mathrm{E}_p} \mu\alpha^\ominus.\langle t_\ominus \| \alpha^\ominus\rangle$. One has:

$$\mathrm{CPS}_\ominus[\![\mu\alpha^\ominus.\langle t_\ominus \| \alpha^\ominus\rangle]\!](M) = \mathsf{let}\, \alpha^\ominus \,\mathsf{be}\, M \,\mathsf{in}\, \mathrm{CPS}_\ominus[\![t_\ominus]\!](\alpha^\ominus) \triangleright_{\mathrm{R}_\lambda} \mathrm{CPS}_\ominus[\![t_\ominus]\!](M)$$

Case $e_+ \triangleright_{\mathrm{E}_p} \tilde{\mu}x^+.\langle x^+ \| e_+\rangle$ *and* $t_\ominus \triangleright_{\mathrm{E}_p} \tilde{\mu}[\alpha^\ominus].\langle [\alpha^\ominus] \| e_+\rangle$. Same reasoning.
Case $t_\ominus \triangleright_{\mathrm{E}_p} \lambda x.\mu\alpha.\langle t_\ominus \| x\cdot\alpha\rangle$. One has depending on the polarity of α:

$$\begin{aligned}
&\mathrm{CPS}_\ominus[\![\lambda x.\mu\alpha^\ominus.\langle t_\ominus \| x\cdot\alpha^\ominus\rangle]\!](M)\\
&= \mathsf{let}\,(x,k)\,\mathsf{be}\, M \,\mathsf{in}\, \mathsf{let}\, \alpha^\ominus \,\mathsf{be}\, k \,\mathsf{in}\, \mathrm{CPS}_\ominus[\![t_\ominus]\!]((x, \lambda k'.(\alpha^\ominus\, k')))\\
&\mathrm{CPS}_\ominus[\![\lambda x.\mu\alpha^+.\langle t_\ominus \| x\cdot\alpha^+\rangle]\!](M)\\
&= \mathsf{let}\,(x,k)\,\mathsf{be}\, M \,\mathsf{in}\, \mathsf{let}\, \alpha^+ \,\mathsf{be}\, \lambda k'.(k\, k') \,\mathsf{in}\, \mathrm{CPS}_\ominus[\![t_\ominus]\!]((x,\alpha^+))
\end{aligned}$$

In both cases:

$$\begin{aligned}
&\mathrm{CPS}_\ominus[\![\lambda x.\mu\alpha.\langle t_\ominus \| x\cdot\alpha\rangle]\!](M)\\
&\simeq_{\mathrm{R}_\lambda} \mathsf{let}\,(x,k)\,\mathsf{be}\, M \,\mathsf{in}\, \mathsf{let}\, y \,\mathsf{be}\, (x, \underline{\lambda k'.(k\, k')}) \,\mathsf{in}\, \mathrm{CPS}_\ominus[\![t_\ominus]\!](y)\\
&\leftarrow_{\mathrm{E}_\lambda} \underline{\mathsf{let}\,(x,k)\,\mathsf{be}\, M \,\mathsf{in}\, \mathsf{let}\, y \,\mathsf{be}\, (x,k) \,\mathsf{in}}\, \mathrm{CPS}_\ominus[\![t_\ominus]\!](y)\\
&\lhd_{\mathrm{R}_\lambda} \mathsf{let}\, y \,\mathsf{be}\, (\underline{\mathsf{let}\,(x,k)\,\mathsf{be}\, M \,\mathsf{in}\,(x,k)}) \,\mathsf{in}\, \mathrm{CPS}_\ominus[\![t_\ominus]\!](y)\\
&\leftarrow_{\mathrm{E}_\lambda} \underline{\mathsf{let}\, y \,\mathsf{be}\, M \,\mathsf{in}}\, \mathrm{CPS}_\ominus[\![t_\ominus]\!](y)\\
&\triangleright_{\mathrm{R}_\lambda} \mathrm{CPS}_\ominus[\![t_\ominus]\!](M)
\end{aligned}$$

Case $\pi_{\ominus} \rhd_{\mathrm{E}_{\lambda}} \tilde{\mu}(x,y).\langle (x,y) \parallel \pi_{\ominus} \rangle$. Same reasoning.

Case $t_{+} \rhd_{\mathrm{E}_{p}} \mu\alpha^{+}.\langle t_{+} \parallel \alpha^{+} \rangle$ *and* $e_{\ominus} \rhd_{\mathrm{E}_{p}} \tilde{\mu}x^{\ominus}.\langle x^{\ominus} \parallel e_{\ominus} \rangle$. Same reasoning as (4.) below.

(4.) *Case* $t_{+} \rhd_{\mathrm{E}_{p}} \mu\alpha^{+}.\langle t_{+} \parallel \alpha^{+} \rangle$. One has:

$$\begin{aligned} \mathrm{CPS}_{+}[\![\mu\alpha^{+}.\langle t_{+} \parallel \alpha^{+} \rangle]\!](\varphi) &= \mathsf{let}\ \alpha^{+}\ \mathsf{be}\ \lambda k.\varphi(k)\ \mathsf{in}\ \mathrm{CPS}_{+}[\![t_{+}]\!](M \mapsto \alpha^{+} M) \\ &\rhd_{\mathrm{R}_{p}} \mathrm{CPS}_{+}[\![t_{+}]\!](M \mapsto \lambda k.\varphi(k)\, M) \\ &\to_{\mathrm{R}_{p}} \mathrm{CPS}_{+}[\![t_{+}]\!](\varphi) \end{aligned}$$

Case $e_{\ominus} \rhd_{\mathrm{E}_{p}} \tilde{\mu}x^{\ominus}.\langle x^{\ominus} \parallel e_{\ominus} \rangle$. Same reasoning.

(5.) We have indeed:

$$\begin{aligned} \mathrm{CPS}[\![\langle \mu\hat{\mathfrak{tp}}.c \parallel \hat{\mathfrak{tp}} \rangle]\!] &= \lambda k.\mathrm{CPS}[\![c]\!]\, k \\ &\lhd_{\mathrm{E}_{\lambda}} \mathrm{CPS}[\![c]\!] \end{aligned}$$

∎

Theorem IV.18 (Simulation)**.** *Let* c *and* c' *be two commands of the calculus* $\mathbf{L}_{pol,\hat{\mathfrak{tp}}^{\ominus}}$.

1. *If* $c \to_{\mathrm{R}_{p}} c'$ *then* $\mathrm{CPS}[\![c]\!] \to^{+}_{\mathrm{R}_{\lambda}} \mathrm{CPS}[\![c']\!]$.
2. *If* $c \simeq_{\mathrm{RE}_{p}} c'$ *then* $\mathrm{CPS}[\![c]\!] \simeq_{\mathrm{RE}_{\lambda}} \mathrm{CPS}[\![c']\!]$.

Proof. By immediate induction on the definitions of $\to_{\mathrm{R}_{p}}$ and $\simeq_{\mathrm{RE}_{p}}$, using Lemma IV.17. ∎

A fortiori, the translation simulates the reduction of machines of $\lambda\ell$. We deduce the coherence of calculi $\lambda\ell$ and $\mathbf{L}_{\mathrm{pol},\hat{\mathfrak{tp}}^{\ominus}}$:

Corollary IV.19 (Coherence)**.** *If* x *and* y *are two distinct polarised variables, then* $x \neq_{\mathrm{RE}_{p}} y$*, and therefore also* $x \neq_{p} y$.

Proof. Let x and y be two distinct variables of the calculus $\mathbf{L}_{\mathrm{pol},\hat{\mathfrak{tp}}^{\ominus}}$ that we can assume to be of the same polarity. In order to show $x \neq_{\mathrm{RE}_{p}} y$, it is sufficient by the rule $\rhd_{\mathrm{E}_{p}}$ to show for some α of the proper polarity: $\langle\, x \parallel \alpha \,\rangle \neq_{\mathrm{RE}_{p}} \langle\, y \parallel \alpha \,\rangle$. Let such an a α. If $z \in \{x, y\}$ is positive then one has $\mathrm{CPS}[\![\, \langle\, z \parallel \alpha \,\rangle \,]\!] = \alpha\, z$; otherwise one has $\mathrm{CPS}[\![\, \langle\, z \parallel \alpha \,\rangle \,]\!] = z\, \alpha$.

Yet according to Proposition IV.15, one has $x \neq_{\mathrm{RE}_\lambda} y$. Thus we also have $\mathrm{CPS}[\![\langle x \,\|\, \alpha\rangle]\!] \neq_{\mathrm{RE}_\lambda} \mathrm{CPS}[\![\langle y \,\|\, \alpha\rangle]\!]$; and therefore $\langle x \,\|\, \alpha\rangle \neq_{\mathrm{RE}_p} \langle y \,\|\, \alpha\rangle$ using Theorem IV.18. ∎

IV.4.3 Expressiveness of pure $\mathbf{L}_{\mathbf{pol},\widehat{\mathrm{tp}}^{\ominus}}$

Call-by-name delimited continuations / Saurin's Λμ

The calculus $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^{\ominus}}$ restricted to fully negative terms corresponds to Herbelin and Ghilezan's $\lambda\mu\widehat{\mathrm{tp}}_n$ calculus [HG08]. This is the calculus of Figure III.7 on page 179 with $\widehat{\mathrm{tp}}^{\ominus}$ renamed as the $\widehat{\mathrm{tp}}$ operator of the current section, and with Herbelin and Ghilezan's equation $\langle \mu\widehat{\mathrm{tp}}.c \,\|\, \widehat{\mathrm{tp}}\rangle \simeq_{\mathrm{E}_n}$ restored.

Herbelin and Ghilezan show that the calculus $\lambda\mu\widehat{\mathrm{tp}}_n$ is in correspondence with the Λμ calculus of De Groote and Saurin [dG94, Sau05]. As Saurin shows, Λμ answers positively the question of a Böhm theorem. This property is false in the λC calculus, as David and Py showed [DP01]. Incidentally, this raises the question of separation in $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^{\ominus}}$ and $\lambda\ell$.

Shift$_0$/Reset$_0$

Let us assume that the $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^{\ominus}}$ calculus is extended with the duploid syntax from Figure II.1 on page 119.

The presence of positives gives more expressiveness compared to $\lambda\mu\widehat{\mathrm{tp}}_n$. Thus, in the $\lambda\mu\widehat{\mathrm{tp}}_n$ calculus , doing $\widehat{\mathrm{tp}}$ followed by $\mu\widehat{\mathrm{tp}}$ does nothing:

$$\mu\widehat{\mathrm{tp}}.\langle t_{\ominus} \,\|\, \widehat{\mathrm{tp}}\rangle \simeq_{\mathrm{RE}_p} t_{\ominus}$$

This explains, according to Herbelin and Ghilezan, the "*at a first glance surprising*" absence of control delimiters in Λμ. In the presence of positives, however, we define a control delimiter as follows:

$$\mu\widehat{\mathrm{tp}}.\big\langle t_{+} \,\big\|\, \tilde{\mu}x^{+}.\langle \mathrm{delay}(x^{+}) \,\|\, \widehat{\mathrm{tp}}\rangle\big\rangle$$

where $\mathrm{delay}(t_{+}) \stackrel{\text{def}}{=} \mu\{\alpha^{+}\}.\langle t_{+} \,\|\, \alpha^{+}\rangle$.

This allows us to define (variants of) the operators *shift*$_0$ $(S_0 k.t)$

and $reset_0$ $(<\cdot>_0)$ of Danvy and Filinski [DF90] and Shan [Sha07]. The two operators satisfy, in the λ calculus in call by value:

$$\begin{array}{rcl} <V>_0 & \triangleright & V \\ <E(S_0 x.t)>_0 & \triangleright & t[\lambda x.<E(x)>_0/x] \end{array}$$

We obtain such operators in the calculus $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^{\ominus}}$ by defining operations that satisfy the following reductions:

$$\begin{array}{lcl} \langle <t_+>_0 \| e_+ \rangle & \triangleright^*_{\mathrm{R}_p} & \langle t_+ \| \widehat{\mathrm{tp}}_0 \rangle [\{e_+\}] \\ \langle S_0 x^+.t_+ \| e_+ \rangle [\{e'_+\}] & \triangleright^*_{\mathrm{R}_p} & \langle t_+ [\{\mathsf{k}_{e_+}\}/x^+] \| e'_+ \rangle \\ \text{where:} & & \\ \langle V_+ \| \widehat{\mathrm{tp}}_0 \rangle [\{e_+\}] & \triangleright^*_{\mathrm{R}_p} & \langle V_+ \| e_+ \rangle \\ \langle \mathsf{k}_{e_+} \| V_+ \cdot e'_+ \rangle & \triangleright^*_{\mathrm{R}_p} & \langle V_+ \| e_+ \rangle [\{e'_+\}] \end{array}$$

In other words we take for any term t_+ and any context e_+:

$$\begin{array}{rcl} \widehat{\mathrm{tp}}_0 & \stackrel{\mathrm{def}}{=} & \tilde{\mu} x^+.\langle \mathrm{delay}(x^+) \| \widehat{\mathrm{tp}} \rangle \\ <t_+>_0 & \stackrel{\mathrm{def}}{=} & \mu \alpha^+.\langle \mu \widehat{\mathrm{tp}}.\langle t_+ \| \widehat{\mathrm{tp}}_0 \rangle \| \{\alpha^+\} \rangle \\ \mathsf{k}_{e_+} & \stackrel{\mathrm{def}}{=} & \lambda x^+.\mu \alpha^+.\langle \mu \widehat{\mathrm{tp}}.\langle x^+ \| e_+ \rangle \| \{\alpha^+\} \rangle \\ S_0 x^+.t_+ & \stackrel{\mathrm{def}}{=} & \mu \alpha^+.\langle \lambda x^+.\mathrm{delay}(t_+) \| \{\mathsf{k}_{\alpha^+}\} \cdot \widehat{\mathrm{tp}} \rangle \end{array}$$

These definitions are inspired from Materzok and Biernacki's CPS translation for $S_0 k.t_+$ and $<t_+>_0$ [MB11].

Reductions are obtained as follows. One has:

$$\begin{array}{rcl} \langle <V_+>_0 \| \alpha^+ \rangle & \triangleright^*_{\mathrm{R}_p} & \langle t_+ \| \widehat{\mathrm{tp}}_0 \rangle [\{\alpha^+\}] \\ & \triangleright^*_{\mathrm{R}_p} & \langle V_+ \| \alpha^+ \rangle \end{array}$$

Besides, let $e_+(\cdot)$ be a context that has an adjoint term ${e_+}^*(\cdot)$ in the

following sense:

$$\langle e_+{}^*(x^+) \,\|\, \alpha^+ \rangle \rhd^*_{\mathrm{R}_p} \langle x^+ \,\|\, e_+(\alpha^+) \rangle$$

One has:

$$\begin{aligned}\langle <e_+{}^*(S_0 k.t_+)>_0 \,\|\, \alpha^+ \rangle &\rhd^*_{\mathrm{R}_p} \langle e_+{}^*(S_0 k.t_+) \,\|\, \widehat{\mathsf{tp}}_0 \rangle [\{\alpha^+\}] \\ &\to^*_{\mathrm{R}_p} \langle S_0 k.t_+ \,\|\, e_+(\widehat{\mathsf{tp}}_0) \rangle [\{\alpha^+\}] \\ &\rhd^*_{\mathrm{R}_p} \langle t_+[\mathsf{k}_{e_+(\widehat{\mathsf{tp}})}/k] \,\|\, \alpha^+ \rangle\end{aligned}$$

where:

$$\mathsf{k}_{e_+(\widehat{\mathsf{tp}})} \simeq_{\mathrm{R}_p} \lambda x^+.<e_+{}^*(x^+)>_0$$

Indeed, one has:

$$\langle \mathsf{k}_{e_+(\widehat{\mathsf{tp}}_0)} \,\|\, y^+ \cdot \alpha^+ \rangle \rhd_{\mathrm{R}_p} \langle y^+ \,\|\, e_+(\widehat{\mathsf{tp}}_0) \rangle [\{\alpha^+\}] \lhd_{\mathrm{R}_p} \langle \lambda x^+.<e_+{}^*(x^+)>_0 \,\|\, y^+ \cdot \alpha^+ \rangle$$

Thus, we implemented in $\mathbf{L}_{\mathrm{pol},\widehat{\mathsf{tp}}^\ominus}$ the reduction behaviour of S_0 as follows:

$$<e_+{}^*(S_0 x.t)>_0 \rhd^*_{\mathrm{R}_p} t[\mathsf{k}_{e_+(\widehat{\mathsf{tp}})}/x]$$

where $\mathsf{k}_{e_+(\widehat{\mathsf{tp}})}$ is equivalent to $\lambda x.<e_+{}^*(x)>_0$ which takes part in the original definition.

A variant of Felleisen's last operator

Felleisen proposed in a note at the end of [AH08] to consider an operator $\mathcal{G}$ that satisfies the following reduction rule:

$$<E(\mathcal{G}\, f)> \rhd <f\ [E]>$$

where $<\cdot>$ represents a control delimited and where $[E]$ represents an encoding of the context E inside the values of the language. This encoding is made at runtime by a "*meta-function*".

The operator $\mathcal{T}$ of λℓ and $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^{\ominus}}$ is a variant of $\mathcal{G}$[7]:

$$\langle \mathcal{T}\, t_{\ominus} \| \pi \rangle [\pi'_{\ominus}, \sigma] \rhd^{*}_{\mathrm{R}_p} \langle t_{\ominus} \| [\pi] \cdot \pi'_{\ominus} \rangle [\sigma]$$

that is to say in Felleisen's notation:

$$<\pi^{*}(\mathcal{T}\, f)> \rhd f\,[\pi]$$

In the calculus λℓ, the operator $\mathcal{T}$ is defined from the operator ℓ and is typable (in a simple setting) with $(\sim A \to \bot) \to A$. Interestingly, we could have derived ℓ starting from $\mathcal{T}$:

$$\ell \simeq_{\mathrm{RE}_p} \lambda x^{\ominus}.(\mathcal{T}\, \lambda y^{+}.(x^{\ominus}\, (\mathsf{send}\, y^{+}\, (\mathcal{T}\, \lambda z^{+}.z^{+}))))$$

However, we cannot derive the type $\ell : (A \to \bot) \to \sim A$ from the one of $\mathcal{T}$. This led us to choose ℓ over $\mathcal{T}$.

