On Subexponentials, Synthetic Connectives, and Multi-level Delimited Control

Chuck Liang, Dale Miller

On Subexponentials, Synthetic Connectives, and Multi-Level Delimited Control

Chuck Liang[1] and Dale Miller[2]

[1] Department of Computer Science, Hofstra University, Hempstead, NY, USA
chuck.liang at hofstra.edu
[2] Inria & LIX/Ecole Polytechnique, Palaiseau, France
dale.miller at inria.fr

Abstract. We construct a partially-ordered hierarchy of delimited control operators similar to those of the CPS hierarchy of Danvy and Filinski [5]. However, instead of relying on nested CPS translations, these operators are directly interpreted in linear logic extended with subexponentials (i.e., multiple pairs of ! and ?). We construct an independent proof theory for a fragment of this logic based on the principle of focusing. It is then shown that the new constraints placed on the permutation of cuts correspond to multiple levels of delimited control.

1 Introduction

This paper formulates a system, motivated by linear logic with multiple pairs of exponentials, with the intent of giving a Curry-Howard style basis for multiple levels of delimited control operators similar to those of Danvy and Filinski [5]. The computational interpretation of classical logic that began with Griffin [8] and Parigot [17] can already explain *undelimited* control operators such as *call/cc.* However, there is nothing in classical logic that can explain directly why the capturing of a continuation should be stopped by a delimiter. Continuation capture is reflected in classical proof theory by the phenomenon of contraction and scope extrusion, which are restricted in intuitionistic logic. The fine-grained control over the capture of continuations suggests a *combination* of classical logic with intuitionistic logic. In [9], delimitation is explained by a *transition from an intuitionistic to a non-intuitionistic mode of derivation,* which necessitates a cut-elimination strategy to deal with these transitions. However, it is known that multiple levels of delimitation can be used to block control operators from crossing programming boundaries and interfering with control operators in different modules (see [10]). We may even wish to have a partially ordered hierarchy of operators. For example, in the term $(f\ \sharp_i(g\ \sharp_j[control^j\ c. \ldots control^k c.s]))$ we can require that $control^j$ is not delimited by $\sharp_j$ but is delimited by $\sharp_i$ if i is stronger or unrelated to j. Suppose also that an external procedure is then called that contains a $control^k$ construct. We may wish to specify what, if any, part of the continuation of the calling program can be captured by this operator by designating the relationship of i and j to k.

At the proof-theoretic level, subexponential linear logic alone is not enough. Simply adding more pairs of ? and ! does not fundamentally change the cut-elimination algorithm of linear logic. We construct a stand-alone fragment of linear logic by using the

principle of focusing (focalisation) in formulating *synthetic connectives.* The existing definitions of focusing [1], [15] in linear logic are inadequate for fully exploiting the power of exponentials. Thus a new proof theory is needed.

Previous studies of Curry-Howard interpretations of classical logic using linear logic are exemplified by the systems LKT and LKQ [3]. These systems represent different focusing strategies for eliminating non-determinism in reduction (call-by-value for LKQ, by name for LKT). These uses of focusing are by now well understood and we shall not reconstruct them. What we emphasize in this paper is a different effect of subexponentials on reduction that can operate orthogonally to the elimination of non-determinism with focusing. The use of focusing in this paper is at a deeper level.

To motivate the need for subexponentials, a naive attempt at combining classical and intuitionistic logics can easily result in a collapse into the former even within the context of linear logic. The single-conclusion characterization of intuitionistic logic was inherited by linear logic. Unfortunately, this characterization leaves little *in between* intuitionistic and classical logic. The representation of intutionistic implication as $!A \multimap B$ is not modular. Since $\multimap$ is equivalent to a disjunction ($\parr$), its intuitionistic strength evaporates in the in the presence of multiple conclusions. However, multiple conclusions represent saved continuations in classical computation. Fortunately, there are also multiple conclusion characterizations of intuitionistic logic:

$$\frac{\Gamma \vdash A, A, \Delta}{\Gamma \vdash A, \Delta}\ C \qquad \frac{\Gamma \vdash \Delta}{\Gamma \vdash A, \Delta}\ W \qquad \frac{A, \Gamma \vdash B}{\Gamma \vdash A \to B, \Delta}\ IL \qquad \frac{A, \Gamma \vdash B, \Delta}{\Gamma \vdash A \to B, \Delta}\ CL$$

Structural rules are allowed and the distinction between classical and intuitionistic logics rests on the $\to$ introduction rule: the "IL" rule prevents scope extrusion since it enforces the scope of A to include only B and not also the formulas in Δ. This perspective offers a new opportunity for combining classical and intuitionistic logics. Informally, we can hope for something of the form

$$\frac{A, \Gamma \vdash B, \Delta_2}{\Gamma \vdash A \overset{2}{\to} B, \Delta_1 \Delta_2}$$

Here, the indices 1 and 2 represent different levels of *modality.* Introducing an implication at level 2 can require forgetting level 1 conclusions while keeping those at level 2 or higher. This is the kind of system that we can build with subexponentials.

2 Subexponential Linear Logic

Subexponential, or *multi-colored* linear logic was suggested by Girard and first described in [4]. Given a preordered set of indices, there is a $!_i$ for each index i with $?_i$ as it's dual: $?_i A = (!_i A^\perp)^\perp$. A $?_i$ does not need to admit contraction or weakening. However, the availability of these structural rules must respect the ordering relation: if $j \geq i$ then $?_j$ must admit all the structural rules admitted by $?_i$. This restriction is required to preserve the cut elimination procedure of linear logic. For all indices k, the usual dereliction rule is allowed for $?_k$ on the right and $!_k$ on the left. The promotion

rule is generalized as follows, which we display in two forms:

$$\frac{\vdash ?_{n_1} A^1, \ldots ?_{n_k} A^k, B}{\vdash ?_{n_1} A^1, \ldots ?_{n_k} A^k, !_j B}\ j \leq n_1, \ldots n_k \qquad \frac{!_{n_1} A^1, \ldots !_{n_k} A^k \vdash B}{!_{n_1} A^1, \ldots !_{n_k} A^k \vdash !_j B}\ j \leq n_1, \ldots n_k$$

The single sided rule is equivalent to the two-sided version with a single conclusion. The second form is closer to what we use. The promotion rule applies dually on the left-hand side since $!_j A$ on one side is equivalent to $?_j A^\perp$ on the other side.

The term *subexponential* was introduced in [15] along with a focused proof system for them. Semantically, subexponentials can be characterized as restrictions to subspaces. In phase semantics it is easily seen that $!_k$ restricts a fact to a submonoid that corresponds to k, with the ordering of submonoids determined by inclusion.

