INTERACTION AND DEPTH
AGAINST NONDETERMINISM IN PROOF SEARCH

OZAN KAHRAMANOĞULLARI

The Microsoft Research – University of Trento Centre for Computational and Systems Biology
e-mail address: ozan@cosbi.eu

ABSTRACT. Deep inference is a proof theoretic methodology that generalizes the standard notion of inference of the sequent calculus, whereby inference rules become applicable at any depth inside logical expressions. Deep inference provides more freedom in the design of deductive systems for different logics and a rich combinatoric analysis of proofs. In particular, construction of exponentially shorter analytic proofs becomes possible, however with the cost of a greater nondeterminism than in the sequent calculus. In this paper, we show that the nondeterminism in proof search can be reduced without losing the shorter proofs and without sacrificing proof theoretic cleanliness. For this, we exploit an interaction and depth scheme in the logical expressions. We demonstrate our method on deep inference systems for multiplicative linear logic and classical logic, discuss its proof complexity and its relation to focusing, and present implementations.

INTRODUCTION

Proof search is of central importance in automated reasoning, especially for a broad range of applications in the fields of automated theorem proving, software verification and artificial intelligence. The development of formalisms, techniques and principles that allow the construction of shorter proofs in different logics is a requirement for the applications in these fields. This requirement gains an even greater emphasis for logics, where highly optimized techniques, such as classical resolution, are not applicable.

Deep inference [18] is a proof theoretic methodology that generalizes the notion of inference in formalisms such as the sequent calculus, natural deduction and analytic tableaux by introducing a top-down symmetry of the inference rules. This symmetry brings about a combinatoric wealth that gives rise to properties that are otherwise not observable, and thereby results in a broad spectrum of theoretical and practical consequences. For example, at the theoretical front, a manifestation of the top-down symmetry of the deep inference rules is observed as the cut rule and the axiom become dual deep inference rules [18, 3, 37].[1] In the standard sequent calculus, this duality remains implicit in the inference rules, and can only be revealed explicitly, for example, by constructing a proof net or an atomic flow graph [19, 21], that is, by removing the deductive information in the proofs that displays their

[1]These are the rules ai↑ and ai↓ in Definition 1.9.

step-wise logical construction. On the practical front, deep inference brings proof theory closer to computation by providing a meaningful deductive interpretation of concepts such as locality and sequentiality [18, 35, 6, 23, 39]. The proof theoretic interpretation of these concepts make available potential applications in logic programming [7, 29, 25].

The top-down symmetry of the deep inference rules has implications that also bridge theoretical concepts with practical ones. This is because the symmetry of the inference rules makes it possible to treat all parts of the proofs and derivations at the same syntactic level as the logic. As a result, it becomes possible to apply the inference rules at any depth inside logical expressions, similar to the application of term rewriting rules [27, 29]. This feature, delivering the name deep inference, contrasts with the notion of inference of the sequent calculus, which we call shallow inference. As an example, consider the following shallow inference proof in multiplicative linear logic (Figure 2) and a deep inference proof of the same formula, where the rule instances are marked:

$$
\mathsf{\otimes}\cfrac{\mathsf{\otimes}\cfrac{\mathsf{id}\cfrac{}{\vdash a \,,\, \bar{a}}}{\vdash a \,\otimes\, \bar{a}} \quad \mathsf{id}\cfrac{}{\vdash b \,,\, \bar{b}}}{\cfrac{\vdash ([a \,\otimes\, \bar{a}] \otimes \bar{b}) \,,\, b}{\vdash [([a \,\otimes\, \bar{a}] \otimes \bar{b}) \,\otimes\, b]}}
\qquad\qquad
\mathsf{ai{\downarrow}}\cfrac{\mathsf{u_2{\downarrow}}\cfrac{\mathsf{ai{\downarrow}}\cfrac{1}{[\bar{b} \,\otimes\, b]}}{[(1 \otimes \bar{b}) \,\otimes\, b]}}{[([a \,\otimes\, \bar{a}] \otimes \bar{b}) \,\otimes\, b]}
$$

Shallow inference proofs are constructed by breaking the logical expressions into smaller components. These components are separated with meta-level operators: the space between two branches in a proof tree denotes a meta-level conjunction and the comma in a sequent denotes a meta-level disjunction. However a separation as in the sequent calculus of meta-level and object-level causes a mismatch for some logics [17] as the object level of the logic and the meta-level correspond to different logical operators. For example, in linear logic, the logical object-level employs multiplicative conjunctions as in the proof above as well as additive conjunctions. However, in the sequent calculus the meta-level conjunction implicitly plays a multiplicative role for some rules and an additive role for others. Deep inference abolishes the meta-level in proofs as the meta-level is integrated into the object-level. Thereby, deep inference provides a solution to this mismatch as it clarifies the deductive reading of the proofs by keeping the inference steps within the logic of the system. For example, in the proof above on the right, each proof step is a linear implication, and there is no meta-level information.

Another consequence of the distinction between meta-level and object-level is observed in term rewriting implementations of shallow inference, as they require additional mechanisms, outside the logic, that implement the meta-level [32]. Absence of meta-level renders such an additional mechanism unnecessary, and makes the term rewriting implementations much simpler [29].

While indicating novel proof theoretic perspectives, the combinatoric wealth offered by deep inference comes with a cost as the traditional methods do not meet the proof theoretic requirements. This is because the long established standard techniques of the sequent calculus for proving cut elimination and completeness are not applicable in the deep inference setting. As a result of this, the potential that came about due to deep inference required the development of proof theoretic methods for addressing the emerging technical challenges. The consequent efforts resulted in cut elimination results, which are very different from the then-available methods for the sequent calculus [18, 36, 4, 5, 39].