In fact, it is hard to find a logical[8] counterpart to delimited continuations and to show that it brings added value compared to non-delimited control, because delimited and non-delimited control are equivalent at the level of provability — and this shows that we could not have chosen $\mathcal{T}$ over ℓ.

IV.5 Negation is involutive in λℓ and $\mathbf{L}_{\mathbf{pol},\widehat{\mathbf{tp}}^{\ominus}}$

We have defined negation depending on the polarity:

$$\neg A = \begin{cases} \sim N & \text{if } A = N \\ P \to \bot & \text{if } A = P \end{cases}$$

We motivate this definition in terms of focusing in Section IV.5.1. Then, in Section IV.5.2 we show how delimited control is responsible

[7] Notation: $\mathcal{T}$ comes after $\mathcal{S}_0$, just as $\mathcal{G}$ comes after $\mathcal{F}$.

[8] By logical, we mean a type system without annotations (see Chapter III on page 153). Murthy introduced "*pseudo-classical types*" inspired by Girard's **LC** [Mur92] but with annotations.

for the good behaviour of $\bot$ in our untyped setting.

In Section IV.5.3, we define and motivate what we mean with *isomorphic* in a polarised setting. Finally, in Section IV.5.4 we show that there is an isomorphism between A and $\neg\neg A$ for both A positive and A negative.

IV.5.1 Rules of negation are focused

As observed in Section IV.1, negation is not involutive in the λC calculus because there are two distinct ways of introducing double negation on the left. In $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^{\ominus}}$, the left-introduction rule of the double negation of a negative is obtained with $[e_\ominus]\cdot\widehat{\mathrm{tp}}$:

$$\dfrac{\dfrac{\Gamma \mid e_\ominus : N \vdash \Delta}{\Gamma \vdash [e_\ominus] : {\sim}N \mid \Delta}\,(\vdash {\sim}) \qquad \dfrac{}{\Gamma \mid \widehat{\mathrm{tp}} : \bot \vdash \Delta}\,(\bot \vdash)}{\Gamma \mid [e_\ominus]\cdot\widehat{\mathrm{tp}} : {\sim}N \to \bot \vdash \Delta}\,(\to \vdash)$$

The version with cuts of this rule coincides with the one without cuts:

$$\boxed{\langle t_\ominus \| [e_\ominus]\cdot\widehat{\mathrm{tp}}\rangle \simeq_{\mathrm{R}_p} \langle \mu\alpha^{\ominus}.\langle t_\ominus \| [e_\ominus]\cdot\widehat{\mathrm{tp}}\rangle \| e_\ominus\rangle}\,.$$

Proof. This is obvious when $e_\ominus$ is a stack. When $e_\ominus$ is not a stack, one has:

$$\begin{aligned}\langle t_\ominus \| \underline{[e_\ominus]\cdot\widehat{\mathrm{tp}}}\rangle &\rhd_{\mathrm{R}_p} \langle \underline{[e_\ominus]} \| \tilde{\mu}x^{+}.\langle t_\ominus \| x^{+}\cdot\widehat{\mathrm{tp}}\rangle\rangle \\ &\rhd_{\mathrm{R}_p} \langle \mu\alpha^{\ominus}.\langle [\alpha^{\ominus}] \| \underline{\tilde{\mu}x^{+}}.\langle t_\ominus \| x^{+}\cdot\widehat{\mathrm{tp}}\rangle\rangle \| e_\ominus\rangle \\ &\to_{\mathrm{R}_p} \langle \mu\alpha^{\ominus}.\langle t_\ominus \| [\alpha^{\ominus}]\cdot\widehat{\mathrm{tp}}\rangle \| e_\ominus\rangle \end{aligned}$$ ∎

This identification is a consequence of the following two focusing reduction rules.

Right-introduction of the negation of a negative hides a cut
In other words:

$$\boxed{\langle [e_\ominus] \| e'\rangle \rhd_{\mathrm{R}_p} \langle \mu\alpha^{\ominus}.\langle [\alpha^{\ominus}] \| e'\rangle \| e_\ominus\rangle} \text{ when } e_\ominus \text{ is not a stack.}$$

This means that if $e_{\ominus}$ is not a stack, then $[e_{\ominus}]$ is not a value. It is a term which is evaluated in a strict fashion into the first stack encountered during the evaluation of $e_{\ominus}$.

Left-introduction of the negation of a positive hides a cut

In other words:

$$\boxed{\langle t_{\ominus} \| u_{+} \cdot \widehat{\mathfrak{tp}} \rangle \rhd_{\mathrm{R}_p} \langle u_{+} \| \tilde{\mu} x^{+} . \langle t_{\ominus} \| x^{+} \cdot \widehat{\mathfrak{tp}} \rangle \rangle} \text{ when } u_{+} \text{ is not a value.}$$

This corresponds to the fact that the arguments to a function are called by value.

For the same reasons, the two ways of introducing the double negation of a positive on the right coincide. That the rules of negation hide cuts constitutes the main idea of the involutive negation.[9]

IV.5.2 Delimited control for falsity

The rules of units are delicate in an untyped setting. Yet $\bot$ is essential to define negation as an implication:

$$\frac{\Gamma \vdash P, \Delta \qquad \dfrac{}{\bot \vdash}\,(\bot \vdash)}{\Gamma, P \to \bot \vdash \Delta}\,(\to \vdash) \qquad \frac{\dfrac{\Gamma, P \vdash \Delta}{\Gamma, P \vdash \bot, \Delta}\,(\vdash \bot)}{\Gamma \vdash P \to \bot, \Delta}\,(\vdash \to)$$

In comparison, the impredicative encoding $(\forall X\, X)$ does not let us derive the rule $(\bot \vdash)$, as already underlined by Ariola and Herbelin [AH03]. As a consequence, negation defined as $A \to \forall X\, X$ would not be the negation of sequent calculus.

[9] This already appears in the careful reading of the works of Girard [Gir91, Gir93] and Danos, Joinet and Schellinx [DJS97].

The rules

The rules for falsity are given with the context $\hat{\mathtt{tp}}$ and its binder:

$$\frac{}{\mid \hat{\mathtt{tp}} : \bot \vdash}\ (\bot\ \vdash) \qquad \frac{c : (\Gamma \vdash \Delta)}{\Gamma \vdash \mu\hat{\mathtt{tp}}.c : \bot \mid \Delta}\ (\vdash \bot)$$

Thus we can derive the rules of negation:

$$\frac{\Gamma \vdash t_+ : P \mid \Delta}{\Gamma \mid t_+ \cdot \hat{\mathtt{tp}} : P \to \bot \vdash \Delta}\ (\neg) \qquad \frac{\Gamma \mid e_+ : P \vdash \Delta}{\Gamma \vdash \lambda x^+.\mu\hat{\mathtt{tp}}.\langle x^+ \| e_+ \rangle : P \to \bot \mid \Delta}\ (\neg)$$

The rewriting rules that concerns $\hat{\mathtt{tp}}$ and $\mu\hat{\mathtt{tp}}$ are the following ones:

$$\begin{array}{lccc} (R_{\hat{\mathtt{tp}}}) & \mu\hat{\mathtt{tp}}.\langle t_\ominus \| \hat{\mathtt{tp}} \rangle & \triangleright_{\mathrm{R}_p} & t_\ominus \\ (E_{\hat{\mathtt{tp}}}) & c & \triangleright_{\mathrm{E}_p} & \langle \mu\hat{\mathtt{tp}}.c \| \hat{\mathtt{tp}} \rangle \end{array}$$

The observation that the presence of a specific constant $\mathtt{tp}$ for the rule $(\bot\ \vdash)$ simplifies and enhances the theory of the C operator ([FFKD87, FH92, Par92]) is due to Ariola, Herbelin and Sabry [AH03, AHS04, AH08]. The extension of $\mathtt{tp}$ into $\hat{\mathtt{tp}}$ and the addition of the operator $\mu\hat{\mathtt{tp}}$ to model delimited control operators in call by value is due to the same authors [AHS04, Her05, AHS09]. Our variant is inspired from Herbelin and Ghilezan [HG08].

A unit in an untyped setting

One can see $\hat{\mathtt{tp}}$ as a corrected unit by removing the hypothesis that the rule $(\bot\ \vdash)$ is given by a stack. Indeed let us consider a language p' obtained by replacing $\hat{\mathtt{tp}}$ by a constant $\mathtt{tp}$ that has the following rules:

$$\begin{array}{lccc} (R_{\mathtt{tp}}) & \langle \mu\mathtt{tp}.c \| \mathtt{tp} \rangle & \triangleright_{\mathrm{R}_{p'}} & c \\ (E_{\mathtt{tp}}) & t_\ominus & \triangleright_{\mathrm{E}_{p'}} & \mu\mathtt{tp}.\langle t_\ominus \| \mathtt{tp} \rangle \\ (R_\mu) & \langle \mu\alpha^\ominus.c \| \mathtt{tp} \rangle & \triangleright_{\mathrm{R}_{p'}} & c[\mathtt{tp}/\alpha^\ominus] \end{array}$$

The first two rules are equivalent to the ones of $\hat{\mathtt{tp}}$, while the third one corresponds to the hypothesis that $\mathtt{tp}$ is a stack. We first deduce

the following:

Proposition IV.20. *For all c and $\pi_{\ominus}$ one has:*

$$\mu\mathsf{tp}.c \simeq_{p'} \mu\alpha^{\ominus}.c \qquad (\alpha^{\ominus} \notin \mathbf{fv}(c))$$
$$\pi_{\ominus} \simeq_{p'} \mathsf{tp}\,.$$

Proof. First we have for any c and $\alpha^{\ominus} \notin \mathbf{fv}(c)$:

$$\mu\mathsf{tp}.c \leftarrow_{\mathrm{R}_{p'}} \mu\mathsf{tp}.\langle \mu\alpha^{\ominus}.c \,\|\, \mathsf{tp}\rangle \lhd_{\mathrm{E}_{p'}} \mu\_.c$$

Now let $\pi_{\ominus}$ be a stack. One has:

$$\begin{aligned}
&\underline{\pi_{\ominus}} \rhd_{\mathrm{E}_{p'}} \tilde{\mu}x^{\ominus}.\langle \underline{x^{\ominus}} \,\|\, \pi_{\ominus}\rangle \\
&\quad \to_{\mathrm{E}_{p'}} \tilde{\mu}x^{\ominus}.\big\langle \underline{\mu\mathsf{tp}.\langle x^{\ominus} \,\|\, \mathsf{tp}\rangle} \,\big\|\, \pi_{\ominus}\big\rangle \\
&\quad \simeq_{p'} \tilde{\mu}x^{\ominus}.\big\langle \underline{\mu\alpha^{\ominus}.\langle x^{\ominus} \,\|\, \mathsf{tp}\rangle} \,\big\|\, \pi_{\ominus}\big\rangle \\
&\quad \to_{\mathrm{R}_{p'}} \underline{\tilde{\mu}x^{\ominus}.\langle x^{\ominus} \,\|\, \mathsf{tp}\rangle} \\
&\quad \lhd_{\mathrm{E}_{p'}} \mathsf{tp}
\end{aligned}$$

∎

These equations are very natural… but only in a typed setting. Indeed, in an untyped setting, all commands become equivalent, and therefore also all the terms.

Proposition IV.21. *For any two commands c and c' one has $c \simeq_{p'} c'$.*

Proof. Let c and c' be two commands. One has:

$$y^{+}\cdot\tilde{\mu}y^{+}.c \simeq_{p'} \mathsf{tp} \simeq_{p'} y^{+}\cdot\tilde{\mu}y^{+}.c'$$

From this we conclude $c \simeq_{p'} c'$:

$$\begin{aligned}
c &\lhd^{*}_{\mathrm{R}_{p'}} \langle \lambda x^{+}.x^{+} \,\|\, y^{+}\cdot\tilde{\mu}y^{+}.c\rangle \\
&\simeq_{p'} \langle \lambda x^{+}.x^{+} \,\|\, y^{+}\cdot\tilde{\mu}y^{+}.c'\rangle \\
&\rhd^{*}_{\mathrm{R}_{p'}} c'
\end{aligned}$$

∎

By removing the hypothesis that $\mathtt{tp}$ is a value, we obtain $\hat{\mathtt{tp}}$ and the calculus becomes coherent, as we showed in Corollary IV.19. It remains to show that we have indeed $\neg\neg A \simeq A$.

IV.5.3 Isomorphisms in a polarised setting

Definition IV.22. Two types (or formulae) A and B are isomorphic if they have the same polarity and there exist two terms:

$$x : A \vdash \phi(x) : B$$
$$y : B \vdash \psi(y) : A$$

with:

$$\mathsf{let}\, y\, \mathsf{be}\, \phi(x)\, \mathsf{in}\, \psi(y) \simeq x$$
$$\mathsf{let}\, x\, \mathsf{be}\, \psi(y)\, \mathsf{in}\, \phi(x) \simeq y$$

In this case we write $A \cong_{\phi,\psi} B$.

In the calculus $\mathbf{L}_{\mathrm{pol},\hat{\mathtt{tp}}^{\ominus}}$, we take $\simeq\, = \simeq_{\mathrm{RE}_p}$ above. In the $\lambda\ell$ calculus, we take $\simeq\, = \approx_p$.

Proposition IV.23. *If we have $A \cong_{\phi,\psi} B$ in the calculus $\boldsymbol{L}_{pol,\hat{\mathtt{tp}}^{\ominus}}$, such that ϕ and ψ are given by quasi-proof terms of $\lambda\ell$, then one has $A \cong_{\phi,\psi} B$ in the calculus $\lambda\ell$.*

Proof. If $\phi(x)$ and $\psi(y)$ are quasi-proof terms, then it is also the case of $\mathsf{let}\, y\, \mathsf{be}\, \phi(x)\, \mathsf{in}\, \psi(y)$ and of $\mathsf{let}\, x\, \mathsf{be}\, \psi(y)\, \mathsf{in}\, \phi(x)$. Therefore by hypothesis and definition one has:

$$\mathsf{let}\, y\, \mathsf{be}\, \phi(x)\, \mathsf{in}\, \psi(y) \approx_p x$$
$$\mathsf{let}\, x\, \mathsf{be}\, \psi(y)\, \mathsf{in}\, \phi(x) \approx_p y$$ ∎

Motivation

Definition IV.22 is motivated in the language of duploids of Chapter II as follows. Suppose that two morphisms $\phi : A \to B$ and $\psi : B \to A$

define an isomorphism in a duploid $\mathscr{D}$, in the following sense:

$$\psi \circ \phi = \mathrm{id}_A$$
$$\phi \circ \psi = \mathrm{id}_B$$

Such an isomorphism is not enough to freely identify A with B: Consider $A = N$ and $B = \Downarrow N$ with $\phi = \mathsf{wrap}_N$ and $\psi = \mathsf{unwrap}_N$, or symmetrically $A = P$ and $B = \Uparrow P$. The issue is that we don't necessarily have:

$$\begin{aligned} &\forall f, g,\ (f \circ \psi) \circ (\phi \circ g) = f \circ g \\ &\forall f, g,\ (f \circ \phi) \circ (\psi \circ g) = f \circ g \end{aligned} \tag{IV.2}$$

Lemma IV.24. *Under the condition* (IV.2)*, ϕ is thunkable if and only if ψ is thunkable, and ϕ is linear if and only if ψ is linear.*

Proof. By symmetry, it is enough to show one of the four implications. Suppose that ϕ is linear. On a:

$$\begin{aligned} \forall f, g:\ \psi \circ (f \circ g) &= \psi \circ (\underline{((\phi \circ \psi) \circ f)} \circ g) \text{ since } \phi \circ \psi = \mathrm{id}_B \\ &= \underline{\psi \circ (\phi \circ ((\psi \circ f) \circ g))} \text{ since } \phi \text{ is linear} \\ &= \underline{(\psi \circ \phi)} \circ (\psi \circ f) \circ g \text{ with (IV.2)} \\ &= (\psi \circ f) \circ g \text{ since } \psi \circ \phi = \mathrm{id}_A \end{aligned}$$

hence ψ is linear. ∎

For such a strengthened notion of isomorphism, there are two possibilities:

- A and B have the same polarity. Then the equations (IV.2) are satisfied by ∘∘- or ••-associativity. This notion coincides with the one of isomorphism in the categories $\mathscr{P}$, $\mathscr{N}$.
- A and B have different polarities, say $A = P$ and $B = N$. Then $\phi : P \to N$ is both thunkable and linear. As a consequence, ψ is also both thunkable and linear according to Lemma IV.24.

This suggests two modifications to the definition of an isomorphism in a duploid:

1. A and B have the same polarity; or:
2. ϕ and ψ are both thunkable and linear.

It is easy to see that condition #2 is also enough to obtain (IV.2).

Laurent, Quatrini and Tortora de Falco [LQTdF05] have studied a notion of isomorphism similar to (a syntactic approximation of) condition #2 in (cut-free) **LC**[10]; for which there is no such isomorphism if A and B do not have the same polarity. We extend their argument with Proposition IV.44: in the context of the $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^{\ominus}}$ calculus, no term $x : N \vdash t : P$ is both linear (in N) and thunkable (in P).

We adopted above the first (weaker) notion of isomorphism.

IV.5.4 Proof of the involution

We prove in this section that there exists an isomorphism between A and $\neg\neg A$ first in $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^{\ominus}}$ and then in $\lambda\ell$, in the sense of IV.22.

We insist that the proofs are entirely algebraic. In particular, we need no hypothesis about the syntax being non-extensible or a particular typed setting being strongly normalising. As a consequence, the isomorphism remains true in all extensions of the calculi and of their type systems. (A counter-example to this property is an isomorphism such as $\forall X\,(A \to X) \to X \cong A$ in System F.)

[10] In fact the authors consider Danos, Joinet and Schellinx's $\mathbf{LK}^{\eta}_{p}$ [DJS97]. It is the same system as **LC** except that the stoup of **LC** is less constrained than the η restriction of $\mathbf{LK}^{\eta}_{p}$. By same, we mean that we can define (in the terminology of the respective authors):

- Connectives of $\mathbf{LK}^{\eta}_{p}$ in **LC** by coercing polarities: $A \wedge_m B \stackrel{\text{def}}{=} (A \wedge V) \wedge (B \wedge V)$, etc.
- Connectives of **LC** in $\mathbf{LK}^{\eta}_{p}$ by case analysis: $A^q \wedge B^q \stackrel{\text{def}}{=} A^q \wedge_m B^q$, $A^t \wedge B^t \stackrel{\text{def}}{=} A^t \wedge_a B^t$, etc.