Subexponentials appear to be a simple generalization of linear logic save for one significant fact. In most proof systems for ordinary linear logic, and for classical and intuitionistic logics, weakening can be pushed to the initial rules. There is never a need to force weakening at other points in a proof and, therefore, it can be ignored. With indexed exponentials, however, *weakening cannot be pushed to the initial rules.* The sequent $!_1 A, !_2 B \vdash !_2 C$ may only be provable if $!_1 A$ is weakened away. If a proof of $!_1 A \multimap !_2 B \multimap !_2 C$ is represented by $\lambda x \lambda y.t$, then x cannot appear free in t. This represents a form of resource control: not how many times but *where* a resource can appear.

3 Extending The Focusing Principle

A central goal of this paper is to derive a refinement of classical logic from subexponential linear logic that is well-behaved and self-contained with respect to cut-elimination. In this regard, the principle of focusing remains important even though our target here is not a focused sequent calculus for cut-free proofs but a natural deduction system for writing programs with possibly many cuts. We wish to synthesize new connectives by combining linear logic connectives, but not every combination can be used. An important test for the integrity of synthetic introduction rules is *initial elimination:* that $A \vdash A$ is provable (in sequent calculus). Attempts to create synthetic connectives that ignore the focusing principle generally end in failure. For example, we may naively wish to consider $(A \otimes B)\ \& \ C$ as a ternary connective with the following introduction rules:

$$\frac{\Gamma_1 \vdash A \quad \Gamma_2 \vdash B \quad \Gamma_1 \Gamma_2 \vdash C}{\Gamma_1 \Gamma_2 \vdash (A \otimes B)\ \& \ C}\ \&R$$

$$\frac{A, B, \Gamma \vdash \mathcal{D}}{(A \otimes B)\ \& \ C, \Gamma \vdash \mathcal{D}}\ \&L_1 \qquad \frac{C, \Gamma \vdash \mathcal{D}}{(A \otimes B)\ \& \ C, \Gamma \vdash \mathcal{D}}\ \&L_2$$

Here, $\otimes$ is positive but $\&$ is negative. But it is easily seen that

$$(A \otimes B)\ \& \ C \vdash (A \otimes B)\ \& \ C$$

is not provable with *these* introduction rules. In contrast, connectives of the same polarity can be combined to form synthetic connectives (e.g. $(A \otimes B) \oplus C$). While this principle is clear with regard to the binary connectives, with exponentials there is more

flexibility. In terms of focusing, ? builds a negative formula and ! builds a positive formula because they have the following properties (with implied duals for !) with respect to the negative (dually positive) binary connectives:

$$?(?A \parr ?B) \equiv ?A \parr ?B \qquad ?(?A \,\&\, ?B) \equiv ?A \,\&\, ?B$$

These properties are not ordinary logical equivalences: they express an important property in the structure of proofs, e.g., *a contraction on* $?(?A \parr ?B)$ *can be recursively replaced by contraction on its immediate subformulas.* This is the basis of focusing in classical logic. To explain focusing in intuitionistic logic in terms of linear logic [12], we also have the following property: $!(!A \multimap B) \equiv !(!A \multimap !B)$. Intuitionistic implication is usually translated as $!A \multimap B$, but it is better to regard it as $!(!A \multimap B)$: the outer ! is excluded because promotion on the right is always possible in simulations of LJ sequents. With subexponentials, however, this promotion may not always be possible. One can regard focusing proof systems as the application of these properties in one direction: removing as many exponentials as possible. Our usage of them is rather in the other direction: *adding more exponentials harmlessly.* A contraction-enabled ? must be present for a programmer to use *call/cc* or similar control operations.

In our system it is of crucial importance to identify the conditions under which the following properties are preserved in the presence of subexponentials:

- $?_{i'}(?_k A \parr ?_{j'} B) \equiv ?_k A \parr ?_{j'} B$ if and only if $i' \leq k$ and $i' \leq j'$
- $!_i(!_k A \multimap !_j B) \equiv !_i(!_k A \multimap B)$ if and only if $j \leq k$ and $j \leq i$.

The index names are chosen to correspond to formulas in Section 4. A consideration of the cut-free proofs of these properties shows that the restrictions on indices are *necessary and sufficient.* Although our use of subexponentials will be effusive, we only admit formulas that satisfy these restrictions.

Proof theoretically, the core of this paper is a rather bold proposition: *a formula of the form* $!_i?_{i'}(!_k?_{k'}A \multimap !_j?_{j'}B)$*, provided that* $i' \leq k, j'$ *and* $j \leq i, k$*, forms a valid synthetic connective.* Clearly it cannot be seen as such according to the current definitions of focusing and the polarities of ! and ?. The consequence of this boldness is that we can no longer rely on the proof theory of (subexponential) linear logic. Although we use linear logic formulas, cut-elimination within our system cannot inherit the cut-elimination procedure of linear logic (the cut-free proofs will not stay within the fragment). What's needed is a new, unique proof theory that is suitable to this new sense of synthetic connective, with its own procedure for cut-elimination that is sensitive to the index restrictions on formulas. The proof theory, presented in a longer version of this paper [13], allows our system to stand independently of linear logic.

4 The Fragment MC: Multi-Colored Classical Logic

For our purpose, all subexponentials $?_i$ admit both weakening and contraction (on the right). The aspect of linear logic that prevents resources from being reused is an orthogonal issue. For clarity, we assume that subexponential indices form a partial order, although some of our examples will simply use natural numbers as indices. We also

assume the existence of finite joins and meets and that there is a maximum index mx and a minimum 0. We write $min(a, b)$ for meets and $max(a, b)$ for joins.

With a single pair of exponentials there are seven equivalence classes of exponential prefixes or *modalities:* ?, !, ?!, !?, ?!?, !?! and the empty prefix. This property extends to any pair of subexponentials $!_i$ and $?_k$. For any prefix ν, $\nu\nu \equiv \nu$. For example, $!_i?_k!_i?_kA \equiv !_i?_kA$. Most studies of linear logic consider only a few of these modalities. The LC fragment of linear logic [7], for example, uses only ! and ? although ?! and !? may appear applied to atoms.