Some of these cut-elimination results were obtained by fragmenting proofs into different phases, where in each phase distinct inference rules are used. The resulting *decomposition* theorems [36, 4, 39] made available new ways for studying proofs, which are impossible in the sequent calculus. These results can be seen as a consequence of the greater freedom in proof construction in deep inference that provides many more proofs in comparison to the sequent calculus.

The concepts listed above qualify deep inference as a proof theoretic methodology that is closer to computation. In fact, deep inference provides the means to design deductive systems for different logics that reveal properties of proofs of computational relevance and otherwise impossible. For example, the deep inference system BV [18, 24] gives a proof theoretic interpretation of sequentiality as modeled in process algebra by extending multiplicative linear logic (MLL) with the rules mix, nullary mix and a self-dual, non-commutative operator. BV and its Turing-complete extension [23, 39, 38], are provably not designable in the sequent calculus [41]. Moreover, the computer science notion of locality, i.e., an operation having a bounded computational cost, also finds a meaningful proof theoretic interpretation in the deep inference setting [6, 35, 40].

Deep inference deductive systems provide advantages also when they are considered for *computation as proof search*: the applicability of the inference rules at arbitrary depths inside logical expressions makes it possible to start the construction of the proofs from subformulae. This capability *provides many more different proofs, some of which are much shorter* than those provided by shallow inference. For example, for a class of formulae called Statman tautologies [33], deep inference provides *polynomially* growing proofs in contrast to the *exponentially* growing sequent calculus proofs [9]. The reason for this drastic difference is because the sequent calculus can access the subformulae only by opening up the formulae at the main connective, and proofs are then constructed by spreading the context to the branches of the proof tree as in the example above. In contrast, applying the inference rules without such a restriction as in deep inference makes it possible to reduce the length of these proofs to an extend that in some cases *provides a polynomial length instead of an exponential one*. In certain cases, the bounded depth formalism with alternative compression mechanisms can be used for recuperating the polynomial size proofs, as it is the case in [13] for classical logic with a cubical transformation. However, in nondeterministic proof search, where the search spaces grow exponentially in the number of proof steps, even *a linear speed up can extend the margin of successful searches*.

Despite the existence of shorter proofs, a naive proof search approach with deep inference results in transposing the problem. This is because deep inference rules can be applied in many more ways, and this gives rise to a larger breadth of the search space. Due to the resulting greater size of the search space, in order to benefit from the shorter proofs provided by deep inference in proof search, *the non-determinism needs to be controlled*.

The trade off between shorter proofs and larger breadth of the search space has been addressed by adapting the focusing technique [1, 2] to deep inference systems [12]. Although focusing provides a satisfactory means for reducing nondeterminism in the sequent calculus, it has limitations in the deep inference setting. This is because in deep inference, context management is very different from the more rigid sequent calculus, which allows the application of the inference rules only to the top-level formulae. Other techniques for rendering sequent calculus based proof search more efficient, for example, for linear logic [10], include methods that use additional tagging mechanisms to account for formulas that are considered as consumed resources in proof search.

We present a technique for controlling nondeterminism in deep inference deductive systems. Our technique, which is complementary to focusing and *pure from a logical point of view*, provides a more immediate access to shorter proofs without sacrificing proof theoretic cleanliness. This is achieved by exploiting an interaction pattern together with depth within the rule instances during proof construction: in deep inference deductive systems, the combinatoric behavior results from the instances of the switch rule, which is the rule responsible for context management. This rule is common to the deep inference systems for different fragments of linear logic [37, 35], classical logic [4, 6], modal logics [15, 34], intuitionistic logic [40], the logic BV [18] and its extension with the exponentials of linear logic [23, 39].

With linear logic operators, the switch rule is given as follows:

$$\mathsf{s} \frac{S([R \,\bindnasrepma\, U] \otimes T)}{S[(R \otimes T) \,\bindnasrepma\, U]}$$

The intuition behind the role of the switch rule in proof construction can be seen by using a communication metaphor [18]: when the formulae that are connected with a disjunction are considered as communicating in a context S, a bottom-up application of the switch rule can be perceived as breaking the communication between the atoms in T and U and this way enhancing the communication between the atoms in R and U. In proof construction, this mechanism is used to manage contexts to bring dual atoms closer as proofs are constructed bottom-up by annihilating communicating dual atoms at the instances of the atomic interaction rule, that is, the rule ai↓ (see Definition 1.9), as illustrated in the example above.

Our method is based on carrying the communication and interaction pattern used by the atomic interaction rule to the switch rule. For this purpose, we redesign the switch rule to enhance the communication of dual atoms that readily have an interaction potential. This way, we manage the contexts during proof construction with respect to the interaction potentials of the formula that are processed at the rule instances. The laziness and depth conditions that we impose make the processing of the formula more gradual, and reduce the size of the information being processed at every proof step, thereby resulting in a reduction in nondeterminism. By combining the notions of laziness and depth, which are dual conditions together with the interaction condition, we introduce a rule that we call *deep lazy interaction switch*. This rule results in a greater reduction in nondeterminism without losing the shorter proofs available due to deep inference, and reduces the cost of applying the interaction condition.

The ideas on controlling nondeterminism in deep inference were initially introduced in [24], where we used the lazy interaction switch rule as a proof theoretic tool in proving NP-hardness of the logic BV. As BV extends multiplicative linear logic with the mix and nullary mix rules, the completeness argument in [24] benefits from the role played by the mix rule: the presence of mix provides proofs for some formulae that are not provable in MLL. This phenomena, which results from the instances of the mix rule, makes it possible to define a notion that we call *independence* on the formulae, which, in [24], simplifies the completeness argument. For example, the formulae $[a \,\bindnasrepma\, \bar{a}]$ and $[b \,\bindnasrepma\, \bar{b}]$ are independent, as the provability of one does not depend on the provability of the other, and their disjunction $[a \,\bindnasrepma\, \bar{a} \,\bindnasrepma\, b \,\bindnasrepma\, \bar{b}]$ can be proved in BV. However, the independence notion does not carry over to multiplicative linear logic directly, because, in contrast to BV, the proofs of two formulae do not deliver a proof of their disjunction. Moreover, the units 1 and $\perp$ pose cases that

require further attention. We thus refine our technique and show that it does not rely on the mix rule.

