Factorize Factorization

Beniamino Accattoli
Inria & LIX, École Polytechnique, UMR 7161, Palaiseau, France

Claudia Faggian
Université de Paris, IRIF, CNRS, F-75013 Paris, France

Giulio Guerrieri
University of Bath, Department of Computer Science, Bath, United Kingdom

Abstract

We present a new technique for proving factorization theorems for compound rewriting systems in a modular way, which is inspired by the Hindley-Rosen technique for confluence. Specifically, our technique is well adapted to deal with extensions of the call-by-name and call-by-value λ-calculi.

The technique is first developed abstractly. We isolate a sufficient condition (called linear swap) for lifting factorization from components to the compound system, and which is compatible with β-reduction. We then closely analyze some common factorization schemas for the λ-calculus.

Concretely, we apply our technique to diverse extensions of the λ-calculus, among which de' Liguoro and Piperno's non-deterministic λ-calculus and—for call-by-value—Carraro and Guerrieri's shuffling calculus. For both calculi the literature contains factorization theorems. In both cases, we give a new proof which is neat, simpler than the original, and strikingly shorter.

2012 ACM Subject Classification Theory of computation → Rewrite systems; Theory of computation → Lambda calculus; Theory of computation → Logic

Keywords and phrases Lambda Calculus, Rewriting, Reduction Strategies, Factorization

Digital Object Identifier 10.4230/LIPIcs.CSL.2021.22

Funding This work is partially supported by ANR JCJC grant "COCA HOLA" (ANR-16-CE40-004-01), ANR PRC project PPS (ANR-19-CE48-0014), and EPSRC Project EP/R029121/1 *Typed lambda-calculi with sharing and unsharing.*

1 Introduction

The λ-calculus underlies functional programming languages and, more generally, the paradigm of higher-order computation. Through the years, more and more advanced features have enriched this paradigm, including control, non-determinism, states, probabilistic or quantum features. The well established way to proceed is to extend the λ-calculus with new operators. Every time, good operational properties, such as confluence, normalization, or termination, need to be proved. It is evident that the more complex and advanced is the calculus under study, the more the ability to *modularize the analyses* of its properties is crucial.

Techniques for modular proofs are available for termination and confluence, with a rich literature which examines under which conditions these properties lift from modules to the compound system—some representative papers are [55, 54, 56, 50, 39, 40, 41, 31, 10, 20, 18, 17, 5, 12], see Gramlich [22] for a survey. Termination and confluence concern the existence and the uniqueness of normal forms, which are the results of a computation. When the focus is on *how to compute* the result, that is, on identifying reduction strategies with good properties, then only few abstract techniques are currently available (we mention [21, 37, 38], [53, Ch. 8], and [2])—this paper proposes a new one.

Factorization. The most basic property about how to compute is *factorization*, whose paradigmatic example is the head factorization theorem of the λ-calculus (theorem 11.4.6 in

Barendregt's book [11]): every β-reduction sequence $t \to_\beta^* u$ can be re-organized/factorized so as to first reducing head redexes and then everything else—in symbols $t \underset{\mathsf{h}}{\to}^* \underset{\neg\mathsf{h}}{\to}^* u$.

The study of factorization in λ-calculus goes back to Rosser [49]. Factorization results are sometimes referred to as *semi-standardization* [42], or *postponement* [53], and often simply called *standardization*—standardization is however a more sophisticated property (sketched below) of which factorization is a basic instance. Here, we adopt Melliès terminology [38].

According to Melliès [38], the meaning of factorization is that the *essential* part of a computation can always be separated from its junk. Let's abstract the role of head reduction, by assuming that computations consists of steps $\underset{\mathsf{e}}{\to}$ which are in some sense *essential*, and steps $\underset{\mathsf{i}}{\to}$ which are not. Factorization says that every rewrite sequence $t \to^* s$ can be factorized as $t \underset{\mathsf{e}}{\to}^* u \underset{\mathsf{i}}{\to}^* s$, *i.e.*, as a sequence of essential steps followed by inessential ones.

Well known examples of essential reductions are head and leftmost-outermost reduction for the λ-calculus (see Barendregt [11]), or left and weak reduction for the call-by-value λ-calculus (see Plotkin [45] and Paolini and Ronchi Della Rocca [48]).

Very much as confluence, factorization for λ-calculi requires non-trivial proof techniques such as finite developments [15, 53], labeling [36, 30], or parallel reduction [52].

Uses of Factorization. Factorization is commonly used as a *building block* in proving more sophisticated properties of the *how-to-compute* kind. It is often the main ingredient in proofs of *normalization* theorems [11, 52, 28, 3], stating that a reduction strategy reaches a normal form whenever one exists. Leftmost-outermost normalization is a well known example.

Another property, *standardization*, generalizes factorization: reduction sequences can be organized with respect to an order on redexes, not just with respect to the distinction essential/inessential. It is an early result that factorization can be used to prove standardization: iterated head factorizations provides what is probably the simplest way to prove Curry and Feys' left-to-right standardization theorem, via Mitschke's argument [42].

Additionally, the independence of some computational tasks, such as garbage collection, is often modeled as a factorization theorem.

Contributions of this Paper. In this paper we propose a technique for proving in a *modular* way factorization theorems for *compound higher-order systems,* such as those obtained by extending the λ-calculus with advanced features. The approach can be seen as an analogous for factorization of the classical technique for confluence based on Hindley-Rosen lemma, which we discuss in the next paragraphs. Mimicking the use of Hindley-Rosen lemma is natural, yet to our knowledge such an approach has never been used before, at least not in the λ-calculus literature. Perhaps this is because a direct transposition of Hindley-Rosen technique does not work with β reduction, as we discuss below and in Sect. 3.

After developing a sharper technique, we apply it to various known extensions of the λ-calculus which do not fit into easily manageable categories of rewriting systems. In all our case studies, our novel proofs are neat, concise, and simpler than the originals.

Confluence via Hindley-Rosen. Let's consider confluence. The simplest modular technique to establish it is based on Hindley-Rosen lemma, which states that the *union* of two confluent reductions $\to_1$ and $\to_2$ is confluent if $\to_1$ and $\to_2$ satisfy a commutation property. This is the technique used in Barendregt's book for proving confluence of $\to_{\beta\eta}$ (Thm. 3.3.9 in [11]), where it is also stressed that the proof is simpler than Curry and Feys' original one.

While the result is basic, Hindley-Rosen technique provides a powerful tool to prove confluence of compound systems. In the literature of the λ-calculus, we mention for instance its use in the linear-algebraic λ-calculus [7], the probabilistic λ-calculus [19], the $\Lambda\mu$-calculus [51], the shuffling calculus [14], the λ-calculus extended with lists [47] or pattern-matching [13], or with a `let` construct [6]. It is worth to spell-out the gain. Confluence is often a non-trivial property to establish—when higher-order is involved, the proof of confluence requires sophisticated techniques. The difficulty *compounds* when extending the λ-calculus with new constructs. Still, the problem is often originated by β reduction itself, which encapsulates the higher-order features of the computation. By using Hindley-Rosen lemma, confluence of β is used as a *black box*: one *relies* on that—without having to prove it again—to show that the extended calculus is confluent.

Hindley-Rosen and Sufficient Conditions. There is a subtle distinction between Hindley-Rosen *lemma*, and what we refer to as Hindley-Rosen *technique*. Hindley-Rosen lemma reduces confluence of a compound system to commutation of the components—the modules. To establish commutation, however, is a non-trivial task, because it is a *global* property, that is, it quantifies over *all sequences* of steps. The success of the lemma in the λ-calculus literature stems from the existence of easy to check conditions which suffice to prove commutation. All the examples mentioned above indeed satisfy Hindley's *strong commutation* property [25] (Lemma 3.3.6 in [11]), where at most one reduction—but not both—may require multiple steps to close a diagram, commutation then follows by a finitary tiling argument. Strong commutation turns Hindley-Rosen lemma into an effective, concrete proof technique.

Modular Factorization, Abstractly. Here, we present a modular approach to factorization inspired by the Hindley-Rosen *technique*. A formulation of Hindley-Rosen lemma for factorization is immediate, and is indeed folklore. But exactly as for confluence, this reduces factorization of a compound system to a property that is difficult to establish, without a real gain. The crucial point is finding suitable conditions that can be used in practice. The *issue* here is that the natural adaptation of strong commutation to factorization is—in general—*not* verified by extensions of the λ-calculi, as it does not interact well with β (see Ex. 3.2 in Sect. 3). We identify an alternative condition—called *linear swap*—which is satisfied by a large variety of interesting examples, turning the approach into an effective, concrete *technique*. Testing the linear swap condition is easy and combinatorial in nature, as it is a *local* property, in the sense that only single steps (rather than sequences of steps) need to be manipulated. This holds true even when the modules are not confluent, or non-terminating. The other key point in our approach is that we *assume* the modules to be factorizing, therefore we can use their factorization—that may require non-trivial proof techniques such as parallel reductions or finite developments—as a black box.

Modular Factorization, Concretely. We then focus on our target, how to establish factorization results for extensions of the λ-calculus. Concretely, we start from β reduction, or its call-by-value counterpart β_v, and allow the calculus to be enriched with extra rules. Here we discover a further striking gain: for common factorization schemas such as head or weak factorization, verifying the required linear swap conditions reduces to checking *a single case*, together with the fact that the new rule behaves well with respect to substitution. The test for modular factorization which we obtain is a ready-to-use and easy recipe that can be applied in a variety of cases.

We illustrate our technique by providing several examples, chosen to stress the independence of the technique from other rewriting properties. In particular, we give a *new* and arguably *simpler* proof of two results from the literature. The first is head factorization for the non-deterministic λ-calculus by de' Liguoro and Piperno [16], that extends the λ-calculus with a choice operator $\oplus$. It is a *non confluent* calculus, and it is representative of the class of λ-calculi extended with a commutative effect—such as *probabilistic* choice—of which presents most features and all issues, see [33] for a thorough discussion.

The second is a new, simplified proof of factorization for the shuffling calculus—a refinement of the call-by-value λ-calculus due to Carraro and Guerrieri [14], whose left factorization is proved by Guerrieri, Paolini, and Ronchi della Rocca in [24]. In this case the λ-calculus is extended with extra rules but they are not associated to a new operator. The resulting calculus is subtle, as it has critical pairs.

In both cases, the new proof is neat, conceptually clear, and strikingly short. The reason why our proofs are only a few lines long, whereas the originals require several pages, is exactly that there is no need to "prove again" factorization of β or β_v. We just show that β (resp. β_v) interacts well with the new rules.

Further Applications: Probabilistic λ-calculi. The investigation in this paper was triggered by concrete needs, namely the study of strategies for probabilistic λ-calculi [19, 35]. The probabilistic structure adds complexity, and indeed makes the study of factorization painful—exposing the need for tools to make such an analysis more manageable. Our technique smoothly applies, providing new concise proofs that are significantly simpler than the originals—indeed surprisingly simple. These results are however only overviewed in this paper: we sketch the application to the call-by-value probabilistic calculus by Faggian and Ronchi della Rocca [19], leaving the technical details in Appendix B. The reason is that, while the application of our technique is simple, the *syntax* of probabilistic λ-calculi is not—because reduction is defined on (monadic) structures representing probability distributions over terms. Aiming at making the paper accessible within the space limits, we prefer to focus on examples in a syntax which is familiar to a wide audience. Indeed, once the technique is understood, its application to other settings is immediate, and in large part automatic.

A Final Remark. Like Hindley-Rosen for confluence, our technique is sufficient but not necessary to factorization. Still, its features and wide range of application make it a remarkable tool to tame the complexity which is often associated to the analysis of advanced compound calculi. By emphasizing the benefits of a modular approach to factorization, we hope to prompt the development of even more techniques.

Related work. To our knowledge, the only result in the literature about modular techniques for factorization is Accattoli's technique for calculi with explicit substitutions [2], which relies on termination hypotheses. Our *linear swap* condition (page 8) is technically the same as his *diagonal-swap* condition. One of the insights at the inception of this work is exactly that termination in [2] is used only to establish factorization of each single module, but not when combining them. Here we *assume* modules to be factorizing, therefore avoiding termination requirements, and obtaining a more widely applicable technique.

Van Oostrom's decreasing diagrams technique [58] is a powerful and *inherently modular* tool to establish confluence and commutation. Surprisingly, it has not yet been used for factorization, but steps in this direction have been presented recently [57].

A divide-and-conquer approach is well-studied for termination. The key point is finding conditions which guarantee that the union of terminating relations is terminating. Several have been studied [10, 20]. The weakest such condition, namely $\rightarrow_2 \cdot \rightarrow_1 \subseteq \rightarrow_1 \cup \rightarrow_2 \cdot (\rightarrow_1 \cup \rightarrow_2)^*$, is introduced by Doornbos and von Karger [18], and then studied by Dershowitz [17], under the name of *lazy commutation*, and by van Oostrom and Zantema [60]. Interestingly, lazy commutation is similar to the linear swap condition.

Another approach to study extensions of a rewriting system is isolating syntactical conditions that induce rewriting properties—for instance orthogonality of the rewriting rules induces confluence, see Terese [53, Ch. 10.4]. Factorization and standardization are also investigated, in particular for left-to-right standardization [53, Ch. 8.5.7].

2 Preliminaries

In this section we recall some standard definitions and notations in rewriting theory (see for instance Terese [53] or Baader and Nipkow [9]), and then provide an overview of commutation, confluence, and factorization. Both *confluence* and *factorization* are forms of commutation.

Basics. An *abstract rewriting system (ARS)* is a pair $\mathcal{A} = (A, \rightarrow)$ consisting of a set A and a binary relation $\rightarrow$ on A whose pairs are written $t \rightarrow s$ and called *steps*. We denote $\rightarrow^*$ (resp. $\rightarrow^=$) the transitive-reflexive (resp. reflexive) closure of $\rightarrow$, and use $\leftarrow$ for the reverse relation of $\rightarrow$, that is, $u \leftarrow t$ if $t \rightarrow u$. If $\rightarrow_1, \rightarrow_2$ are binary relations on A then $\rightarrow_1 \cdot \rightarrow_2$ denotes their composition, *i.e.* $t \rightarrow_1 \cdot \rightarrow_2 s$ if there exists $u \in A$ such that $t \rightarrow_1 u \rightarrow_2 s$. We write $(A, \{\rightarrow_1, \rightarrow_2\})$ to denote the ARS $(A, \rightarrow)$ where $\rightarrow \; = \; \rightarrow_1 \cup \rightarrow_2$. We freely use the fact that the transitive-reflexive closure of a relation is a closure operator, that is, it satisfies

$$\rightarrow \subseteq \rightarrow^*, \qquad (\rightarrow^*)^* \;=\rightarrow^*, \qquad \rightarrow_1 \subseteq \rightarrow_2 \text{ implies } \rightarrow_1^* \subseteq \rightarrow_2^* . \qquad (\textbf{Closure})$$

The following property is an immediate consequence:

$$(\rightarrow_1 \cup \rightarrow_2)^* \;=\; (\rightarrow_1^* \cup \rightarrow_2^*)^*. \qquad (\textbf{TR})$$

Local vs Global Properties. An important distinction in rewriting theory is between local and global properties. A property of term t is *local* if it is quantified over only *one-step reductions* from t, while it is *global* if it is quantified over all *rewrite sequences* from t. Local properties are easier to test, because the analysis (usually) involves a finite number of cases.

Commutation. Two relations $\rightarrow_1$ and $\rightarrow_2$ on A *commute* if ${\leftarrow_1}^* \cdot {\rightarrow_2}^* \subseteq {\rightarrow_2}^* \cdot {\leftarrow_1}^*$.