Since our main connective is implication and we wish to capture (at least) classical computations, let us review how classical implication can be represented in linear logic. The most straightforward translation is $!A \multimap ?B$ (equivalently $?A^{\perp} \parr ?B$). This representation is sufficient for cut-free proofs, but for proofs with cuts it is clearly inadequate: one cannot form a cut with a right-side $?B$ and a left-side $!B$. In the terminology of [3], we require an *adequate inductive decoration.* Two well-known ones are the T-translation: $!?A \multimap ?B$, and the Q-translation $!A \multimap ?!B$. In each case one modality is a suffix of the other. With subexponentials however, we need a more flexible way to switch between left and right modalities because a promotion of $?_kB$ to $!_i?_kB$ may not always be possible.

The modalities we use will be $!_i?_k$ and $?_k!_i?_k$ (for every pair of i and k). Note that since $!_i?_k!_i?_kA \equiv !_i?_kA$, *each modality can be seen as a suffix of the other.* Promotion and dereliction will be restricted to forms that render them *inverse operations:*

$$\frac{!_i?_kA}{?_k!_i?_kA}\ derelict \qquad \frac{?_k!_i?_kA}{!_i?_kA}\ promote$$

Adding a $!_i$ or $?_k$ before $!_i?_k$ or $?_k!_i?_k$ will still result in something equivalent to one of the two forms. We only use equivalence classes of modalities and never write $!_i?_k!_i?_k$.

The T and Q translations gave rise to LKT and LKQ respectively, which are semi-focused sequent calculi, and cut-elimination for these systems distinguish between a call-by-name (T) and call-by-value (Q) strategies. On the surface, our system does not have such properties because we rely on *logical equivalences,* such as between $!_i?_kA$ and $!_i?_k!_i?_kA$, and between $?A \parr ?B$ and $?(?A \parr ?B)$. The correspondence between our system and (subexponential) linear logic exists only at the level of provability, but not in the structure of cut-elimination and cut-free proofs. However, our "fragment" of linear logic in fact forms a logical system in its own right, with its own notion of what *"cut free proof"* means. As we show in the longer version of this paper [13], cut-elimination within this fragment requires a more delicate procedure than for unrestricted subexponential linear logic. The non-determinism eliminated by polarization and focusing can also be achieved here, but that is not the purpose of this paper. Rather it is to demonstrate the effect on cut permutation of something quite different from polarity information.

Only the form $!_i?_k$ may appear on the left side of $\multimap$ (left-side of sequents), thus we effectively use three modalities for each pair of indices i and k ($?_i!_k$ as well).

As already indicated, our "arrow" appears as follows:

$$!_i?_{i'}(!_k?_{k'}A \multimap !_j?_{j'}B) \text{ or } ?_{i'}!_i?_{i'}(!_k?_{k'}A \multimap !_j?_{j'}B) \quad \textbf{such that } i' \leq k, j' \textbf{ and } j \leq k, i$$

The index restrictions are not ad-hoc: they are motivated by focusing. Under these restrictions, we can show that the above formula is equivalent to several other forms:

$$!_i?_{i'}(!_k?_{k'}A \multimap !_j?_{j'}B) \;\equiv\; !_i(!_k?_{k'}A \multimap !_j?_{j'}B) \;\equiv\; !_i(!_k?_{k'}A \multimap ?_{j'}B)$$

The first condition, $i' \leq k, j'$, makes $?_{i'}$ gratuitous and allows us to write a synthetic introduction rule as long as the rule does not break apart $!_k?_{k'}$ in the premise. The second condition, $j \leq k, i$, means that a conclusion $?_{j'}!_j?_{j'}B$ *can always be promoted to* $!_j?_{j'}B$ (thus it does not matter if we write $!_j?_{j'}B$ or $?_{j'}!_j?_{j'}B$ on the right of $\multimap$). All of these conditions are required for cut elimination.

The following also hold: $!_j?_{j'}(!_k?_{k'}A \multimap !_m?_{m'}B) \;\multimap\; (!_k?_{k'}A \multimap !_m?_{m'}B)$ (to construct a proof bottom-up, first promote $!_m$, then derelict $!_j$, then promote $?_{j'}$). This will allow us to form an adequate $\rightarrow$-elimination rule.

For this presentation we restrict to the adequately decorated arrow as our only connective, although other connectives can be added along similar lines[3]. It is also possible to interpret sequents as formulas: a (two-sided) sequent such as $\Gamma, !_k?_{k'}A \vdash !_j?_{j'}B$ can be interpreted recursively as $\Gamma \vdash !_{mx}?_{min(k,j')}(!_k?_{k'}A \multimap !_j?_{j'}B)$. This being overly cumbersome, we simplify the interpretation of sequents by also using the empty modality and the connective $\parr$, but only for sequents. To formally define the language of MC, we use the following grammar:

$$\begin{array}{l}
S \;\longrightarrow\; F \mid F_1^{\perp} \mid F_1^{\perp} \parr S \mid F_2 \parr S \\
F \;\longrightarrow\; F_1 \mid F_2 \\
F_1 \;\longrightarrow\; !_i?_{i'}C. \qquad F_2 \;\longrightarrow\; ?_{i'}!_i?_{i'}C \\
C \;\longrightarrow\; (!_k?_{k'}C \multimap !_j?_{j'}C) \mid p
\end{array}$$

The syntactic variable p ranges over atomic formulas; S ranges over formulas that represent sequents; F ranges over formulas preceded by the modalities $!_i?_{i'}$ or $?_{i'}!_i?_{i'}$ for any i, i'; and C ranges over formulas that are not preceded by these modalities. It is also required that the restrictions on implication $i' \leq k, j'$ and $j \leq k, i$ are imposed recursively on $!_i?_{i'}C$ and on all $?_{i'}!_i?_{i'}C$. Furthermore, *all end-sequents are of the form* $\vdash F$. When we speak of a *formula of the MC fragment,* we are referring to a F-generated formula of the above grammar, since such formulas form end-sequents. Notice that the grammar generates at most one F_1 formula in a sequent, which means there is at most one $!_i?_{i'}C$ on the right-hand side (left side occurrences are represented by $F_1^{\perp}$). This is the only kind of sequent that will appear in proofs starting from valid end-sequents.

Informally speaking, the index of an exponential operator indicates a *resource class.* A proof of $!_kA$ can only contain resources (free variables) of class k or higher. One useful analogy is that $?$ represents a *producer* and $!$ represents a *consumer.* A formula prefixed by $!_i?_kA$ or $?_k!_i?_kA$ has both a *consumer level* i and a *producer level* k. The special forms of promotion and dereliction switch the formula between the producer and consumer modes. For example, a level-2 consumer will not consume level-1 products. As long as $?_k$ hides behind $!_i$, it does not affect what can be consumed. But once

[3] Adding second-order quantifiers will, however, encounter problems similar to those found in polarized settings: how can one enforce the index restrictions on the bound variable when they are instantiated. First-order $\forall$ can be represented by $!_i?_{i'}\forall x.!_k?_{k'}A$ where $i' \leq k'$ and $k \leq i$. These restrictions guarantee $?_{i'}\forall x.?_{k'}A \;\equiv\; \forall x.?_{k'}A$ and $!_i\forall x.!_kA \;\equiv\; !_i\forall x.A$.

revealed, it produces a resource, which can only be used by consumers at level k or below, i.e., appear in subproofs outside of the context of a promotion to a level higher than k. A producer has the ability to replicate itself (contraction). On the left hand side of $\multimap$, $!_i?_k$ becomes a level i producer.