In the following, we give a fragmented elaboration of different deep inference systems for multiplicative linear logic, where the nondeterminism is gradually reduced. For the presentation of the ideas, we use the standard notions and notations as presented in [18, 36, 4, 37, 35, 9, 23, 39]. In the recent years, deep inference has grown to encompass a family of proof theoretic formalisms [20], and the calculus of structures remained as the simpler broadly-used deep inference framework. We therefore use the calculus of structures as the presentation framework. We first show on a deep inference system for multiplicative linear logic that the nondeterminism due to the units can be easily controlled by explicit inference rules. Following this, within a lattice representation, we introduce the restrictions on the switch rule, which result in eight different systems. We show the completeness of these systems for multiplicative linear logic. We give a constructive cut elimination proof, which justifies that our technique is clean from a proof theoretic point of view. We show that our technique does not introduce any additional complexity in comparison to the sequent calculus with respect to the size of the proofs. Following the approach in [9, 13, 14], we show that the systems obtained with our technique polynomially simulate shallow inference systems. We then discuss the relation of our method with the focusing technique [1, 2, 12].

Availability of a constructive cut-elimination procedure for the system with deep lazy interaction switch shows that this system is independently meaningful from a proof theoretic point of view, in isolation from any other system for multiplicative linear logic. Moreover, the technique and the rules that we use here are common to all the deep inference systems, thus they should be applicable to other deep inference systems. As an evidence for this, we show that deep lazy interaction switch can be replaced with the switch rule in a system for classical logic without breaking the completeness of this system.

All the systems presented in this paper are implemented in Maude [29].[2] An interactive prover, written in OCaml, is also available for download.[3]

1. Proof Theory with Deep Inference

In deep inference deductive systems, the laws such as associativity and commutativity are imposed on the logical expressions by means of an underlying equational system [18, 35, 37, 3, 40, 6, 23, 39]. This is done at every inference step by working with congruence classes of formulae that are called structures: each structure, as a syntactic entity, denotes a set of formulae that are equivalent under a congruence relation, and inference rules are applied on these congruence classes. The notion of structure originates from graphs called relation webs [18]. The non-commutative multiplicative linear logic BV, introduced in [18], was obtained by preserving certain geometric properties of structures at every inference step. These inference rules gave rise to systems for various logics.

Structures, which we define for multiplicative linear logic, share properties with formulae and sequents, and they can be easily mapped to these entities.

Definition 1.1. There are countably many *atoms*, denoted by $a, b, c, \ldots$ The *structures* P, Q, R, $S \ldots$ of multiplicative linear logic are generated by the grammar

$$R ::= a \mid 1 \mid \bot \mid [R \,\bindnasrepma\, R] \mid (R \otimes R) \mid \bar{R} \quad ,$$

---

[2]http://sites.google.com/site/ozankahramanogullari/software/di_in_maude
[3]http://sites.google.com/site/ozankahramanogullari/software/interana

where a stands for any atom. 1 and $\bot$ are special atoms, called *one* and *bottom*. A structure $[R \bindnasrepma R]$ is a *par structure*, $(R \otimes R)$ is a *copar (times) structure* and $\bar{R}$ is the *negation* of R. Structures are considered to be equivalent modulo the relation $\approx$, which is the smallest congruence relation induced by the equational system consisting of the equations for *associativity* and *commutativity* for par and copar, and the *equations*

$$\overline{[R \bindnasrepma T]} \approx (\bar{R} \otimes \bar{T}), \quad \overline{(R \otimes T)} \approx [\bar{R} \bindnasrepma \bar{T}], \quad \bar{\bar{R}} \approx R, \quad \bar{\bot} \approx 1, \quad \bar{1} \approx \bot$$

for negation. We denote the structures in the same equivalence class by picking a structure from the equivalence class. If there is no ambiguity, we drop the superfluous brackets in the structures by resorting to the equations for associativity.

Remark 1.2. Following [37], in the definition of the structures above, we include the equations for negation in the congruence relation imposed on structures. However, in the systems discussed below, the inference rules do not introduce any new negation symbols on the structures in proof construction. Because of this, we consider the structures to be in negation normal form: every structure can be considered in negation normal form by applying the equations for negation in Definition 1.1 from left to right exhaustively as term rewriting rules to push the negation symbol to the atoms. Given that a proof search is initiated with a structure in negation normal form, no new negation symbol is introduced by the inference rules. This allows us to disregard the equations for negation in the discussions below by assuming that proof construction is initiated with a structure in negation normal form.

Notation 1.3. In deep inference, the convention is using different types of parentheses to denote different logical operators by using the comma as an infix separator. For example, the expression $a \bindnasrepma b$ is often denoted by $[a, b]$. In this paper, we adapt this notation to the usual notation of the logical operators by replacing the comma with the logical operators, for example, as in $[a \bindnasrepma b]$.

Example 1.4. With respect to the congruence relation $\approx$ on the structures, we have

$$[(\bar{b} \otimes (\bar{a} \otimes \bot)) \bindnasrepma [b \bindnasrepma [a \bindnasrepma 1]]] \approx [[b \bindnasrepma ((\bar{a} \otimes \bot) \otimes \bar{b})] \bindnasrepma [1 \bindnasrepma a]]$$

and we can denote both of them with $[(\bar{a} \otimes \bar{b} \otimes \bot) \bindnasrepma a \bindnasrepma b \bindnasrepma 1]$.

Definition 1.5. A *structure context*, denoted as in $S\{\ \}$, is a structure with a hole that does not appear in the scope of negation. The structure R is a *substructure* of $S\{R\}$ and $S\{\ \}$ is its *context*. A context is empty if it is $\{\ \}$. Context braces are omitted if no ambiguity is possible. That is, whenever structural brackets of any kind surround the context of a hole, hole braces are omitted.