Confluence. A relation $\rightarrow$ on A is confluent if it commutes with itself. A classic tool to modularize the proof of confluence is Hindley-Rosen lemma. Confluence of two relations $\rightarrow_1$ and $\rightarrow_2$ does not imply confluence of $\rightarrow_1 \cup \rightarrow_2$, however it does if they commute.

▶ **Lemma** (Hindley-Rosen)**.** *Let $\rightarrow_1$ and $\rightarrow_2$ be relations on the set A. If $\rightarrow_1$ and $\rightarrow_2$ are confluent and commute with each other, then $\rightarrow_1 \cup \rightarrow_2$ is confluent.*

Easy-to-Check Conditions for Hindley Rosen. Commutation is a global condition, which is difficult to test. What turns Hindley-Rosen lemma into an effective, usable *technique*, is the availability of local, *easy-to-check* sufficient conditions. One of the simplest but most useful such conditions is Hindley's strong commutation [25]:

$$\leftarrow_1 \cdot \rightarrow_2 \subseteq \; {\rightarrow_2}^* \cdot {\leftarrow_1}^= \qquad (\textbf{Strong Commutation})$$

▶ **Lemma 2.1** (Local test for commutation [25])**.** *Strong commutation implies commutation.*

All the extensions of λ-calculus we cited at page 2 (namely [11, 7, 19, 51, 14, 47, 13, 6]) prove confluence by using Hindley-Rosen lemma via strong commutation (possibly in its weaker diamond-like form $\leftarrow_1 \cdot \rightarrow_2 \subseteq \; {\rightarrow_2}^= \cdot {\leftarrow_1}^=$).

Factorization. We now recall definitions and basic facts on the rewriting property at the center of this paper, factorization. Let $\mathcal{A} = (A, \{\underset{e}{\to}, \underset{i}{\to}\})$ be an ARS.

- The relation $\to\; = \underset{e}{\to} \cup \underset{i}{\to}$ satisfies **e-factorization**, written $\texttt{Fact}(\underset{e}{\to}, \underset{i}{\to})$, if

$$\texttt{Fact}(\underset{e}{\to}, \underset{i}{\to}): \quad (\underset{e}{\to} \cup \underset{i}{\to})^* \subseteq \underset{e}{\to}^* \cdot \underset{i}{\to}^* \qquad \textbf{(Factorization)}$$

- The relation $\underset{i}{\to}$ **postpones** after $\underset{e}{\to}$, written $\texttt{PP}(\underset{e}{\to}, \underset{i}{\to})$, if

$$\texttt{PP}(\underset{e}{\to}, \underset{i}{\to}): \quad \underset{i}{\to}^* \cdot \underset{e}{\to}^* \subseteq \underset{e}{\to}^* \cdot \underset{i}{\to}^*. \qquad \textbf{(Postponement)}$$

Postponement can be formulated in terms of commutation, and vice versa, since clearly ($\underset{i}{\to}$ postpones after $\underset{e}{\to}$) if and only if ($\underset{i}{\leftarrow}$ commutes with $\underset{e}{\to}$). Note that reversing $\underset{i}{\to}$ introduce an asymmetry between the two relations. It is an easy result that e-factorization is equivalent to postponement, which is a more convenient way to express it. The following equivalences—which we shall use freely—are all well known.

▶ **Lemma 2.2.** *For any two relations $\underset{e}{\to}, \underset{i}{\to}$ the following statements are equivalent:*

1. Semi-local postponement*: $\underset{i}{\to}^* \cdot \underset{e}{\to} \subseteq \underset{e}{\to}^* \cdot \underset{i}{\to}^*$ (and its dual $\underset{i}{\to} \cdot \underset{e}{\to}^* \subseteq \underset{e}{\to}^* \cdot \underset{i}{\to}^*$).*
2. Postponement*: $\texttt{PP}(\underset{e}{\to}, \underset{i}{\to})$.*
3. Factorization*: $\texttt{Fact}(\underset{e}{\to}, \underset{i}{\to})$.*

Another property that we shall use freely is the following, which is immediate by the definition of postponement and property **TR** (page 5).

▶ **Property 2.3.** *Given a relation $\underset{i}{\circ\!\!\to}$ such that $\underset{i}{\circ\!\!\to}^* = \underset{i}{\to}^*$, $\texttt{PP}(\underset{e}{\to}, \underset{i}{\to})$ if and only if $\texttt{PP}(\underset{e}{\to}, \underset{i}{\circ\!\!\to})$.*

A well-known use of the above is to instantiate $\underset{i}{\circ\!\!\to}$ with a notion of parallel reduction [52].

Easy-to-Check Sufficient Condition for Postponement. Hindley first noted that a local property implies postponement, hence factorization [25]. It is immediate to recognize that the property below is exactly the postponement analog of strong commutation in Lemma 2.1 (it is the same expression, with $\underset{i}{\to} := \leftarrow_1$ and $\underset{e}{\to} := \to_2$).

We say that $\underset{i}{\to}$ **strongly postpones** after $\underset{e}{\to}$, if

$$\texttt{SP}(\underset{e}{\to}, \underset{i}{\to}): \quad \underset{i}{\to} \cdot \underset{e}{\to} \subseteq \underset{e}{\to}^* \cdot \underset{i}{\to}^= \qquad \textbf{(Strong Postponement)}$$

▶ **Lemma 2.4** (Local test for postponement [25])**.** *Strong postponement implies postponement:*

$$\texttt{SP}(\underset{e}{\to}, \underset{i}{\to}) \textit{ implies } \texttt{PP}(\underset{e}{\to}, \underset{i}{\to}), \textit{ and so } \texttt{Fact}(\underset{e}{\to}, \underset{i}{\to}).$$

Strong postponement is at the heart of several factorization proofs. However (similarly to the diamond property for confluence) it can rarely be used *directly*, because most interesting relations—*e.g.* β reduction in λ calculus—do not satisfy it. Still, its range of application hugely widens by using Lemma 2.3.

It is instructive to examine strong postponement with respect to β reduction, as it allows us to also recall why it is difficult to establish head factorization for the λ-calculus.

▶ **Example 2.5** (λ-calculus and strong postponement)**.** In view of head factorization, the β reduction is decomposed in head reduction $\underset{h}{\to}_\beta$ and its dual $\underset{\neg h}{\to}_\beta$, that is $\to_\beta = \underset{h}{\to}_\beta \cup \underset{\neg h}{\to}_\beta$. To prove head factorization is non trivial precisely because $\texttt{SP}(\underset{h}{\to}_\beta, \underset{\neg h}{\to}_\beta)$ *does not* hold.

Consider the following example: $(\lambda x.xxx)(Iz) \underset{\neg h}{\to}_\beta (\lambda x.xxx)z \underset{h}{\to}_\beta zzz$. The sequence $\underset{\neg h}{\to}\underset{h}{\to}$ can only postpone to a reduction sequence of shape $\underset{h}{\to}\underset{h}{\to}\underset{\neg h}{\to}\underset{\neg h}{\to}$

$$(\lambda x.xxx)(Iz) \xrightarrow[\mathsf{h}]{}_\beta (Iz)(Iz)(Iz) \xrightarrow[\mathsf{h}]{}_\beta z(Iz)(Iz) \xrightarrow[\neg\mathsf{h}]{}_\beta zz(Iz) \xrightarrow[\neg\mathsf{h}]{}_\beta zzz$$

A solution is to compress sequences of $\xrightarrow[\neg\mathsf{h}]{}$ by introducing an intermediate relation $\overset{\circ}{\underset{\neg\mathsf{h}}{\Rightarrow}}$ (*internal parallel reduction*) such that $\overset{\circ}{\underset{\neg\mathsf{h}}{\Rightarrow}}^* = \xrightarrow[\neg\mathsf{h}]{}_\beta^*$ and which does verify strong postponement. This is indeed the core of Takahashi's technique [52]. All the work in [52] goes into defining parallel reductions, and proving $\mathtt{SP}(\xrightarrow[\mathsf{h}]{}_\beta, \overset{\circ}{\underset{\neg\mathsf{h}}{\Rightarrow}})$. One indeed has $\overset{\circ}{\underset{\neg\mathsf{h}}{\Rightarrow}} \cdot \xrightarrow[\mathsf{h}]{}_\beta \subseteq \xrightarrow[\mathsf{h}]{}_\beta \cdot \xrightarrow[\mathsf{h}]{}_\beta^* \cdot \overset{\circ}{\underset{\neg\mathsf{h}}{\Rightarrow}}$.

3 Modularizing Factorization

All along this section, we assume to have two relations $\to_\alpha, \to_\gamma$ on the same set A, such that

$$\to_\alpha \;=\; \xrightarrow[\mathsf{e}]{}_\alpha \cup \xrightarrow[\mathsf{i}]{}_\alpha \text{ and } \to_\gamma \;=\; \xrightarrow[\mathsf{e}]{}_\gamma \cup \xrightarrow[\mathsf{i}]{}_\gamma.$$

We define $\xrightarrow[\mathsf{i}]{} := (\xrightarrow[\mathsf{i}]{}_\alpha \cup \xrightarrow[\mathsf{i}]{}_\gamma)$ and $\xrightarrow[\mathsf{e}]{} := (\xrightarrow[\mathsf{e}]{}_\alpha \cup \xrightarrow[\mathsf{e}]{}_\gamma)$. Clearly $\to_\alpha \cup \to_\gamma \;=\; \xrightarrow[\mathsf{i}]{} \cup \xrightarrow[\mathsf{e}]{}$. Our goal is obtaining a technique in the style of Hindley-Rosen's for confluence, to establish that if $\to_\alpha, \to_\gamma$ are e-factorizing then their union also is, that is, $\mathtt{Fact}(\xrightarrow[\mathsf{e}]{}, \xrightarrow[\mathsf{i}]{})$ holds.

Issues. In spite of the large and fruitful use in the λ-calculus literature of Hindley-Rosen technique to simplify the analysis of confluence, we are not aware of any similar technique in the analysis of factorization. In this section we explain why a transposition of the technique is not immediate when β reduction is involved.

A direct equivalent of Hindley-Rosen lemma for commutation is folklore. An explicit proof is in [58]. Formulated in terms of postponement we obtain the following statement.

▶ **Lemma 3.1** (Hindley-Rosen transposed to factorization). *Assume $\to_\alpha$ and $\to_\gamma$ are* e-*factorizing relations. Their union $\to_\alpha \cup \to_\gamma$ satisfies* $\mathtt{Fact}(\xrightarrow[\mathsf{e}]{}, \xrightarrow[\mathsf{i}]{})$ *if*

$$\mathtt{PP}(\xrightarrow[\mathsf{e}]{}_\gamma, \xrightarrow[\mathsf{i}]{}_\alpha): \; \xrightarrow[\mathsf{i}]{}_\alpha^* \cdot \xrightarrow[\mathsf{e}]{}_\gamma^* \subseteq \xrightarrow[\mathsf{e}]{}_\gamma^* \cdot \xrightarrow[\mathsf{i}]{}_\alpha^* \quad \textit{and} \quad \mathtt{PP}(\xrightarrow[\mathsf{e}]{}_\alpha, \xrightarrow[\mathsf{i}]{}_\gamma): \; \xrightarrow[\mathsf{i}]{}_\gamma^* \cdot \xrightarrow[\mathsf{e}]{}_\alpha^* \subseteq \xrightarrow[\mathsf{e}]{}_\alpha^* \cdot \xrightarrow[\mathsf{i}]{}_\gamma^* \quad (\#)$$

Exactly as Hindley-Rosen lemma, the modularization lemma above is of no practical use by itself, as the pair of conditions (#) one has to test are as global as the original problem. What we need is to have local conditions (akin to strong commutation) to turn the lemma into a usable technique. One obvious choice is *strong postponement*:

$$\mathtt{SP}(\xrightarrow[\mathsf{e}]{}_\gamma, \xrightarrow[\mathsf{i}]{}_\alpha): \; \xrightarrow[\mathsf{i}]{}_\alpha \cdot \xrightarrow[\mathsf{e}]{}_\gamma \subseteq \xrightarrow[\mathsf{e}]{}_\gamma^* \cdot \xrightarrow[\mathsf{i}]{}_\alpha^= \quad \text{and} \quad \mathtt{SP}(\xrightarrow[\mathsf{e}]{}_\alpha, \xrightarrow[\mathsf{i}]{}_\gamma): \; \xrightarrow[\mathsf{i}]{}_\gamma \cdot \xrightarrow[\mathsf{e}]{}_\alpha \subseteq \xrightarrow[\mathsf{e}]{}_\alpha^* \cdot \xrightarrow[\mathsf{i}]{}_\gamma^= . \quad (\#\#)$$

Clearly, ## implies # (Lemma 2.4). We may hope to have all the elements for a postponement analog of Hindley-Rosen technique, but it is not the case. Unfortunately, conditions ## usually *do not hold* in extensions of the λ-calculus. Let us illustrate the issue with an example, the non-deterministic λ-calculus, that we shall develop formally in Sect. 5.

▶ **Example 3.2** (Issues). Consider the extension of the language of λ-terms with a construct $\oplus$ which models non-deterministic choice. The term $\oplus pq$ non-deterministically reduces to either p or q, that is, $\oplus pq \to_\oplus p$ or $\oplus pq \to_\oplus q$. The calculus $(\Lambda, \{\to_\beta, \to_\oplus\})$ has two reduction rules, $\to_\beta$ and $\to_\oplus$. For both, we define head and non-head steps as usual.

Consider the following sequence: $(\lambda x.xxx)(\oplus pq) \xrightarrow[\neg\mathsf{h}]{}_\oplus (\lambda x.xxx)p \xrightarrow[\mathsf{h}]{}_\beta ppp$. This sequence $\xrightarrow[\neg\mathsf{h}]{}_\oplus \cdot \xrightarrow[\mathsf{h}]{}_\beta$ can only postpone to a reduction sequence of shape $\xrightarrow[\mathsf{h}]{}_\beta \cdot \xrightarrow[\mathsf{h}]{}_\oplus \cdot \xrightarrow[\neg\mathsf{h}]{}_\oplus \cdot \xrightarrow[\neg\mathsf{h}]{}_\oplus$:

$$(\lambda x.xxx)(\oplus pq) \xrightarrow[\mathsf{h}]{}_\beta (\oplus pq)(\oplus pq)(\oplus pq) \xrightarrow[\mathsf{h}]{}_\oplus p(\oplus pq)(\oplus pq) \xrightarrow[\neg\mathsf{h}]{}_\oplus pp(\oplus pq) \xrightarrow[\neg\mathsf{h}]{}_\oplus ppp.$$

Since the β-step duplicates the redex $\oplus pq$, the condition $\mathtt{SP}(\xrightarrow[\mathsf{h}]{}_\beta, \xrightarrow[\neg\mathsf{h}]{}_\oplus) : \xrightarrow[\neg\mathsf{h}]{}_\oplus \xrightarrow[\mathsf{h}]{}_\beta \subseteq \xrightarrow[\mathsf{h}]{}_\beta^* \xrightarrow[\neg\mathsf{h}]{}_\oplus^=$ *does not hold.* The phenomenon is similar to Ex. 2.5, but now moving to parallel reduction is not a solution: the problem here is not just compressing steps, but the fact that by swapping $\underset{\neg\mathsf{h}}{\to}_{\oplus}$ and $\underset{\mathsf{h}}{\to}_{\beta}$, a *third* relation $\underset{\mathsf{h}}{\to}_{\oplus}$ appears.

Note that the problem above is specific to factorization, and does not appear with confluence.

A Robust Condition for Modular Factorization. Inspired by Accattoli's study of factorization for λ-calculi with explicit substitutions [2], we consider an alternative sufficient condition for modular factorization, which holds in many examples, as the next sections shall show.