For example, the equivalence between $!_i(!_k A \multimap !_j B)$ and $!_i(!_k A \multimap B)$ when $j \leq k, i$ can be understood using this analogy as follows. In a term $\lambda x.t$ representing the proof of a formula such as $!_3(!_4 A \multimap !_2 B)$, the outer $!_3$ states that no products lower than level-3 will be consumed by $\lambda x.t$. The only new product in t is the variable x, which is at level 4. So of course stating that t does not use products lower than level 2 is redundant.

5 Natural Deduction and Proof Terms in MC

In the following we adopt the convenient notations $!_{ij} = !_{i,j} = !_i?_j$ and $?_{jij} = ?_{j,i,j} = ?_j!_i?_j$. So $?_{jij}A$ promotes to $!_{ij}A$, for example. We revert to the unabbreviated forms for clarity when needed.

Single sided sequents suffice for classical linear logic. However, since our principal connective is an arrow, using one-sided sequents will appear awkward. Thus we shall make a small concession to Gentzen style systems and use two sided sequents with at most one formula on the right. This means that instead of $\Gamma \vdash ?_{iki}A, ?_{jmj}B$ we write $!_j?_m!_j B^{\perp}, \Gamma \vdash ?_{iki}A$. This concession is superficial since negation in linear logic is involutive. The interpretation of a sequent $A_1, A_2, \ldots, A_n \vdash B$ is the formula $A_1^{\perp} \parr A_2^{\perp}, \ldots, \parr A_n^{\perp} \parr B$, or just $A_1^{\perp} \parr A_2^{\perp}, \ldots, \parr A_n^{\perp}$ if the right side is empty. The concession is only one of *notation.* In this two-sided scenario the modalities that can appear to the right of $\vdash$ are $!_i?_k$ and $?_k!_i?_k$ while those that can appear on the left are $!_i?_k$ and $!_k?_i!_k$. We store multiple conclusions on the left. Since contractions are allowed, the left-hand side of a sequent can be considered a set. The notation B, Γ does not preclude the possibility that $B \in \Gamma$. Certain rules, such as explicit weakening and the renaming of variable indices are excluded from the system because they can be shown to be admissible. The set Γ consists of formulas $!_{n_1}A_1, \ldots !_{n_m}A_m$. We write $\Gamma^{(k)}$ to indicate that k is the smallest n_i. In other words, a promotion (on the right) to $!_m$ is allowed by $\Gamma^{(k)}$ if $m \leq k$. We write $\Gamma^{(n_1,\ldots,n_k)}$ to mean $\Gamma^{(min(n_1,\ldots,n_k))}$. We refer to n and m as the *maximum promotion level* of their contexts. The empty context has maximum promotion level mx. The natural deduction style proof system for MC, along with their term annotations, appear in Figure 1.

We prefer natural deduction to present delimited controls operators in direct style. We refer to the implication introduction and elimination rules as Abs and App. Because of the index restrictions already imposed on formulas, no further conditions are required in App. In addition to the $Produce$ rule, promotion has been folded into App and, implicitly, the Id rule. $Produce$ does not cross $!_i?_k$ but switches the formula from consumer to producer mode going upward. In each rule that requires promotion, implicit or otherwise, weakening is also folded in. The dereliction rule $Consume$ can also be simulated by $Name$ followed immediately by $Unname$. However, we have included it as a separate rule for convenience. The $!DR$ rule is only needed for atoms. The restrictions on implication means that the $?_{i'}$ in $!_i?_{i'}(!_k?_{k'}A \multimap !_j?_{j'}B)$ does not interfere with its introduction rule (i.e., it does not destroy focus, despite appearance).

$$\frac{u : \Gamma \vdash !_i?_k A}{[d]u : (!_k?_i!_k A^{\perp})^d, \Gamma \vdash}\ Name \qquad \frac{t :\ (!_k?_i!_k A^{\perp})^d, \Gamma \vdash}{\mu^k d.t : \Gamma \vdash ?_k!_i?_k A}\ Unname$$

$$\frac{f : \Gamma_1^{(n)} \vdash !_i?_{i'}(!_k?_{k'}A \multimap !_j?_{j'}B) \qquad t : \Gamma_2^{(m)} \vdash !_k?_{k'}A}{(f\ \sharp_{n'} t) : \Gamma_1^{(n)}\Gamma_2^{(m)} \vdash !_j?_{j'}B}\ App,\ n' \leq min(n, j')$$

$$\frac{t :\ !_k?_{k'}A^x, \Gamma^{(n)} \vdash !_j?_{j'}B}{\lambda x.t : \Gamma'\Gamma^{(n)} \vdash !_i?_{i'}(!_k?_{k'}A \multimap !_j?_{j'}B)}\ Abs,\ i \leq n$$

$$\frac{t : \Gamma^{(n)} \vdash ?_i!_k?_i A}{t : \Gamma'\Gamma^{(n)} \vdash !_k?_i A}\ Produce\ (k \leq n) \qquad \frac{t : \Gamma \vdash !_j?_i A}{t\ (\equiv \mu^i d.[d]t) :\ \Gamma \vdash ?_i!_j?_i A}\ Consume$$

$$\frac{}{x :!_i?_j C^x, \Gamma \vdash !_i?_j C}\ Id \qquad \frac{t : \Gamma \vdash !_a?_b q}{t : \Gamma \vdash !_c?_d q}\ !DR\ (a \geq c,\ b \leq d,\ q \text{ atomic})$$

Fig. 1. Natural Deduction in MC

However, $!_a?_b q$ when q is an atom posses a slight problem. This is the only rule that violates focusing boundaries, but it is required for completeness, by which we mean the following (see [13] for proof):

Theorem 1. *A formula of the MC fragment is provable by natural deduction in MC if and only if it is provable in subexponential linear logic.*

The correctness of MC depends on not just this theorem but on proof theoretic results for its sequent calculus, with the most important difference being the restriction of Id to atoms C, plus the replacement of App with a left introduction rule:

$$\frac{\Gamma^{(n)} \vdash !_k?_{k'}A \qquad !_j?_{j'}B, \Gamma^{(n)} \vdash \mathcal{C}}{!_i?_{i'}(!_k?_{k'}A \multimap !_j?_{j'}B), \Gamma^{(n)} \vdash \mathcal{C}}$$

Cut elimination in this system requires all the index restrictions on formulas to succeed. Furthermore, one can check that instances of this rule can be *stacked,* i.e., focused (consider B to be another implication), by verifying the initial elimination test for synthetic connectives. These results are presented in detail in [13].