Example 1.6. Let $S\{\ \} = [\{\ \} \bindnasrepma a \bindnasrepma b \bindnasrepma 1]$, $R = \bar{a}$ and $T = (\bar{b} \otimes \bot)$. Then we have that

$$S(R \otimes T) = [(\bar{a} \otimes \bar{b} \otimes \bot) \bindnasrepma a \bindnasrepma b \bindnasrepma 1] \,.$$

Definition 1.7. An *inference rule* is a scheme of the kind

$$\rho \, \frac{T}{R} \,,$$

where ρ is the *name* of the rule, T is its *premise* and R is its *conclusion*. A typical deep inference rule has the shape

$$\rho \, \frac{S\{T\}}{S\{R\}} \quad,$$

$$\mathsf{ai}{\downarrow}\,\frac{S\{1\}}{S[a \,\bindnasrepma\, \bar{a}]} \qquad \mathsf{s}\,\frac{S([U \,\bindnasrepma\, R] \otimes T)}{S[U \,\bindnasrepma\, (R \otimes T)]} \qquad \mathsf{ai}{\uparrow}\,\frac{S[R \,\bindnasrepma\, (a \otimes \bar{a})]}{S\{R\}}$$

$$\mathsf{u}_1{\downarrow}\,\frac{S\{R\}}{S[R \,\bindnasrepma\, \bot]} \qquad \mathsf{u}_1{\uparrow}\,\frac{S(R \otimes 1)}{S\{R\}} \qquad \mathsf{u}_2{\downarrow}\,\frac{S\{R\}}{S(R \otimes 1)} \qquad \mathsf{u}_2{\uparrow}\,\frac{S[R \,\bindnasrepma\, \bot]}{S\{R\}}$$

Figure 1: Deep Inference System SMLS

which is determined by the implication $T \Rightarrow R$ inside a generic context $S\{\ \}$. In the case of multiplicative linear logic, this is linear implication. In an instance of ρ, we say that R is the *redex* and T is the *contractum*. A system $\mathscr{S}$ is a set of inference rules.

An inference rule of the form in Definition 1.7 specifies a step of rewriting. In this paper, these rewritings are those that rewrite the conclusion to the premise. This is because we consider the inference rules for proof-search, thus we consider their bottom-up applications, which result in proofs that grow from the conclusion to the premise.

Remark 1.8. In deductive systems with deep inference, it is common to include a set of equations for the treatment of the units of the logic in the congruence relation in addition to those for associativity, commutativity and negation. Here, with a focus on proof search, we choose a treatment of units by means of inference rules of the deductive system. This results in an equivalent proof theoretic consideration, while providing more control over the units in proof construction as we demonstrate below.

Definition 1.9. The rules of the Symmetric deep inference Multiplicative Linear logic System or SMLS are depicted in Figure 1. The rules are called *atomic interaction* ($\mathsf{ai}{\downarrow}$), *switch* (s), *atomic cut* ($\mathsf{ai}{\uparrow}$), unit one down ($\mathsf{u}_1{\downarrow}$), unit one up ($\mathsf{u}_1{\uparrow}$), unit two down ($\mathsf{u}_2{\downarrow}$) and unit two up ($\mathsf{u}_2{\uparrow}$), respectively.

Definition 1.10. The system MLS is obtained by removing the cut rule ($\mathsf{ai}{\uparrow}$) from SMLS.

Definition 1.11. The system MLSu is obtained by removing the rules $\mathsf{u}_1{\uparrow}$, $\mathsf{u}_2{\uparrow}$ and the cut rule from SMLS.

Remark 1.12. In deep inference, deductive systems for different logics employ different treatments of the units in accordance with the underlying deductive setting. For example, in the logic BV [18, 24], the units $\bot$ and 1 of multiplicative linear logic merge into one unit, as they are logically equivalent in the presence of the mix rule (mix) and the nullary mix (mix0). In their sequent calculus presentation, the rules mix and mix0 are given as follows:

$$\mathsf{mix}\,\frac{\vdash \Gamma \qquad \vdash \Delta}{\vdash \Gamma, \Delta} \qquad\qquad \mathsf{mix0}\,\frac{}{\vdash}$$

When multiplicative linear logic in the sequent calculus (Definition 1.16, Figure 2) is extended with the rules mix and mix0, it becomes possible to prove the logical equivalence of the units $\bot$ and 1, and as a result of this replace the two units with a single unit. In BV this equivalence is realized by such a single unit that replaces the units $\bot$ and 1 [18]. In fact, due to this equivalence, the unit can be removed completely from the deductive system of the logic BV [26]. However, in multiplicative linear logic, the units play a different role, similar

to other atoms, which provides an expressive power for the unit-only fragment: as shown by Lincoln and Winkler [31], the fragment of MLL with only units and no other atoms is NP-complete.

Definition 1.13. A *derivation* Δ is a finite chain of instances of inference rules. A derivation can consist of just one structure. The top-most structure in a derivation is called the *premise*, and the bottom-most structure is called its *conclusion*. A derivation Δ whose premise is T, conclusion is R, and inference rules are in $\mathscr{S}$ will be written as

$$\Delta \left\|\mathscr{S} \begin{array}{c} T \\ \\ R \end{array}\right. .$$

A multiplicative linear logic *proof* Π in deep inference is either the unit 1 or a finite derivation whose premise is the unit 1. The size $|\Pi|$ of a proof Π is the number of unit and atom occurrences appearing in it.