We say that $\underset{\mathsf{i}}{\to}_{\alpha}$ **linearly swaps** with $\underset{\mathsf{e}}{\to}_{\gamma}$ if

$$\texttt{lSwap}(\underset{\mathsf{i}}{\to}_{\alpha}, \underset{\mathsf{e}}{\to}_{\gamma}) : \ \underset{\mathsf{i}}{\to}_{\alpha} \cdot \underset{\mathsf{e}}{\to}_{\gamma} \subseteq \underset{\mathsf{e}}{\to}_{\gamma} \cdot \to_{\alpha}^{*} \qquad \textbf{(Linear Swap)}$$

Note that, on the right-hand side, the relation is $\to_{\alpha}^{*}$, not $\underset{\mathsf{i}}{\to}_{\alpha}$. This small change will make a big difference, and overcome the issue we have seen in Ex. 3.2 (note that there $\underset{\mathsf{i}}{\to}_{\beta}\underset{\mathsf{h}}{\to}_{\oplus} \subseteq \underset{\mathsf{h}}{\to}_{\beta} \to_{\oplus}^{*}$ holds). Perhaps surprisingly, this easy-to-check condition, which is *local* and *linear* in $\underset{\mathsf{e}}{\to}_{\gamma}$, suffices, and holds in a large variety of cases. Moreover, it holds *directly* (even with β) that is, without the mediating role of parallel reductions (as it is the instead the case of Takahashi's technique, see Ex. 2.5).

We finally obtain a modular factorization technique, via the following easy property.

▶ **Lemma 3.3.** $\to_{a} \cdot \to_{b} \subseteq \to_{b} \cdot \to_{c}^{*}$ *implies* $\to_{a}^{*} \cdot \to_{b} \subseteq \to_{b} \cdot \to_{c}^{*}$.

▶ **Theorem 3.4** (Modular factorization). *Let* $\to_{\alpha} = (\underset{\mathsf{e}}{\to}_{\alpha} \cup \underset{\mathsf{i}}{\to}_{\alpha})$ *and* $\to_{\gamma} = (\underset{\mathsf{e}}{\to}_{\gamma} \cup \underset{\mathsf{i}}{\to}_{\gamma})$ *be* e*-factorizing relations. The union* $\to_{\alpha} \cup \to_{\gamma}$ *satisfies* e*-factorization* $\texttt{Fact}(\underset{\mathsf{e}}{\to}, \underset{\mathsf{i}}{\to})$*, for* $\underset{\mathsf{e}}{\to} := \underset{\mathsf{e}}{\to}_{\alpha} \cup \underset{\mathsf{e}}{\to}_{\gamma}$*, and* $\underset{\mathsf{i}}{\to} := \underset{\mathsf{i}}{\to}_{\alpha} \cup \underset{\mathsf{i}}{\to}_{\gamma}$*, if the following linear swaps hold:*

$$\texttt{lSwap}(\underset{\mathsf{i}}{\to}_{\alpha}, \underset{\mathsf{e}}{\to}_{\gamma}) : \ \underset{\mathsf{i}}{\to}_{\alpha} \cdot \underset{\mathsf{e}}{\to}_{\gamma} \subseteq \underset{\mathsf{e}}{\to}_{\gamma} \cdot \to_{\alpha}^{*} \qquad \textit{and} \qquad \texttt{lSwap}(\underset{\mathsf{i}}{\to}_{\gamma}, \underset{\mathsf{e}}{\to}_{\alpha}) : \ \underset{\mathsf{i}}{\to}_{\gamma} \cdot \underset{\mathsf{e}}{\to}_{\alpha} \subseteq \underset{\mathsf{e}}{\to}_{\alpha} \cdot \to_{\gamma}^{*}$$

Proof. We prove that the assumptions imply $\texttt{SP}(\underset{\mathsf{e}}{\to}, \underset{\mathsf{i}}{\to}_{\alpha}{}^{*} \cup \underset{\mathsf{i}}{\to}_{\gamma}{}^{*})$, hence $\texttt{PP}(\underset{\mathsf{e}}{\to}, \underset{\mathsf{i}}{\to}_{\alpha}{}^{*} \cup \underset{\mathsf{i}}{\to}_{\gamma}{}^{*})$ (by Lemma 2.4). Therefore $\texttt{PP}(\underset{\mathsf{e}}{\to}, \underset{\mathsf{i}}{\to})$ by Lemma 2.3 (because $(\underset{\mathsf{i}}{\to}_{\alpha} \cup \underset{\mathsf{i}}{\to}_{\gamma})^{*} = (\underset{\mathsf{i}}{\to}_{\alpha}{}^{*} \cup \underset{\mathsf{i}}{\to}_{\gamma}{}^{*})^{*}$ by property **TR**), and so $\texttt{Fact}(\underset{\mathsf{e}}{\to}, \underset{\mathsf{i}}{\to})$ holds.

To verify $\texttt{SP}(\underset{\mathsf{e}}{\to}_{\alpha} \cup \underset{\mathsf{e}}{\to}_{\gamma}, \underset{\mathsf{i}}{\to}_{\alpha}{}^{*} \cup \underset{\mathsf{i}}{\to}_{\gamma}{}^{*})$, we observe that the following holds:

$$\underset{\mathsf{i}}{\to}_{k}{}^{*} \cdot \underset{\mathsf{e}}{\to}_{j} \subseteq (\underset{\mathsf{e}}{\to}_{j} \cup \underset{\mathsf{e}}{\to}_{k})^{*} \cdot \underset{\mathsf{i}}{\to}_{k}{}^{*} \quad \text{for all } k, j \in \{\alpha, \gamma\}$$

- **Case** $j = k$**.** This is immediate by e-factorization of $\to_{\alpha}$ and $\to_{\gamma}$, and by Lemma 2.2.1.
- **Case** $j \neq k$**.** $\texttt{lSwap}(\underset{\mathsf{i}}{\to}_{\alpha}, \underset{\mathsf{e}}{\to}_{\gamma})$ implies $(\underset{\mathsf{i}}{\to}_{\alpha})^{*} \cdot \underset{\mathsf{e}}{\to}_{\gamma} \subseteq \underset{\mathsf{e}}{\to}_{\gamma} \cdot \to_{\alpha}^{*}$, by Lemma 3.3. Since $\to_{\alpha}$ e-factorizes, we obtain $(\underset{\mathsf{i}}{\to}_{\alpha})^{*} \cdot \underset{\mathsf{e}}{\to}_{\gamma} \subseteq \underset{\mathsf{e}}{\to}_{\gamma} \cdot \underset{\mathsf{e}}{\to}_{\alpha}{}^{*} \cdot \underset{\mathsf{i}}{\to}_{\alpha}{}^{*}$. Similarly for $\texttt{lSwap}(\underset{\mathsf{i}}{\to}_{\gamma}, \underset{\mathsf{e}}{\to}_{\alpha})$. ◀

Note that in the proof of Theorem 3.4, the assumption that $\to_{\alpha}$ and $\to_{\gamma}$ factorize is crucial. Using that, together with Lemma 3.3, we obtain $\texttt{SP}(\underset{\mathsf{e}}{\to}, \underset{\mathsf{i}}{\to}_{\alpha}{}^{*})$, that is, $\underset{\mathsf{i}}{\to}_{\alpha}{}^{*}$ postpones after both e-steps, (and similarly for $\underset{\mathsf{i}}{\to}_{\gamma}{}^{*}$). Note also that $\texttt{lSwap}(\underset{\mathsf{i}}{\to}, \underset{\mathsf{e}}{\to})$—taken alone—does not imply $\texttt{PP}(\underset{\mathsf{e}}{\to}, \underset{\mathsf{i}}{\to})$. For instance, let's consider again Ex. 2.5. It is clear that $\texttt{lSwap}(\underset{\neg\mathsf{h}}{\to}_{\beta}, \underset{\mathsf{h}}{\to}_{\beta})$ holds and yet it does not imply $\texttt{Fact}(\underset{\mathsf{h}}{\to}_{\beta}, \underset{\neg\mathsf{h}}{\to}_{\beta})$. Stronger tools, such as parallel reduction or finite development are needed here—there is no magic.

The next sections apply the modularization result to various λ-calculus extensions.

Linear Postponement. We collect here two easy properties which shall simplify the proof of factorization in several of the case studies (use Lemma 3.3).

▶ **Lemma 3.5** (Linear postponement).

1. $(\underset{\mathsf{i}}{\rightarrow} \cdot \underset{\mathsf{e}}{\rightarrow} \subseteq \underset{\mathsf{e}}{\rightarrow} \cdot \underset{\mathsf{i}}{\rightarrow}^*) \;\Rightarrow\; \mathtt{SP}(\underset{\mathsf{e}}{\rightarrow}, \underset{\mathsf{i}}{\rightarrow}^*) \;\Rightarrow\; \mathtt{Fact}(\underset{\mathsf{e}}{\rightarrow}, \underset{\mathsf{i}}{\rightarrow})$.
2. $(\underset{\mathsf{i}}{\rightarrow} \cdot \underset{\mathsf{e}}{\rightarrow} \subseteq \underset{\mathsf{e}}{\rightarrow} \cdot \rightarrow^=) \;\Rightarrow\; \mathtt{SP}(\underset{\mathsf{e}}{\rightarrow}, \underset{\mathsf{i}}{\rightarrow}) \;\Rightarrow\; \mathtt{Fact}(\underset{\mathsf{e}}{\rightarrow}, \underset{\mathsf{i}}{\rightarrow})$.

Factorization vs. Confluence. Factorization and confluence are *independent* properties. In Sect. 5 we apply our modular factorization technique to a non-confluent calculus. Conversely, $\beta\eta$, which is confluent, does not verify head nor leftmost factorization, even though both β and η—separately—do.

4 Extensions of the Call-by-Name λ-Calculus: Head Factorization

We shall study factorization theorems for extensions of both of the call-by-name (shortened to CbN) and of the call-by-value (CbV) λ-calculus. The CbN λ-calculus—also simply known as λ-calculus—is the set of λ-terms Λ, equipped with the β-reduction, while the CbV λ-calculus is the set of λ-terms Λ, equipped with the β_v-reduction.

In this section, we first revise the language of the λ-calculus and then consider the case when the calculus is *enriched with new operators*, such as a non-deterministic choice or a fix-point operator—so, together with β, we have other reduction rules. We study in this setting *head factorization*, which is by far the most important and common factorization scheme in λ-calculus. We show that here our modular technique further simplifies, providing an easy, *ready-to-use* test for head factorization of compound systems (Proposition 4.5). Indeed, verifying the two linear swap conditions of Theorem 3.4 now reduces to a single, simple test. Such a simplification only relies on β and on the properties of the contextual closure, that is, it holds independently of the specific form of the extra rule.

4.1 The (Applied) λ-Calculus

Since in the next sections we shall extend the λ-calculus with new operators, such as a non-deterministic choice $\oplus$ or a fix-point Y, we allow in the syntax a set of constants, meant to represent such operators. So, for instance, in Sect. 5 we shall see $\oplus$ as a constant. This way factorization results with respect to β-reduction can be seen as holding also in the λ-calculus with extended syntax—this is absolutely harmless.

Note that despite the fact that the classic Barendregt's book [11] defines the λ-calculus without constants (the calculus is pure), other classic references such as Hindley and Seldin's book [27] or Plotkin [45] do include constants in the language of terms—thus there is nothing exotic in our approach. Following Hindley and Seldin, when the set of constants is empty, the calculus is called *pure*, otherwise *applied*.

The Language. The following grammars generate λ-terms and contexts.

$$t, p, q, r, s ::= x \mid c \mid \lambda x.t \mid ts \quad \textbf{(terms } \Lambda\textbf{)} \qquad\qquad \mathsf{C} ::= \langle\,\rangle \mid t\mathsf{C} \mid \mathsf{C}t \mid \lambda x.\mathsf{C} \quad \textbf{(contexts)}$$

where x ranges over a countable set of *variables*, c over a disjoint (finite, infinite or empty) set of constants. Variables and constants are *atoms*, terms of shape pq are *applications*, and $\lambda x.p$ *abstractions*. If the constants are $c_1, \ldots, c_n$, the set of terms is sometimes noted as $\Lambda_{c_1 \ldots c_n}$.

The plugging $\mathsf{C}\langle t\rangle$ of a term t into a context is the operation replacing the only occurrence of a hole $\langle\,\rangle$ in C with t, potentially capturing free variables of C.

A reduction step $\to_\gamma$ is defined as the contextual closure of a *root* relation $\mapsto_\gamma$ on Λ, which is called a *rule*. Explicitly, $t \to_\gamma s$ if $t = \mathsf{C}\langle r\rangle$ and $s = \mathsf{C}\langle r'\rangle$, for some context C with $r \mapsto_\gamma r'$. The term r is called a γ-redex. Given two rules $\mapsto_\alpha, \mapsto_\gamma$ on Λ, the relation $\to_{\alpha\gamma}$ is $\to_\alpha \cup \to_\gamma$, which can equivalently be defined as the contextual closure of $\mapsto_\alpha \cup \mapsto_\gamma$.

The **(CbN) $\boldsymbol{\lambda}$-calculus is** $(\Lambda, \to_\beta)$, the set Λ of terms together with **$\boldsymbol{\beta}$-reduction** $\to_\beta$, defined as the contextual closure of the β-rule: $(\lambda x.p)q \mapsto_\beta p\{x{:=}q\}$ where $p\{x{:=}q\}$ denotes capture-avoiding substitution. We silently work modulo α-equivalence.

Properties of the Contextual Closure. Here we recall basic properties about contextual closures and substitution, preparing the ground for the simplifications studied next.

A relation $\looparrowright$ on terms is *substitutive* if

$$r \looparrowright r' \text{ implies } r\{x{:=}q\} \looparrowright r'\{x{:=}q\}. \qquad \textbf{(substitutive)}$$

An obvious induction on the shape of terms shows the following (see Barendregt [11], p. 54).

▶ **Property 4.1** (Substitutive). *Let $\to_\gamma$ be the contextual closure of $\mapsto_\gamma$.*

1. *If $\mapsto_\gamma$ is substitutive then $\to_\gamma$ is substitutive: $p \to_\gamma p'$ implies $p\{x{:=}q\} \to_\gamma p'\{x{:=}q\}$.*
2. *If $q \to_\gamma q'$ then $t\{x{:=}q\} \to_\gamma^* t\{x{:=}q'\}$.*

We recall a basic but key property of contextual closures. If a step $\to_\gamma$ is obtained by closure under *non-empty context* of a rule $\mapsto_\gamma$, then it preserves the shape of the term:

▶ **Property 4.2** (Shape preservation). *Assume $t = C\langle r\rangle \to C\langle r'\rangle = t'$ and that context C is* non-empty. *The term t' is an application (resp. an abstraction) if and only if t is.*

Notice that since the closure under *empty* context of $\mapsto_\gamma$ is always an *essential* step (whatever head, left, or weak), Property 4.2 implies that non-essential steps always preserve the shape of terms—we spell this out in Properties A.1 and A.2 in the Appendix. Notice that we shall often write $\mapsto_\gamma$ to indicate the step $\to_\gamma$ which is obtained by *empty contextual closure.*

Head Reduction. Head contexts are defined as follows:

$$\mathsf{H} ::= \lambda x_1 \ldots \lambda x_k.\langle\ \rangle t_1 \ldots t_n \qquad \textbf{(head contexts)}$$

where $k \geq 0$ and $n \geq 0$. A *non-head context* is a context which is not head. A *head* step $\underset{\mathsf{h}}{\to}_\gamma$ (resp. *non-head* step $\underset{\neg\mathsf{h}}{\to}_\gamma$) is defined as the closure under head contexts (resp. non-head contexts) of rule $\mapsto_\gamma$. Obviously, $\to_\gamma = \underset{\mathsf{h}}{\to}_\gamma \cup \underset{\neg\mathsf{h}}{\to}_\gamma$.