Case $\neg\neg P \cong P$

Definition IV.25. We define the following notation:

$$\tilde{\mu}[x\cdot\widehat{\mathrm{tp}}].c \stackrel{\text{def}}{=} \tilde{\mu}[\alpha^{\ominus}].\langle\lambda x.\mu\widehat{\mathrm{tp}}.c \,\|\, \alpha^{\ominus}\rangle$$

This construct has the following introduction rule:

$$\frac{c : (\Gamma, x : A \vdash \Delta)}{\Gamma \mid \tilde{\mu}[x\cdot\widehat{\mathrm{tp}}].c : {\sim}(A \to \bot) \vdash \Delta}$$

Lemma IV.26. *One has:*

$$\langle [V\cdot\widehat{\mathrm{tp}}] \,\|\, \tilde{\mu}[x\cdot\widehat{\mathrm{tp}}].c\rangle \simeq_{\mathrm{RE}_p} c[V/x]$$
$$\tilde{\mu}[x\cdot\widehat{\mathrm{tp}}].\langle [x\cdot\widehat{\mathrm{tp}}] \,\|\, e_+\rangle \simeq_{\mathrm{RE}_p} e_+$$

Proof. One has:

$$\begin{aligned}
&\langle [V\cdot\underline{\widehat{\mathrm{tp}}}] \,\|\, \tilde{\mu}[x\cdot\widehat{\mathrm{tp}}].c\rangle\\
&\to^{*}_{\mathrm{R}_p} \langle \mu\alpha^{\ominus}.\langle [V\cdot\alpha^{\ominus}] \,\|\, \underline{\tilde{\mu}[x\cdot\widehat{\mathrm{tp}}].c}\rangle \,\|\, \widehat{\mathrm{tp}}\rangle\\
&\to_{\mathrm{R}_p} \langle \mu\alpha^{\ominus}.\langle \underline{\lambda x.\mu\widehat{\mathrm{tp}}.c} \,\|\, V\cdot\alpha^{\ominus}\rangle \,\|\, \widehat{\mathrm{tp}}\rangle\\
&\to_{\mathrm{R}_p} \langle \underline{\mu\alpha^{\ominus}.\langle \mu\widehat{\mathrm{tp}}.c[V/x] \,\|\, \alpha^{\ominus}\rangle} \,\|\, \widehat{\mathrm{tp}}\rangle\\
&\leftarrow_{\mathrm{E}_p} \underline{\langle \mu\widehat{\mathrm{tp}}.c[V/x] \,\|\, \widehat{\mathrm{tp}}\rangle}\\
&\leftarrow_{\mathrm{E}_p} c[V/x]
\end{aligned}$$

$$\begin{aligned}
&\tilde{\mu}[x\cdot\widehat{\mathrm{tp}}].\langle [x\cdot\widehat{\mathrm{tp}}] \,\|\, e_+\rangle\\
&= \tilde{\mu}[\alpha^{\ominus}].\langle \lambda x.\mu\widehat{\mathrm{tp}}.\langle [x\cdot\underline{\widehat{\mathrm{tp}}}] \,\|\, e_+\rangle \,\|\, \alpha^{\ominus}\rangle\\
&\to^{*}_{\mathrm{R}_p} \tilde{\mu}[\alpha^{\ominus}].\Big\langle \lambda x.\underline{\mu\widehat{\mathrm{tp}}.\langle \mu\beta^{\ominus}.\langle [x\cdot\beta^{\ominus}] \,\|\, e_+\rangle \,\|\, \widehat{\mathrm{tp}}\rangle} \,\Big\|\, \alpha^{\ominus}\Big\rangle\\
&\to_{\mathrm{R}_p} \tilde{\mu}[\alpha^{\ominus}].\langle \lambda x.\mu\beta^{\ominus}.\langle \underline{[x\cdot\beta^{\ominus}]} \,\|\, e_+\rangle \,\|\, \alpha^{\ominus}\rangle\\
&\leftarrow_{\mathrm{R}_p} \tilde{\mu}[\alpha^{\ominus}].\Big\langle \underline{\lambda x.\mu\beta^{\ominus}.\langle \mu\alpha^{\ominus}.\langle [\alpha^{\ominus}] \,\|\, e_+\rangle \,\|\, x\cdot\beta^{\ominus}\rangle} \,\Big\|\, \alpha^{\ominus}\Big\rangle\\
&\to_{\mathrm{R}_p} \tilde{\mu}[\alpha^{\ominus}].\langle \mu\alpha^{\ominus}.\langle [\alpha^{\ominus}] \,\|\, e_+\rangle \,\|\, \alpha^{\ominus}\rangle\\
&\simeq_{\mathrm{RE}_p} e_+
\end{aligned}$$

∎

Proposition IV.27 ($P \cong \neg\neg P$ in $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^{\ominus}}$)**.** *We define:*

$$\phi_+(x^+) \stackrel{\text{def}}{=} [x^+\cdot\widehat{\mathrm{tp}}]$$
$$\psi_+(y^+) \stackrel{\text{def}}{=} \mu\alpha^+.\langle y^+ \| \tilde{\mu}[x^+\cdot\widehat{\mathrm{tp}}].\langle x^+ \| \alpha^+\rangle\rangle$$

In $\mathbf{L}_{pol,\widehat{\mathrm{tp}}^{\ominus}}$*, one has* $P \cong_{\phi_+,\psi_+} {\sim}(P \to \bot)$.

Proof. Indeed one has:

$$x^+ : P \vdash [x^+\cdot\widehat{\mathrm{tp}}] : {\sim}(P \to \bot)$$
$$y^+ : {\sim}(P \to \bot) \vdash \mu\alpha^+.\langle y^+ \| \tilde{\mu}[x^+\cdot\widehat{\mathrm{tp}}].\langle x^+ \| \alpha^+\rangle\rangle : P$$

Besides, one has:

$$\begin{aligned}
&\langle \mathsf{let}\, y^+ \,\mathsf{be}\, \phi_+(x^+) \,\mathsf{in}\, \psi_+(y^+) \| \alpha^+\rangle \\
&\rhd_{\mathrm{R}_p} \left\langle [x^+\cdot\widehat{\mathrm{tp}}] \middle\| \tilde{\mu}y^+.\langle \underline{\mu\alpha^+.\langle y^+ \| \tilde{\mu}[x^+\cdot\widehat{\mathrm{tp}}].\langle x^+ \| \alpha^+\rangle\rangle} \middle\| \alpha^+ \right\rangle \\
&\to_{\mathrm{R}_p} \left\langle [x^+\cdot\widehat{\mathrm{tp}}] \middle\| \underline{\tilde{\mu}y^+.\langle y^+ \| \tilde{\mu}[x^+\cdot\widehat{\mathrm{tp}}].\langle x^+ \| \alpha^+\rangle\rangle} \right\rangle \\
&\to_{\mathrm{E}_p} \langle [x^+\cdot\widehat{\mathrm{tp}}] \| \underline{\tilde{\mu}[x^+\cdot\widehat{\mathrm{tp}}].\langle x^+ \| \alpha^+\rangle} \rangle \\
&\simeq_{\mathrm{RE}_p} \langle x^+ \| \alpha^+\rangle
\end{aligned}$$

$$\begin{aligned}
&\langle \mathsf{let}\, x^+ \,\mathsf{be}\, \psi_+(y^+) \,\mathsf{in}\, \phi_+(x^+) \| \beta^+\rangle \\
&\rhd_{\mathrm{R}_p} \left\langle \underline{\mu\alpha^+.\langle y^+ \| \tilde{\mu}[x^+\cdot\widehat{\mathrm{tp}}].\langle x^+ \| \alpha^+\rangle\rangle} \middle\| \tilde{\mu}x^+.\langle [x^+\cdot\widehat{\mathrm{tp}}] \| \beta^+\rangle \right\rangle \\
&\simeq_{\mathrm{RE}_p} \langle y^+ \| \underline{\tilde{\mu}[x^+\cdot\widehat{\mathrm{tp}}].\langle [x^+\cdot\widehat{\mathrm{tp}}] \| \beta^+\rangle} \rangle \\
&\leftarrow^{*}_{\mathrm{E}_p} \langle y^+ \| \beta^+\rangle
\end{aligned}$$

∎

Corollary IV.28 ($P \cong \neg\neg P$ in $\lambda\ell$)**.** *One takes, referring to the terms defined in Figure IV.2e on page 210:*

$$\phi'_+(x^+) \stackrel{\text{def}}{=} (E\, x^+)_+$$
$$\psi'_+(y^+) \stackrel{\text{def}}{=} (D_\neg\, y^+)_+$$

In $\lambda\ell$*, one has* $P \cong_{\phi'_+,\psi'_+} {\sim}(P \to \bot)$.

Proof. We proved in sections IV.3.4 and IV.3.3:

$$x^{+} : P \vdash (E\, x^{+})_{+} : {\sim}(P \to \bot)$$
$$y^{+} : {\sim}(P \to \bot) \vdash (D_{\neg}\, y^{+})_{+} : P$$

We also proved in particular:

$$\langle (E\, x^{+})_{+} \| \alpha^{+} \rangle [\beta^{\ominus}] \succ_{p}^{*} \langle [x^{+} \cdot \beta^{\ominus}] \| \alpha^{+} \rangle$$
$$\langle (D_{\neg}\, [x^{+} \cdot \alpha^{\ominus}])_{+} \| \beta^{+} \rangle \succ_{p}^{*} \langle x^{+} \| \beta^{+} \rangle [\alpha^{\ominus}]$$

According to Proposition IV.11, these reductions are equivalences between commands of $\mathbf{L}_{\mathrm{pol},\hat{\mathfrak{tp}}^{\ominus}}$. As a consequence one has by extensionality in $\mathbf{L}_{\mathrm{pol},\hat{\mathfrak{tp}}^{\ominus}}$:

$$(E\, x^{+})_{+} \simeq_{\mathrm{RE}_{p}} [x^{+} \cdot \hat{\mathfrak{tp}}] = \phi_{+}(x^{+})$$
$$(D_{\neg}\, y^{+})_{+} \simeq_{\mathrm{RE}_{p}} \mu\beta^{+}.\langle y^{+} \| \tilde{\mu}[x^{+} \cdot \hat{\mathfrak{tp}}].\langle x^{+} \| \beta^{+} \rangle \rangle = \psi_{+}(y)$$

According to Proposition IV.27, one therefore has $P \cong_{\phi'_{+},\psi'_{+}} {\sim}(P \to \bot)$ in $\mathbf{L}_{\mathrm{pol},\hat{\mathfrak{tp}}^{\ominus}}$. Since both ϕ'_{+} and ψ'_{+} are quasi-proof terms, we have according to Proposition IV.23 $P \cong_{\phi'_{+},\psi'_{+}} {\sim}(P \to \bot)$ in $\lambda\ell$. ∎

Case $\neg\neg N \cong N$

Definition IV.29. We define the following notation:

$$\lambda[\alpha^{\ominus}].t \stackrel{\text{def}}{=} \lambda x^{+}.\mu\beta.\langle x^{+} \| \tilde{\mu}[\alpha^{\ominus}].\langle t \| \beta \rangle \rangle$$

This construction has the following derived rule:

$$\frac{\Gamma \vdash t : A \mid \alpha^{\ominus} : N, \Delta}{\Gamma \vdash \lambda[\alpha^{\ominus}].t : (\sim N) \to A}$$

Lemma IV.30. *One has:*

$$\langle \lambda[\alpha^{\ominus}].\mu\hat{\mathfrak{tp}}.c \| [\pi_{\ominus}] \cdot \hat{\mathfrak{tp}} \rangle \simeq_{\mathrm{RE}_{p}} c[\pi_{\ominus}/\alpha^{\ominus}]$$
$$\lambda[\alpha^{\ominus}].\mu\hat{\mathfrak{tp}}.\langle t_{+} \| [\alpha^{\ominus}] \cdot \hat{\mathfrak{tp}} \rangle \simeq_{\mathrm{RE}_{p}} t_{+}$$

Proof. Indeed:

$$
\begin{aligned}
&\langle \lambda[\alpha^{\ominus}].\mu\hat{\mathrm{tp}}.c \,\|\, [\pi_{\ominus}]\cdot\hat{\mathrm{tp}}\rangle\\
&= \left\langle \lambda x^{+}.\mu\beta^{\ominus}.\langle x^{+} \,\|\, \tilde{\mu}[\alpha^{\ominus}].\langle \mu\hat{\mathrm{tp}}.c \,\|\, \beta^{\ominus}\rangle\right\rangle \,\|\, [\pi_{\ominus}]\cdot\underline{\hat{\mathrm{tp}}}\right\rangle\\
&\to_{\mathrm{R}_p} \left\langle \mu\beta^{\ominus}.\left\langle \underline{\lambda x^{+}.\mu\beta^{\ominus}.}\langle x^{+} \,\|\, \tilde{\mu}[\alpha^{\ominus}].\langle \mu\hat{\mathrm{tp}}.c \,\|\, \beta^{\ominus}\rangle\right\rangle \,\|\, [\pi_{\ominus}]\cdot\beta^{\ominus}\right\rangle \,\|\, \hat{\mathrm{tp}}\right\rangle\\
&\to^{*}_{\mathrm{R}_p} \left\langle \mu\beta^{\ominus}.\langle [\pi_{\ominus}] \,\|\, \underline{\tilde{\mu}[\alpha^{\ominus}].}\langle \mu\hat{\mathrm{tp}}.c \,\|\, \beta^{\ominus}\rangle\right\rangle \,\|\, \hat{\mathrm{tp}}\right\rangle\\
&\to_{\mathrm{R}_p} \langle \underline{\mu\beta^{\ominus}.\langle \mu\hat{\mathrm{tp}}.c[\pi_{\ominus}/\alpha^{\ominus}] \,\|\, \beta^{\ominus}\rangle} \,\|\, \hat{\mathrm{tp}}\rangle\\
&\leftarrow_{\mathrm{E}_p} \underline{\langle \mu\hat{\mathrm{tp}}.c[\pi_{\ominus}/\alpha^{\ominus}] \,\|\, \hat{\mathrm{tp}}\rangle}\\
&\leftarrow_{\mathrm{E}_p} c[\pi_{\ominus}/\alpha^{\ominus}]
\end{aligned}
$$

$$
\begin{aligned}
&\lambda[\alpha^{\ominus}].\mu\hat{\mathrm{tp}}.\langle t_{+} \,\|\, [\alpha^{\ominus}]\cdot\underline{\hat{\mathrm{tp}}}\rangle\\
&\to_{\mathrm{R}_p} \lambda[\alpha^{\ominus}].\underline{\mu\hat{\mathrm{tp}}.\langle \mu\beta^{\ominus}.\langle t_{+} \,\|\, [\alpha^{\ominus}]\cdot\beta^{\ominus}\rangle \,\|\, \hat{\mathrm{tp}}\rangle}\\
&\to_{\mathrm{R}_p} \lambda[\alpha^{\ominus}].\mu\beta^{\ominus}.\langle t_{+} \,\|\, [\alpha^{\ominus}]\cdot\beta^{\ominus}\rangle\\
&\to_{\mathrm{R}_p} \lambda x^{+}.\mu\beta^{\ominus}.\langle x^{+} \,\|\, \tilde{\mu}[\alpha^{\ominus}].\langle t_{+} \,\|\, \underline{[\alpha^{\ominus}]}\cdot\beta^{\ominus}\rangle\rangle\\
&\leftarrow_{\mathrm{R}_p} \lambda x^{+}.\mu\beta^{\ominus}.\left\langle x^{+} \,\middle\|\, \underline{\tilde{\mu}[\alpha^{\ominus}].\langle [\alpha^{\ominus}] \,\|\, \tilde{\mu}x^{+}.\langle t_{+} \,\|\, x^{+}\cdot\beta^{\ominus}\rangle\rangle}\right\rangle\\
&\leftarrow_{\mathrm{E}_p} \lambda x^{+}.\mu\beta^{\ominus}.\langle x^{+} \,\|\, \underline{\tilde{\mu}x^{+}.}\langle t_{+} \,\|\, x^{+}\cdot\beta^{\ominus}\rangle\rangle\\
&\to_{\mathrm{R}_p} \underline{\lambda x^{+}.\mu\beta^{\ominus}.\langle t_{+} \,\|\, x^{+}\cdot\beta^{\ominus}\rangle}\\
&\leftarrow_{\mathrm{E}_p} t_{+}
\end{aligned}
$$

∎

Proposition IV.31 ($N \cong \neg\neg N$ in $\mathbf{L}_{\mathrm{pol},\hat{\mathrm{tp}}^{\ominus}}$)**.** *Let us take:*

$$
\begin{aligned}
\phi_{\ominus}(x^{\ominus}) &\stackrel{\text{def}}{=} \lambda[\alpha^{\ominus}].\mu\hat{\mathrm{tp}}.\langle x^{\ominus} \,\|\, \alpha^{\ominus}\rangle\\
\psi_{\ominus}(y^{\ominus}) &\stackrel{\text{def}}{=} \mu\alpha^{\ominus}.\langle y^{\ominus} \,\|\, [\alpha^{\ominus}]\cdot\hat{\mathrm{tp}}\rangle
\end{aligned}
$$

In the calculus $\boldsymbol{L}_{pol,\hat{\mathrm{tp}}^{\ominus}}$*, one has:*

$$N \cong_{\phi_{\ominus},\psi_{\ominus}} (\sim N) \to \bot$$

Proof. Indeed one has:

$$x^{\ominus} : N \vdash \lambda[\alpha^{\ominus}].\mu\hat{\mathsf{tp}}.\langle x^{\ominus} \| \alpha^{\ominus}\rangle : (\sim N) \to \bot$$
$$y^{\ominus} : (\sim N) \to \bot \vdash \mu\alpha^{\ominus}.\langle y^{\ominus} \| [\alpha^{\ominus}]\cdot\hat{\mathsf{tp}}\rangle$$

One also has:

$$\begin{aligned}
&\langle \mathsf{let}\, y^{\ominus}\, \mathsf{be}\, \phi_{\ominus}(x^{\ominus})\, \mathsf{in}\, \psi_{\ominus}(y^{\ominus}) \| \alpha^{\ominus}\rangle \\
&\rhd_{\mathrm{R}_p} \left\langle \lambda[\alpha^{\ominus}].\mu\hat{\mathsf{tp}}.\langle x^{\ominus} \| \alpha^{\ominus}\rangle \middle\| \underline{\tilde{\mu} y^{\ominus}.\langle \mu\alpha^{\ominus}.\langle y^{\ominus} \| [\alpha^{\ominus}]\cdot\hat{\mathsf{tp}}\rangle} \middle\| \alpha^{\ominus}\right\rangle \\
&\to^{*}_{\mathrm{R}_p} \langle \underline{\lambda[\alpha^{\ominus}].\mu\hat{\mathsf{tp}}.\langle x^{\ominus} \| \alpha^{\ominus}\rangle} \| [\alpha^{\ominus}]\cdot\hat{\mathsf{tp}}\rangle \\
&\simeq_{\mathrm{RE}_p} \langle x^{\ominus} \| \alpha^{\ominus}\rangle
\end{aligned}$$

$$\begin{aligned}
&\langle \mathsf{let}\, x^{\ominus}\, \mathsf{be}\, \psi_{\ominus}(y^{\ominus})\, \mathsf{in}\, \phi_{\ominus}(x^{\ominus}) \| \beta^{\ominus}\rangle \\
&\rhd_{\mathrm{R}_p} \left\langle \mu\alpha^{\ominus}.\langle y^{\ominus} \| [\alpha^{\ominus}]\cdot\hat{\mathsf{tp}}\rangle \middle\| \underline{\tilde{\mu} x^{\ominus}.\langle \lambda[\alpha^{\ominus}].\mu\hat{\mathsf{tp}}.\langle x^{\ominus} \| \alpha^{\ominus}\rangle} \middle\| \beta^{\ominus}\right\rangle \\
&\to^{*}_{\mathrm{R}_p} \langle \underline{\lambda[\alpha^{\ominus}].\mu\hat{\mathsf{tp}}.\langle y^{\ominus} \| [\alpha^{\ominus}]\cdot\hat{\mathsf{tp}}\rangle} \| \beta^{\ominus}\rangle \\
&\simeq_{\mathrm{RE}_p} \langle y^{\ominus} \| \beta^{\ominus}\rangle
\end{aligned}$$

∎

Corollary IV.32 ($N \cong \neg\neg N$ in $\lambda\ell$)**.** *We define according to Figure IV.2e on page 210:*

$$\phi'_{\ominus}(x^{\ominus}) \stackrel{\text{def}}{=} \lambda y^{+}.(\mathsf{send}\, y^{+}\, x^{\ominus})_{\ominus}$$
$$\psi'_{\ominus}(y^{\ominus}) \stackrel{\text{def}}{=} (\mathcal{T}\, y^{\ominus})_{\ominus}$$

In $\lambda\ell$, one has $N \cong_{\phi'_{\ominus},\psi'_{\ominus}} \sim N \to \bot$.