In contrast to proofs and derivations in the sequent calculus, which are trees, the deep inference proofs are chains of instances of the inference rules. In deep inference, the inference rules are not restricted to be applied at the main connective. That is, when the the formula is seen as a term, they can be applied not only at the root position, but also at other positions of the term. As it has been shown by Bruscoli and Guglielmi [9], in some cases this provides shorter proofs than those that are provided by the notion of inference of the sequent calculus that we call *shallow inference*: shallow inference permits the application of the inference rules only at the top level connective. However, applicability of the inference rules at any depth inside a structure makes it possible to start the construction of a proof from the substructures. This brings about many more proofs, including those available with shallow inference. Some of these proofs are much shorter than the proofs in any bounded depth system [13, 14]: bounded depth systems are those that retain the top-down symmetry of deep inference, but can otherwise be designed at the same depth level of sequent calculus systems. In some cases, deep inference proofs are exponentially shorter than shallow inference proofs [9], which we discuss in Section 3. Let us see the reason for this on an example.

Example 1.14. Consider the two proofs below of the structure $[([a \,\mathscr{V}\, \bar{a}] \otimes \bar{b}) \,\mathscr{V}\, b]$ in MLS. In the proof on the left, we use shallow inference rules, where the context $S\{\ \}$ is $\{\ \}$, resembling the application of the sequent calculus rules. This requires prior rule instances at the top-level in order to access other instances. In the proof on the right, we use deep inference rules.

$$\mathsf{ai}{\downarrow};\, \mathsf{u}_2{\downarrow}\, \cfrac{\mathsf{s}\, \cfrac{\mathsf{ai}{\downarrow}\, \cfrac{1}{[b \,\mathscr{V}\, \bar{b}]}}{([a \,\mathscr{V}\, \bar{a}] \otimes [b \,\mathscr{V}\, \bar{b}])}}{[([a \,\mathscr{V}\, \bar{a}] \otimes \bar{b}) \,\mathscr{V}\, b]} \qquad\qquad \mathsf{ai}{\downarrow};\, \mathsf{u}_2{\downarrow}\, \cfrac{\mathsf{ai}{\downarrow}\, \cfrac{1}{[b \,\mathscr{V}\, \bar{b}]}}{[([a \,\mathscr{V}\, \bar{a}] \otimes \bar{b}) \,\mathscr{V}\, b]}$$

Despite the availability of shorter proofs, deep inference causes a greater non-determinism in proof search: the inference rules can be applied at many more positions than in the sequent calculus. This results in a much larger breadth of the search space in proof search. Some of this increased non-determinism is due to the treatment of units, which provides a rich proof theory, however it results in redundant rule instances.

$$\mathsf{id}\,\frac{}{\vdash A, \bar{A}} \qquad \otimes\,\frac{\vdash A, \Phi \quad \vdash B, \Psi}{\vdash (A \otimes B), \Phi, \Psi} \qquad \parr\,\frac{\vdash A, B, \Phi}{\vdash [A \parr B], \Phi} \qquad \perp\,\frac{\vdash \Phi}{\vdash \perp, \Phi} \qquad 1\,\frac{}{\vdash 1}$$

Figure 2: MLL in the sequent calculus

Example 1.15. The following derivations involve rule instances, which do not contribute to the construction of proofs. In particular, the two derivations on the left result in the premises and the conclusions that are the same structures, whereas the derivation on the right transforms a provable structure in the conclusion to a structure in the premise that is not provable. These derivations are instantiated with the application of $\mathsf{u_1}{\uparrow}$ and $\mathsf{u_2}{\uparrow}$.

$$
\mathsf{u_2}{\downarrow}\,\cfrac{[a \parr b]}{\mathsf{u_1}{\uparrow}\,\cfrac{\mathsf{s}\,\cfrac{([a \parr b] \otimes 1)}{[a \parr (b \otimes 1)]}}{[a \parr b]}}
\qquad
\mathsf{u_1}{\downarrow}\,\cfrac{(a \otimes b)}{\mathsf{u_2}{\uparrow}\,\cfrac{\mathsf{s}\,\cfrac{([\perp \parr a] \otimes b)}{[\perp \parr (a \otimes b)]}}{(a \otimes b)}}
\qquad
\mathsf{s}\,\cfrac{(a \otimes [1 \parr \bar{a}])}{\mathsf{u_1}{\uparrow}\,\cfrac{[(a \otimes 1) \parr \bar{a}]}{[a \parr \bar{a}]}}
$$

The system MLS contains the rules $\mathsf{u_1}{\uparrow}$ and $\mathsf{u_2}{\uparrow}$, which are not included in MLSu. In order to avoid derivations of the form in Example 1.15 in proof search, we thus remove the rules $\mathsf{u_1}{\uparrow}$ and $\mathsf{u_2}{\uparrow}$ from MLS, and obtain the system MLSu. The completeness of MLSu for multiplicative linear logic can be easily shown by inductively translating MLL proofs in the sequent calculus into MLSu proofs, or alternatively via cut elimination. We use the proof in [37], where MLL proofs in the sequent calculus are transformed into proofs in MLS. Here the key argument is the fact that the rules $\mathsf{u_1}{\uparrow}$ and $\mathsf{u_2}{\uparrow}$ do not play a role in the translation.

Definition 1.16. The Multiplicative Linear Logic system in the sequent calculus or MLL is depicted in Figure 2.

The system MLL is defined on formulae, which are different from structures as the equalities for associativity and commutativity do not apply to formulae. Here, for simplicity we use the same notation for structures and formulae, and allow the equalities for associativity and commutativity to operate on structures. The following theorem, which makes use of this simplification, states that MLS is complete for multiplicative linear logic. The proof of this statement can be found in [37].

Theorem 1.17. *A structure R has a proof in* MLS *if and only if the expression obtained by translating R into a formula has a proof in* MLL.

Definition 1.18. Two systems $\mathscr{S}$ and $\mathscr{S}'$ are *equivalent* if they prove the same structures. A rule ρ is *admissible* for a system $\mathscr{S}$ if for every proof in $\mathscr{S} \cup \{\rho\}$ there is a proof in $\mathscr{S}$ with the same conclusion.