Note that the *empty context* $\langle\ \rangle$ is a *head context.* Therefore $\mapsto_\gamma \subseteq \underset{\mathsf{h}}{\to}_\gamma$ holds (a fact which we shall freely use) and Property 4.2 always applies to non-head steps.

4.2 Call-by-Name: Head Factorization, Modularly.

Head factorization is of great importance for the theory of the CbN λ-calculus, which is why head factorization for $\to_\beta$ is well studied. If we consider a calculus $(\Lambda, \to_\beta \cup \to_\gamma)$, where $\to_\gamma$ is a new reduction added to β, our modular technique (Theorem 3.4) states that the compound system $\to_\beta \cup \to_\gamma$ satisfies head factorization if $\to_\gamma$ does, and both $\texttt{lSwap}(\underset{\neg\mathsf{h}}{\to}_\beta, \underset{\mathsf{h}}{\to}_\gamma)$ and $\texttt{lSwap}(\underset{\neg\mathsf{h}}{\to}_\gamma, \underset{\mathsf{h}}{\to}_\beta)$ hold. We show that in the head case our technique simplifies even more, reducing to the test in Proposition 4.5.

First, we observe that in this case, *any* linear swap condition can be tested by considering for the head step only the root relation $\mapsto$, that is, only the closure of $\mapsto$ under *empty* context, which is a head step by definition. This is expressed in the following lemma, where we include also a variant, that shall be useful later on.

▶ **Lemma 4.3** (Lifting root linear swaps). *Let $\mapsto_\alpha, \mapsto_\gamma$ be root relations on Λ.*

1. $\underset{\neg h}{\to}_\alpha \cdot \mapsto_\gamma \subseteq \underset{h}{\to}_\gamma \cdot \to_\alpha^*$ *implies* $\mathtt{lSwap}(\underset{\neg h}{\to}_\alpha, \underset{h}{\to}_\gamma)$.
2. *Similarly,* $\underset{\neg h}{\to}_\alpha \cdot \mapsto_\gamma \subseteq \underset{h}{\to}_\gamma \cdot \to_\alpha^=$ *implies* $\underset{\neg h}{\to}_\alpha \cdot \to_\gamma \subseteq \underset{h}{\to}_\gamma \cdot \to_\alpha^=$.

Second, since we are studying $\to_\beta \cup \to_\gamma$, one of the linear swaps is $\mathtt{lSwap}(\underset{\neg h}{\to}_\gamma, \underset{h}{\to}_\beta)$. We show that, whatever is $\to_\gamma$, it linearly swaps with $\underset{h}{\to}_\beta$ as soon as $\mapsto_\gamma$ is *substitutive.*

▶ **Lemma 4.4** (Swap with $\underset{h}{\to}_\beta$). *If $\mapsto_\gamma$ is substitutive then $\mathtt{lSwap}(\underset{\neg h}{\to}_\gamma, \underset{h}{\to}_\beta)$ holds.*

The proofs of these two lemmas are in Appendix A.1.

Summing up, since head factorization for β is known, we obtain the following test to verify that the compound system $\to_\beta \cup \to_\gamma$ satisfies head factorization $\mathtt{Fact}(\underset{h}{\to}_\beta \cup \underset{h}{\to}_\gamma, \underset{\neg h}{\to}_\beta \cup \underset{\neg h}{\to}_\gamma)$.

▶ **Proposition 4.5** (A test for modular head factorization). *Let $\to_\beta$ be β-reduction and $\to_\gamma$ be the contextual closure of a rule $\mapsto_\gamma$. Their union $\to_\beta \cup \to_\gamma$ satisfies head factorization if:*

1. Head factorization of $\to_\gamma$*:* $\mathtt{Fact}(\underset{h}{\to}_\gamma, \underset{\neg h}{\to}_\gamma)$.
2. Root linear swap*:* $\underset{\neg h}{\to}_\beta \cdot \mapsto_\gamma \subseteq \underset{h}{\to}_\gamma \cdot \to_\beta^*$.
3. Substitutivity*:* $\mapsto_\gamma$ *is substitutive.*

Note that none of the properties concerns $\to_\beta$ alone, as we already know that head factorization of $\to_\beta$ holds. In Sect. 5 we shall use our test (Proposition 4.5) to prove head factorization for the non-deterministic λ-calculus. The full proof is only a few lines long.

We conclude by observing that Lemma 4.3 gives either a proof that the swap conditions hold, or a counter-example. Let us give an example of this latter use.

▶ **Example 4.6** (Finding counter-examples). The test of Proposition 4.5 can also be used to provide a counter-example to head factorization when it fails. Let's instantiate $\to_\gamma$ with $\to_\eta$, that is, the contextual closure of rule $\lambda x.tx \mapsto_\eta t$ if $x \notin \mathsf{fv}(t)$. Now, consider the root linear swap: $t := \lambda x.(II)(Ix) \underset{\neg h}{\to}_\beta \lambda x.(II)x \mapsto_\eta II =: s$, where $I := \lambda z.z$. Note that t has no $\underset{h}{\to}_\eta$ step, and so the two steps cannot be swapped. The reduction sequence above is a *counter-example to both head and leftmost factorization* for $\beta\eta$. Start with the head (and leftmost) redex II: $\lambda x.(II)(Ix) \underset{h}{\to}_{\beta\eta} \lambda x.I(Ix)$. From $\lambda x.I(Ix)$, there is no way to reach s. We recall that $\beta\eta$ still satisfies leftmost normalization—the proof is non-trivial [30, 52, 59, 29].

5 The Non-Deterministic λ-Calculus $\Lambda_\oplus$

De' Liguoro and Piperno's non-deterministic λ-calculus $\Lambda_\oplus$ is defined in [16] by extending the λ-calculus with a new operator $\oplus$ whose rule models *non-deterministic choice.* Intuitively, $t \oplus p$ non-deterministically rewrites to either t or p. Notably, $\Lambda_\oplus$ is *not confluent*, hence it is a good example of the fact that confluence and factorization are independent properties.

We briefly recall $\Lambda_\oplus$ and its features, then use our technique to give a *novel and neat* proof of de' Liguoro and Piperno's *head factorization* result [16, Cor. 2.10].

Syntax. We slightly depart from the presentation in [16], as we consider $\oplus$ as a constant, and write $\oplus tp$ rather than $t \oplus p$, working as usual for the λ-calculus with constants (see *e.g.*,

[27], or [11, Sec. 15.3]).[1] Terms and contexts are generated by:

$$t, p, q, r ::= x \mid \oplus \mid \lambda x.t \mid tp \quad (\textbf{terms } \Lambda_\oplus) \qquad \mathsf{C} ::= \langle\ \rangle \mid t\mathsf{C} \mid \mathsf{C}t \mid \lambda x.\mathsf{C} \quad (\textbf{contexts})$$

As before, $\to_\beta$ denotes β-reduction, while the rewrite step $\to_\oplus$ is the contextual closure of the following non-deterministic rule: $\oplus tp \mapsto_\oplus t$ and $\oplus tp \mapsto_\oplus p$.

Subtleties. The calculus $(\Lambda_\oplus, \to_\beta \cup \to_\oplus)$ is non trivial. Of course, $\to$ is not confluent. Moreover, the following examples from [16] show that permuting β and $\oplus$ steps is delicate.

- $\to_\oplus$ *creates β-redexes.* For instance, $((\lambda x.x) \oplus y)z \to_\oplus (\lambda x.x)z \to_\beta z$, hence the $\to_\oplus$-step cannot be postponed after $\to_\beta$.
- *Choice duplication.* Postponing $\to_\beta$ after $\to_\oplus$ is also problematic, because β-steps may multiply choices, introducing new results: flipping a coin and duplicating the result is not equivalent to duplicating the coin and then flipping twice. For instance, let $t = (\lambda x.xx)(\oplus pq)$. Duplicating first one may have $t \to_\beta (\oplus pq)(\oplus pq) \to_\oplus q(\oplus pq) \to_\oplus qp$ while flipping first one has $t \to_\oplus (\lambda x.xx)p \to_\beta pp$ or $t \to_\oplus (\lambda x.xx)q \to_\beta qq$ but in both cases qp cannot be reached.

These examples are significant as the same issues impact any calculus with choice effects.

Head Factorization. The head (resp. non-head)[2] rewrite steps $\underset{\mathsf{h}}{\to}_\beta$ and $\underset{\mathsf{h}}{\to}_\oplus$ (resp. $\underset{\neg\mathsf{h}}{\to}_\beta$ and $\underset{\neg\mathsf{h}}{\to}_\oplus$) are defined as the closure by head (resp. non-head) contexts of rules $\mapsto_\beta$ and $\mapsto_\oplus$, respectively. We also set $\underset{\mathsf{h}}{\to} := \underset{\mathsf{h}}{\to}_\beta \cup \underset{\mathsf{h}}{\to}_\oplus$ and $\underset{\neg\mathsf{h}}{\to} := \underset{\neg\mathsf{h}}{\to}_\beta \cup \underset{\neg\mathsf{h}}{\to}_\oplus$.

De' Liguoro and Piperno prove that despite the failure of confluence, $\Lambda_\oplus$ satisfies head factorization. They prove this result via standardization, following Klop's technique [30].

▶ **Theorem 5.1** (Head factorization, Cor. 2.10 in [16]). $\mathtt{Fact}(\underset{\mathsf{h}}{\to}, \underset{\neg\mathsf{h}}{\to})$ *holds in the non-deterministic λ-calculus $\Lambda_\oplus$.*

A New Proof, Modularly. We give a novel, strikingly simple proof of $\mathtt{Fact}(\underset{\mathsf{h}}{\to}, \underset{\neg\mathsf{h}}{\to})$, simply by proving that $\to_\beta$ and $\to_\oplus$ satisfy the hypotheses of the test for modular head factorization (Proposition 4.5). All the ingredients we need are given by the following easy lemma.

▶ **Lemma 5.2** (Root linear swaps). **1.** $t \underset{\neg\mathsf{h}}{\to}_\beta p \mapsto_\oplus q$ *implies* $t \mapsto_\oplus \cdot \to_\beta^= q$.
2. $t \underset{\neg\mathsf{h}}{\to}_\oplus p \mapsto_\oplus q$ *implies* $t \mapsto_\oplus \cdot \to_\oplus^= q$.

Proof. **1.** Let $p = \oplus p_1 p_2$ and assume $\oplus p_1 p_2 \mapsto_\oplus p_i = q$, with $i \in \{1, 2\}$. Since $t \underset{\neg\mathsf{h}}{\to}_\beta p$, by Property 4.2 (as spelled-out in Property A.1), t has shape $\oplus t_1 t_2$, with $\oplus t_1 t_2 \underset{\neg\mathsf{h}}{\to}_\beta \oplus p_1 p_2$. Therefore, either $t_1 \to_\beta p_1$ or $t_2 \to_\beta p_2$, from which $t = \oplus t_1 t_2 \mapsto_\oplus t_i \to_\beta^= p_i = q$.
2. The proof is the same as above, just replace β with $\oplus$. ◀

▶ **Theorem 5.3** (Testing head factorization). *We have* $\mathtt{Fact}(\underset{\mathsf{h}}{\to}, \underset{\neg\mathsf{h}}{\to})$ *because we have:*
1. Head factorization of $\to_\oplus$*:* $\mathtt{Fact}(\underset{\mathsf{h}}{\to}_\oplus, \underset{\neg\mathsf{h}}{\to}_\oplus)$.
2. Root linear swap*:* $\underset{\neg\mathsf{h}}{\to}_\beta \cdot \mapsto_\oplus \subseteq \underset{\mathsf{h}}{\to}_\oplus \cdot \to_\beta^=$.
3. Substitutivity*:* $\mapsto_\oplus$ *is substitutive.*

[1] Note that there is no loss with respect to the syntax in [16], where $\oplus$ comes with exactly two arguments, because in our formalism such a restriction defines a sub-system that is closed under reduction.
[2] Non-head steps are called *internal* ($\underset{\mathsf{i}}{\to}$) in [16].

Proof. We prove the hypotheses of Proposition 4.5.

1. $\underset{\neg\mathsf{h}}{\to}_{\oplus}$ linearly postpones after $\underset{\mathsf{h}}{\to}_{\oplus}$ because lifting the swap in Lemma 5.2.2 via Lemma 4.3.2 (with $\alpha = \gamma := \oplus$) gives $t \underset{\neg\mathsf{h}}{\to}_{\oplus} p \underset{\mathsf{h}}{\to}_{\oplus} q \;\subseteq\; t \underset{\mathsf{h}}{\to}_{\oplus} \cdot \to_{\oplus}^{=} q$. Lemma 3.5.2 gives Factorization.
2. This is exactly Lemma 5.2.1.
3. By definition of substitution $(\oplus p_1 p_2)\{x{:=}q\} = \oplus p_1\{x{:=}q\}\, p_2\{x{:=}q\} \mapsto_{\oplus} p_i\{x{:=}q\}$. ◀

6 Extensions of the CbV λ-Calculus: Left and Weak Factorization

Plotkin's call-by-value (CbV) λ-calculus [45] is the restriction of the λ-calculus where β-redexes can be fired only when the argument is a *value*, where values are defined by:

$$v \quad ::= \quad x \mid c \mid \lambda x.t \qquad (\textbf{values } \mathcal{V})$$

The CbV λ-calculus is given by the pair $(\Lambda, \to_{\beta_v})$, where *β_v-reduction* $\to_{\beta_v}$ is the contextual closure of the following rule $\mapsto_{\beta_v}$: $(\lambda x.t)v \mapsto_{\beta_v} t\{x{:=}v\}$, where v is a value.

Left and Weak Reduction. In the literature on the CbV λ-calculus, factorization is considered with respect to various essential reductions. Usually, the essential reduction is *weak*, that is, it does not act under abstractions. There are three main weak schemes: reducing from left to right, as originally done by Plotkin [45], from right to left, as done for instance by Leroy's ZINC abstract machine [34], or in an unspecified non-deterministic order, used for example by Dal Lago and Martini [32].

Here we focus on the left(-to-right) and the (unspecified) weak schemes. *Left* contexts L and *weak* contexts W are respectively defined by

$$\mathsf{L} ::= \langle\,\rangle \mid \mathsf{L}t \mid v\mathsf{L} \qquad\qquad \mathsf{W} ::= \langle\,\rangle \mid \mathsf{W}t \mid t\mathsf{W}$$

Given a rule $\mapsto_\gamma$, a *left* step $\underset{\mathsf{l}}{\to}_\gamma$ (resp., a *weak* step $\underset{\mathsf{w}}{\to}_\gamma$) is its closure by left (resp. weak) context. A *non-left* step $\underset{\neg\mathsf{l}}{\to}_\gamma$ (resp. *non-weak* step $\underset{\neg\mathsf{w}}{\to}_\gamma$) is a step obtained as the closure by a context which is not left (resp. not weak).

Left/Weak Factorization, Modularly. For both left and weak reductions, we derive a test for modular factorization analogous to the test for head factorization (Proposition 4.5). Note that we already know that $(\Lambda, \to_{\beta_v})$ satisfies left and weak factorization: the former was proved by Plotkin [45], the latter is folklore—a proof can be found in our previous work [3].