Proof. We proved in sections IV.3.4 and IV.3.3:

$$x^{\ominus} : N \vdash \lambda y^{+}.(\mathsf{send}\, y^{+}\, x^{\ominus})_{\ominus} : \sim N \to \bot$$
$$y^{\ominus} : \sim N \to \bot \vdash (\mathcal{T}\, y^{\ominus})_{\ominus} : N$$

Besides one has:

$$\langle \lambda y^{+}.(\mathsf{send}\, y^{+}\, x^{\ominus})_{\ominus} \| [\alpha^{\ominus}]\cdot\beta^{\ominus}\rangle \succ^{*}_{p} \langle x^{\ominus} \| \alpha^{\ominus}\rangle[\beta^{\ominus}]$$
$$\langle (\mathcal{T}\, y^{\ominus})_{\ominus} \| \alpha^{\ominus}\rangle[\beta^{\ominus}, \gamma^{\ominus}] \succ^{*}_{p} \langle y^{\ominus} \| [\alpha^{\ominus}]\cdot\beta^{\ominus}\rangle[\gamma^{\ominus}]$$

According to Proposition IV.11, these reductions are equivalences of commands of $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^{\ominus}}$. As a consequence one again has by extensionality in $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^{\ominus}}$:

$$\lambda y^{+}.(\mathsf{send}\, y^{+}\, x^{\ominus})_{\ominus} \simeq_{\mathrm{RE}_p} \lambda[\alpha^{\ominus}].\mu\widehat{\mathrm{tp}}.\langle x^{\ominus} \| \alpha^{\ominus}\rangle = \phi_{\ominus}(x^{\ominus})$$
$$(\mathcal{T}\, y^{\ominus})_{\ominus} \simeq_{\mathrm{RE}_p} \mu\alpha^{\ominus}.\langle y^{\ominus} \| [\alpha^{\ominus}]\cdot\widehat{\mathrm{tp}}\rangle = \psi_{\ominus}(y^{\ominus})$$

According to Proposition IV.31, one therefore has $N \cong_{\phi'_{\ominus},\psi'_{\ominus}} {\sim}N \to \bot$ in $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^{\ominus}}$. Again, since $\phi'_{\ominus}$ and $\psi'_{\ominus}$ are quasi-proof terms, one has according to Proposition IV.23 $N \cong_{\phi'_{\ominus},\psi'_{\ominus}} {\sim}N \to \bot$ in $\lambda\ell$. ∎

IV.6 A semantic characterisation of the stoup

This section is a study of the equations of the untyped calculus $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^{\ominus}}$, and in particular the notion of thunkability of terms and linearity of contexts from Chapter II. We should mention that the development of this section is not specific to $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^{\ominus}}$ and can be generalised, for instance to the calculus $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^{+}}$.

Linearity and thunkability are typically approximated in various ways. The notion of value and stack is the simplest, and has the least structure. For instance, we do not ask that the syntactic composite $\mathsf{let}\, x\, \mathsf{be}\, V_1\, \mathsf{in}\, V_2$ of two values be a value. The *stoup* is a syntactic device which, in **LC** [Gir91], describes with hindsight a generalisation of values and stacks. It amounts (modulo duality) to add, to values made from products and sums, an operation of composition and operations such as distributivity. In counterpart, this syntactic notion is more delicate to manipulate formally [LQTdF05].

We use the notions of linearity and thunkability to define a *semantic* notion of stoup, in the sense that the notion is attached to properties rather than to the form, and also that it is likely not recursive as we will conjecture. This is a concise syntactic account, as well as an extension to the polarised case, of the structure of thunkable

and central morphisms studied separately by Führmann [Füh99] and Selinger [Sel01].

First, in IV.6.1 we explain why restrictions knows as "*A-normal forms*", "*monadic normal forms*" or "*η restriction*" are harmful in our context. Then in sections IV.6.2 and IV.6.3 we characterise thunkable and linear terms in terms of commutation properties and substitution properties. Then in Section IV.6.4 we apply the results to prove that additional expansion rules are derivable with certain restrictions. Last, in Section IV.6.5 we show that the notion of thunkable terms and linear context provide an over-approximation of Girard's stoup.

IV.6.1 On A-normal forms

In the study of systems with focalisation, rules of reduction ς (Figure IV.4b), such as the following ones:

$$\begin{aligned}&\langle (t_+, u_+) \| e_+ \rangle \\ &\triangleright_{\mathrm{R}_p} \langle t_+ \| \tilde{\mu} x^+.\langle (x^+, u_+) \| e_+ \rangle \rangle \\ &\to_{\mathrm{R}_p} \left\langle t_+ \middle\| \tilde{\mu} x^+.\left\langle u_+ \middle\| \tilde{\mu} y^+.\langle (x^+, y^+) \| e_+ \rangle \right\rangle \right\rangle\end{aligned}$$

(when t_+ and u_+ are not values), were from the beginning replaced with a restriction of the constructs to values (e.g. (V, W)), with the exceptions of works such as Wadler's [Wad03].

The restriction appeared under various formulations [Gir91, FSDF93, HD94, DJS97, CH00]. In particular, Flanagan *et al.* introduced to this effect the terminology *A-normal form* [FSDF93]; Hatcliff and Danvy understood the role of Moggi's monadic model and called it *monadic normal form* [HD94]; and Danos, Joinet and Schellinx understood that ς-like rules are a necessary consequence of η rules and coined the terminology *η restriction* [DJS97].

The argument in favour of the restriction is that the proofs that correspond to (t, u), $[e_\ominus]$ and $t{\cdot}e$ are still derivable in the restricted system. Indeed, when the pair (V, W) is restricted to values, it is

possible to define (t, u) for any t and u as follows:

$$(t,u) \stackrel{\text{def}}{=} \mu\alpha^{+}.\Big\langle t \,\Big\|\, \tilde{\mu}x.\big\langle u \,\big\|\, \tilde{\mu}y.\langle (x,y) \,\|\, \alpha^{+}\rangle\big\rangle\Big\rangle$$

In what follows, the restriction is not acceptable. Indeed, we are going to substitute variables with thunkable and linear terms. For want of a recursive characterisation of these notions, substitution must therefore be defined for arbitrary terms and contexts, which is not the case in the presence of a restriction to A-normal forms.

IV.6.2 Equalities of commutation

Let us define the following *shifting* operations.

Definition IV.33. We take:

$$\begin{aligned}
\mathrm{delay}(t_\varepsilon) &\stackrel{\text{def}}{=} \lambda_.t_\varepsilon \text{ (with _ negative)}\\
\mathrm{force}_\varepsilon(t_\ominus) &\stackrel{\text{def}}{=} t\ \lambda x.x \stackrel{\text{def}}{=} \mu\alpha^{\varepsilon}.\langle t_\ominus \,\|\, \lambda x.x\cdot\alpha^{\varepsilon}\rangle\\
\mathrm{wrap}^{*}_{\varepsilon}(e_+) &\stackrel{\text{def}}{=} \tilde{\mu}x^{\varepsilon}.\langle (\lambda y.y, x^{\varepsilon}) \,\|\, e_+\rangle\\
\mathrm{unwrap}^{*}(e_\varepsilon) &\stackrel{\text{def}}{=} \tilde{\mu}(_, x^{\varepsilon}).\langle x^{\varepsilon} \,\|\, e_\varepsilon\rangle \text{ (with _ negative)}
\end{aligned}$$

For t_ε one thus has $t_\varepsilon \simeq_{\mathrm{RE}_p} \mathrm{force}_\varepsilon(\mathrm{delay}(t_\varepsilon))$; while for any e_ε one has $e_\varepsilon \simeq_{\mathrm{RE}_p} \mathrm{wrap}^{*}_{\varepsilon}(\mathrm{unwrap}^{*}(e_\varepsilon))$. Indeed:

$$\begin{aligned}
\mathrm{force}_\varepsilon(\mathrm{delay}(t_\varepsilon)) &= \mu\alpha^{\varepsilon}.\langle \underline{\lambda_.t_\varepsilon} \,\|\, \lambda x.x\cdot\alpha^{\varepsilon}\rangle\\
&\to_{\mathrm{R}_p} \underline{\mu\alpha^{\varepsilon}.\langle t_\varepsilon \,\|\, \alpha^{\varepsilon}\rangle}\\
&\leftarrow_{\mathrm{E}_p} t_\varepsilon\\
\mathrm{wrap}^{*}_{\varepsilon}(\mathrm{unwrap}^{*}(e_\varepsilon)) &= \tilde{\mu}x^{\varepsilon}.\Big\langle (\lambda y.y, x^{\varepsilon}) \,\Big\|\, \underline{\tilde{\mu}(_, x^{\varepsilon}).\langle x^{\varepsilon} \,\|\, e_\varepsilon\rangle}\Big\rangle\\
&\to_{\mathrm{R}_p} \underline{\tilde{\mu}x^{\varepsilon}.\langle x^{\varepsilon} \,\|\, e_\varepsilon\rangle}\\
&\leftarrow_{\mathrm{E}_p} e_\varepsilon
\end{aligned}$$

These are not the shifting operations of the duploid but rather the

operations of *weak* shifts, in the sense that we can have:

$$\mathrm{delay}(\mathrm{force}_\varepsilon(x^\ominus)) \not\simeq_{\mathrm{RE}_p} x^\ominus$$
$$\mathrm{unwrap}^*(\mathrm{wrap}^*_\varepsilon(x^+)) \not\simeq_{\mathrm{RE}_p} x^+$$

Still, weak shifts are enough for our purposes.

Freshness clauses

In what follows μq refers to a binder among $\mu\alpha, \lambda x.\mu\alpha$ and $\tilde{\mu} q$ refers to one among $\tilde{\mu} x, \tilde{\mu}(x, y), \tilde{\mu}[\alpha^\ominus]$. We define $\mathbf{v}(q)$ as the set of variables that appear in μq or $\tilde{\mu} q$ (depending on the case). We define pairwise freshness as follows:

- $q \# f$ when $\mathbf{v}(q) \cap \mathbf{fv}(f) = \varnothing$;
- $f \# f'$ when $\mathbf{fv}(f) \cap \mathbf{fv}(f') = \varnothing$;
- $q \# q'$ when $\mathbf{v}(q) \cap \mathbf{v}(q') = \varnothing$.

We now introduce a definition to simplify the statement of freshness clauses for variables.

Definition IV.34 (Bindings)**.**

1. A *simple binding* (notation $\{f/q\}$) is given by a pair (f, q) of the form $(t, \tilde{\mu} q)$ or $(e, \mu q)$ together with one of the following:
   - f is a positive term and $\tilde{\mu} q$ is of the form $\tilde{\mu} x^+, \tilde{\mu}(x, y), \tilde{\mu}[\alpha^\ominus]$;
   - f is a negative term and $\tilde{\mu} q$ is of the form $\tilde{\mu} x^\ominus$;
   - f is a positive context and μq is of the form $\mu\alpha^+$;
   - f is a negative context and μq is of the form $\mu\alpha^\ominus, \lambda x.\mu\alpha$.
2. A *double binding* (notation $\{f_1/q_1, f_2/q_2\}$) is a pair $(f_1, q_1), (f_2, q_2)$ of simple bindings that satisfies $q_1 \# f_2$, $q_2 \# f_1$ and $q_1 \# q_2$.

Equalities of commutation for values

Values and stacks verify certain equalities of commutation:

Lemma IV.35. *Let c be a command. For any double binding $\{V/q, e/q'\}$, one has:*

$$\langle \mu q'.\langle V \| \tilde{\mu} q.c\rangle \| e\rangle \simeq_{\mathrm{RE}_p} \langle V \| \tilde{\mu} q.\langle \mu q'.c \| e\rangle\rangle$$

For any double binding of the form $\{\pi/q, t/q'\}$, one has:

$$\langle \mu q.\langle t \| \tilde{\mu} q'.c\rangle \| \pi\rangle \simeq_{\mathrm{RE}_p} \langle t \| \tilde{\mu} q'.\langle \mu q.c \| \pi\rangle\rangle$$

Proof. If $\{V/q, e/q'\}$ is a double binding then one has:

$$\begin{aligned}
\langle \mu q'.\langle \underline{V} \| \tilde{\mu} q.c\rangle \| e\rangle &\lhd_{\mathrm{R}_p} \langle V \| \underline{\tilde{\mu} x.\langle \mu q'.\langle x \| \tilde{\mu} q.c\rangle \| e\rangle}\rangle && \text{since } q' \# V \\
&\to_{\mathrm{E}_p} \left\langle V \middle\| \tilde{\mu} q.\left\langle q \middle\| \underline{\tilde{\mu} x.}\langle \mu q'.\langle x \| \tilde{\mu} q.c\rangle \| e\rangle\right\rangle\right\rangle && \text{since } q \# e \\
&\to^2_{\mathrm{R}_p} \langle V \| \tilde{\mu} q.\langle \mu q'.c \| e\rangle\rangle && \text{since } q' \# q
\end{aligned}$$

Same reasoning for $\{\pi/q, t/q'\}$. ∎

Thunkable terms, in the terminology of Chapter II, and linear contexts, behave like values and stacks. So we are mainly interested in commutation properties of the form:

$$\langle \mu q'.\langle t \| \tilde{\mu} q.c\rangle \| e\rangle \simeq_{\mathrm{RE}_p} \langle t \| \tilde{\mu} q.\langle \mu q'.c \| e\rangle\rangle \qquad \text{(IV.3)}$$

for a given t or a given e.

Bound variables convention. As is standard, we implicitly assumed so far Barendregt's variable convention: *bound variables are chosen to be distinct from free variables*. We stress that in the context of an equation of the form (IV.3), this implies that $\{t/q, e/q'\}$ is a double binding. Therefore the hypothesis that we consider double bindings remains implicit in the remainder of the section.

IV.6.3 Thunkable terms, linear contexts

The properties/definitions that follow characterise thunkable terms and linear contexts.

Proposition IV.36 (Characterisation of thunkable terms)**.** *Let t be a term. The following properties are equivalent:*

1. *For all c, e, q, q',* $\boxed{\langle \mu q'.\langle t \| \tilde{\mu} q.c\rangle \| e\rangle \simeq_{\mathrm{RE}_p} \langle t \| \tilde{\mu} q.\langle \mu q'.c \| e\rangle\rangle}$;
2. *For all $e_\ominus$,* $\boxed{\langle \mathrm{delay}(t) \| e_\ominus\rangle \simeq_{\mathrm{RE}_p} \langle t \| \tilde{\mu} x.\langle \mathrm{delay}(x) \| e_\ominus\rangle\rangle}$;
3. *For all c, x,* $\boxed{\langle t \| \tilde{\mu} x.c\rangle \simeq_{\mathrm{RE}_p} c[t/x]}$.

Definition IV.37. A term is *thunkable* if it satisfies one of the above equivalent properties.