Theorem 1.19. *The systems* MLS *and* MLSu *are equivalent.*

Proof. Every proof in MLSu is a proof in MLS. For the proof of the other direction, by Theorem 1.17, given an MLL proof ∇, by structural induction we construct a proof Π in MLSu, omitting the obvious translation from the sequents to structures. The only difference between this construction and the proof in [37] is in the use of the rules $\mathsf{u_1}{\downarrow}$ and $\mathsf{u_2}{\downarrow}$ instead of the equalities for unit.

- If ∇ is an instance of the rule id in MLL then take the proof given with i↓, which is the generic version of ai↓, and derivable for ai↓, s and u_2↓. That is, for every instance of the rule id, take an instance of the rule

$$
\text{i↓} \frac{S\{1\}}{S[R \,\bindnasrepma\, \bar{R}]} \text{ , which we use as a short-hand for a derivation } \quad \begin{array}{c} S\{1\} \\ \Delta \big\| \{\,\text{ai↓}, \text{s}, \text{u}_2\text{↓}\,\} \\ S[R \,\bindnasrepma\, \bar{R}] \end{array} .
$$

The derivation Δ is constructed by structural induction on R. The base case is given with an instance of ai↓ or u_2↓ for the cases where R is an atom or a unit. For the inductive case, where $R = [R_1 \,\bindnasrepma\, R_2]$ or $R = (R_1 \otimes R_2)$, apply the induction hypothesis to

$$
\text{s} \cfrac{\text{i↓} \cfrac{\text{u}_2\text{↓} \cfrac{\text{i↓} \cfrac{S\{1\}}{S[R_2 \,\bindnasrepma\, \bar{R}_2]}}{S[R_2 \,\bindnasrepma\, (1 \otimes \bar{R}_2)]}}{S[R_2 \,\bindnasrepma\, ([R_1 \,\bindnasrepma\, \bar{R}_1] \otimes \bar{R}_2)]}}{S[R_1 \,\bindnasrepma\, R_2 \,\bindnasrepma\, (\bar{R}_1 \otimes \bar{R}_2)]} \quad .
$$

- If the rule $\otimes$ in MLL is the last rule applied in ∇, then take the following proof, where Π_1 and Π_2 are given by induction hypothesis.

$$
\text{s} \cfrac{\text{s} \cfrac{\begin{array}{c} 1 \\ \Pi_2 \big\| \\ \text{u}_2\text{↓} \cfrac{[A \,\bindnasrepma\, \Phi]}{([A \,\bindnasrepma\, \Phi] \otimes 1)} \\ \Pi_1 \big\| \\ ([A \,\bindnasrepma\, \Phi] \otimes [B \,\bindnasrepma\, \Psi]) \end{array}}{[([A \,\bindnasrepma\, \Phi] \otimes B) \,\bindnasrepma\, \Psi]}}{[(A \otimes B) \,\bindnasrepma\, \Phi \,\bindnasrepma\, \Psi]}
$$

- If the rule $\bindnasrepma$ in MLL is the last rule applied in ∇, then the proof of the premise exists by induction hypothesis.

- If the rule $\bot$ in MLL is the last rule applied in ∇, then take the following proof, where Π_1 is given by induction hypothesis.

$$
\text{u}_1\text{↓} \cfrac{\begin{array}{c} 1 \\ \Pi_1 \big\| \\ \Phi \end{array}}{[\Phi \,\bindnasrepma\, \bot]}
$$

- if ∇ is an instance of the rule 1 in MLL then take the proof given with 1.

Alternatively, a completeness argument for MLSu that does not rely on the sequent calculus is given by removing the instances of the rules u_1↑ and u_2↑ from the proofs in MLS by using a cut elimination argument: given a proof in MLS, we replace each instance of u_1↑ and u_2↑ with the following derivations, which introduce the instances of the cut rule ai↑.

$$\mathsf{u_1}{\downarrow}\cfrac{S(R \otimes 1)}{\mathsf{s}\cfrac{S([\perp \mathbin{\bindnasrepma} R] \otimes 1)}{\mathsf{ai}{\uparrow}\cfrac{S[R \mathbin{\bindnasrepma} (1 \otimes \perp)]}{S\{R\}}}} \qquad\qquad \mathsf{u_2}{\downarrow}\cfrac{S[R \mathbin{\bindnasrepma} \perp]}{\mathsf{ai}{\uparrow}\cfrac{S[R \mathbin{\bindnasrepma} (1 \otimes \perp)]}{S\{R\}}}$$

Then, to remove the instances of the cut rule, we apply a cut elimination procedure, which does not introduce any new instances of $\mathsf{u_1}{\uparrow}$ and $\mathsf{u_2}{\uparrow}$. In Subsection 2.3, we give such a cut elimination procedure for a more restricted system for multiplicative linear logic than MLSu, that is, MLSdli. This proof can be extended to MLSu by considering a similar case analysis, however without the need to use Lemma 2.26.   □

The treatment of units 1 and $\perp$ by inference rules as in MLSu becomes useful in controlling the non-determinism in proof search, in contrast to the case, where the units are treated with respect to a congruence relation.