▶ **Proposition 6.1** (A test for modular left/weak factorization). *Let $\to_{\beta_v}$ be β_v-reduction, $\to_\gamma$ the contextual closure of a rule $\mapsto_\gamma$, and $\mathsf{e} \in \{\mathsf{l}, \mathsf{w}\}$. Their union $\to_{\beta_v} \cup \to_\gamma$ satisfies* e*-factorization* $\mathtt{Fact}(\underset{\mathsf{e}}{\to}_{\beta_v} \cup \underset{\mathsf{e}}{\to}_\gamma, \underset{\neg\mathsf{e}}{\to}_{\beta_v} \cup \underset{\neg\mathsf{e}}{\to}_\gamma)$ *if:*

1. e-factorization of $\to_\gamma$*:* $\mathtt{Fact}(\underset{\mathsf{e}}{\to}_\gamma, \underset{\neg\mathsf{e}}{\to}_\gamma)$.
2. Root linear swap*:* $\underset{\neg\mathsf{e}}{\to}_{\beta_v} \cdot \mapsto_\gamma \;\subseteq\; \underset{\mathsf{e}}{\to}_\gamma \cdot \to_{\beta_v}^{*}$.
3. Substitutivity*: $\mapsto_\gamma$ is substitutive.*

The easy proof is in Appendix A.2.

7 The Shuffling Calculus

Plotkin's CbV λ-calculus is usually considered on closed terms. When dealing with open terms, it is well known that a mismatch between the operational and the denotational semantics arises, as first pointed out by Paolini and Ronchi della Rocca [44, 43, 48]. The literature contains several proposals of extensions of β_v reduction to overcome this issue, see Accattoli and Guerrieri for discussions [4]. One such refinement is Carraro and Guerrieri's *shuffling calculus* [14], which extends Plotkin's λ-calculus with extra rules (without adding new operators). These rules are inspired by linear logic proof nets, and are the CbV analogous of Regnier's σ rules [46]. Left factorization for the shuffling calculus is studied by Guerrieri, Paolini, and Ronchi della Rocca in [24, 23], by adapting Takahashi's technique [52].

We recall the calculus, then use our technique to give a new proof of factorization, both left (as in [24]) and weak (new). Remarkably, our proofs are *very short*, whereas the original requires several pages (to define parallel reductions and prove their properties).

The Syntax. The *shuffling calculus* is simply Plotkin's calculus extended with *σ-reduction* $\to_\sigma$, that is, the contextual closure of the root relation $\mapsto_\sigma = \mapsto_{\sigma_1} \cup \mapsto_{\sigma_3}$, where

$$(\lambda x.t)us \mapsto_{\sigma_1} (\lambda x.ts)u \;\text{ if } x \notin \mathsf{fv}(s) \qquad\qquad v((\lambda x.t)u) \mapsto_{\sigma_3} (\lambda x.vt)u \;\text{ if } x \notin \mathsf{fv}(v)$$

We write $\to_{\sigma_i}$ for the contextual closure of $\mapsto_{\sigma_i}$ (so $\to_\sigma = \to_{\sigma_1} \cup \to_{\sigma_3}$), and $\to_{\mathsf{sh}} = \to_{\beta_v} \cup \to_\sigma$.

Subtleties. From a rewriting perspective, the shuffling calculus is an interesting extension of the λ-calculus because its intricate rules do not fit into easy to manage classes of rewriting systems. Orthogonal systems have only simple forms of overlaps of redexes. While the λ-calculus is an orthogonal system, the σ-rules introduce non-trivial overlaps such as the following ones. Setting $I := \lambda x.x$ and $\delta := \lambda x.xx$, the term $\delta I \delta$ is a σ_1-redex and contains the β_v-redex δI, while the term $\delta(I\delta)(xI)$ is a σ_1-redex and contains the σ_3-redex $\delta(I\delta)$, which contains in turn the β_v-redex $I\delta$.

Left and Weak Factorization. Despite all these traits, the shuffling calculus has good properties, such as confluence [14], and left factorization [24]. Moreover, $\to_\sigma$ is terminating [14]. The tests developed in the previous section allow us to easily prove both left and weak factorization. We check the hypotheses of Proposition 6.1—all the ingredients we need are in Lemma 7.1 (the easy details are in Appendix A.3). Note that the *empty context is both a left and a weak* context, hence $t \mapsto_{\sigma_i} u$ implies both $t \underset{\mathsf{l}}{\to}_{\sigma_i} u$ and $t \underset{\mathsf{w}}{\to}_{\sigma_i} u$.

▶ **Lemma 7.1** (Root linear swaps). *Let* $\mathsf{e} \in \{\mathsf{l}, \mathsf{w}\}$ *and* $i \in \{1, 3\}$. *Then:*

1. $\underset{\neg\mathsf{e}}{\to}_{\beta_v} \cdot \mapsto_{\sigma_i} \subseteq \mapsto_{\sigma_i} \cdot \to_{\beta_v}$.
2. $\underset{\neg\mathsf{e}}{\to}_{\sigma} \cdot \mapsto_{\sigma_i} \subseteq \mapsto_{\sigma_i} \cdot \to_{\sigma}$.

▶ **Theorem 7.2** (Testing left (weak) factorization). *Let* $\mathsf{e} \in \{\mathsf{l}, \mathsf{w}\}$. $\mathtt{Fact}(\underset{\mathsf{e}}{\to}_{\mathsf{sh}}, \underset{\neg\mathsf{e}}{\to}_{\mathsf{sh}})$ *holds, as:*

1. Left (resp. weak) factorization of $\to_\sigma$*:* $\mathtt{Fact}(\underset{\mathsf{e}}{\to}_{\sigma}, \underset{\neg\mathsf{e}}{\to}_{\sigma})$.
2. Root linear swap*:* $\underset{\neg\mathsf{e}}{\to}_{\beta_v} \cdot \mapsto_\sigma \subseteq \underset{\mathsf{e}}{\to}_{\sigma} \cdot \to_{\beta_v}$.
3. Substitutivity*:* $\mapsto_{\sigma_i}$ *is substitutive, for* $i \in \{1, 3\}$.

Proof. We prove the hypotheses of Proposition 6.1:

1. Left (resp. weak) factorization of $\to_\sigma$ holds because $\underset{\neg\mathsf{e}}{\to}_{\sigma}$ linearly postpones after $\underset{\mathsf{e}}{\to}_{\sigma}$: indeed, by Lemma 7.1.2 $\underset{\neg\mathsf{e}}{\to}_{\sigma} \cdot \mapsto_\sigma \subseteq \underset{\mathsf{e}}{\to}_{\sigma} \cdot \to_\sigma$ and by contextual closure (Lemma A.3 with $\alpha = \gamma = \sigma$) we have that $\underset{\neg\mathsf{e}}{\to}_{\sigma} \cdot \underset{\mathsf{e}}{\to}_{\sigma} \subseteq \underset{\mathsf{e}}{\to}_{\sigma} \cdot \to_{\sigma}^{=}$. We conclude by Lemma 3.5.2.
2. This is Lemma 7.1.1.
3. By definition of substitution. The immediate proof is in Appendix A.3. ◀

8 Non-Terminating Relations

In this section we provide examples of the fact that our technique does not rest on termination hypotheses. We consider *fixpoint operators* in both CbN and CbV, which have non-terminating reductions. Obviously, when terms are not restricted by types, the operator is definable, so the example is slight artificial, but we hope clarifying.

There are also cases where the modules are terminating but the compound system is not. The technique, surprisingly, still works. Accattoli gives various examples based on λ-calculi with explicit substitutions in [2]. An insight of this paper—not evident in [2]—is that *termination is not needed to lift factorization* from the modules to the compound system.

CbN Fixpoint, Head Factorization. We first consider the calculus $\beta Y := (\Lambda_Y, \rightarrow_\beta \cup \rightarrow_Y)$ [26][3]. It extends the CbN λ-calculus with a constant Y and a reduction $\rightarrow_Y$, the contextual closure of the rule $Yt \mapsto_Y t(Yt)$. Points 1-3 below are immediate—details in Appendix A.4.

▶ **Proposition 8.1** (Testing head factorization for βY). *$\rightarrow_\beta \cup \rightarrow_Y$ satisfies head factorization:*

1. Head factorization of $\rightarrow_Y$*:* $\mathtt{Fact}(\xrightarrow[\mathsf{h}]{}_{\mathsf{Y}}, \xrightarrow[\neg\mathsf{h}]{}_{\mathsf{Y}})$.
2. Root linear swap*:* $\xrightarrow[\neg\mathsf{h}]{}_\beta \cdot \mapsto_Y \subseteq \xrightarrow[\mathsf{h}]{}_Y \cdot \rightarrow_\beta^*$.
3. Substitutivity*: $\mapsto_Y$ is substitutive.*

CbV Fixpoint, Weak Factorization. We now consider weak factorization and a CbV counterpart of the previous example. We follow Abramsky and McCusker [1], who study a call-by-value PCF with a fixpoint operator Z. Similarly, we extend the CbV λ-calculus with a constant Z and its reduction $\rightarrow_Z$, which is the contextual closure of the rule $Zv \mapsto_Z \lambda x.v(Zv)x$ where v is a value. The calculus $\beta_v Z$ is therefore $(\Lambda_Z, \rightarrow_{\beta_v} \cup \rightarrow_Z)$. Points 1-3 below are all immediate—details in Appendix A.4.

▶ **Proposition 8.2** (Testing weak factorization for $\beta_v Z$). *$\rightarrow_\beta \cup \rightarrow_Z$ satisfies weak factorization:*

1. Weak factorization of $\rightarrow_Z$*:* $\mathtt{Fact}(\xrightarrow[\mathsf{w}]{}_{\mathsf{Z}}, \xrightarrow[\neg\mathsf{w}]{}_{\mathsf{Z}})$.
2. Root linear swap*:* $\xrightarrow[\neg\mathsf{w}]{}_{\beta_v} \cdot \mapsto_Z \subseteq \xrightarrow[\mathsf{w}]{}_Z \cdot \rightarrow_{\beta_v}^*$.
3. Substitutivity*: $\mapsto_Z$ is substitutive.*

9 Further Applications: Probabilistic Calculi

In this paper, we present our technique using examples which are within the familiar language of λ-calculus. However the core of the technique—Theorem 3.4—is independent from a specific syntax. It can be used in calculi whose objects are richer than λ-terms. The probabilistic λ-calculus is a prime example.

A recent line of research is developing probabilistic calculi where evaluation is not limited to a deterministic strategy. Faggian and Ronchi della Rocca [19] define two calculi—$\Lambda_\oplus^{\mathsf{cbv}}$ and $\Lambda_\oplus^{\mathsf{cbn}}$—which model respectively CbV and CbN probabilistic higher-order computation, while being *conservative extensions* of the CbV and CbN λ-calculus. For both calculi confluence and factorization (called *standardization* in [19]) hold. There is however a deep *asymmetry* between the two results. *Confluence* is neatly proved via Hindley-Rosen technique, by relying

[3] Head factorization of βY is easily obtained by a high-level argument, as consequence of left-normality, see Terese [53, Ch. 8.5]. The point that we want to stress here is that the validity of *linear swaps* is not limited to *terminating* reduction, and βY provides a simple, familiar example.

on the fact the β (resp. β_v) reduction is confluent. The proof of *factorization* is instead laborious: the authors define a notion of parallel reduction for the new calculus, and then adapt Takahashi's technique [52]. Leventis work [35] on the call-by-name probabilistic λ-calculus suffers a similar problem. He proves factorization by relying on the finite developments method, but the proof is equally laborious.

Our technique allows for *a neat, concise proof of factorization*, which reduces to only testing a single linear swap, with no need of parallel reductions or finite developments. To prove factorization turns out to be in fact *easier* than proving confluence. The technical details—that is, the definition of the calculus and the proof—are in Appendix B.

10 Conclusions and Discussions

Summary. A well-established approach to model higher-order computation with advanced features, is starting from the call-by-name or call-by-value λ-calculus, and enrich it with new constructs. We propose a sharp technique to establish factorization of a compound system from factorization of its components. As we point out, the natural transposition of Hindley-Rosen technique for confluence does not work here, because the obtained conditions are—in general—not validated by extensions of the λ-calculus. The turning point is the identification of an alternative sufficient condition, called linear swap. Moreover, on common factorization schemes such as head or weak factorization, our technique reduces to a straightforward test. Concretely, we apply our technique to various examples, stressing its independence from common simplifying hypotheses such as confluence, orthogonality, and termination.

Black Box and Elementary Commutations. A key feature of our technique is to take factorization of the core relations—the modules—as *black boxes*. The focus is then on the analysis of the *interaction* between the modules. The benefit is both practical and conceptual: we disentangle the components—and the issues—under study. This is especially appealing when dealing with extensions of the λ-calculus, built on top of β or β_v reduction, because often most of the difficulties come from the higher-order component, that is, β or β_v itself—whose factorization is non-trivial to prove but known to hold—rather than from the added features.

Good illustrations of these points are our proofs of factorization. We stress that:

- the proof of factorization of the compound system is independent from the specific technique (finite developments, parallel reduction, etc.) used to prove factorization of the modules.
- to verify good interaction between the modules, it often suffices to check *elementary, local commutations*—the linear swaps.

These features provide a neat proof-technique *supporting the development and the analysis* of complex compound systems.

Conclusions. When one wants to model new computational features, the calculus is often not given, but it has to be designed, in such a way that it satisfies confluence and factorization. The process of *developing* the calculus and the process of *proving* its good properties, go hand in hand. If the latter is difficult and prone to errors, the former also is. The black-box approach makes our technique efficient and accessible also to working scientists who are not specialists in rewriting. And even for the λ-calculus expert who masters tools such as finite developments, labeling or parallel reduction, it still appears desirable to limit the amount of difficulties. The more advanced and complex are the computational systems we study, the more crucial it is to have reasoning tools as simple to use as possible.

References

1 Samson Abramsky and Guy McCusker. Call-by-value games. In *Computer Science Logic, 11th International Workshop, CSL '97, Annual Conference*, volume 1414 of *Lecture Notes in Computer Science*, pages 1–17. Springer, 1997. doi:10.1007/BFb0028004.

2 Beniamino Accattoli. An abstract factorization theorem for explicit substitutions. In *23rd International Conference on Rewriting Techniques and Applications, RTA 2012*, volume 15 of *LIPIcs*, pages 6–21, 2012. doi:10.4230/LIPIcs.RTA.2012.6.

3 Beniamino Accattoli, Claudia Faggian, and Giulio Guerrieri. Factorization and normalization, essentially. In *Programming Languages and Systems - 17th Asian Symposium, APLAS 2019, Proceedings*, volume 11893 of *Lecture Notes in Computer Science*, pages 159–180. Springer, 2019. doi:10.1007/978-3-030-34175-6_9.

4 Beniamino Accattoli and Giulio Guerrieri. Open call-by-value. In *Programming Languages and Systems - 14th Asian Symposium, APLAS 2016, Proceedings*, volume 10017 of *Lecture Notes in Computer Science*, pages 206–226, 2016. doi:10.1007/978-3-319-47958-3_12.

5 Yohji Akama. On Mints' reduction for ccc-calculus. In *Typed Lambda Calculi and Applications, International Conference on Typed Lambda Calculi and Applications, TLCA '93, Utrecht, The Netherlands, March 16-18, 1993, Proceedings*, pages 1–12, 1993. doi:10.1007/BFb0037094.

6 Zena M. Ariola, Matthias Felleisen, John Maraist, Martin Odersky, and Philip Wadler. The call-by-need lambda calculus. In Ron K. Cytron and Peter Lee, editors, *Conference Record of POPL'95: 22nd ACM Symposium on Principles of Programming Languagesn*, pages 233–246. ACM Press, 1995. doi:10.1145/199448.199507.

7 Pablo Arrighi and Gilles Dowek. Lineal: A linear-algebraic lambda-calculus. *Log. Methods Comput. Sci.*, 13(1), 2017. doi:10.23638/LMCS-13(1:8)2017.

8 Martin Avanzini, Ugo Dal Lago, and Akihisa Yamada. On probabilistic term rewriting. In *Functional and Logic Programming - 14th International Symposium, FLOPS 2018*, volume 10818 of *Lecture Notes in Computer Science*, pages 132–148. Springer, 2018. doi:10.1007/978-3-319-90686-7_9.