Proof. (1. ⇒ 2.) Let $e_\ominus$ be a negative context. One has:

$$\begin{aligned}
&\langle t \| \tilde{\mu} x.\langle \underline{\mathrm{delay}(x)} \| e_\ominus\rangle\rangle \\
&\to_{\mathrm{E}_p} \langle t \| \tilde{\mu} x.\langle \lambda_.\mu\alpha.\langle x \| \alpha\rangle \| e_\ominus\rangle\rangle \\
&\simeq_{\mathrm{RE}_p} \langle \lambda_.\underline{\mu\alpha.\langle t \| \tilde{\mu} x.\langle x \| \alpha\rangle\rangle} \| e_\ominus\rangle \text{ according to 1.} \\
&\leftarrow^*_{\mathrm{E}_p} \langle \mathrm{delay}(t) \| e_\ominus\rangle
\end{aligned}$$

(2. ⇒ 3.) Let c, x. By considering $\mathrm{force}_\varepsilon(\mathrm{delay}(u_\varepsilon)) \simeq_{\mathrm{RE}_p} u_\varepsilon$ one has:

$$\begin{aligned}
&\langle t \| \tilde{\mu} x.c\rangle \\
&\simeq_{\mathrm{RE}_p} \langle t \| \tilde{\mu} x.c[\mathrm{force}(\underline{\mathrm{delay}(x)})/x]\rangle \\
&\leftarrow_{\mathrm{R}_p} \langle t \| \underline{\tilde{\mu} x.}\langle \mathrm{delay}(x) \| \tilde{\mu} y^\ominus.c[\mathrm{force}(y^\ominus)/x]\rangle\rangle \\
&\simeq_{\mathrm{RE}_p} \langle \mathrm{delay}(t) \| \underline{\tilde{\mu} y^\ominus.}c[\mathrm{force}(y^\ominus)/x]\rangle \quad \text{en supposant 2.} \\
&\to_{\mathrm{R}_p} c[\mathrm{force}(\mathrm{delay}(t))/x] \\
&\simeq_{\mathrm{RE}_p} c[t/x]
\end{aligned}$$

whence the result.

(3. ⇒ 1.) Let c, e, q, q' such that $\{t/q, e/q'\}$ is a double binding. Let $x \notin \mathbf{fv}(c, e)$. One has $x \# e$, hence $\{x/q, e/q'\}$ is a double binding. Thus using Lemma IV.35 one has:

$$\langle \mu q'.\langle x \| \tilde{\mu} q.c\rangle \| e\rangle \simeq_{\mathrm{RE}_p} \langle x \| \tilde{\mu} q.\langle \mu q'.c \| e\rangle\rangle$$

Therefore:

$$\begin{aligned}
&\langle \mu q'.\langle \underline{t} \| \tilde{\mu} q.c\rangle \| e\rangle \\
&= \langle \mu q'.\langle x \| \tilde{\mu} q.c\rangle \| e\rangle \underline{[t/x]} \text{ since } q' \# t, x \# e, x \# c \\
&\simeq_{\mathrm{RE}_p} \left\langle t \,\middle\|\, \tilde{\mu} x.\underline{\langle \mu q'.\langle x \| \tilde{\mu} q.c\rangle \| e\rangle}\right\rangle \text{ assuming 3.} \\
&\simeq_{\mathrm{RE}_p} \left\langle t \,\middle\|\, \underline{\tilde{\mu} x.\langle x \| \tilde{\mu} q.\langle \mu q'.c \| e\rangle\rangle}\right\rangle \text{ using the above} \\
&\leftarrow_{\mathrm{E}_p} \langle t \| \tilde{\mu} q.\langle \mu q'.c \| e\rangle\rangle
\end{aligned}$$

whence the result. ∎

Proposition IV.38 (Characterisation of linear contexts)**.** *Let e be a context. The following properties are equivalent:*

1. *For all c, t, q, q',* $\boxed{\langle t \| \tilde{\mu} q'.\langle \mu q.c \| e\rangle\rangle \simeq_{\mathrm{RE}_p} \langle \mu q.\langle t \| \tilde{\mu} q'.c\rangle \| e\rangle}$;
2. *For all t_+,* $\boxed{\langle t_+ \| \mathrm{unwrap}^*(e)\rangle \simeq_{\mathrm{RE}_p} \langle \mu\alpha.\langle t_+ \| \mathrm{unwrap}^*(\alpha)\rangle \| e\rangle\,.}$;
3. *For all c, α,* $\boxed{\langle \mu\alpha.c \| e\rangle \simeq_{\mathrm{RE}_p} c[e/\alpha]}$.

Definition IV.39. A context is linear if it satisfies one of the above equivalent properties.

Proof. The proof is symmetric to the previous one. (1. ⇒ 2.) Let t_+ be a positive term. Equation 2. corresponds to the following case of property 1.:

$$\langle \mu\alpha.\langle t_+ \| \tilde{\mu}(\_, x).\langle x \| \alpha\rangle\rangle \| e\rangle \simeq_{\mathrm{RE}_p} \langle t_+ \| \tilde{\mu}(\_, x).\langle \mu\alpha.\langle x \| \alpha\rangle \| e\rangle\rangle$$

(2. ⇒ 3.) Consider c, α. By considering $\mathrm{wrap}^*_\varepsilon(\mathrm{unwrap}^*(e'_\varepsilon)) \simeq_{\mathrm{RE}_p} e'_\varepsilon$ one

has:

$$\begin{aligned}
&\langle \mu\alpha.c \| e\rangle \\
&\simeq_{\mathrm{RE}_p} \langle \mu\alpha.c[\mathrm{wrap}^*(\underline{\mathrm{unwrap}^*(\alpha)})/\alpha] \| e\rangle \\
&\leftarrow_{\mathrm{R}_p} \langle \underline{\mu\alpha}.\langle \mu\beta^+.c[\mathrm{wrap}^*(\beta^+)/\alpha] \| \mathrm{unwrap}^*(\alpha)\rangle \| e\rangle \\
&\simeq_{\mathrm{RE}_p} \langle \underline{\mu\beta^+}.c[\mathrm{wrap}^*(\beta^+)/\alpha] \| \mathrm{unwrap}^*(e)\rangle \quad \text{d'après 2.} \\
&\to_{\mathrm{R}_p} c[\mathrm{wrap}^*(\mathrm{unwrap}^*(e))/\alpha] \\
&\simeq_{\mathrm{RE}_p} c[e/\alpha]
\end{aligned}$$

whence the result.

(3. ⇒ 1.) Let c, t, q, q' such that $\{e/q, t/q'\}$ is a double binding. One has:

$$\begin{aligned}
&\langle t \| \tilde{\mu}q'.\langle \mu q.c \| \underline{e}\rangle\rangle \\
&\simeq_{\mathrm{RE}_p} \left\langle \mu\alpha.\underline{\langle t \| \tilde{\mu}q'.\langle \mu q.c \| \alpha\rangle\rangle} \middle\| e \right\rangle \text{ en supposant 3., car } q' \# e \\
&\simeq_{\mathrm{RE}_p} \left\langle \underline{\mu\alpha.\langle \mu q.\langle t \| \tilde{\mu}q'.c\rangle \| \alpha\rangle} \middle\| e \right\rangle \text{ par le lemme IV.35} \\
&\leftarrow_{\mathrm{E}_p} \langle \mu q.\langle t \| \tilde{\mu}q'.c\rangle \| e\rangle
\end{aligned}$$

Whence the result. ∎

The above properties are an extension to terms with pattern-matching of Proposition II.15 about duploids, for which we only need weak shifts already, if we examine the proof.

Definition IV.40. (Terminology inspired from Führmann [Füh99], Selinger [Sel01])

- t is *central* if one has $\boxed{\langle u \| \tilde{\mu}q'.\langle t \| \tilde{\mu}q.c\rangle\rangle \simeq_{\mathrm{RE}_p} \langle t \| \tilde{\mu}q.\langle u \| \tilde{\mu}q'.c\rangle\rangle}$ $(\forall c, u, q, q')$.
- e is *central* if one has $\boxed{\langle \mu q'.\langle \mu q.c \| e\rangle \| e'\rangle \simeq_{\mathrm{RE}_p} \langle \mu q.\langle \mu q'.c \| e'\rangle \| e\rangle}$ $(\forall c, e', q, q')$.

Now we define the following notations:

$$\boxed{\begin{array}{l} c\{t/q\} \stackrel{\text{def}}{=} \langle t \parallel \tilde{\mu} q.c \rangle \\ c\{e/q\} \stackrel{\text{def}}{=} \langle \mu q.c \parallel e \rangle \end{array}}$$

for all simple bindings $\{t/q\}$ and $\{e/q'\}$. Notice that this notation is less useful, because it artificially distinguishes $c\{\mu q'.c'/q\}$ and $c'\{\mu q.c/q'\}$ although it is the same command.

With these definitions, the previous properties are stated as the commutation of double bindings:

- t thunkable $\Leftrightarrow\ \forall (c, e, q, q')\, c\{t/q\}\{e/q'\} \simeq_{\mathrm{RE}_p} c\{e/q'\}\{t/q\}$.
- t central $\Leftrightarrow\ \forall (c, u, q, q')\, c\{t/q\}\{u/q'\} \simeq_{\mathrm{RE}_p} c\{u/q'\}\{t/q\}$.
- e linear $\Leftrightarrow\ \forall (c, t, q, q')\, c\{e/q\}\{t/q'\} \simeq_{\mathrm{RE}_p} c\{t/q'\}\{e/q\}$.
- e central $\Leftrightarrow\ \forall (c, e', q, q')\, c\{e/q\}\{e'/q'\} \simeq_{\mathrm{RE}_p} c\{e'/q'\}\{e/q\}$.

Proposition IV.41.

1. *If t is thunkable and $t \simeq_{\mathrm{RE}_p} u$ then u is thunkable. If e is linear and $e \simeq_{\mathrm{RE}_p} e'$ then e' is linear;*
2. *Any value is thunkable and central; any stack is linear and central;*
3. *Any thunkable term and any linear context is central;*
4. (Thanks to the operations $\lambda x.t$ and $t \cdot \pi$:) *Any central context is linear.*
5. (Thanks to the operations $[e^{\ominus}]$ and $\tilde{\mu}[a^{\ominus}].c$:) *Any central term is thunkable.*
6. *Let $\{f/\kappa\}$ be a simple binding with f linear/thunkable (depending on the case) and κ a variable.*
   a) *If t is thunkable then $t[f/\kappa]$ is thunkable;*
   b) *If e is linear then $e[f/\kappa]$ is linear.*

Proof.

1. Immediate through the first characterisation of thunkable/linear.
2. This is lemma IV.35 which extends to the cases of t, e central with same proofs.
3. Let t be thunkable. Then let c, u, q, q' be such that $\{t/q, u/q'\}$ is a double binding. One has:

$$\begin{aligned}
c\{t/q\}\{u/q'\} &= c\{x/q\}\{u/q'\}\underline{[t/x]} \text{ since } q' \# t \\
&\simeq_{\mathrm{RE}_p} c\underline{\{x/q\}\{u/q'\}}\{t/x\} \text{ since } t \text{ is thunkable} \\
&\simeq_{\mathrm{RE}_p} c\{u/q'\}\{x/q\}\underline{\{t/x\}} \text{ since } x \text{ is central} \\
&\simeq_{\mathrm{RE}_p} c\{u/q'\}\{x/q\}[t/x] \text{ since } t \text{ is thunkable} \\
&= c\{u/q'\}\{t/q\}
\end{aligned}$$

   Hence t is central. The proof that linear e is central is the same.
4. Let e be central. Let c, e, q, q' be such that $\{t/q, e/q'\}$ is a double binding. If t is a value then one has $c\{t/q\}\{e/q'\} \simeq_{\mathrm{RE}_p} c\{e/q'\}\{t/q\}$ by thunkability of t. Now assume that t is a positive term which is not a value. We consider π a closed stack (for instance $\pi = \tilde{\mu}x^+.\langle\, \lambda y.y \parallel \hat{\mathfrak{tp}} \,\rangle$). One has:

$$\begin{aligned}
&\langle \mu q'.\langle t \parallel \underline{\tilde{\mu} q.c}\rangle \parallel e\rangle \\
&\to_{\mathrm{E}_p} \underline{\langle \mu q'.\langle t \parallel \tilde{\mu}x^+.\langle x^+ \parallel \tilde{\mu} q.c\rangle\rangle \parallel e\rangle} \\
&\leftarrow^2_{\mathrm{R}_p} \underline{\langle \mu q'.\langle \lambda x^+.\mu_.\langle x^+ \parallel \tilde{\mu} q.c\rangle \parallel t\cdot\pi\rangle \parallel e\rangle} \\
&\simeq_{\mathrm{RE}_p} \langle \lambda x^+.\mu_.\underline{\langle \mu q'.\langle x^+ \parallel \tilde{\mu} q.c\rangle \parallel e\rangle} \parallel t\cdot\pi\rangle \text{ since } e \text{ is central} \\
&\simeq_{\mathrm{RE}_p} \langle \underline{\lambda x^+.\mu_.}\langle x^+ \parallel \tilde{\mu} q.\langle \mu q'.c \parallel e\rangle\rangle \parallel t\cdot\pi\rangle \text{ since } x^+ \text{ is thunkable} \\
&\to^2_{\mathrm{R}_p} \langle t \parallel \underline{\tilde{\mu}x^+.\langle x^+ \parallel \tilde{\mu} q.\langle \mu q'.c \parallel e\rangle\rangle}\rangle \\
&\leftarrow_{\mathrm{E}_p} \langle t \parallel \tilde{\mu} q.\langle \mu q'.c \parallel e\rangle\rangle
\end{aligned}$$

   Hence e is linear.
5. Let t be central. Let c, e, q, q' be such that $\{t/q, e/q'\}$ is a double

binding. If e is a stack then one has $c\{t/q\}\{e/q'\} \simeq_{\mathrm{RE}_p} c\{e/q'\}\{t/q\}$ by linearity of e. Now assume that e is a negative context which is not a stack. One has:

$$\begin{aligned}
&\langle \underline{\mu q'.\langle t \| \tilde{\mu} q.c\rangle} \| e\rangle \\
&\to_{\mathrm{E}_p} \underline{\langle \mu\alpha^{\ominus}.\langle \mu q'.\langle t \| \tilde{\mu} q.c\rangle \| \alpha^{\ominus}\rangle \| e\rangle} \\
&\leftarrow^{2}_{\mathrm{R}_p} \langle [e] \| \tilde{\mu}[\alpha^{\ominus}].\underline{\langle \mu q'.\langle t \| \tilde{\mu} q.c\rangle \| \alpha^{\ominus}\rangle}\rangle \\
&\simeq_{\mathrm{RE}_p} \underline{\langle [e] \| \tilde{\mu}[\alpha^{\ominus}].\langle t \| \tilde{\mu} q.\langle \mu q'.c \| \alpha^{\ominus}\rangle\rangle\rangle} \text{ since } \alpha^{\ominus} \text{ is lienar} \\
&\simeq_{\mathrm{RE}_p} \langle t \| \tilde{\mu} q.\underline{\langle [e] \| \tilde{\mu}[\alpha^{\ominus}].\langle \mu q'.c \| \alpha^{\ominus}\rangle\rangle}\rangle \text{ since } t \text{ is central} \\
&\to^{2}_{\mathrm{R}_p} \langle t \| \tilde{\mu} q.\langle \underline{\mu\alpha^{\ominus}.\langle \mu q'.c \| \alpha^{\ominus}\rangle} \| e\rangle\rangle \\
&\leftarrow_{\mathrm{E}_p} \langle t \| \tilde{\mu} q.\langle \mu q'.c \| e\rangle\rangle
\end{aligned}$$

Hence t is thunkable.

6. Let t be thunkable. Assume that $\kappa \notin \mathbf{fv}(c)$in the following:

$$\begin{aligned}
c\{t[f/\kappa]/x\} &= c\{t/x\}[f/\kappa] \\
&\simeq_{\mathrm{RE}_p} c\{t/x\}\{f/\kappa\} \text{ since } f \text{ is linear/thunkable} \\
&\simeq_{\mathrm{RE}_p} c[t/x]\{f/\kappa\} \text{ since } t \text{ is thunkable} \\
&\simeq_{\mathrm{RE}_p} c[t/x][f/\kappa] \text{ since } f \text{ is linear/thunkable} \\
&= c[t[f/\kappa]/x]
\end{aligned}$$

Hence the result. Same reasoning for e linear. ∎

IV.6.4 Application to dual expansions

The goal is to show that the rules of $\mathbf{L}_{\mathrm{pol},\widehat{\mathrm{tp}}^{\ominus}}$ imply expansions dual to the expansions $\rhd_{\mathrm{E}_p}$, however restricted to thunkable or linear terms. To begin with we define constructs hd (*head*), tl (*tail*), fst (*first*), snd

(*second*) and $[\,]^{-1}$ by solving the following equations in $\mathbf{L}_{\mathrm{pol},\hat{\mathrm{tp}}^{\ominus}}$:

$$\begin{array}{rcl}
\langle \mathsf{hd}(V\cdot\pi)\,\|\,e\rangle & \to^{*}_{\mathrm{R}_p} & \langle V\,\|\,e\rangle \\
\langle t\,\|\,\mathsf{tl}(V\cdot\pi)\rangle & \to^{*}_{\mathrm{R}_p} & \langle t\,\|\,\pi\rangle \\
\langle \mathsf{fst}(V,W)\,\|\,e\rangle & \to^{*}_{\mathrm{R}_p} & \langle V\,\|\,e\rangle \\
\langle \mathsf{snd}(V,W)\,\|\,e\rangle & \to^{*}_{\mathrm{R}_p} & \langle W\,\|\,e\rangle \\
\langle t\,\|\,[\,]^{-1}([\pi_{\ominus}])\rangle & \to^{*}_{\mathrm{R}_p} & \langle t\,\|\,\pi_{\ominus}\rangle
\end{array}$$

These constructs have to be relativised to polarities as follows:

Definition IV.42. For any term t_{+}, any context $e_{\ominus}$ and any $\varepsilon, \varepsilon_1, \varepsilon_2 \in \{+,\ominus\}$, we define the following terms:

$$\begin{array}{rcl}
\mathsf{hd}_{\varepsilon_1\varepsilon_2}(e_{\ominus}) & \stackrel{\mathrm{def}}{=} & \mu\beta^{\varepsilon_1}.\langle \mu(x^{\varepsilon_1}\cdot\alpha^{\varepsilon_2}).\langle x^{\varepsilon_1}\,\|\,\beta^{\varepsilon_1}\rangle\,\|\,e_{\ominus}\rangle \\
\mathsf{tl}_{\varepsilon_1\varepsilon_2}(e_{\ominus}) & \stackrel{\mathrm{def}}{=} & \tilde{\mu}y^{\varepsilon_2}.\langle \mu(x^{\varepsilon_1}\cdot\alpha^{\varepsilon_2}).\langle y^{\varepsilon_2}\,\|\,\alpha^{\varepsilon_2}\rangle\,\|\,e_{\ominus}\rangle \\
\mathsf{fst}_{\varepsilon_1\varepsilon_2}(t_{+}) & \stackrel{\mathrm{def}}{=} & \mu\alpha^{\varepsilon_1}.\langle t_{+}\,\|\,\tilde{\mu}(x^{\varepsilon_1},y^{\varepsilon_2}).\langle x^{\varepsilon_1}\,\|\,\alpha^{\varepsilon_1}\rangle\rangle \\
\mathsf{snd}_{\varepsilon_1\varepsilon_2}(\pi_{\ominus}) & \stackrel{\mathrm{def}}{=} & \mu\alpha^{\varepsilon_2}.\langle t_{+}\,\|\,\tilde{\mu}(x^{\varepsilon_1},y^{\varepsilon_2}).\langle y^{\varepsilon_2}\,\|\,\beta^{\varepsilon_2}\rangle\rangle \\
[\,]^{-1}(V_{+}) & \stackrel{\mathrm{def}}{=} & \tilde{\mu}x^{\ominus}.\langle V_{+}\,\|\,\mu[\alpha^{\ominus}].\langle x^{\ominus}\,\|\,\alpha^{\ominus}\rangle\rangle
\end{array}$$

Now we show that the expansion rules of the $\mathbf{L}_{\mathrm{pol},\hat{\mathrm{tp}}^{\ominus}}$ calculus:

$$\begin{array}{rcl}
t_{\ominus} & \rhd_{\mathrm{E}_p} & \lambda x.\mu\alpha.\langle t_{\ominus}\,\|\,x\cdot\alpha\rangle \\
\pi_{\ominus} & \rhd_{\mathrm{E}_p} & \tilde{\mu}(x,y).\langle (x,y)\,\|\,\pi_{\ominus}\rangle \\
e_{+} & \rhd_{\mathrm{E}_p} & \tilde{\mu}[\alpha^{\ominus}].\langle [\alpha^{\ominus}]\,\|\,e_{+}\rangle
\end{array}$$

respectively imply dual rules which are restricted to linear and thunkable terms:

$$\begin{array}{rcll}
e_{\ominus} & \rhd_{\mathrm{F}_p} & (\mathsf{hd}_{\varepsilon_1\varepsilon_2}e_{\ominus})\cdot(\mathsf{tl}_{\varepsilon_1\varepsilon_2}e_{\ominus}) & \text{if } e_{\ominus} \text{ is linear} \\
t_{+} & \rhd_{\mathrm{F}_p} & (\mathsf{fst}_{\varepsilon_1\varepsilon_2}t_{+},\mathsf{snd}_{\varepsilon_1\varepsilon_2}t_{+}) & \text{if } t_{+} \text{ is thunkable} \\
t_{+} & \rhd_{\mathrm{F}_p} & [[\,]^{-1}t_{+}] & \text{if } t_{+} \text{ is thunkable}
\end{array}$$

Proposition IV.43 (Dual expansions). *One has* $\triangleright_{\mathrm{F}_p} \subseteq \simeq_{\mathrm{RE}_p}$.