Example 1.20. Consider the structure $[(\bar{a} \otimes \bar{b}) \mathbin{\bindnasrepma} a \mathbin{\bindnasrepma} b]$, which is provable in MLSu. When the units are treated by the congruence relation that contains the equalities for associativity and commutativity as well as the equalities

$$[R \mathbin{\bindnasrepma} \perp] \approx R \qquad \text{and} \qquad (R \otimes 1) \approx R$$

for unit as in [37], the switch rule can be applied bottom-up to this structure in 28 different ways. However, only 2 of these instances can result in a proof. This can be checked by using the implementation of these systems, given in [29]. Controlling the treatment of units with inference rules as in MLSu reduces the number of instances of the switch rule from 28 to 6, listed below, where those resulting in a proof are the cases given with $(iii.)$ and $(vi.)$ below.

$$(i.)\,\mathsf{s}\,\cfrac{(\bar{b} \otimes [\bar{a} \mathbin{\bindnasrepma} a \mathbin{\bindnasrepma} b])}{[(\bar{a} \otimes \bar{b}) \mathbin{\bindnasrepma} a \mathbin{\bindnasrepma} b]} \quad (ii.)\,\mathsf{s}\,\cfrac{[([b \mathbin{\bindnasrepma} \bar{a}] \otimes \bar{b}) \mathbin{\bindnasrepma} a]}{[(\bar{a} \otimes \bar{b}) \mathbin{\bindnasrepma} a \mathbin{\bindnasrepma} b]} \quad (iii.)\,\mathsf{s}\,\cfrac{[([a \mathbin{\bindnasrepma} \bar{a}] \otimes \bar{b}) \mathbin{\bindnasrepma} b]}{[(\bar{a} \otimes \bar{b}) \mathbin{\bindnasrepma} a \mathbin{\bindnasrepma} b]}$$

$$(iv.)\,\mathsf{s}\,\cfrac{(\bar{a} \otimes [\bar{b} \mathbin{\bindnasrepma} a \mathbin{\bindnasrepma} b])}{[(\bar{a} \otimes \bar{b}) \mathbin{\bindnasrepma} a \mathbin{\bindnasrepma} b]} \quad (v.)\,\mathsf{s}\,\cfrac{[([a \mathbin{\bindnasrepma} \bar{b}] \otimes \bar{a}) \mathbin{\bindnasrepma} b]}{[(\bar{a} \otimes \bar{b}) \mathbin{\bindnasrepma} a \mathbin{\bindnasrepma} b]} \quad (vi.)\,\mathsf{s}\,\cfrac{[([b \mathbin{\bindnasrepma} \bar{b}] \otimes \bar{a}) \mathbin{\bindnasrepma} a]}{[(\bar{a} \otimes \bar{b}) \mathbin{\bindnasrepma} a \mathbin{\bindnasrepma} b]}$$

2. Lazy and Deep Interaction

A controlled treatment of units is useful for reducing the non-determinism in proof search. However, an important part of the non-determinism in proof search with deep inference systems is due to the switch rule, which is the rule responsible for context management.

Example 2.1. Consider the structure $[(([(\bar{c} \otimes \bar{d}) \mathbin{\bindnasrepma} c \mathbin{\bindnasrepma} d] \otimes \bar{a} \otimes \bar{b}) \mathbin{\bindnasrepma} a \mathbin{\bindnasrepma} b]$, which is provable in MLSu. This structure is obtained by nesting the structure in Example 1.20 in itself. In the system, where the units are treated with the congruence relation, the switch rule can be applied bottom-up to this structure in 70 different ways. While MLSu prunes 46 of these instances, leaving 24 of them, only 4 of these instances provide proofs. The 2 of these instances at the deeper positions provide shorter proofs in comparison to the 2 others.

2.1. **Controlling Context Management.** With the following definitions, we re-design the switch rule to control the non-determinism in proof search due to context management.

Definition 2.2. Given a structure R, $\mathsf{at}\,R$ is the set of the atoms in R.

Example 2.3. For $R = [a \,\invamp\, \bar{a} \,\invamp\, b \,\invamp\, (\bar{a} \otimes \bot) \,\invamp\, (a \otimes \bar{b})]$, we have $\mathsf{at}\,R = \{a, \bar{a}, b, \bar{b}, \bot\}$.

Definition 2.4. Consider the switch rule.

$$\mathsf{s}\,\frac{S([U \,\invamp\, R] \otimes T)}{S[U \,\invamp\, (R \otimes T)]}$$

(i) We say that it is *interaction switch (*is*)* if $\mathsf{at}\,\overline{R} \cap \mathsf{at}\,U \neq \emptyset$, that is, R and U contain complementary atoms. In this case, we say R and U can interact.

(ii) It is *lazy switch (*ls*)* if U is a copar or an atom different from $\bot$.

(iii) It is *deep switch (*ds*)* if R is a par or an atom different from 1.

By combining these restrictions, we thus obtain 8 different inference rules. For example, by imposing all these restrictions, we obtain the rule *deep lazy interaction switch* (dlis).

Example 2.5. Consider the structure $[(\bar{a} \otimes \bar{b} \otimes \bot) \,\invamp\, a \,\invamp\, b \,\invamp\, 1]$ with the following instances of the switch rule where the R and U structures are shaded:

$$(i.)\ \frac{[([\bar{a} \,\invamp\, b \,\invamp\, 1] \otimes \bar{b} \otimes \bot) \,\invamp\, a]}{[(\bar{a} \otimes \bar{b} \otimes \bot) \,\invamp\, a \,\invamp\, b \,\invamp\, 1]} \qquad (ii.)\ \frac{[([a \,\invamp\, \bar{a}] \otimes \bar{b} \otimes \bot) \,\invamp\, b \,\invamp\, 1]}{[(\bar{a} \otimes \bar{b} \otimes \bot) \,\invamp\, a \,\invamp\, b \,\invamp\, 1]}$$

(i) is an instance of ds but not an instance of ls or is. (ii) is an instance of dlis because it is an instance of the both ls and ds, and also $\mathsf{at}\,\bar{R} = \{a\}$ and $\mathsf{at}\,U = \{a\}$, and thus $\mathsf{at}\,\bar{R} \cap \mathsf{at}\,U = \{a\} \neq \emptyset$.