9 Franz Baader and Tobias Nipkow. *Term rewriting and all that.* Cambridge University Press, 1998.

10 Leo Bachmair and Nachum Dershowitz. Commutation, transformation, and termination. In *8th International Conference on Automated Deduction, CADE 1986*, volume 230 of *Lecture Notes in Computer Science*, pages 5–20. Springer, 1986. doi:10.1007/3-540-16780-3_76.

11 Hendrik Pieter Barendregt. *The Lambda Calculus – Its Syntax and Semantics*, volume 103 of *Studies in logic and the foundations of mathematics.* North-Holland, 1984.

12 Frédéric Blanqui. Size-based termination of higher-order rewriting. *J. Funct. Program.*, 28:e11, 2018. doi:10.1017/S0956796818000072.

13 Antonio Bucciarelli, Delia Kesner, and Simona Ronchi Della Rocca. Observability = typability + inhabitation. *CoRR*, 2018. URL: http://arxiv.org/abs/1812.06009.

14 Alberto Carraro and Giulio Guerrieri. A semantical and operational account of call-by-value solvability. In *Foundations of Software Science and Computation Structures, 17th International Conference, FoSSaCS 2014*, volume 8412 of *Lecture Notes in Computer Science*, pages 103–118, 2014. doi:10.1007/978-3-642-54830-7_7.

15 Haskell B. Curry and Robert Feys. *Combinatory Logic, Volume 1.* Studies in logic and the foundations of mathematics. North-Holland, 1958.

16 Ugo de' Liguoro and Adolfo Piperno. Non deterministic extensions of untyped lambda-calculus. *Inf. Comput.*, 122(2):149–177, 1995. doi:10.1006/inco.1995.1145.

17 Nachum Dershowitz. On lazy commutation. In *Languages: From Formal to Natural, Essays Dedicated to Nissim Francez on the Occasion of His 65th Birthday*, pages 59–82, 2009. doi:10.1007/978-3-642-01748-3_5.

18 Henk Doornbos and Burghard von Karger. On the union of well-founded relations. *Logic Journal of the IGPL*, 6(2):195–201, 1998. doi:10.1093/jigpal/6.2.195.

19 Claudia Faggian and Simona Ronchi Della Rocca. Lambda calculus and probabilistic computation. In *34th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2019, Vancouver, BC, Canada, June 24-27, 2019*, pages 1–13, 2019. doi:10.1109/LICS.2019.8785699.

20 Alfons Geser. *Relative Termination.* PhD thesis, University of Passau, Germany, 1990. URL: http://vts.uni-ulm.de/docs/2012/8146/vts_8146_11884.pdf.

21 Georges Gonthier, Jean-Jacques Lévy, and Paul-André Melliès. An abstract standardisation theorem. In *Proceedings of the Seventh Annual Symposium on Logic in Computer Science (LICS '92)*, pages 72–81. IEEE Computer Society, 1992. doi:10.1109/LICS.1992.185521.

22 Bernhard Gramlich. Modularity in term rewriting revisited. *Theor. Comput. Sci.*, 464:3–19, 2012. doi:10.1016/j.tcs.2012.09.008.

23 Giulio Guerrieri. Head reduction and normalization in a call-by-value lambda-calculus. In *2nd International Workshop on Rewriting Techniques for Program Transformations and Evaluation, WPTE 2015*, volume 46 of *OASICS*, pages 3–17. Schloss Dagstuhl, 2015. doi:10.4230/OASIcs.WPTE.2015.3.

24 Giulio Guerrieri, Luca Paolini, and Simona Ronchi Della Rocca. Standardization and conservativity of a refined call-by-value lambda-calculus. *Logical Methods in Computer Science*, 13(4), 2017. doi:10.23638/LMCS-13(4:29)2017.

25 J. Roger Hindley. *The Church-Rosser Property and a Result in Combinatory Logic.* PhD thesis, University of Newcastle-upon-Tyne, 1964.

26 J. Roger Hindley. Reductions of residuals are finite. *Transactions of the American Mathematical Society*, 240:345–361, 1978.

27 J. Roger Hindley and Jonathan P. Seldin. *Lambda-Calculus and Combinators: An Introduction.* Cambridge University Press, New York, NY, USA, 2 edition, 2008.

28 Nao Hirokawa, Aart Middeldorp, and Georg Moser. Leftmost outermost revisited. In *26th International Conference on Rewriting Techniques and Applications, RTA 2015*, volume 36 of *LIPIcs*, pages 209–222. Schloss Dagstuhl, 2015. doi:10.4230/LIPIcs.RTA.2015.209.

29 Katsumasa Ishii. A proof of the leftmost reduction theorem for $\lambda\beta\eta$-calculus. *Theor. Comput. Sci.*, 747:26–32, 2018. doi:10.1016/j.tcs.2018.06.003.

30 Jan Willem Klop. *Combinatory Reduction Systems.* Phd thesis, Mathematisch Centrum, Amsterdam, 1980.

31 Masahito Kurihara and Ikuo Kaji. Modular term rewriting systems and the termination. *Inf. Process. Lett.*, 34(1):1–4, 1990. doi:10.1016/0020-0190(90)90221-I.

32 Ugo Dal Lago and Simone Martini. The weak lambda calculus as a reasonable machine. *Theor. Comput. Sci.*, 398(1-3):32–50, 2008. doi:10.1016/j.tcs.2008.01.044.

33 Ugo Dal Lago and Margherita Zorzi. Probabilistic operational semantics for the lambda calculus. *RAIRO Theor. Informatics Appl.*, 46(3):413–450, 2012. doi:10.1051/ita/2012012.

34 Xavier Leroy. The ZINC experiment: an economical implementation of the ML language. Technical report 117, INRIA, 1990. URL: http://gallium.inria.fr/~xleroy/publi/ZINC.pdf.

35 Thomas Leventis. A deterministic rewrite system for the probabilistic λ-calculus. *Math. Struct. Comput. Sci.*, 29(10):1479–1512, 2019. doi:10.1017/S0960129519000045.

36 Jean-Jacques Lévy. *Réductions corrcectes et optimales dans le lambda calcul.* PhD thesis, University of Paris 7, 1978.

37 Paul-André Melliès. Typed lambda-calculi with explicit substitutions may not terminate. In *Typed Lambda Calculi and Applications, Second International Conference on Typed Lambda Calculi and Applications, TLCA '95*, volume 902 of *Lecture Notes in Computer Science*, pages 328–334. Springer, 1995. doi:10.1007/BFb0014062.

38 Paul-André Melliès. A factorisation theorem in rewriting theory. In *Category Theory and Computer Science, 7th International Conference, CTCS '97*, volume 1290 of *Lecture Notes in Computer Science*, pages 49–68, 1997. doi:doi.org/10.1007/BFb0026981.

39 Aart Middeldorp. Modular aspects of properties of term rewriting systems related to normal forms. In *Rewriting Techniques and Applications, 3rd International Conference, RTA-89*,

volume 355 of *Lecture Notes in Computer Science*, pages 263–277. Springer, 1989. doi:10.1007/3-540-51081-8_113.

40 Aart Middeldorp. A sufficient condition for the termination of the direct sum of term rewriting systems. In *Proceedings of the Fourth Annual Symposium on Logic in Computer Science (LICS '89)*, pages 396–401. IEEE Computer Society, 1989. doi:10.1109/LICS.1989.39194.

41 Aart Middeldorp. Confluence of the disjoint union of conditional term rewriting systems. In *Conditional and Typed Rewriting Systems, 2nd International Workshop, CTRS 1990f*, volume 516 of *Lecture Notes in Computer Science*, pages 295–306. Springer, 1990. doi:10.1007/3-540-54317-1_99.

42 Gerd Mitschke. The standardization theorem for λ-calculus. *Mathematical Logic Quarterly*, 25(1-2):29–31, 1979. doi:10.1002/malq.19790250104.

43 Luca Paolini. Call-by-value separability and computability. In *Theoretical Computer Science, 7th Italian Conference, ICTCS 2001*, volume 2202 of *Lecture Notes in Computer Science*, pages 74–89. Springer, 2001. doi:10.1007/3-540-45446-2_5.

44 Luca Paolini and Simona Ronchi Della Rocca. Call-by-value solvability. *RAIRO Theor. Informatics Appl.*, 33(6):507–534, 1999. doi:10.1051/ita:1999130.

45 Gordon D. Plotkin. Call-by-name, call-by-value and the lambda-calculus. *Theor. Comput. Sci.*, 1(2):125–159, 1975. doi:10.1016/0304-3975(75)90017-1.

46 Laurent Regnier. Une équivalence sur les lambda-termes. *Theor. Comput. Sci.*, 126(2):281–292, 1994. doi:10.1016/0304-3975(94)90012-4.

47 György E. Révész. A list-oriented extension of the lambda-calculus satisfying the church-rosser theorem. *Theor. Comput. Sci.*, 93(1):75–89, 1992. doi:10.1016/0304-3975(92)90212-X.

48 Simona Ronchi Della Rocca and Luca Paolini. *The Parametric Lambda Calculus - A Metamodel for Computation*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2004. URL: https://doi.org/10.1007/978-3-662-10394-4.

49 J. Barkley Rosser. A mathematical logic without variables. *Duke Mathematical Journal*, 1(3):328–355, 09 1935. doi:10.1215/S0012-7094-35-00123-5.

50 Michaël Rusinowitch. On termination of the direct sum of term-rewriting systems. *Inf. Process. Lett.*, 26(2):65–70, 1987. doi:10.1016/0020-0190(87)90039-1.

51 Alexis Saurin. On the relations between the syntactic theories of lambda-mu-calculi. In *Computer Science Logic, 22nd International Workshop, CSL 2008, 17th Annual Conference*, pages 154–168, 2008. doi:10.1007/978-3-540-87531-4_13.

52 Masako Takahashi. Parallel reductions in lambda-calculus. *Inf. Comput.*, 118(1):120–127, 1995. doi:10.1006/inco.1995.1057.

53 Terese. *Term Rewriting Systems*, volume 55 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 2003.

54 Yoshihito Toyama. Counterexamples to termination for the direct sum of term rewriting systems. *Inf. Process. Lett.*, 25(3):141–143, 1987. doi:10.1016/0020-0190(87)90122-0.

55 Yoshihito Toyama. On the church-rosser property for the direct sum of term rewriting systems. *J. ACM*, 34(1):128–143, 1987. doi:10.1145/7531.7534.

56 Yoshihito Toyama, Jan Willem Klop, and Hendrik Pieter Barendregt. Termination for the direct sum of left-linear term rewriting systems -preliminary draft-. In *Rewriting Techniques and Applications, 3rd International Conference, RTA-89*, volume 355 of *Lecture Notes in Computer Science*, pages 477–491. Springer, 1989. doi:10.1007/3-540-51081-8_127.

57 Vincent Van Oostrom. Some symmetries of commutation diamonds. Talk at the International Workshop on Confluence, 30 June 2020.

58 Vincent van Oostrom. Confluence by decreasing diagrams. In *Rewriting Techniques and Applications, 19th International Conference, RTA 2008,*, volume 5117 of *Lecture Notes in Computer Science*, pages 306–320. Springer, 2008. doi:10.1007/978-3-540-70590-1_21.

59 Vincent van Oostrom and Yoshihito Toyama. Normalisation by random descent. In *1st International Conference on Formal Structures for Computation and Deduction, FSCD 2016*,

volume 52 of *LIPIcs*, pages 32:1–32:18. Schloss Dagstuhl, 2016. `doi:10.4230/LIPIcs.FSCD.2016.32`.

60 Vincent van Oostrom and Hans Zantema. Triangulation in rewriting. In *23rd International Conference on Rewriting Techniques and Applications (RTA'12) , RTA 2012, May 28 - June 2, 2012, Nagoya, Japan*, pages 240–255, 2012. `doi:10.4230/LIPIcs.RTA.2012.240`.

APPENDIX

Appendix A collects omitted details of proofs. In Appendix B we illustrate a more *advanced example* of application of our technique, namely to the *probabilistic λ-calculus.*

A Appendix: Omitted Proofs

A.1 Head Factorization (Sect. 4)

Consequences of Property 4.2. Note that the empty context $\langle\,\rangle$ is a *head context*. Hence, for a non-head step $\mathsf{C}\langle r\rangle \underset{\neg\mathsf{h}}{\to} \mathsf{C}\langle r'\rangle$, necessarily $\mathsf{C} \neq \langle\,\rangle$, and Property 4.2 applies. Consequently:

▶ **Property A.1** (Shape preservation). *By Property 4.2, $\underset{\neg\mathsf{h}}{\to}_\gamma$ preserves the shapes of terms:*
1. Atoms*: there is no t such that $t \underset{\neg\mathsf{h}}{\to}_\gamma a$, for any variable or constant a.*
2. *$t \underset{\neg\mathsf{h}}{\to}_\gamma \lambda x.u_1$ implies $t = \lambda x.t_1$ and $t_1 \underset{\neg\mathsf{h}}{\to}_\gamma u_1$.*
3. *$t \underset{\neg\mathsf{h}}{\to}_\gamma u_1u_2$ implies $t = t_1t_2$, with $t_1 \underset{\neg\mathsf{h}}{\to}_\gamma u_1$ (and $t_2 = u_2$), or $t_2 \to_\gamma u_2$ (and $t_1 = u_1$).*
4. Redex*: if $t \underset{\neg\mathsf{h}}{\to}_\gamma u$, and u is a β-redex, then t is a β-redex.*
   Similarly, if $t \underset{\neg\mathsf{h}}{\to}_\gamma u$ and u has shape $Kt_1...t_k$ (K a constant), t has the same shape.

Point 4. follows from points 1. to 3. Note that, in particular, if u is a $\oplus$-redex, so is t.

Head Factorization, Modularly.