Despite the index restrictions on implication formulas, it is possible to have Γ contain formulas preceded by any $!_i$. For example, $!_4?_0(!_5?_1 A \multimap !_3?_2(!_8?_6 B \multimap !_2?_2 C))$ is a legal formula. Observe that in the sequent

$$\vdash\ !_i?_{i'}(!_k?_{k'}A \multimap !_j?_{j'}(!_m?_{m'}C \multimap !_r?_{r'}D)),$$

the restriction $j \leq k, i$ means that the second, inner $\multimap$ can be introduced immediately above the introduction of the first, because the addition of $!_k?_{k'}A$ to the left-side context will not prevent the promotion of $!_j$. Thus instances of Abs can always be stacked, i.e., focused as well. Thus focusing is possible should one pursue the extension of the proof system in that direction.

The proof terms here are referred to as *bounded $\lambda\mu$-terms* because we have adopted several aspects of the original $\lambda\mu$ calculus as presented in [17]. First, we prefer to associate the proof term with the entire subproof, and not just the single formula on the right

of a sequent. Secondly, Parigot referred to $[d]t$ as a *named term,* which is then unnamed, or bound, by μ. If one wishes to make sense of $\lambda\mu$ calculus in terms of intuitionistic logic, then μ must be considered a non-logical constant of type $((A \to \bot) \to \bot) \to A$, which would of course require a double negation/CPS translation to become intuitionistically admissible. Under this interpretation, $[d]t$ is just $(d\ t)$ where d is of type $A \to \bot$. This means that the answer type of a captured continuation can only be $\bot$, which is a problem if the continuation is to be used as a procedure. In a logic with involutive negation, such an interpretation becomes unnecessary. The only meaningful operation that $Name$ embeds is a dereliction[4]. We have adopted the strategy that neither promotion nor dereliction are reflected in proof terms: they do not appear to serve any purpose. The only extra notation we add are the *bounded μ^k binder,* superscripted by the producer class/level of its type, and the *bounded reset indicator* $\sharp_n$, which is used to decorate every application term, with its subscript index indicating whether this continuation may be captured by some μ^k. Unlike in other formulations of delimited control operators, $\sharp_n$ is not an independent operator but a form of type annotation.

To illustrate the system we show a generalized proof of a Peirce-like formula:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{!_{b'}?_b!_{b'}Q^\bot, !_{aa'}P \vdash !_{aa'}P}\ Name}{!_{b'}?_b!_{b'}Q^\bot, !_{a'}?_a!_{a'}P^\bot, !_{aa'}P \vdash}\ Unname}{!_{a'}?_a!_{a'}P^\bot, !_{aa'}P \vdash ?_{b'}!_b?_{b'}Q}\ Produce,\ b \le a, a'}{!_{a'}?_a!_{a'}P^\bot, !_{aa'}P \vdash !_{bb'}Q}\ Abs,\ k \le a'}{\ldots \vdash !_{jj'}(!_{kk'}(!_{aa'}P \multimap !_{bb'}Q) \multimap !_{aa'}P) \quad !_{a'}?_a!_{a'}P^\bot \vdash !_{kk'}(!_{aa'}P \multimap !_{bb'}Q)}\ App}{!_{a'}?_a!_{a'}P^\bot, !_{jj'}(!_{kk'}(!_{aa'}P \multimap !_{bb'}Q) \multimap !_{aa'}P) \vdash !_{aa'}P}\ Name - Unname}{!_{jj'}(!_{kk'}(!_{aa'}P \multimap !_{bb'}Q) \multimap !_{aa'}P) \vdash ?_{a',a,a'}P}\ Produce,\ a \le j\ (automatic)}{!_{jj'}(!_{kk'}(!_{aa'}P \multimap !_{bb'}Q) \multimap !_{aa'}P) \vdash !_{aa'}P}\ Abs,\ i \le max}{\vdash !_{ii'}(!_{jj'}(!_{kk'}(!_{aa'}P \multimap !_{bb'}Q) \multimap !_{aa'}P) \multimap !_{aa'}P)}$$

There are only a few restrictions in addition to those already imposed on formulas ($k \le a'$ and $b \le a'$). One can easily choose indices that would make this proof valid, including using the same index everywhere. The proof term is $\lambda x.\mu^{a'} d.[d](x\ \sharp_m(\lambda y.\mu^{b'} e.[d]y))$ where $m \le min(j, a')$. As we shall show, the $\mu^{a'}$ term is guaranteed to be able to catch continuations up to the nearest $\sharp_n$ with $n \not\ge a'$. In a linear ordering, if $n = k - 1$, for example, then capturing the continuation beyond $\sharp_n$ would mean that the promotion to $!_k$ (as part of the upper $Produce$ rule) cannot be duplicated in the proof.

6 Intuitionistic Logic in MC

The resource control aspect of MC generalizes intuitionistic logic. Restrict all formulas to use only the modalities $!_2?_1$ and $?_1!_2?_1$. Then any copies of $?_1!_2?_1$ must be weakened away before an implication can be introduced with the Abs rule. This corresponds to the multiple-conclusion version of intuitionistic sequent calculus, at least when restricted to the $\to$ fragment. Indeed that proof system shares the rare property with subexponential

[4] When restricted to the modalities $!_i?_{i'}$ and $?_{i'}!_i?_{i'}$, dereliction can be expressed by the axiom $!A \to A$: this is an intuitionistic implication, which surely has proof $\lambda x.x$.

linear logic that weakening is forced beneath initial rules. Peirce's formula, $((P \to Q) \to P) \to P$, cannot be proved using only $!_{ik}$ and $?_{kik}$ with $i \not\leq k$:

$$!_{21}(!_{21}(!_{21}(!_{21}P \multimap !_{21}Q) \multimap !_{21}P) \multimap !_{21}P)$$

Although promotion is applicable (backwards) on $!_{21}P$, any copy of P created by contraction, which will appear as $?_1!_2?_1P$ (or $!_1?_2!_1P^{\perp}$ on the left), must be weakened away upon introduction of $!_{21}(!_{21}P \multimap !_{21}Q)$. The only producers that do not labor in vain are those on the left side of (an odd number of occurrences of) $\multimap$. In terms of the generalized proof of Peirce's formula shown earlier, the condition $k \leq a'$ would fail. Contracting the entire formula first will similarly fail.