Proof. (Case $(\mathsf{hd}_{\varepsilon_1\varepsilon_2} e_\ominus)\cdot(\mathsf{tl}_{\varepsilon_1\varepsilon_2} e_\ominus)$) Let $e_\ominus$ be a linear context. One has, omitting polarity annotations:

$$
\begin{aligned}
&(\mathsf{hd}\ e_\ominus)\cdot(\mathsf{tl}\ e_\ominus)\\
&\triangleright_{\mathrm{E}_p} \tilde{\mu}x.\langle x \,\|\, (\mathsf{hd}\ e_\ominus)\cdot(\mathsf{tl}\ e_\ominus)\rangle\\
&\simeq_{\mathrm{RE}_p} \tilde{\mu}x.\left\langle \mathsf{hd}\ e_\ominus \,\middle\|\, \tilde{\mu}y.\langle \mu\alpha.\langle x \| y\cdot\alpha\rangle \,\|\, \mathsf{tl}\ e_\ominus\rangle\right\rangle && \text{(IV.4)}\\
&\simeq_{\mathrm{RE}_p} \tilde{\mu}x.\left\langle \mu(y\cdot\beta).\langle \mu(z\cdot\alpha).\langle x \| y\cdot\alpha\rangle \,\|\, e_\ominus\rangle \,\middle\|\, e_\ominus\right\rangle && \text{(IV.5)}\\
&\simeq_{\mathrm{RE}_p} \tilde{\mu}x.\left\langle \mu\underline{\gamma}.\left\langle \mu(y\cdot\beta).\langle \mu(z\cdot\alpha).\langle x \| y\cdot\alpha\rangle \,\|\, \underline{\gamma}\right\rangle \,\middle\|\, \underline{\gamma}\right\rangle \,\middle\|\, e_\ominus\right\rangle && \text{(IV.6)}\\
&\to_{\mathrm{E}_p}\to_{\mathrm{R}_p} \tilde{\mu}x.\left\langle \mu(y\cdot\alpha).\underline{\left\langle \mu(y\cdot\beta).\underline{\langle \mu(z\cdot\alpha).\langle x \| y\cdot\alpha\rangle \,\|\, y\cdot\alpha\rangle}\right\rangle \,\middle\|\, y\cdot\alpha}\right\rangle \,\middle\|\, e_\ominus\right\rangle\\
&\to^2_{\mathrm{R}_p} \tilde{\mu}x.\left\langle \underline{\mu(y\cdot\alpha).\langle x \| y\cdot\alpha\rangle} \,\middle\|\, e_\ominus\right\rangle\\
&\leftarrow^2_{\mathrm{E}_p} e_\ominus
\end{aligned}
$$

(IV.4) is by $\tilde{\mu}$ expansion if $\varepsilon_1 = \ominus$, ς reduction otherwise, and by μ expansion if $\varepsilon_2 = +$, ς reduction otherwise.
(IV.5) uses the commutation of double bindings by applying Proposition IV.38 once with each body of $\mathsf{hd}\ e_\ominus$ and $\mathsf{tl}\ e_\ominus$, using the hypothesis that $e_\ominus$ is linear.
(IV.6) is also obtained with Proposition IV.38 with $e_\ominus$ linear.
The cases of $(\mathsf{fst}_{\varepsilon_1\varepsilon_2} t_+, \mathsf{snd}_{\varepsilon_1\varepsilon_2} t_+)$ and $[[\,]^{-1} t_+]$ are proved in the same way. ∎

IV.6.5 Stoup

Proposition IV.44 (Stoup). *Let c be a command. The set:*

$$\left\{\alpha^+ \,\middle|\, \mu\alpha^+.c \text{ is thunkable}\right\} \cup \left\{x^\ominus \,\middle|\, \tilde{\mu}x^\ominus.c \text{ is linear}\right\}$$

has at most one element.

Proof. By contradiction. Assume that $\mu\alpha^+.c$ is thunkable and $\tilde{\mu}x^\ominus.c$ is linear at the same time. (The cases of two thunkable terms and two

linear contexts are the same using the property that linear and thunkable implies central.) We take c_1 and c_2 two commands such that $c_1 \not\simeq_{\mathrm{RE}_p} c_2$; for instance $\langle y_1 \parallel \beta \rangle$ and $\langle y_2 \parallel \beta \rangle$ for $y_1 \neq y_2$ (see Corollary IV.19). One therefore has $\alpha^+, x^\ominus \notin \mathbf{fv}(c_1, c_2)$. We consider the following commands:

$$c_1' \stackrel{\text{def}}{=} \langle \mu\alpha^+.\langle \mu\_.c_1 \parallel \tilde{\mu}x^\ominus.c\rangle \parallel \tilde{\mu}\_.c_2 \rangle$$
$$c_2' \stackrel{\text{def}}{=} \langle \mu\_.c_1 \parallel \tilde{\mu}x^\ominus.\langle \mu\alpha^+.c \parallel \tilde{\mu}\_.c_2\rangle\rangle$$

One has:

$$\begin{aligned}
c_1' &= \langle \mu\alpha^+.\langle \underline{\mu_}.c_1 \parallel \tilde{\mu}x^\ominus.c\rangle \parallel \tilde{\mu}_.c_2 \rangle \\
&\simeq_{\mathrm{RE}_p} \langle \underline{\mu\alpha^+}.c_1 \parallel \tilde{\mu}_.c_2\rangle && \text{by linearity of } \tilde{\mu}x^\ominus.c \\
&\simeq_{\mathrm{RE}_p} c_1 && \text{since } \alpha^+ \notin \mathbf{fv}(c_1) \\
c_2' &= \langle \mu_.c_1 \parallel \tilde{\mu}x^\ominus.\langle \mu\alpha^+.c \parallel \underline{\tilde{\mu}_}.c_2\rangle\rangle \\
&\simeq_{\mathrm{RE}_p} \langle \mu_.c_1 \parallel \underline{\tilde{\mu}x^\ominus}.c_2\rangle && \text{by thunkability of } \mu\alpha^+.c \\
&\simeq_{\mathrm{RE}_p} c_2 && \text{since } x^\ominus \notin \mathbf{fv}(c_2).
\end{aligned}$$

Yet:

$$\begin{aligned}
c_2' &= \langle \mu_.c_1 \parallel \tilde{\mu}x^\ominus.\langle \mu\alpha^+.c \parallel \underline{\tilde{\mu}_}.c_2\rangle\rangle \\
&\triangleleft_{\mathrm{R}_p} \langle \mu\alpha^+.\langle \mu_.c_1 \parallel \tilde{\mu}x^\ominus.\langle \underline{\mu\alpha^+}.c \parallel \alpha^+\rangle\rangle \parallel \tilde{\mu}_.c_2\rangle \\
&\to_{\mathrm{R}_p} \langle \mu\alpha^+.\langle \mu_.c_1 \parallel \tilde{\mu}x^\ominus.c\rangle \parallel \tilde{\mu}_.c_2\rangle = c_1'
\end{aligned}$$

whence $c_1 \simeq_{\mathrm{RE}_p} c_2$, which is impossible. ∎

This motivates a semantic redefinition of **LC**'s stoup.

Definition IV.45 (**£** predicate)**.** *Short definition:* Whenever a negative variable (respectively a positive co-variable) is underlined in $\Gamma \mathbin{\pounds} \Delta$, where Γ is a set of variables and Δ a set of co-variables, it means that the associated command, associated term or associated context is thunkable (resp. linear) in this variable.

Long definition: we define the predicate $\mathbf{£}$ as follows:

$$c : \left(\Gamma, \underline{\Lambda} \mathbf{£} \underline{\Pi}, \Delta\right) \overset{\text{def}}{\iff} \begin{cases} \Lambda = \{x^{\ominus}\}, \Pi = \varnothing \text{ and } \tilde{\mu}x^{\ominus}.c \text{ is linear} \\ \text{or} \\ \Lambda = \varnothing, \Pi = \{\alpha^{+}\} \text{ and } \mu\alpha^{+}.c \text{ is thunkable} \end{cases}$$
$$\text{and } \mathbf{fv}(c) \subseteq \biguplus\{\Lambda, \Gamma, \Delta, \Pi\}$$

$$\Gamma, \underline{\Lambda} \mathbf{£} t \,\big|\, \underline{\Pi}, \Delta \overset{\text{def}}{\iff} \langle t \,\|\, \alpha\rangle : \left(\Gamma, \underline{\Lambda} \mathbf{£} \underline{\Pi}, \alpha, \Delta\right) \quad (\alpha \notin \mathbf{fv}(t))$$

$$\Gamma, \underline{\Lambda} \,\big|\, e \mathbf{£} \underline{\Pi}, \Delta \overset{\text{def}}{\iff} \langle x \,\|\, e\rangle : \left(\Gamma, x, \underline{\Lambda} \mathbf{£} \underline{\Pi}, \Delta\right) \quad (x \notin \mathbf{fv}(e))$$

$$\Gamma \mathbf{£} \underline{t} \,\big|\, \Delta \overset{\text{def}}{\iff} t \text{ is thunkable and } \mathbf{fv}(t) \subseteq \Gamma \cup \Delta$$

$$\Gamma \,\big|\, \underline{e} \mathbf{£} \Delta \overset{\text{def}}{\iff} e \text{ is thunkable and } \mathbf{fv}(e) \subseteq \Gamma \cup \Delta$$

With this definition we can take advantage of the characterisations IV.36 and IV.38. We show that we retrieve an over-approximation of the stoup of **LC** [Gir91]:

Proposition IV.46. *The predicate* $\mathbf{£}$ *satisfies the rules of Figure IV.8 on the next page.*

Proof. In the proof, we implicitly use the property that thunkable terms and linear contexts are closed under $\simeq_{\mathrm{RE}_p}$ and under substitution with linear/thunkable terms/contexts (Proposition IV.41).

Properties of identity type.

Axiom: one has $\Gamma, \underline{x^{\ominus}} \mathbf{£} x^{\ominus} \,\big|\, \underline{\varnothing}, \Delta$ because this is equivalent to $\tilde{\mu}x^{\ominus}.\langle x^{\ominus} \,\|\, \alpha^{\ominus}\rangle$ linear; yet one indeed has $\tilde{\mu}x^{\ominus}.\langle\, x^{\ominus} \,\|\, \alpha^{\ominus} \,\rangle \simeq_{\mathrm{RE}_p} \alpha^{\ominus}$, which is linear. Same reasoning for the variable α^{+}.

Activation, underlined case: by definition.

Activation, non-underlined case: for $\tilde{\mu}x.c$, one has $\langle x \,\|\, \tilde{\mu}x.c\rangle : \left(\Gamma, x, \underline{\Lambda} \mathbf{£} \underline{\Pi}, \Delta\right)$ through equivalence with hypothesis $c : \left(\Gamma, x, \underline{\Lambda} \mathbf{£} \underline{\Pi}, \Delta\right)$. Same reasoning with $\mu\alpha.c$.

Cut: assume that $\Gamma, \underline{\Lambda} \mathbf{£} t \,\big|\, \underline{\Pi}, \Delta$ and $\Gamma' \,\big|\, \underline{e} \mathbf{£} \Delta'$ (the case of a cut between $\Gamma \mathbf{£} \underline{t} \,\big|\, \Delta$ and $\Gamma', \underline{\Lambda} \,\big|\, e \mathbf{£} \underline{\Pi}, \Delta'$ is symmetric). If $\Pi = \{\alpha^{+}\}$, one has to show $\mu\alpha^{+}.\,\langle\, t \,\|\, e \,\rangle$ thunkable (the case $\Lambda = \{x^{\ominus}\}$ is identical). One has by hypothesis $\mu\alpha^{+}.\langle t \,\|\, \beta\rangle$ thunkable for $\beta \notin \mathbf{fv}(t)$ and also e linear. Besides, one has $\mu\alpha^{+}.\langle t \,\|\, e\rangle = \mu\alpha^{+}.\langle t \,\|\, \beta\rangle[e/\beta]$ since the writing of the conclusion implies

$$\frac{}{\Gamma, \underline{x^{\ominus}} \mathbin{\pounds} x^{\ominus} \mid \Delta} \qquad \frac{c : (\Gamma, \underline{x^{\ominus}} \mathbin{\pounds} \underline{\varnothing}, \Delta)}{\Gamma \mid \underline{\tilde{\mu} x^{\ominus}.c} \mathbin{\pounds} \Delta} \qquad \frac{c : (\Gamma, x, \underline{\Lambda} \mathbin{\pounds} \underline{\Pi}, \Delta)}{\Gamma, \underline{\Lambda} \mid \tilde{\mu} x.c \mathbin{\pounds} \underline{\Pi}, \Delta}$$

$$\frac{}{\Gamma \mid \alpha^{+} \mathbin{\pounds} \underline{\alpha^{+}}, \Delta} \qquad \frac{c : (\Gamma, \underline{\varnothing} \mathbin{\pounds} \underline{\alpha^{+}}, \Delta)}{\Gamma \mathbin{\pounds} \underline{\mu \alpha^{+}.c} \mid \Delta} \qquad \frac{c : (\Gamma, \underline{\Lambda} \mathbin{\pounds} \underline{\Pi}, \alpha, \Delta)}{\Gamma, \underline{\Lambda} \mathbin{\pounds} \mu \alpha.c \mid \underline{\Pi}, \Delta}$$

$$\frac{\Gamma \mathbin{\pounds} \underline{t} \mid \Delta \qquad \Gamma', \underline{\Lambda} \mid e \mathbin{\pounds} \underline{\Pi}, \Delta'}{\langle t \| e \rangle : (\Gamma \cup \Gamma', \underline{\Lambda} \mathbin{\pounds} \underline{\Pi}, \Delta \cup \Delta')} \qquad \frac{\Gamma, \underline{\Lambda} \mathbin{\pounds} t \mid \underline{\Pi}, \Delta \qquad \Gamma' \mid \underline{e} \mathbin{\pounds} \Delta'}{\langle t \| e \rangle : (\Gamma \cup \Gamma', \underline{\Lambda} \mathbin{\pounds} \underline{\Pi}, \Delta \cup \Delta')}$$

(a) Properties of identity type

$$\frac{}{\Gamma \mathbin{\pounds} \underline{V} \mid \Delta}\ \mathbf{fv}(V) \subseteq \Gamma \cup \Delta \qquad \frac{}{\Gamma \mid \underline{\pi} \mathbin{\pounds} \Delta}\ \mathbf{fv}(\pi) \subseteq \Gamma \cup \Delta$$

$$\frac{c : (\Gamma, \underline{\Lambda} \mathbin{\pounds} \underline{\Pi}, \Delta)}{c : (\Gamma, x, \underline{\Lambda} \mathbin{\pounds} \underline{\Pi}, \Delta)} \qquad \frac{c : (\Gamma, \underline{\Lambda} \mathbin{\pounds} \underline{\Pi}, \Delta)}{c : (\Gamma, \underline{\Lambda} \mathbin{\pounds} \underline{\Pi}, \Delta, \alpha)}$$

$$\frac{c : (\Gamma, x^{\varepsilon}, y^{\varepsilon}, \underline{\Lambda} \mathbin{\pounds} \underline{\Pi}, \Delta)}{c[x^{\varepsilon}/y^{\varepsilon}] : (\Gamma, x^{\varepsilon}, \underline{\Lambda} \mathbin{\pounds} \underline{\Pi}, \Delta)} \qquad \frac{c : (\Gamma, \underline{\Lambda} \mathbin{\pounds} \underline{\Pi}, \Delta, \alpha^{\varepsilon}, \beta^{\varepsilon})}{c[\alpha^{\varepsilon}/\beta^{\varepsilon}] : (\Gamma, \underline{\Lambda} \mathbin{\pounds} \underline{\Pi}, \Delta, \alpha^{\varepsilon})}$$

(b) Properties of structure type

$$\frac{\Gamma \mathbin{\pounds} \underline{t} \mid \Delta \qquad \Gamma' \mathbin{\pounds} \underline{u} \mid \Delta'}{\Gamma \cup \Gamma' \mathbin{\pounds} \underline{(t, u)} \mid \Delta \cup \Delta'} \qquad \frac{c : (\Gamma, x, y, \underline{\Lambda} \mathbin{\pounds} \underline{\Pi}, \Delta)}{\Gamma, \underline{\Lambda} \mid \tilde{\mu}(x, y).c \mathbin{\pounds} \underline{\Pi}, \Delta}$$

$$\frac{\Gamma \mathbin{\pounds} \underline{t} \mid \Delta \qquad \Gamma \mid \underline{e} \mathbin{\pounds} \Delta}{\Gamma \cup \Gamma' \mid \underline{t \cdot e} \mathbin{\pounds} \Delta \cup \Delta'} \qquad \frac{c : (\Gamma, x, \underline{\Lambda} \mathbin{\pounds} \underline{\Pi}, \Delta, \alpha)}{\Gamma, \underline{\Lambda} \mathbin{\pounds} \mu(x \cdot \alpha).c \mid \underline{\Pi}, \Delta}$$

$$\frac{\Gamma \mid \underline{e_{\ominus}} \mathbin{\pounds} \Delta}{\Gamma \mathbin{\pounds} \underline{[e_{\ominus}]} \mid \Delta} \qquad \frac{c : (\Gamma, \underline{\Lambda} \mathbin{\pounds} \underline{\Pi}, \Delta, \alpha^{\ominus})}{\Gamma, \underline{\Lambda} \mid \mu[\alpha^{\ominus}].c \mathbin{\pounds} \underline{\Pi}, \Delta}$$

(c) Property of logic type

Figure IV.8: Properties of linear and thunkable terms

assumes that $\Delta \cup \Delta' \supseteq \mathbf{fv}(e)$ and $\Pi \ni \alpha$ are disjoint. Hence the result by substitution.