▶ **Lemma** (4.3, Lifting root linear swaps). *Let $\mapsto_\alpha, \mapsto_\gamma$ be root relations on Λ.*
1. *$\underset{\neg\mathsf{h}}{\to}_\alpha \cdot \mapsto_\gamma \subseteq \underset{\mathsf{h}}{\to}_\gamma \cdot \to_\alpha^*$ implies* $\mathtt{lSwap}(\underset{\neg\mathsf{h}}{\to}_\alpha, \underset{\mathsf{h}}{\to}_\gamma)$.
2. *Similarly, $\underset{\neg\mathsf{h}}{\to}_\alpha \cdot \mapsto_\gamma \subseteq \underset{\mathsf{h}}{\to}_\gamma \cdot \to_\alpha^=$ implies $\underset{\neg\mathsf{h}}{\to}_\alpha \cdot \to_\gamma \subseteq \underset{\mathsf{h}}{\to}_\gamma \cdot \to_\alpha^=$.*

Proof.
1. We prove that $t \underset{\neg\mathsf{h}}{\to}_\alpha u \underset{\mathsf{h}}{\to}_\gamma s$ implies $t \underset{\mathsf{h}}{\to}_\gamma \cdot \to_\alpha^* s$ by induction on the head context $\mathsf{H} = \lambda x_1 \dots \lambda x_k.\langle\,\rangle t_1 \dots t_n$ of the reduction step $u \underset{\mathsf{h}}{\to}_\gamma s$. Cases:
   a. u is a γ-redex (*i.e.* $\mathsf{H} = \langle\,\rangle$, $k = 0$, $n = 0$) and hence $t \underset{\neg\mathsf{h}}{\to}_\alpha u \mapsto_\gamma s$. Thus, $t \underset{\mathsf{h}}{\to}_\gamma \cdot \to_\alpha^* s$ by the hypothesis $\underset{\neg\mathsf{h}}{\to}_\alpha \cdot \mapsto_\gamma \subseteq \underset{\mathsf{h}}{\to}_\gamma \cdot \to_\alpha^*$.
   b. $u = \lambda x.u_1$ (*i.e.* $k > 0$) with $u_1 \underset{\mathsf{h}}{\to}_\gamma s_1$ and $u = \lambda x.u_1 \underset{\mathsf{h}}{\to}_\gamma \lambda x.s_1 = s$. By shape preservation (Property A.1.2) $t = \lambda x.t_1$ and $t_1 \underset{\neg\mathsf{h}}{\to}_\alpha u_1$. By *i.h.*, we can conclude.
   c. $u = u_1u_2$ (*i.e.*, $k = 0$, $n > 0$) with $u_1 \underset{\mathsf{h}}{\to}_\gamma u_1'$ and $u = u_1u_2 \underset{\mathsf{h}}{\to}_\gamma u_1'u_2 = s$. By shape preservation (Property A.1.3), there are two sub-cases for $t \underset{\neg\mathsf{h}}{\to}_\alpha u_1u_2$.
      i. $t = t_1u_2$ and $t_1 \underset{\neg\mathsf{h}}{\to}_\alpha u_1$. By *i.h.*, $t_1 \underset{\mathsf{h}}{\to}_\gamma t_1' \to_\alpha^* u_1'$ so $t = t_1u_2 \underset{\mathsf{h}}{\to}_\gamma t_1'u_2 \to_\alpha^* u_1'u_2 = s$.
      ii. $t = u_1t_2$ and $t_2 \to_\alpha u_2$. Then, $t = u_1t_2 \underset{\mathsf{h}}{\to}_\gamma u_1't_2 \to_\alpha u_1'u_2 = s$.
2. The proof is similar to Point 1. ◀

▶ **Lemma** (4.4, Swap with $\underset{\mathsf{h}}{\to}_\beta$). *If $\mapsto_\gamma$ is substitutive then* $\mathtt{lSwap}(\underset{\neg\mathsf{h}}{\to}_\gamma, \underset{\mathsf{h}}{\to}_\beta)$ *holds.*

Proof. First, note that by Property 4.1, we have that $\to_\gamma$ is substitutive. We prove ($t \underset{\neg\mathsf{h}}{\to}_\gamma u \underset{\mathsf{h}}{\to}_\beta s$ implies $t \underset{\mathsf{h}}{\to}_\beta \cdot \to_\gamma^* s$) by using Lemma 4.3.

Let $u = (\lambda x.u_1)u_2 \mapsto_\beta u_1\{x{:=}u_2\}$. By Property A.1, either (i) $t = (\lambda x.p)u_2$ and $p \underset{\neg\mathsf{h}}{\to}_\gamma u_1$ or (ii) $t = (\lambda x.u_1)q$ and $q \to_\gamma u_2$. Case (i): $(\lambda x.p)u_2 \underset{\mathsf{h}}{\to}_\beta p\{x{:=}u_2\}$. By Property 4.1 (point 1) $p\{x{:=}u_2\} \to_\gamma u_1\{x{:=}u_2\}$. Case (ii): $(\lambda x.u_1)q \underset{\mathsf{h}}{\to}_\beta u_1\{x{:=}q\}$. By using Property 4.1 (point 2), $u_1\{x{:=}q\} \to_\gamma^* u_1\{x{:=}u_2\}$. ◀

A.2 Call-by-Value λ-Calculus (Sect. 6)

Consequences of Property 4.2. The empty context $\langle\rangle$ is *both a left and a weak context.* Hence Property 4.2 always applies to non-left and non-weak steps. Consequently:

▶ **Property A.2** (Shape preservation). *Fixed* $\mathsf{e} \in \{\mathsf{l}, \mathsf{w}\}$, $\underset{\neg\mathsf{e}}{\to}_\gamma$ *preserves the shape of terms:*

1. Atoms*: there is no t such that $t \underset{\neg\mathsf{e}}{\to}_\gamma a$, for any variable or constant a;*
2. Abstraction*: $t \underset{\neg\mathsf{e}}{\to}_\gamma \lambda x.u_1$ implies $t = \lambda x.t_1$ and $t_1 \to_\gamma u_1$;*
3. Application*: $t \underset{\neg\mathsf{e}}{\to}_\gamma u_1u_2$ implies $t = t_1t_2$, with either (i) $t_1 \underset{\neg\mathsf{e}}{\to}_\gamma u_1$ and $t_2 = u_2$, or (ii) $t_2 \to_\gamma u_2$ and $t_1 = u_1$. Moreover, in case (ii): if $vt_2 \underset{\neg\mathsf{l}}{\to} vu_2$ (v a value), then $t_2 \underset{\neg\mathsf{l}}{\to} u_2$; if $t_1t_2 \underset{\neg\mathsf{w}}{\to}_\gamma u_1u_2$, then $t_2 \underset{\neg\mathsf{w}}{\to}_\gamma u_2$ (always).*
4. Redex*: if $t \underset{\neg\mathsf{e}}{\to}_\gamma u$, and u is a β_v-redex, then t also is (as a consequence of points 1. to 3.)*

Left and Weak Factorization, Modularly. To prove Proposition 6.1 we proceed similarly to Sect. 4.2.

▶ **Lemma A.3** (Lifting root linear swaps). *Let $\mapsto_\alpha, \mapsto_\gamma$ be root relations on Λ.*

1. *If $\underset{\neg\mathsf{l}}{\to}_\alpha \cdot \mapsto_\gamma \subseteq \underset{\mathsf{l}}{\to}_\gamma \cdot \to_\alpha^*$ then* `lSwap`$(\underset{\neg\mathsf{l}}{\to}_\alpha, \underset{\mathsf{l}}{\to}_\gamma)$.
2. *If $\underset{\neg\mathsf{w}}{\to}_\alpha \cdot \mapsto_\gamma \subseteq \underset{\mathsf{w}}{\to}_\gamma \cdot \to_\alpha^*$ then* `lSwap`$(\underset{\neg\mathsf{w}}{\to}_\alpha, \underset{\mathsf{w}}{\to}_\gamma)$.
3. *Similarly, $\underset{\neg\mathsf{e}}{\to}_\alpha \cdot \mapsto_\gamma \subseteq \underset{\mathsf{e}}{\to}_\gamma \cdot \to_\alpha^=$ implies $\underset{\neg\mathsf{e}}{\to}_\alpha \cdot \to_\gamma \subseteq \underset{\mathsf{e}}{\to}_\gamma \cdot \to_\alpha^=$, with $\mathsf{e} \in \{\mathsf{l}, \mathsf{w}\}$*

Proof.

1. We prove that $t \underset{\neg\mathsf{l}}{\to}_\alpha u \underset{\mathsf{l}}{\to}_\gamma s$ implies $t \underset{\mathsf{l}}{\to}_\gamma \cdot \to_\alpha^* s$, by induction on the context L of $u \underset{\mathsf{l}}{\to}_\gamma s$.
   a. $\mathsf{L} = \langle\rangle$, *i.e.* $u \mapsto_\gamma s$. The claim holds by hypothesis.
   b. $\mathsf{L} = \mathsf{L}'u_2$, *i.e.* $u = u_1u_2 \underset{\mathsf{l}}{\to}_\gamma s_1u_2 = s$ with $u_1 \underset{\mathsf{l}}{\to}_\gamma s_1$. By Property A.2.3, $t = t_1t_2$ and
      i. either $t_2 \to_\alpha u_2$ (and $t_1 = u_1$), so $t = u_1t_2 \underset{\mathsf{l}}{\to}_\gamma s_1t_2 \to_\alpha s_1u_2 = s$;
      ii. or $t_1 \underset{\neg\mathsf{l}}{\to}_\alpha u_1$ (and $t_2 = u_2$); by *i.h.*, $t_1 \underset{\mathsf{l}}{\to}_\gamma \cdot \to_\alpha^* s_1$, so $t = t_1u_2 \underset{\mathsf{l}}{\to}_\gamma \cdot \to_\alpha^* s_1u_2 = s$.
   c. $\mathsf{L} = v\mathsf{L}'$, *i.e.* $u = vu_2 \underset{\mathsf{l}}{\to}_\gamma vs_2 = s$ with $u_2 \underset{\mathsf{l}}{\to}_\gamma s_2$. By Property A.2.3, $t = t_1t_2$ and
      i. either $t_2 \underset{\neg\mathsf{l}}{\to}_\alpha u_2$ (and $t_1 = v$); by *i.h.*, $t_2 \underset{\mathsf{l}}{\to}_\gamma \cdot \to_\alpha^* s_2$, so $t = vt_2 \underset{\mathsf{l}}{\to}_\gamma \cdot \to_\alpha^* vs_2 = s$;
      ii. or $t_1 \underset{\neg\mathsf{l}}{\to}_\alpha v$ (and $t_2 = u_2$); since v is a value, by Property A.2.1-2, t_1 is also a value and so $t = t_1u_2 \underset{\mathsf{l}}{\to}_\gamma t_1s_2 \to_\alpha vs_2 = s$.
2. The proof is similar Point 1, but simpler. Case $\mathsf{W} = \langle\rangle$ is the same. Case $\mathsf{W} = \mathsf{W}'u_2$ is exactly like $\mathsf{L} = \mathsf{L}'u_2$, and case $\mathsf{W} = u_1\mathsf{W}'$ is symmetric to case $\mathsf{W} = \mathsf{W}'u_2$.
3. The proof is similar to Points 1-2. ◀

▶ **Lemma A.4** (Swap with $\underset{\mathsf{e}}{\to}_{\beta_v}$). *If $\mapsto_\gamma$ is substitutive then* `lSwap`$(\underset{\neg\mathsf{e}}{\to}_\gamma, \underset{\mathsf{e}}{\to}_{\beta_v})$, *for* $\mathsf{e} \in \{\mathsf{l}, \mathsf{w}\}$.

Proof. We prove `lSwap`$(\underset{\neg\mathsf{l}}{\to}_\gamma, \underset{\mathsf{l}}{\to}_{\beta_v})$, the other swap is similar. By Lemma A.3, it is enough to prove that $t \underset{\neg\mathsf{l}}{\to}_\gamma u \mapsto_{\beta_v} s$ implies $t \underset{\mathsf{l}}{\to}_{\beta_v} \cdot \to_\gamma^* s$. We use Property 4.1 (substitutivity). Let $u = (\lambda x.u_1)v \mapsto_{\beta_v} u_1\{x{:=}v\} = s$. By Property A.2 (3. and 4.) we have:

1. either $t = (\lambda x.t_1)v$ and $t_1 \to_\gamma u_1$; thus, $t = (\lambda x.t_1)v \underset{\mathsf{l}}{\to}_{\beta_v} t_1\{x{:=}v\} \to_\gamma u_1\{x{:=}v\} = s$, where the $\to_\gamma$ step takes place by Property 4.1.1 since $\to_\gamma$ is substitutive.
2. or $t = (\lambda x.u_1)w$ where $w \to_\gamma v$ and w is a value; hence, $t = (\lambda x.u_1)w \underset{\mathsf{l}}{\to}_{\beta_v} u_1\{x{:=}w\} \to_\gamma^* u_1\{x{:=}v\}$, where the $\to_\gamma$ steps take place by Property 4.1.2. ◀

A.3 The Shuffling Calculus (Sect. 7)

▶ **Property A.5** (Values are closed under substitution). *If v and w are values, so is $v\{x{:=}w\}$.*

▶ **Lemma** (Lemma 7.1, Root linear swaps). *Let $\mathsf{e} \in \{\mathsf{l}, \mathsf{w}\}$ and $i \in \{1, 3\}$. Then:*
1. $\underset{\neg\mathsf{e}}{\to}_{\beta_v} \cdot \mapsto_{\sigma_i} \subseteq \mapsto_{\sigma_i} \cdot \to_{\beta_v}$.
2. $\underset{\neg\mathsf{e}}{\to}_{\sigma} \cdot \mapsto_{\sigma_i} \subseteq \mapsto_{\sigma_i} \cdot \to_{\sigma}$.

Proof. We prove the following, more general, statement: Let $\mathsf{e} \in \{\mathsf{l}, \mathsf{w}\}$ and $i \in \{1, 3\}$, let $\to_\gamma$ be the contextual closure of *any* rule $\mapsto_\gamma$ on Λ. Then, $t \underset{\neg\mathsf{e}}{\to}_{\gamma} u \mapsto_{\sigma_i} s \subseteq t \mapsto_{\sigma_i} \cdot \to_\gamma s$.

First, let us consider $\mathsf{e} = \mathsf{l}$. We examine the two cases for $u \mapsto_{\sigma_i} s$ ($i = 1$ or $i = 3$).

σ_1: By hypothesis, $u = (\lambda x.q)pr \mapsto_{\sigma_1} (\lambda x.qr)p = s$ with $x \notin \mathsf{fv}(r)$. We can assume $x \notin \mathsf{fv}(t)$. Since $t \underset{\neg\mathsf{l}}{\to}_{\gamma} u$, by Property A.2 (iterated), we have $t = (\lambda x.q')p'r'$ and moreover:
- either $q' \to_\gamma q$ and $r' = r$ and $p' = p$,
- or $r' \to_\gamma r$ and $q' = q$ and $p' = p$,
- or $p' \underset{\neg\mathsf{l}}{\to}_{\gamma} p$ and $q' = q$ and $r' = r$.

In any case, $t = (\lambda x.q')p'r' \mapsto_{\sigma_1} (\lambda x.q'r')p' \underset{\neg\mathsf{l}}{\to}_{\gamma} (\lambda x.qr)p = s$, since $x \notin \mathsf{fv}(t) \supseteq \mathsf{fv}(r')$.

σ_3: By hypothesis, $u = v((\lambda x.r)p) \mapsto_{\sigma_3} (\lambda x.vr)p = s$ with $x \notin \mathsf{fv}(v)$. We can assume $x \notin \mathsf{fv}(t)$. As $t \underset{\neg\mathsf{l}}{\to}_{\gamma} u$, by Property A.2 (iterated), we have $t = v'((\lambda x.r')p')$ and moreover:
- either $v' \underset{\neg\mathsf{l}}{\to}_{\gamma} v$ and $r' = r$ and $p' = p$,
- or $r' \to_\gamma r$ and $v' = v$ and $p' = p$,
- or $p' \underset{\neg\mathsf{l}}{\to}_{\gamma} p$ and $v' = v$ and $r' = r$.

In any case, $t = v'((\lambda x.r')p') \mapsto_{\sigma_3} (\lambda x.v'r')p' \underset{\neg\mathsf{l}}{\to}_{\gamma} (\lambda x.vr)p = s$, as $x \notin \mathsf{fv}(t) \supseteq \mathsf{fv}(v')$.

As usual, the proof in the case $\mathsf{e} = \mathsf{w}$ is similar, and simpler. ◀

▶ **Lemma A.6** (Substitutivity of $\to_\sigma$). *If $t \mapsto_{\sigma_i} t'$ then $t\{x{:=}v\} \mapsto_{\sigma_i} t'\{x{:=}v\}$, for $i \in \{1, 3\}$.*

Proof. σ_1: $t = (\lambda y.r)su \mapsto_{\sigma_1} (\lambda x.ru)s = t'$ with $y \notin \mathsf{fv}(u)$ and we can suppose without loss of generality that $y \notin \mathsf{fv}(v) \cup \{x\}$. Therefore, $t\{x{:=}v\} = (\lambda y.r\{x{:=}v\})s\{x{:=}v\}u\{x{:=}v\} \mapsto_{\sigma_1} (\lambda y.r\{x{:=}v\}u\{x{:=}v\})s\{x{:=}v\} = t'\{x{:=}v\}$ since $y \notin (\mathsf{fv}(u) \smallsetminus \{x\}) \cup \mathsf{fv}(v) = \mathsf{fv}(u\{x{:=}v\})$.