It is simple to verify that any formula where the maximum producer level is lower than the minimum consumer level can only have an intuitionistic proof. From this perspective, classical and intuitionistic logics are at opposite ends of a spectrum. Intuitionistic logic allows no scope extrusion, whereas classical logic make no such restrictions. With the appropriate indexing scheme in MC, we can choose to extrude into the scope of some $\to$ while keeping others intact. MC represents a new way to combine these logics without one collapsing into the other.

7 Reductions for Bounded $\lambda\mu$

When proofs are combined by cut, the potential danger of merging two contexts $\Gamma_1^{(n)}$ and $\Gamma_2^{(m)}$ into $\Gamma_1\Gamma_2^{(n,m)}$ is that the lowering of the maximum promotion level will mean that some promotions can no longer be duplicated. To determine what remains as valid reduction strategies, first we note the following, which is easily proved:

Lemma 2. *If $\Gamma^{(n)} \vdash !_i?_kA$ is provable without weakening (all formulas in Γ appear as free variables in the proof term), then $i \leq n$.*

Using this property, first we verify that β-reduction is still a valid strategy:

$$\frac{\dfrac{s : !_{uu'}A^x, \Gamma_1^{(n,u)} \vdash !_{rr'}B}{\lambda x.s : \Gamma_1^{(n)} \vdash !_{vv'}(!_{uu'}A \multimap !_{rr'}B)}\ v \leq n \qquad t : \Gamma_2^{(m)} \vdash !_{uu'}A}{s[t/x] : \Gamma_1\Gamma_2^{(n,m)} \vdash !_{rr'}B}$$

We are only concerned with those branches of the left subproof where $!_{uu'}A$ persists (has not been weakened away). Clearly in these branches there cannot be any promotion higher than to $!_u$. But by Lemma 2, the right subproof either ends in weakening, which permutes easily with cut, or it holds that $u \leq m$, and thus $min(n, m)$ is not lower than $min(n, u)$, which means these promotions can still occur after $\Gamma_2^{(m)}$ has been added to the left subproof. Thus **β-reduction is still a valid strategy**.

The restriction $n' \leq min(n, j')$ on the *App* rule can be strengthened to $n' \leq min(i, j')$ since $i \leq n$ by Lemma 2.

Next, we examine the possibility of capturing the continuation in the style of the original $\lambda\mu$ calculus. In order to not clash with β-reduction, the original $\lambda\mu$ calculus

only allowed the continuation to the right of μ to be captured, by which we mean the following scenario:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{w : \Gamma' \vdash !_{vv'}(!_{uu'}A \multimap !_{rr'}B)}{[d]w : \ (!_{v'vv'}(!_{uu'}A \multimap !_{rr'}B)^{\perp})^{d}, \Gamma' \vdash}}{\vdots}}{s : \ (!_{v'vv'}(!_{uu'}A \multimap !_{rr'}B)^{\perp})^{d}, \Gamma_1^{(n)} \vdash}}{\mu^{v'}d.s : \Gamma_1^{(n)} \vdash ?_{v'vv'}(!_{uu'}A \multimap !_{rr'}B)}}{\Gamma_1^{(n)} \vdash !_{vv'}(!_{uu'}A \multimap !_{rr'}B)} \ v \leq n \qquad t : \Gamma_2^{(m)} \vdash !_{uu'}A}{\mu^{v'}d.s\{[d]wt/[d]w\} : \Gamma_1\Gamma_2^{(n,m)} \vdash !_{rr'}B}$$

Again relevant are the branches of the left subproof that contains $!_{v'vv'}(!_{uu'}A \multimap !_{rr'}B)^{\perp}$, which means that there can only be promotions up to level $!_{v'}$. But $v' \leq u$ is required of well-formed formulas and by Lemma 2, $u \leq m$. So once again substituting $\Gamma_2^{(m)}$ into the left subproof will not prevent any promotions from being duplicated. **This type of continuation capture is also valid.**

However, it was soon recognized (e.g., [16]) that continuation capture need not clash with β-reduction as long as we define a (call-by-value) reduction strategy carefully. The resulting form of continuation capture can be generalized to the capture of an entire evaluation context:

$$\dfrac{f : \Gamma_1^{(n)} \vdash !_{vv'}(!_{uu'}A \multimap !_{rr'}B) \qquad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{t : \Gamma' \vdash !_{uu'}A}{[d]t : \ (!_{u'uu'}A^{\perp})^{d}, \Gamma' \vdash}}{\vdots}}{s : \ (!_{u'uu'}A^{\perp})^{d}, \Gamma_2^{(m)} \vdash}}{\mu^{u'}d.s : \Gamma_2^{(m)} \vdash ?_{u'uu'}A}}{\mu^{u'}d.s : \Gamma_2^{(m)} \vdash !_{uu'}A} \ u \leq m}{(f \ \sharp_k \ \mu^{u'}d.s) : \Gamma_1\Gamma_2^{(n,m)} \vdash !_{rr'}B}$$

In order to permute the cut with f into the right subproof, we need to be able to retain promotions up to level $!_{u'}$. Unlike the two previous cases, however, now it would possible for this condition to be violated if n or r' is smaller than u' (see the proof of Peirce's formula). Thus here we mark the redex with $\sharp_k$ where $k \leq min(n, r')$ (or $k \leq min(v, r')$). The continuation f can be captured by $\mu^{u'}$ only if $min(n, r') \geq u'$. We allow k to be less than $min(n, r')$ because we may decide to force β-reduction anyway. We can reserve the minimum index 0 for this purpose, and require $k > 0$ in all terms $\mu^k d.t$. Then $(f \ \sharp_0 s)$ will always force β-reduction: we can just write $(f \ s)$ in that

case. If capturing f is legal, then the resulting proof can have the following form:

$$\dfrac{\dfrac{\dfrac{\dfrac{(f\ \sharp_k t) : \Gamma_1^{(n)}\Gamma' \vdash !_{rr'}B}{!_{r'rr'}B^{\perp}, \Gamma_1^{(n)}\Gamma' \vdash}\ Consume}{\vdots}}{\dfrac{!_{r'rr'}B^{\perp}, \Gamma_1\Gamma_2^{(n,m)} \vdash}{\Gamma_1\Gamma_2^{(n,m)} \vdash ?_{r'rr'}B}\ null}}{s\{(f\ \sharp_k t)/[d]t\} : \Gamma_1\Gamma_2^{(n,m)} \vdash !_{rr'}B}\ r \leq m, n$$

Recall that merely moving a formula from one side of the sequent to the other is a null operation in linear logic due to involutive negation. It is important to note that the final promotion from $?_{r'}!_r?_{r'}B$ to $!_r?_{r'}B$ is always valid because it is required of legal formulas that $r \leq u, v$ in $!_{vv'}(!_{uu'}A \multimap !_{rr'}B)$. But by Lemma 2, $v \leq n$ and $u \leq m$, and thus $r \leq n, m$ which makes the promotion valid. This is a consequence of the focusing-related property $!(!A \multimap !B) \equiv !(!A \multimap B)$ when generalized to subexponentials.