Properties of structure type.

A value is thunkable and a stack is linear: see Proposition IV.41.

Weakening: trivial.

Contraction: we have to show that if $\tilde{\mu}z^{\ominus}.c$ is linear with $z \notin \{x^{\varepsilon}, y^{\varepsilon}\}$ then $\tilde{\mu}z.c[x^{\varepsilon}/y^{\varepsilon}] = (\tilde{\mu}z.c)[x^{\varepsilon}/y^{\varepsilon}]$ is too; this is immediate by substitution since x^{ε} is thunkable. The cases $\mu\gamma^{+}.c$ instead of $\tilde{\mu}z^{\ominus}.c$ and/or $\alpha^{\varepsilon}, \beta^{\varepsilon}$ instead of $x^{\varepsilon}, y^{\varepsilon}$ are identical.

Properties of logic type.

Cases (t,u), $t{\cdot}e$ and $[e^{\ominus}]$: immediate by substitution of x, y and α in (x,y) (thunkable), $x{\cdot}\alpha$ and $[\alpha]$ (linear).

Case $\tilde{\mu}(x,y).c$: Suppose $c : (\Gamma, x, y, \underline{\Lambda} \mathbin{\pounds} \underline{\Pi}, \Delta)$. If $\Pi = \{\alpha^{+}\}$, one has to show $\mu\alpha^{+}.\langle z^{+} \parallel \tilde{\mu}(x,y).c \rangle$ thunkable for $z^{+} \notin \mathbf{fv}(c)$ (the case $\Lambda = \{x^{\ominus}\}$ is similar). By hypothesis, $\mu\alpha^{+}.\langle (x,y) \parallel \tilde{\mu}(x,y).c \rangle \simeq_{\mathrm{RE}_p} \mu\alpha^{+}.c$ is thunkable. If we consider $c_1 \simeq_{\mathrm{RE}_p} c_2$ the equation of commutation that characterises the thunkability of the term, then this shows $c_1[(x,y)/z^{+}] \simeq_{\mathrm{RE}_p} c_2[(x,y)/z^{+}]$. Thus we also have $\langle z^{+} \parallel \tilde{\mu}(x,y).\langle (x,y) \parallel \tilde{\mu}z^{+}.c_1 \rangle \rangle \simeq_{\mathrm{RE}_p} \langle z^{+} \parallel \tilde{\mu}(x,y).\langle (x,y) \parallel \tilde{\mu}z^{+}.c_2 \rangle \rangle$, hence the result by extensionality.

Cases $\tilde{\mu}[\alpha^{\ominus}].c$ and $\mu(x{\cdot}\alpha).c$: same reasoning. ∎

Remark IV.47. Contrarily to [Gir91], the following properties are false:

$$\left| \underline{\widehat{\mathrm{tp}}} \mathbin{\pounds} \qquad\qquad x^{\ominus} \mathbin{\pounds} \underline{\mu\alpha^{+}.\langle x^{\ominus} \parallel \widehat{\mathrm{tp}} \rangle} \right|$$

The sensible conjecture that follows shows that the notion of stoup that we just defined cannot be captured by syntax. The idea is that a term that terminates without side-effects is equivalent to the returned value, while a term that does not return would never be thunkable; yet, termination is not recursively discriminable.

Conjecture IV.48. *The set of thunkable terms is not recursive.*

Proof sketch. Since all the negative terms are thunkable, this amounts to showing that the set of thunkable positive terms is not recursive.

Let us consider Λ the set of λ-terms (see for instance Barendregt [Bar84]) as well as:

$$\mathbf{I} = \left\{ M \in \Lambda \,\middle|\, M \simeq_{\beta\eta} \lambda x.x \right\}$$
$$\mathbf{\Omega} = \left\{ M \in \Lambda \,\middle|\, M \simeq_{\beta\eta} \omega\omega = (\lambda x.xx)(\lambda x.xx) \right\}$$

They are two sets of λ-terms closed under $\simeq_{\beta\eta}$ and disjoint, so they are recursively inseparable [Bar84].

To each λ-term M we associate a positive term $\mathrm{test}(M)$ defined with:

$$\mathrm{test}(M) \stackrel{\text{def}}{=} \mu\alpha^{+}.\langle t \,\|\, \mu\beta^{\ominus}.\langle () \,\|\, \alpha^{+}\rangle \cdot \gamma \rangle$$

where t is the negative term that corresponds to M through the encoding of Figure I.13c on page 82. There is a converse function that associates M to every positive term of the form $\mathrm{test}(M)$ which is recursive; thus the sets $\mathrm{test}(\mathbf{I})$ and $\mathrm{test}(\mathbf{\Omega})$ are recursively inseparable.

Since the encoding of Figure I.13c preserves $\simeq_{\beta\eta}$ equivalence, one has for all $M \in I$:

$$\mathrm{test}(M) \simeq_{\mathrm{RE}_p} \mu\alpha^{+}.\langle \lambda x^{\ominus}.x^{\ominus} \,\|\, \mu\beta^{\ominus}.\langle () \,\|\, \alpha^{+}\rangle \cdot \gamma \rangle \simeq_{\mathrm{RE}_p} ()$$

Thus $\mathrm{test}(M)$ is thunkable.

Conversely, it is reasonable to think that if $M \in \Lambda$ is such that $\mathrm{test}(M)$ is thunkable, then M is solvable. (The development of rewriting techniques for the calculi **L**, from which such a result would follow, is an open problem.)

We conclude by noticing that if $M \in \mathbf{\Omega}$, then $\mathrm{test}(M)$ is not thunkable, since $\omega\omega$ and therefore M are not solvable. The set of thunkable terms separates $\mathrm{test}(\mathbf{I})$ from $\mathrm{test}(\mathbf{\Omega})$, so it is not recursive. ∎

That $\mathrm{test}(M)$ is thunkable whenever $M \simeq_{\beta\eta} \lambda x.x$ shows that the semantic notion of stoup cannot be reduced to a notion of value as developed through continuation calculi. The remainder of the proof aims at showing that it can be reduced to no syntactic notion.

Bibliography

[Abr03] Samson Abramsky, *Sequentiality vs. Concurrency In Games And Logic*, Math. Struct. Comput. Sci. **13** (2003), no. 4, 531–565. 14, 40, 85, 107

[ABDM03] Mads Sig Ager, Dariusz Biernacki, Olivier Danvy, and Jan Midtgaard, *A Functional Correspondence between Evaluators and Abstract Machines*, PPDP, 2003, pp. 8–19. 84

[And92] Jean-Marc Andreoli, *Logic Programming with Focusing Proof in Linear Logic*, Journal of Logic and Computation **2** (1992), no. 3, 297–347. 14, 40, 84

[AH03] Zena M. Ariola and Hugo Herbelin, *Minimal Classical Logic and Control Operators*, ICALP '03, LNCS, vol. 2719, Springer, 2003, pp. 871–885. 243, 244

[AH08] _____, *Control Reduction Theories: the Benefit of Structural Substitution*, Journal of Functional Programming **18** (2008), no. 3, 373–419. 35, 85, 98, 240, 244

[AHS04] Zena M. Ariola, Hugo Herbelin, and Amr Sabry, *A Type-Theoretic Foundation of Continuations and Prompts*, ICFP '04, ACM, 2004, pp. 40–53. 32, 93, 154, 171, 214, 244

[AHS09] _____, *A type-theoretic foundation of delimited continuations*, Higher-Order and Symbolic Computation **22** (2009), no. 3, 233–273. 36, 99, 154, 244

[BBG+63] John W. Backus, Friedrich L. Bauer, Julien Green, C. Katz, John McCarthy, Alan J. Perlis, Heinz Rutishauser, Klaus Samelson, Bernard Vauquois, Joseph Henry Wegstein, Adriaan van Wijngaarden, Michael Woodger, and Peter Naur, *Revised report*

on the algorithm language ALGOL *60*, Commun. ACM **6** (1963), no. 1, 1–17. 93

[Bar84] Hendrik Pieter Barendregt, *The Lambda Calculus – Its Syntax and Semantics*, Studies in Logic and the Foundations of Mathematics, vol. 103, North-Holland, 1984. 271

[BW95] Nick Benton and Philip Wadler, *Linear Logic, Monads, and the Lambda Calculus*, LICS '95, IEEE Computer Society Press, 1995, pp. 420–431. 111, 155, 197

[BB09] Małgorzata Biernacka and Dariusz Biernacki, *Context-based proofs of termination for typed delimited-control operators*, PPDP, 2009, pp. 289–300. 154, 175

[BD07] Małgorzata Biernacka and Olivier Danvy, *A Syntactic Correspondence between Context-Sensitive Calculi and Abstract Machines*, no. 1-3, 76–108, Extended version available as the research report BRICS RS-06-18. 84

[BDS06] Dariusz Biernacki, Olivier Danvy, and Chung-chieh Shan, *On the Static and Dynamic Extents of Delimited Continuations*, no. 3, 274–297. 154

[Bla92] Andreas Blass, *A Game Semantics for Linear Logic*, Ann. Pure Appl. Logic **56** (1992), no. 1-3, 183–220. 85, 107

[Bru12] Aloïs Brunel, *Quantitative classical realizability*, CoRR **abs/1201.4307** (2012). 70

[CK99] Serenella Cerrito and Delia Kesner, *Pattern Matching as Cut Elimination*, LICS, 1999, pp. 98–108. 85

[CK04] _____, *Pattern matching as cut elimination*, Theor. Comput. Sci. **323** (2004), no. 1-3, 71–127. 85

[Cle06] John Clements, *Portable and high-level access to the stack with Continuation Marks*, Ph.D. thesis, Northeastern University, 2006. 35, 98, 215

[CH00] Pierre-Louis Curien and Hugo Herbelin, *The duality of computation*, ACM SIGPLAN Notices **35** (2000), 233–243. 15, 17, 30, 36, 40, 84, 91, 99, 100, 155, 162, 207, 208, 255

[CM10] Pierre-Louis Curien and Guillaume Munch-Maccagnoni, *The duality of computation under focus*, Proc. IFIP TCS, 2010, Extended version. 17, 47, 85

[DJS95] Vincent Danos, Jean-Baptiste Joinet, and Harold Schellinx, *LKQ and LKT: sequent calculi for second order logic based upon dual linear decompositions of the classical implication*, London Mathematical Society Lecture Notes **1** (1995), 222. 29, 91, 99, 100, 101

[DJS97] _____, *A New Deconstructive Logic: Linear Logic*, Journal of Symbolic Logic **62 (3)** (1997), 755–807. 26, 29, 30, 84, 88, 91, 92, 99, 100, 101, 135, 157, 161, 182, 184, 202, 243, 248, 255

[Dan00] Olivier Danvy, *Formalizing Implementation Strategies for First-Class Continuations*, ESOP, 2000, pp. 88–103. 155

[Dan08] _____, *Defunctionalized Interpreters for Programming Languages*, ICFP, 2008, Invited talk, pp. 131–142. 84

[DF90] Olivier Danvy and Andrzej Filinski, *Abstracting Control*, LISP and Functional Programming, 1990, pp. 151–160. 31, 37, 93, 94, 99, 154, 232, 239

[DL92] Olivier Danvy and Julia L. Lawall, *Back to Direct Style II: First-Class Continuations*, LISP and Functional Programming, 1992, pp. 299–310. 155

[DM08] Olivier Danvy and Kevin Millikin, *A Rational Deconstruction of Landin's SECD Machine with the J Operator*, Logical Methods in Computer Science **4** (2008), no. 4. 93

[DN05] Olivier Danvy and Lasse R. Nielsen, *CPS transformation of beta-redexes*, Inf. Process. Lett. **94** (2005), no. 5, 217–224. 232

[DP01] René David and Walter Py, *λμ-Calculus and Böhm's Theorem*, J. Symb. Log. **66** (2001), no. 1, 407–413. 238

[dG94] Philippe de Groote, *A CPS-Translation of the λμ-Calculus*, CAAP, 1994, pp. 85–99. 37, 99, 238

[Dem12] Delphine Demange, *Semantic Foundations of Intermediate Program Representations*, Ph.D. thesis, ENS Cachan - Bretagne, 2012. 94

[DDFR12] Jean-Guillaume Dumas, Dominique Duval, Laurent Fousse, and Jean-Claude Reynaud, *A duality between exceptions and states*, Mathematical Structures in Computer Science **22** (2012), no. 4, 719–722. 178

[DL06] Roy Dyckhoff and Stéphane Lengrand, *LJQ, a strongly Focused Calculus for Intuitionistic Logic*, Proceedings of the 2nd Conference on Computability in Europe (CiE'06), 2006. 85

[DL07] _____, *Call-by-Value lambda-calculus and LJQ*, J. Log. Comput. **17** (2007), no. 6, 1109–1134. 85

[ER04] Martin Erwig and Deling Ren, *Monadification of functional programs*, Sci. Comput. Program. **52** (2004), no. 1-3, 101–129. 110

[Fel88] Matthias Felleisen, *The Theory and Practice of First-Class Prompts*, POPL, 1988, pp. 180–190. 153, 154

[FFKD87] Matthias Felleisen, Daniel P. Friedman, Eugene E. Kohlbecker, and Bruce F. Duba, *A Syntactic Theory of Sequential Control*, Theor. Comput. Sci. **52** (1987), 205–237. 32, 84, 85, 95, 244

[FH92] Matthias Felleisen and Robert Hieb, *The Revised Report on the Syntactic Theories of Sequential Control and State*, Theor. Comput. Sci. **103** (1992), no. 2, 235–271. 85, 244

[FWFD88] Matthias Felleisen, Mitchell Wand, Daniel P. Friedman, and Bruce F. Duba, *Abstract Continuations: A Mathematical Semantics for Handling Full Jumps*, LISP and Functional Programming, 1988, pp. 52–62. 154

[Fil89] Andrzej Filinski, *Declarative Continuations and Categorical Duality*, Master's thesis, DIKU, Computer Science Department, University of Copenhagen, Aug 1989, DIKU Rapport 89/11. 15, 41, 100, 208

[Fil94] _____, *Representing Monads*, Proc. POPL, ACM Press, 1994, pp. 446–457. 94

[Fil96] _____, *Controlling Effects*, Ph.D. thesis, School of Computer Science, Carnegie Mellon University, Pittsburgh, Pennsylvania, May 1996, Technical Report CMU-CS-96-119. 94

[Fil99] _____, *Representing Layered Monads*, POPL, 1999, pp. 175–188. 94

[FSDF93] Cormac Flanagan, Amr Sabry, Bruce F. Duba, and Matthias Felleisen, *The Essence of Compiling with Continuations*, PLDI, 1993, pp. 237–247. 161, 255

[Fri78] Harvey Friedman, *Classically and intuitionistically provably recursive functions*, Higher Set Theory (D. S. Scott and G. H. Muller, eds.), Lecture Notes in Mathematics, vol. 699, Springer-Verlag, 1978, pp. 21–28. 36, 99

[Füh99] Carsten Führmann, *Direct Models for the Computational Lambda Calculus*, Electr. Notes Theor. Comput. Sci. **20** (1999), 245–292. 14, 25, 40, 87, 105, 107, 109, 137, 140, 152, 207, 255, 261

[Gen35] Gerhard Gentzen, *Untersuchungen über das logische Schließen*, Mathematische Zeitschrift **39** (1935), 176–210,405–431. 15, 21, 40, 49, 55, 56, 77, 287

[Gen69] Gerhard Gentzen, *The collected papers of Gerhard Gentzen*, North-Holland Pub. Co, 1969. 49, 56, 287

[Gir72] Jean-Yves Girard, *Interprétation fonctionnelle et élimination des coupures dans l'arithmétique d'ordre supérieur*, Ph.D. thesis, Université Paris VII, 1972. 16, 42, 71

[Gir87] _____, *Linear Logic*, Theoretical Computer Science **50** (1987), 1–102. 14, 15, 40, 41, 63, 70, 99, 110, 111, 155, 161, 197, 208

[Gir91] _____, *A new constructive logic: Classical logic*, Math. Struct. Comp. Sci. **1** (1991), no. 3, 255–296. 15, 26, 29, 30, 32, 33, 35, 41, 84, 88, 91, 92, 94, 95, 96, 98, 99, 100, 107, 135, 157, 161, 184, 202, 243, 254, 255, 268, 270

[Gir93] _____, *On the Unity of Logic*, Ann. Pure Appl. Logic **59** (1993), no. 3, 201–217. 137, 184, 243