σ_3: $t = w((\lambda y.u)s) \mapsto_{\sigma_3} (\lambda y.wu)s = t'$ with $y \notin \mathsf{fv}(w)$ and we can suppose without loss of generality that $y \notin \mathsf{fv}(v) \cup \{x\}$. Therefore, $t\{x{:=}v\} = w((\lambda y.u\{x{:=}v\})s\{x{:=}v\}) \mapsto_{\sigma_3} (\lambda y.w\{x{:=}v\}u\{x{:=}v\})s\{x{:=}v\} = t'\{x{:=}v\}$ as $w\{x{:=}v\}$ is a value (Property A.5) and $y \notin (\mathsf{fv}(w) \smallsetminus \{x\}) \cup \mathsf{fv}(v) = \mathsf{fv}(w\{x{:=}v\})$. ◀

A.4 Non-Terminating Relations (Sect. 8)

▶ **Prop** (8.1. Testing head factorization for βY). *$\to_\beta \cup \to_Y$ satisfies head factorization:*
1. Head factorization of $\to_Y$*:* $\mathtt{Fact}(\underset{\mathsf{h}}{\to}_{\mathsf{Y}}, \underset{\neg\mathsf{h}}{\to}_{\mathsf{Y}})$.
2. Root linear swap*:* $\underset{\neg\mathsf{h}}{\to}_{\beta} \cdot \mapsto_Y \subseteq \underset{\mathsf{h}}{\to}_{Y} \cdot \to_\beta^*$.
3. Substitutivity*: $\mapsto_Y$ is substitutive.*

Proof. We verify the hypotheses of Proposition 4.5:
1. To verify that the reduction $\to_Y$ satisfies head factorization is routine.
2. Assume $t \underset{\neg\mathsf{h}}{\to}_{\beta} Yp \mapsto_Y p(Yp)$. By Property 4.2 (as spelled-out in Property A.1), if $t \underset{\neg\mathsf{h}}{\to}_{\beta} Yp$ then $t = Yq$ and $q \to_\beta p$. Hence $t = Yq \underset{\mathsf{h}}{\to}_{Y} q(Yq) \to_\beta^* p(Yp)$.
3. Simply $(Yp)\{x{:=}q\} = Y(p\{x{:=}q\}) \mapsto_Y (p\{x{:=}q\})(Y(p\{x{:=}q\})) = (p(Yp))\{x{:=}q\}$. ◀

▶ **Prop** (8.2. Testing weak factorization for $\beta_v Z$). *$\to_\beta \cup \to_Z$ satisfies weak factorization:*

1. Weak factorization of $\rightarrow_Z$*:* $\mathtt{Fact}(\underset{\mathsf{w}}{\rightarrow}_{\mathsf{Z}}, \underset{\neg\mathsf{w}}{\rightarrow}_{\mathsf{Z}})$.
2. Root linear swap*:* $\underset{\neg\mathsf{w}}{\rightarrow}_{\beta_v} \cdot \mapsto_Z \subseteq \underset{\mathsf{w}}{\rightarrow}_Z \cdot \rightarrow^*_{\beta_v}$.
3. Substitutivity*:* $\mapsto_Z$ *is substitutive.*

Proof. We prove the hypotheses of Proposition 4.5:
1. It is easy to verify that $\underset{\neg\mathsf{w}}{\rightarrow}_Z \cdot \underset{\mathsf{w}}{\rightarrow}_Z \subseteq \underset{\mathsf{w}}{\rightarrow}_Z \cdot \underset{\neg\mathsf{w}}{\rightarrow}_Z{}^*$. Then apply Lemma 3.5.1.
2. Assume $t \underset{\neg\mathsf{w}}{\rightarrow}_{\beta_v} Zv \mapsto_Z \lambda x.v(Zv)x$. By Property 4.2 (as spelled-out in Property A.2), if $t \underset{\neg\mathsf{w}}{\rightarrow}_{\beta_v} Zv$ then $t = Zw$ and $w \rightarrow_{\beta_v} v$. So, $t = Zw \underset{\mathsf{w}}{\rightarrow}_Z \lambda x.w(Zw)x \rightarrow^*_{\beta_v} \lambda x.v(Zv)x$.
3. Simply $(Zv)\{x{:=}q\} = Z(v\{x{:=}q\}) \mapsto_Z \lambda y.v\{x{:=}q\}(Z(v\{x{:=}q\})y = (\lambda y.v(Zv)y)\{x{:=}q\}$.

◀

B Appendix: Factorizing Factorization in Probabilistic λ-calculus

Faggian and Ronchi della Rocca [19] define two calculi—$\Lambda^{\mathsf{cbn}}_{\oplus}$ and $\Lambda^{\mathsf{cbv}}_{\oplus}$—which model respectively CbV and CbN probabilistic higher-order computation, and are conservative extensions of the CbN and CbV λ-calculi. Here we focus on CbV, which is the most relevant paradigm for calculi with effects, but the same approach applies to CbN.

We first recall the syntax of $\Lambda^{\mathsf{cbv}}_{\oplus}$ (we refer to [19] for background and details), and then give a new proof of weak factorization, using our technique and obtaining a neat, compact proof of factorization, which only requires a few lines.

Terms. $\Lambda^{\mathsf{cbv}}_{\oplus}$ is a rewrite system where the objects to be rewritten are not terms, but monadic structures on terms, namely multi-distributions [8]. Intuitively, a multi-distribution represents a probability distribution on the possible reductions from a term. Terms and contexts are the same as for the non-deterministic λ-calculus, but here we write the $\oplus$ infix, to facilitate reference to [19]. *Terms and values* are generated by the grammars

$$M ::= x \mid \lambda x.M \mid MM \mid M \oplus M \qquad (\textbf{Terms } \Lambda_{\oplus})$$
$$V := x \mid \lambda x.M \qquad (\textbf{Values } \mathcal{V})$$

where x ranges over a countable set of *variables*. *Contexts* and *weak contexts* are given by:

$$\mathsf{C} ::= \langle\,\rangle \mid \mathsf{C}M \mid M\mathsf{C} \mid \lambda x.\mathsf{C} \mid \mathsf{C} \oplus M \mid M \oplus \mathsf{C} \qquad (\textbf{Contexts})$$
$$\mathsf{W} ::= \langle\,\rangle \mid \mathsf{W}M \mid M\mathsf{W} \qquad (\textbf{Weak Contexts})$$

where $\langle\,\rangle$ denotes the *hole* of the context.

The intended behaviour of $M \oplus N$ is to reduce to either M or N, *with equal probability* $\frac{1}{2}$. This is formalized by means of *multi-distributions.*

Multi-distributions. A *multi-distribution* $\mathtt{m} = [p_i M_i \mid i \in I]$ is a multiset of pairs of the form pM, with $p \in]0,1]$, $M \in \Lambda_{\oplus}$, and $\sum p_i \leq 1$. We denote by $\mathcal{M}(\Lambda_{\oplus})$ the set of all multi-distributions. The sum of multi-distributions is denoted by $+$. The product $q \cdot \mathtt{m}$ of a scalar q and a multi-distribution $\mathtt{m}$ is defined pointwise $q[p_i M_i]_{i\in I} := [(qp_i)M_i]_{i\in I}$.

The calculus $(\mathcal{M}(\Lambda_{\oplus}), \Rightarrow)$. The calculus $\Lambda^{\mathsf{cbv}}_{\oplus}$ is the rewrite system $(\mathcal{M}(\Lambda_{\oplus}), \Rightarrow)$ where $\mathcal{M}(\Lambda_{\oplus})$ is the set of multi-distributions on $\Lambda_{\oplus}$ and the relation $\Rightarrow \subseteq \mathcal{M}(\Lambda_{\oplus}) \times \mathcal{M}(\Lambda_{\oplus})$ is defined in Fig. 1 and Fig. 2. First, we define one-step reductions from terms to multi-distributions—so for example, $M \oplus N \rightarrow [\frac{1}{2}M, \frac{1}{2}N]$. Then, we lift the definition of reduction to a binary relation on $\mathcal{M}(\Lambda_{\oplus})$, in the natural way—for instance $[\frac{1}{2}(\lambda x.x)z, \frac{1}{2}(M \oplus N)] \Rightarrow [\frac{1}{2}z, \frac{1}{4}M, \frac{1}{4}N]$. Precisely:

$$\mathsf{C}\langle(\lambda x.M)V\rangle \to_{\beta_v} [\mathsf{C}\langle M\{x{:=}V\}\rangle] \qquad \mathsf{W}\langle M\oplus N\rangle \to_{\oplus} [\tfrac{1}{2}\mathsf{W}(M), \tfrac{1}{2}\mathsf{W}(N)] \qquad \begin{array}{l} \to := \to_{\beta_v} \cup \to_{\oplus} \\ \underset{\mathsf{w}}{\to} := \underset{\mathsf{w}}{\to}_{\beta_v} \cup \to_{\oplus} \end{array}$$

Figure 1 Reduction Steps

$$\frac{}{[M] \Rightarrow_r [M]} \qquad \frac{M \to_r \mathtt{m}}{[M] \Rightarrow_r \mathtt{m}} \qquad \frac{([M_i] \Rightarrow_r \mathtt{m}_i)_{i\in I}}{[p_i M_i \mid i \in I] \Rightarrow_r +_{i\in I}\, p_i \cdot \mathtt{m}_i}$$

Figure 2 Lifting $\to_r$ to $\Rightarrow_r$

1. The reductions $\to_{\beta_v}, \to_{\oplus} \subseteq \Lambda_{\oplus} \times \mathcal{M}(\Lambda_{\oplus})$ are defined in Fig. 1. Observe that the $\oplus$ rule—probabilistic choice—is closed only under weak contexts (no reduction in the body of a function nor in the scope of an operator $\oplus$). Instead, the β_v rule is closed under general contexts. Its restriction to closure under weak context is denoted $\underset{\mathsf{w}}{\to}_{\beta}$. The relation $\to$ is the union $\to_{\beta} \cup \to_{\oplus}$, while weak[4] reduction $\underset{\mathsf{w}}{\to}$ is the union of the weak reductions $\underset{\mathsf{w}}{\to}_{\beta_v} \cup \to_{\oplus}$. A $\to$-step which is not weak is noted $\underset{\neg\mathsf{w}}{\to}$.
2. The lifting of a relation $\to_r \subseteq \Lambda_{\oplus} \times \mathcal{M}(\Lambda_{\oplus})$ to a reduction on multi-distribution is defined in Fig. 2. In particular, $\to, \to_{\beta_v}, \to_{\oplus}, \underset{\mathsf{w}}{\to}, \underset{\neg\mathsf{w}}{\to}$ lift to $\Rightarrow, \Rightarrow_{\beta_v}, \Rightarrow_{\oplus}, \underset{\mathsf{w}}{\Rightarrow}, \underset{\neg\mathsf{w}}{\Rightarrow}$.

The restriction of $\to_{\oplus}$ to weak contexts is necessary to have confluence, see [19] for a discussion. The fact that reduction $\to_{\beta_v}$ is unrestricted guarantees that the new calculus is a *conservative extension* of CbV λ-calculus.

Factorization, Modularly. Faggian and Ronchi della Rocca prove—by defining suitable notions of parallel reduction and internal parallel reduction with respect to $\Rightarrow_{\beta_v} \cup \Rightarrow_{\oplus}$—that $\Lambda_{\oplus}^{\mathsf{cbv}}$ satisfies $\mathtt{Fact}(\underset{\mathsf{w}}{\Rightarrow}, \underset{\neg\mathsf{w}}{\Rightarrow})$, that is $\mathtt{m} \Rightarrow^* \mathtt{n}$ implies $\mathtt{m} \underset{\mathsf{w}}{\Rightarrow}^* \cdot \underset{\neg\mathsf{w}}{\Rightarrow}^* \mathtt{n}$.

This result—there called *finitary surface standardization*—is central in [19] because it is the base of the asymptotic constructions which are the core of that paper.

We now give a novel, strikingly short proof of the same result, by using Theorem 3.4. It turns out that we only need to verify the following swap, which is immediate to check.

▶ **Lemma B.1.** *$M \underset{\neg\mathsf{w}}{\Rightarrow}_{\beta_v} \cdot \to_{\oplus} \mathtt{n}$ implies $M \to_{\oplus} \cdot \Rightarrow_{\beta_v} \mathtt{n}$.*

▶ **Theorem B.2** (Factorization of $\Rightarrow$)**.** *Let $\underset{\mathsf{w}}{\Rightarrow} := (\underset{\mathsf{w}}{\Rightarrow}_{\beta_v} \cup \Rightarrow_{\oplus})$ and $\underset{\neg\mathsf{w}}{\Rightarrow} := (\underset{\neg\mathsf{w}}{\Rightarrow}_{\beta_v})$. Then $(\mathcal{M}(\Lambda_{\oplus}), \{\Rightarrow_{\beta_v}, \Rightarrow_{\oplus}\})$ satisfies* w*-factorization* $\mathtt{Fact}(\underset{\mathsf{w}}{\Rightarrow}, \underset{\neg\mathsf{w}}{\Rightarrow})$.

Proof. We verify that the conditions of Theorem 3.4 hold. Note that $\oplus$ has no internal steps, therefore, it suffices to verify only two conditions:

1. weak factorization of $\Rightarrow_{\beta_v}$: $\mathtt{Fact}(\underset{\mathsf{w}}{\Rightarrow}_{\beta_v}, \underset{\neg\mathsf{w}}{\Rightarrow}_{\beta_v})$.
2. $\underset{\neg\mathsf{w}}{\Rightarrow}_{\beta_v}$ linearly swaps with $\Rightarrow_{\oplus}$: $\underset{\neg\mathsf{w}}{\Rightarrow}_{\beta_v} \cdot \Rightarrow_{\oplus} \subseteq \Rightarrow_{\oplus} \cdot \Rightarrow_{\beta_v}^*$.

The other two conditions of Theorem 3.4 hold vacuously, because $\underset{\neg\mathsf{w}}{\Rightarrow}_{\oplus} = \emptyset$ (and $\underset{\mathsf{w}}{\Rightarrow}_{\oplus} = \Rightarrow_{\oplus}$).

1. $\mathtt{Fact}(\underset{\mathsf{w}}{\Rightarrow}_{\beta_v}, \underset{\neg\mathsf{w}}{\Rightarrow}_{\beta_v})$ follows from weak factorization of the CbV λ-calculus $\mathtt{Fact}(\underset{\mathsf{w}}{\to}_{\beta_v}, \underset{\neg\mathsf{w}}{\to}_{\beta_v})$ (see Sect. 6) because clearly $([1M] \Rightarrow_{\beta_v} [1N]$ if and only if $M \to_{\beta_v} N)$, $([1M] \underset{\mathsf{w}}{\Rightarrow}_{\beta_v} [1N]$ if and only if $M \underset{\mathsf{w}}{\to}_{\beta_v} N)$, and similarly $([1M] \underset{\neg\mathsf{w}}{\Rightarrow}_{\beta_v} [1N]$ if and only if $M \underset{\neg\mathsf{w}}{\to}_{\beta_v} N)$.
2. Lemma B.1 implies $\mathtt{m} \underset{\neg\mathsf{w}}{\Rightarrow}_{\beta_v} \cdot \Rightarrow_{\oplus} \mathtt{n} \subseteq \mathtt{m} \Rightarrow_{\oplus} \cdot \Rightarrow_{\beta_v} \mathtt{n}$, by the definition of lifting. ◀

[4] In [19], a *weak* reduction (resp. weak context) is called *surface*, and hence noted $\underset{\mathsf{s}}{\to}$.