8 A Call-by-Value Reduction Strategy

In a term such as $(f\ \sharp_2(g\ \sharp_5(h\ \sharp_4\ \mu^3 d.s)))$, μ^3 should be able to capture both h and g but not f. To formalize an evaluation strategy, we define the following.

Terms and Values:

λ-variables $x, \ldots$ and μ-variables $d, \ldots$
Values $V \longrightarrow x \mid \lambda x.T$
Terms $T \longrightarrow V \mid (T_1\ \sharp_i T_2) \mid \mu^k d.T \mid [d]T$

Evaluation Contexts:

$F^k \longrightarrow [\,] \mid (F^k\ \sharp_n T) \mid (V\ \sharp_j F^k)\ \ (j \geq k)$
$E\ \ \longrightarrow [\,] \mid (V\ \sharp_m E) \mid (E\ \sharp_n T)$

E is an evaluation context while F^k is a *level-k* context that represents a continuation that be captured by $\mu^k d.t$ in the "hole" of the context. Note that in the definition of F^k there is no restriction on the index n, because *forward* capture is always allowed. The rules for F^k implies that terms such as $(\mu^m d.s)\ \sharp_i(\mu^k d.t)$ will have the form $F^m[\mu^m d.s]$ where $F^m = [\,]\sharp_i(\mu^k d.t)$ since μ-terms are not values: the μ^k term will be part of the context captured by μ^m regardless of whether $i \geq k$.

Evaluation Rules:

$$\begin{array}{lcl} E[(\lambda x.t)\ \sharp_n V)] & \longrightarrow & E[t\{V/x\}] \\ E[V\ \sharp_i F^k[\mu^k d.t]] & \longrightarrow & E[V\ \sharp_i\ t\{F^k[u]/[d]u\}]\ \ (i \not\geq k) \end{array}$$

A term of the form $(\lambda x.u)\ \sharp_n V$ or $(V\ \sharp_i F^k[\mu^k d.r])$ with $i \not\geq k$ is called a *redex.*

There is no evaluation rule for when $i \geq k$, which forces F^k to represent the maximum context that can be captured. If no μ^k appears in a term, then the second evaluation rule will never be used, the $\sharp_k$ labels are universally ignored and standard call-by-value reduction takes place.

All application terms include a $\sharp_i$, which can act as a delimiter, stopping the capture of continuations by μ^k with $i \not\geq k$. Instead of a null evaluation rule $\sharp_i(V) \longrightarrow V$, which is found in most other systems, in our system β-reduction simply ignores the symbol.

We are missing an evaluation rule for when the entire term is of the form $F^n[\mu^n d.t]$. However, a $\lambda x.x$ can always be added in front of such a term. We can require that the minimum index 0, or some reserved index unrelated to any k that may appear in $\mu^k d.t$, is reserved for the purpose of forcing β-reduction. For example, $(\lambda x.x)\ \sharp_0\ \mu^n d.t$, with $n > 0$, reduces to $(\lambda x.x)\ \sharp_0\ t\{u/[d]u\}$ (because here $F^n = [\,]$). In other words it simply deletes the annotations placed on t.

The following key lemma illustrates the workings of the contexts and evaluation rules. A proof term is closed if all variables are bound by some λ or μ. Our results are for closed terms (which are the only proofs possible for end-sequents of the form $\vdash F$), but they can also be generalized.

Lemma 3. *Decomposition. For all non-value, closed terms T, either T is of the form $F^k[\mu^k d.s]$ or of the form $E[r]$ where r is a redex. Furthermore, E or F^k is uniquely determined.*

The proof is by induction on the structure of T. It follows easily from the property established by the lemma that if we placed an extra $\lambda x.x$ before a term t then all non-value, closed terms have uniquely determined redexes. If t has type $!_k?_{k'}A$, then $(\lambda x.x)\ \sharp_n\ t$ is well-typed for all $n \leq k'$. The lemma also implies the following:

Corollary 4. *Progress. If $s = (\lambda x.x)\ \sharp_0\ t$ is a closed proof term, then $s = E[r]$ where r is a redex. Furthermore, r is unique.*

Thus *evaluation is deterministic.* The following lemma shows that evaluation is type-safe, and forms part of the Subject Reduction proof.

Lemma 5. *Let C represent either a context E or F^k,*

1. *if $s : \Gamma \vdash A$, $s' : \Gamma \vdash A$ and $C[s] : \Gamma' \vdash A'$ are provable, then $C[s'] : \Gamma' \vdash A'$ is also provable.*
2. *if $F^k[\mu^k d.s] : \Gamma \vdash A$ is provable, then $s\{F^k[w]/[d]w\} : \Gamma \vdash A$ is also provable.*

Each part is proved by induction on the context. The difference between sequents $\Gamma \vdash A$ and $A^\perp, \Gamma\ \vdash$ is merely notational in classical linear logic. The central argument is similar to the reductions at the end of Section 7. Type soundness then follows:

Theorem 6. *Subject Reduction. If $s : \Gamma \vdash A$ is provable and $s \longrightarrow t$ using the evaluation rules, then $t : \Gamma \vdash A$ is also provable.*

In terms of the existing literature on delimited control operators, the behavior of our operators is dynamic as opposed to static: they are closer to the *control/prompt* of [6]. Since we do not interpret $\sharp_k$ as an independent operator, we cannot use it to

guarantee a static behavior. How μ-terms in the body of the substitution term $F^k[u]$ is to be delimited would depend on its surrounding context, which is not statically known. It is known that such dynamic, delimited control operators can have non-terminating behavior, even in a typed setting (see [2], [11]). The following term, adopted from [11], confirms this:

$$(\lambda x.x)\ \sharp_0\ ((\lambda z.\mu^i d.(\lambda y.[d]t)\ \sharp_i[d]t)\ \sharp_i(\mu^i d.(\lambda y.[d]t)\ \sharp_i[d]t))$$