[Gir01] _____, *Locus Solum: From the Rules of Logic to the Logic of Rules*, Mathematical Structures in Computer Science **11** (2001), 301–506. 15, 41, 84, 85

[Gir06] _____, *Le Point Aveugle, Cours de logique, Tome I: Vers la Perfection*, Vision des Sciences, Hermann, 2006, published subsequently in English [Gir11]. 67, 84

[Gir07] _____, *Le Point Aveugle, Cours de logique, Tome II: Vers l'imperfection*, Visions des Sciences, Hermann, 2007, published subsequently in English [Gir11]. 137

[Gir11] _____, *The Blind Spot: Lectures on Logic*, European Mathematical Society, 2011. 278

[GSS92] Jean-Yves Girard, Andre Scedrov, and Philip J. Scott, *Normal Forms and Cut-Free Proofs as Natural Transformations*, in : Logic From Computer Science, Mathematical Science Research Institute Publications 21, Springer-Verlag, 1992, pp. 217–241. 16, 42

[Gri90] Timothy G. Griffin, *A Formulae-as-Types Notion of Control*, Seventeenth Annual ACM Symposium on Principles of Programming Languages, ACM Press, 1990, pp. 47–58. 17, 32, 79, 84, 95

[HL91] Robert Harper and Mark Lillibridge, *ML with callcc is unsound*, Announcement on the TYPES electronic forum, July 1991. 14, 40

[HD94] John Hatcliff and Olivier Danvy, *A Generic Account of Continuation-Passing Styles*, POPL, 1994, pp. 458–471. 161, 255

[HD97] _____, *Thunks and the lambda-Calculus*, J. Funct. Program. **7** (1997), no. 3, 303–319. 14, 25, 40, 87, 95, 109, 175

[Her05] Hugo Herbelin, *C'est maintenant qu'on calcule, au cœur de la dualité*, 2005, Habilitation thesis. 154, 244

[Her08] _____, *Duality of computation and sequent calculus: a few more remarks*, Manuscript, 2008. 84

[Her10] _____, *An Intuitionistic Logic that Proves Markov's Principle*, LICS, 2010, pp. 50–56. 36, 94, 99

[HG08] Hugo Herbelin and Silvia Ghilezan, *An Approach to Call-by-Name Delimited Continuations*, Proc. POPL '08, ACM, 2008. 32, 36, 93, 99, 154, 155, 171, 176, 178, 180, 184, 214, 230, 238, 244

[HS02] Martin Hofmann and Thomas Streicher, *Completeness of continuation models for λμ-calculus*, Inf. Comput. **179** (2002), no. 2, 332–355. 36, 99, 228

[Ili10] Danko Ilik, *Delimited control operators prove Double-negation Shift*, Annals of Pure and Applied logic **163** (2010), no. 11. 36, 94, 99, 154

[Jac93] Bart Jacobs, *Comprehension Categories and the Semantics of Type Dependency*, Theor. Comput. Sci. **107** (1993), no. 2, 169–207. 138, 139

[KH03] Yukiyoshi Kameyama and Masahito Hasegawa, *A sound and complete axiomatization of delimited continuations*, Proc. ICFP '03, ACM, 2003. 154, 155, 171

[KT10] Yukiyoshi Kameyama and Asami Tanaka, *Equational axiomatization of call-by-name delimited control*, PPDP, 2010, pp. 77–86. 154, 177

[KS07] Oleg Kiselyov and Chung-chieh Shan, *A substructural type system for delimited continuations*, Proc. TLCA'07, 2007. 186

[Kri90] Jean-Louis Krivine, *Lambda-calcul, types et modèles*, Masson, 1990, Published in English by Ellis Horwood [Kri93]. 94

[Kri93] _____, *Lambda-calculus, types and models*, Ellis Horwood, 1993. 279

[Kri08] _____, *Structures de réalisabilité, RAM et ultrafiltre sur N*, To appear, 2008. 35, 98

[Kri09] _____, *Realizability in Classical Logic*, Panoramas et Synthèses **27** (2009), 197–229, Circulated since 2004. 14, 15, 33, 34, 35, 36, 40, 41, 62, 70, 79, 85, 95, 96, 98

[Kri12] _____, *Realizability algebras II : new models of ZF + DC*, Logical Methods in Computer Science **8** (2012), no. 1. 81

[LRS93] Yves Lafont, B. Reus, and Thomas Streicher, *Continuation Semantics or Expressing Implication by Negation*, Tech. report, University of Munich, 1993. 17, 25, 29, 79, 87, 91, 94, 99, 100

[LS86] J. Lambek and P. J. Scott, *Introduction to higher order categorical logic*, Cambridge University Press, New York, NY, USA, 1986. 214

[Lan64] Peter J. Landin, *The mechanical evaluation of expressions*, The Computer Journal **6** (1964), no. 4, 308–320. 15, 41, 84

[Lan65] _____, *A correspondence between* ALGOL *60 and Church's Lambda-notation*, Commun. ACM **8** (1965), no. 2, 89–101 and 158–165. 15, 41, 84, 93

[Lan98] _____, *A Generalization of Jumps and Labels*, Higher-Order and Symbolic Computation **11** (1998), no. 2, 125–143. 84, 93

[Lau02] Olivier Laurent, *Etude de la polarisation en logique*, Thèse de doctorat, Université Aix-Marseille II, mar 2002. 17, 22, 26, 29, 30, 42, 88, 91, 92, 99, 100, 101, 131, 135, 188, 194

[Lau05] _____, *Classical isomorphisms of types*, Mathematical Structures in Computer Science **15** (2005), no. 5, 969–1004. 99, 101

[Lau11] _____, *Intuitionistic Dual-intuitionistic Nets*, J. Log. Comput. **21** (2011), no. 4, 561–587. 99, 100

[LQTdF05] Olivier Laurent, Myriam Quatrini, and Lorenzo Tortora de Falco, *Polarized and focalized linear and classical proofs*, Ann. Pure Appl. Logic **134** (2005), no. 2-3, 217–264. 99, 100, 107, 248, 254

[LR03] Olivier Laurent and Laurent Regnier, *About Translations of Classical Logic into Polarized Linear Logic*, Proc. LICS '03, IEEE Computer Society, 2003. 99, 198

[Ler12] Xavier Leroy, *Mechanized Semantics for Compiler Verification*, CPP, 2012, pp. 4–6. 94

[Lev99] Paul Blain Levy, *Call-by-Push-Value: A Subsuming Paradigm*, Proc. TLCA '99, 1999, pp. 228–242. 26, 85, 88, 100, 101

[Lev04] _____, *Call-By-Push-Value: A Functional/Imperative Synthesis*, Semantics Structures in Computation, vol. 2, Springer, 2004. 26, 85, 88

[Lev05] _____, *Adjunction models for call-by-push-value with stacks*, Proc. Cat. Th. and Comp. Sci., ENTCS, vol. 69, 2005. 26, 85, 88

[LM07] Chuck Liang and Dale Miller, *Focusing and Polarization in Intuitionistic Logic*, CSL, 2007, pp. 451–465. 137

[Lod06] Jean-Louis Loday, *Generalized bialgebras and triples of operads*, arXiv preprint math/0611885 (2006). 24, 86, 134

[ML71] Saunders Mac Lane, *Categories for the working mathematician*, Springer-Verlag, New York, 1971, Graduate Texts in Mathematics, Vol. 5. MR MR0354798 (50 #7275) 45, 140

[MOTW94] John Maraist, Martin Odersky, David N. Turner, and Philip Wadler, *Call-by-Name, Call-by-Value, Call-by-Need, and the Linear Lambda Calculus*, Proc. MFPS '95, 1994. 111, 155

[MB11] Marek Materzok and Dariusz Biernacki, *Subtyping delimited continuations*, ICFP, 2011, pp. 81–93. 239

[MB12] _____, *A Dynamic Interpretation of the CPS Hierarchy*, APLAS, 2012, pp. 296–311. 37, 99

[Mel05] Paul-André Melliès, *Asynchronous Games 3 An Innocent Model of Linear Logic*, Electr. Notes Theor. Comput. Sci. **122** (2005), 171–192. 14, 40, 85, 107, 117, 135, 137

[Mel09] Paul-André Melliès, *Categorical semantics of linear logic*, Panoramas et Synthèses, vol. 27, ch. 1, pp. 15–215, Société Mathématique de France, 2009. 136

[Mel12] Paul-André Melliès, *Game Semantics in String Diagrams*, LICS, 2012, pp. 481–490. 136

[MT10] Paul-André Melliès and Nicolas Tabareau, *Resource modalities in tensor logic*, Ann. Pure Appl. Logic **161** (2010), no. 5, 632–653. 86, 100, 136, 137, 207

[Miq09] Alexandre Miquel, *Relating Classical Realizability and Negative Translation for Existential Witness Extraction*, TLCA, 2009, pp. 188–202. 36, 98

[Mog89] Eugenio Moggi, *Computational Lambda-Calculus and Monads*, IEEE Computer Society Press, 1989, pp. 14–23. 15, 25, 41, 85, 87, 155

[Mog91] _____, *Notions of computation and monads*, Inf. Comput. **93** (1991), no. 1, 55–92. 24

[Mun09] Guillaume Munch-Maccagnoni, *Focalisation and Classical Realisability*, Proc. CSL '09, LNCS, Springer-Verlag, 2009. 14, 17, 40, 47, 70, 85, 86

[Mun11] _____, *λ-calcul, machines et orthogonalité*, Unpublished Manuscript. GdT Logique 9-10-2011, October 2011. 47

[Mun12] _____, *Calcul L pour les séquents*, Unpublished manuscript. GdT Logique 24-10-2012, April 2012. 47

[Mur91] Chetan R. Murthy, *An Evaluation Semantics for Classical Proofs*, LICS, 1991, pp. 96–107. 33, 96

[Mur92] _____, *A Computational Analysis of Girard's Translation and LC*, LICS, IEEE Computer Society, 1992, pp. 90–101. 99, 100, 101, 218, 241

[Nip91] Tobias Nipkow, *Higher-Order Critical Pairs*, Proc. 6th IEEE Symp. Logic in Computer Science, IEEE Press, 1991, pp. 342–349. 70

[Oga02] Ichiro Ogata, *A Proof Theoretical Account of Continuation Passing Style*, CSL, 2002, pp. 490–505. 85

[Par92] Michel Parigot, *Lambda-Mu-Calculus: An Algorithmic Interpretation of Classical Natural Deduction*, LPAR, 1992, pp. 190–201. 84, 244

[Par00] _____, *On the Computational Interpretation of Negation*, CSL, 2000, pp. 472–484. 100

[Pit05] Andrew M. Pitts, *Typed Operational Reasoning*, Advanced Topics in Types and Programming Languages (B. C. Pierce, ed.), The MIT Press, 2005, pp. 245–289. 14, 40, 70

[Plo75] Gordon D. Plotkin, *Call-by-name, call-by-value and the lambda-calculus*, Theoretical Computer Science **1** (1975), no. 2, 125–159. 29, 91, 94

[PR01] David Pym and Eike Ritter, *On the semantics of classical disjunction*, Journal of pure and applied algebra **159** (2001), no. 2, 315–338. 208

[QTDF96] Myriam Quatrini and Lorenzo Tortora De Falco, *Polarisation Des Preuves Classiques Et Renversement*, Sciences, Paris t.322, Serie I, 1996. 99

[Rey72] John C. Reynolds, *Definitional Interpreters for Higher-Order Programming Languages*, Proceedings of 25th ACM National Conference (Boston, Massachusetts), 1972, Reprinted in Higher-Order and Symbolic Computation 11(4):363-397, 1998, with a foreword [Rey98], pp. 717–740. 84

[Rey93] _____, *The Discoveries of Continuations*, Lisp and Symbolic Computation **6** (1993), no. 3-4, 233–248. 93

[Rey98] _____, *Definitional Interpreters Revisited*, Higher-Order and Symbolic Computation **11** (1998), no. 4, 355–361. 283

[Sab96] Amr Sabry, *Note on axiomatizing the semantics of control operators*, Tech. Report CIS-TR-96-03, Department of Computer and Information Science, University of Oregon, 1996. 156

[Sau05] Alexis Saurin, *Separation with Streams in the Λμ-calculus*, LICS, IEEE Computer Society, 2005, pp. 356–365. 37, 99, 238

[Sel99] Peter Selinger, *Re: co-exponential question*, Message to the Category Theory mailing list, July 1999, http://permalink.gmane.org/gmane.science.mathematics.categories/1181. 25, 87

[Sel01] _____, *Control Categories and Duality: On the Categorical Semantics of the Lambda-Mu Calculus*, Math. Struct in Comp. Sci. **11** (2001), no. 2, 207–260. 25, 30, 87, 91, 100, 110, 136, 140, 155, 207, 208, 255, 261

[Sha04] Chung-Chieh Shan, *Delimited continuations in natural language: quantification and polarity sensitivity*, CW'04, 2004. 94

[Sha07] _____, *A static simulation of dynamic delimited control*, Higher-Order and Symbolic Computation **20** (2007), no. 4, 371–401. 37, 99, 154, 239

[Shi91] Olin Shivers, *Control-Flow Analysis of Higher-Order Languages or Taming Lambda*, Ph.D. thesis, School of Computer Science, Carnegie Mellon University, Pittsburgh, Pennsylvania, May 1991, Technical Report CMU-CS-91-145. 94

[Ste78] Guy L. Steele Jr., *RABBIT: A Compiler for SCHEME*, Master's thesis, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, Massachusetts, May 1978, Technical report AI-TR-474. 94

[Stø06] Kristian Støvring, *Extending the Extensional Lambda Calculus with Surjective Pairing is Conservative*, Logical Methods in Computer Science **2** (2006), no. 2. 232

[SW74] Christopher Strachey and Christopher P. Wadsworth, *Continuations: A Mathematical Semantics for Handling Full Jumps*, Technical Monograph PRG-11, Oxford University Computing Laboratory, Programming Research Group, Oxford, England, 1974, Reprinted in Higher-Order and Symbolic Computation 13(1/2):135–152, 2000, with a foreword [Wad00]. 94

[SR98] Thomas Streicher and Bernhard Reus, *Classical Logic, Continuation Semantics and Abstract Machines*, J. Funct. Program. **8** (1998), no. 6, 543–572. 17, 79, 100, 207

[SS75] Gerald J. Sussman and Guy L. Steele Jr., *Scheme: An Interpreter for Extended Lambda Calculus*, AI Memo 349, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, Massachusetts, December 1975, Reprinted in Higher-Order and Symbolic Computation 11(4):405–439, 1998, with a foreword [SS98]. 93

[SS98] _____, *The First Report on Scheme Revisited*, Higher-Order and Symbolic Computation **11** (1998), no. 4, 399–404. 285

[Thi97] Hayo Thielecke, *Categorical Structure of Continuation Passing Style*, Ph.D. thesis, University of Edinburgh, 1997. 100, 105, 136

[Wad94] Philip Wadler, *Monads and composable continuations*, Lisp Symb. Comput. **7** (1994), 39–56. 154

[Wad03] _____, *Call-by-value is dual to call-by-name*, SIGPLAN Not. **38** (2003), no. 9, 189–201. 68, 207, 255

[Wad00] Christopher P. Wadsworth, *Continuations revisited*, Higher-Order and Symbolic Computation **13** (2000), no. 1/2, 131–133. 284

[WF92] Andrew K. Wright and Matthias Felleisen, *A Syntactic Approach to Type Soundness*, Information and Computation **115** (1992), 38–94. 115

[Zei08] Noam Zeilberger, *On the unity of duality*, Ann. Pure and App. Logic **153:1** (2008). 85, 131

[Zei09] _____, *The logical basis of evaluation order*, Ph.D. thesis, Carnegie Mellon University, 2009. 85

[Zei10] _____, *Polarity and the logic of delimited continuations*, Proceedings of LICS '10, 2010. 197

List of Tables and Figures

Syntaxe et modèles d'une composition non-associative des programmes et des preuves

La thèse contribue à la compréhension de la nature, du rôle et des mécanismes de la polarisation dans les langages de programmation, en théorie de la preuve et dans les modèles catégoriels. La polarisation correspond à l'idée que la condition d'associativité de la composition peut être relâchée, comme on le montre à travers un résultat qui relie les *duploïdes*, notre modèle direct de la polarisation, aux adjonctions. En conséquence, la polarisation sous-tend de nombreux modèles du calcul, ce que l'on souligne encore en montrant comment les modèles par passage de continuation pour des opérateurs de contrôle délimité se décomposent en trois étapes fondamentales. Elle explique également des phénomènes de constructivité en théorie de la démonstration, ce que l'on illustre en donnant une interprétation selon le principe de la formule comme type à la polarisation en général et à une négation involutive en particulier.

Notre approche est basée sur une représentation interactive des démonstrations et des programmes à base de termes (*calcul* ***L***), qui met en évidence la structure des polarités. Celle-ci est basée sur la correspondance entre les machines abstraites et les calculs de séquents, et vise à synthétiser diverses directions : la modélisation du contrôle, de l'ordre d'évaluation et des effets dans les langages de programmation, la quête d'un lien entre la dualité catégorielle et les continuations, et l'approche interactive de la constructivité en théorie de la preuve. On introduit notre technique en supposant uniquement une connaissance élémentaire du λ-calcul simplement typé et de la réécriture.

Syntax and Models of a non-Associative Composition of Programs and Proofs

The thesis is a contribution to the understanding of the nature, role, and mechanisms of polarisation in programming languages, proof theory and categorical models. Polarisation corresponds to the idea that we can relax the associativity of composition, as we show by relating *duploids*, our direct model of polarisation, to adjunctions. As a consequence, polarisation underlies many models of computation, which we further show by decomposing continuation-passing-style models of delimited control in three fundamental steps. It also explains constructiveness-related phenomena in proof theory, which we illustrate by providing a formulae-as-types interpretation for polarisation in general and for an involutive negation in particular.

The cornerstone of our approach is an interactive term-based representation of proofs and programs (***L*** *calculi*) which exposes the structure of polarities. It is based on the correspondence between abstract machines and sequent calculi, and it aims at synthesising various trends: the modelling of control, evaluation order and effects in programming languages, the quest for a relationship between categorical duality and continuations, and the interactive notion of construction in proof theory. We give a gentle introduction to our approach which only assumes elementary knowledge of simply-typed λ calculus and rewriting.