Here, t can be any value of type $!_j?_i A$ while y and z are vacuous. Let V represent the value $\lambda z.(\mu^i d.(\lambda y.[d]t)\ \sharp_i[d]t)$, then $F^i = V\sharp_i[\,]$ and the term reduces to

$$(\lambda x.x)\ \sharp_0\ ((\lambda y.V\sharp_i t)\ \sharp_i(\mu^i d.(\lambda y.[d]t)\ \sharp_i[d]t))$$

leading to an infinite sequence of reductions. However, the term is well-typed. Also, it does have a normal form, namely t, but this is not reachable using the call-by-value strategy. This phenomenon does not contradict cut-elimination. The reduction steps still correspond to valid proof transformations. The possibility of non-termination is hardly cause for alarm from a programming perspective, and it is entirely consistent with what we already know to be possible with delimited control operators. A static behavior can be simulated using $(\lambda x.x)\sharp_0[\,]$, changing the continuation capture rule to:

$$E[V\ \ \sharp_i F^k[\mu^k d.t]] \longrightarrow \quad E[V\ \ \sharp_i\ t\{(\lambda x.x)\sharp_0 F^k[u]/[d]u\}]\ \ (i \not\geq k)$$

A call-by-name strategy can likewise avoid non-termination but then we can only capture continuations in the form of the original $\lambda\mu$-calculus, which is very limited for direct style programs. Call-by-value offers a much more general way to capture continuations. Forcing a particular evaluation strategy proof theoretically can be accomplished through focusing and this is well known. What we have shown here is that, however a continuation capturing strategy was chosen, *the presence of subexponentials in MC forces it to be delimited.*

9 Conclusion

The casual reader who opens this article to an arbitrary page may become dismayed by the large numbers of $?_i!_k?_i$ and $!_j?_{j'}$ that appear in formulas. Beneath this apparent chaos, however, are the fundamental proof-theoretic principles of adequate inductive decoration, cut-elimination, and focusing. We have extended these principles to a fragment of subexponential linear logic that enhances classical logic. In the MC fragment intuitionistic logic is found not as a restriction on proofs but as a restriction on formulas. This represents a new way to combine classical with intuitionistic logic which is quite different from the polarization approach of other systems. Although MC is defined using the formulas of subexponential linear logic, and is consistent with it in terms of provability, it is self-contained as a logical system, with its own, rather unique proof theoretic properties. The hierarchy of subexponentials naturally leads to a hierarchy of delimited control operators.

Acknowledgments. The authors wish to thank the reviewers of this paper for their comments, and to Danko Ilik for valuable discussion. This work was funded by the ERC Advanced Grant ProofCert.

References

1. Jean-Marc Andreoli. Logic programming with focusing proofs in linear logic. *J. of Logic and Computation*, 2(3):297–347, 1992.
2. Zena M. Ariola, Hugo Herbelin, and Amr Sabry. A type-theoretic foundation of delimited continuations. *Higher Order and Symbolic Computation*, 22(3):233–273, September 2009.
3. V. Danos, J.-B. Joinet, and H. Schellinx. LKT and LKQ: sequent calculi for second order logic based upon dual linear decompositions of classical implication. In J.-Y. Girard, Y. Lafont, and L. Regnier, editors, *Advances in Linear Logic*, number 222 in London Mathematical Society Lecture Note Series, pages 211–224. Cambridge University Press, 1995.
4. Vincent Danos, Jean-Baptiste Joinet, and Harold Schellinx. The structure of exponentials: Uncovering the dynamics of linear logic proofs. In Georg Gottlob, Alexander Leitsch, and Daniele Mundici, editors, *Kurt Gödel Colloquium*, volume 713 of *LNCS*, pages 159–171. Springer, 1993.
5. Olivier Danvy and Andrzej Filinski. Abstracting control. In *LISP and Functional Programming*, pages 151–160, 1990.
6. Mattias Felleisen. The theory and practice of first-class prompts. In *15th ACM Symp. on Principles of Programming Languages*, pages 180–190, New York, NY, USA, 1988. ACM.
7. Jean-Yves Girard. A new constructive logic: classical logic. *Math. Structures in Comp. Science*, 1:255–296, 1991.
8. Timothy Griffin. The formulae-as-types notion of control. In *17th Annual ACM Symp. on Principles of Programming Languages*, pages 47–57, 1990.
9. Danko Ilik. Delimited control operators prove double-negation shift. *Annals of Pure and Applied Logic*, 163(11):1549–1559, 2012.
10. Yukiyoshi Kameyama. Axioms for delimited continuations in the cps hierarchy. In *Computer Science Logic, 18th International Workshop*, volume 3210 of *Lecture Notes in Computer Science*, pages 442–457. Springer, 2004.
11. Yukiyoshi Kameyama and Takuo Yonezawa. Typed dynamic control operators for delimited continuations. In *Symposium on Functional and Logic Programming*, pages 239–254, 2008.
12. Chuck Liang and Dale Miller. Focusing and polarization in linear, intuitionistic, and classical logics. *Theoretical Computer Science*, 410(46):4747–4768, 2009.
13. Chuck Liang and Dale Miller. On subexponentials, synthetic connectives, and multi-level delimited control - long version. Unpublished manuscript available online at http://www.lix.polytechnique.fr/Labo/Dale.Miller/papers/subdelim-long.pdf, 2015.
14. M. Materzok and D. Biernacki. A dynamic interpretation of the CPS hierarchy. In *Programming Languages and Systems - 10th Asian Symposium, APLAS*, pages 296–311, 2012.
15. Vivek Nigam and Dale Miller. Algorithmic specifications in linear logic with subexponentials. In *ACM SIGPLAN Conference on Principles and Practice of Declarative Programming (PPDP)*, pages 129–140, 2009.
16. C.H. Luke Ong and Charles Stewart. A Curry-Howard foundation for functional computation with control. In *Symposium on Principles of Programming Languages*, pages 215–227, 1997.
17. Michel Parigot. $\lambda\mu$-calculus: An algorithmic interpretation of classical natural deduction. In *LPAR: Logic Programming and Automated Reasoning, International Conference*, volume 624 of *LNCS*, pages 190–201. Springer, 1992.
18. Alexis Saurin. A hierarchy for delimited continuations in call-by-name. In *Foundations of Software Science and Computational Structures, 13th International Conference, FOSSACS. Proceedings*, pages 374–388, 2010.