Remark 2.6. Replacing the interaction condition $\mathsf{at}\,\bar{R} \cap \mathsf{at}\,U \neq \emptyset$ in Definition 2.4 with $\mathsf{at}\,\bar{U} = \mathsf{at}\,R$ or $\mathsf{at}\,\bar{U} \subseteq \mathsf{at}\,R$ results in an incomplete system for multiplicative linear logic. To see this on an example, consider the structure:

$$[(b \otimes \bar{c}) \,\invamp\, (a \otimes c) \,\invamp\, ([\bar{a} \,\invamp\, \bar{b}] \otimes [\bar{d} \,\invamp\, \bar{e}]) \,\invamp\, (d \otimes f) \,\invamp\, (\bar{f} \otimes e)]$$

This structure is provable in MLSu, however it cannot be proved in a system where the switch rule is modified such that $\mathsf{at}\,\bar{U} = \mathsf{at}\,R$ or $\mathsf{at}\,\bar{U} \subseteq \mathsf{at}\,R$.

The rule *deep switch* was introduced in [36] to simplify the cut elimination procedure by means of permutations of the inference rules. The intuition behind the naming of this rule is that it minimizes the extra depth introduced by the switch rule as a result of its bottom up application.

The restrictions on lazy and deep switch are dual restrictions with respect to the structures that they are imposed on: in its instances, lazy switch imposes U in a par context not to be a par structure, whereas deep switch imposes R in a copar context not to be a copar structure. Figure 3 shows the relation between the 8 rules obtained, where the arrow denotes the inclusion relation of their instances, for example, every instance of the rule dis is an instance of ds and it is also an instance of is.

Figure 3 left: partial order diagram with nodes dlis (top); lis, dls, dis (second level); ls, is, ds (third level); s (bottom).

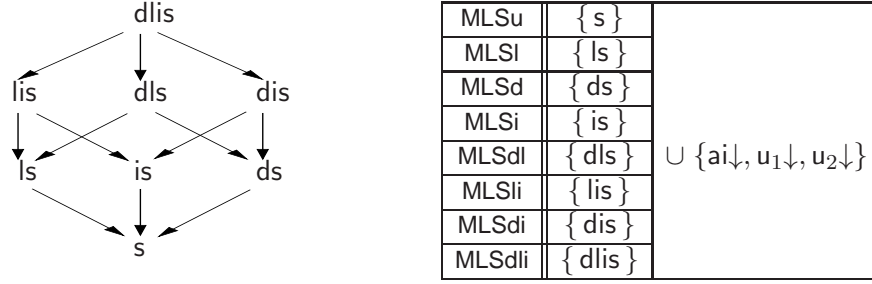| MLSu	{ s }	
MLSl	{ ls }	
MLSd	{ ds }	
MLSi	{ is }	
MLSdl	{ dls }	$\cup$ { ai↓, u$_1$↓, u$_2$↓ }
MLSli	{ lis }	
MLSdi	{ dis }	
MLSdli	{ dlis }	

Figure 3: **Left:** A partial order representation of the inclusion relation of the different switch rule instances, for example, every instance of the rule dlis is an instance of the rule lis, which is an instance of the rule is. **Right:** The systems that are obtained by replacing the switch rule in MLSu with its restricted versions given in Definition 2.4.

Definition 2.7. *System* MLSu *with deep lazy interaction switch*, or MLSdli is the system { ai↓, dlis, u$_1$↓, u$_2$↓ }. We define the other systems obtained by replacing in MLSu the switch rule with one of the rules given in Figure 3 similarly by adding the prefix of the switch rule as a suffix to the name of the system. For example, the *system* MLSu *with deep interaction switch*, or MLSdi is the system { ai↓, dis, u$_1$↓, u$_2$↓ }.

In the sequent calculus, context management is performed by the inference rule R⊗, depicted below, which spreads the context of a copar formula at the top level to the branches of the proof. The switch rule simulates this rule as follows.

$$\otimes \frac{\vdash R, \phi \quad \vdash T, \psi}{\vdash \phi, (R \otimes T), \psi} \qquad \rightsquigarrow \qquad \mathsf{s} \frac{\mathsf{s} \dfrac{([R \,\bindnasrepma\, \phi] \otimes [T \,\bindnasrepma\, \psi])}{[([R \,\bindnasrepma\, \phi] \otimes T) \,\bindnasrepma\, \psi]}}{[(R \otimes T) \,\bindnasrepma\, \phi \,\bindnasrepma\, \psi]}$$

Once the rule ⊗ is applied, the sequents in the two branches cannot exchange any formula, thus the communication between the formula at the two branches becomes impossible while going up in the proofs. However, in order for a proof to be constructed both of these two branches must contain dual atoms.

The rule interaction switch (is) exploits this observation while preserving the possibility to be applied at arbitrary depths inside logical expressions. Thus, the rule is can simulate the sequent calculus proofs, but also provide shorter proofs due to deep instances.

The rule dls exploits the capability of the switch rule to manage contexts gradually. This is done by constraining the size of the formulae that are brought together for interaction, where interaction is eventually realized by the instances of the rule ai↓ that correspond to the axiom of sequent calculus. In its shallow instances, the rule dls is similar to constraining the rule ⊗ such that the sequent ϕ consists of a single non-par formula, and R a non-copar formula. In fact, such a restriction in the sequent calculus would not affect the size of the proofs. However, as we show below, the system with the rule dlis polynomially simulates the sequent calculus proofs, while providing shorter proofs due to deep inference. In fact, in classical logic, the gain in proof length becomes exponential in certain cases. Because the rule dlis does not impose any restriction on the deep applicability of the inference rules, it does not cause a loss in shorter proofs that are available due to deep inference. However,

	1.	2.	3.	4.	5.	6.	7.	8.
MLS	22	28	70	114	112	112	112	118
MLSu	6	8	24	42	42	42	42	42
MLSl	2	5	16	18	28	28	28	18
MLSd	3	8	15	21	24	24	24	42
MLSi	6	6	15	30	26	31	36	26
MLSdl	1	5	10	9	16	16	16	18
MLSli	2	3	8	9	14	18	22	7
MLSdi	3	6	8	12	12	15	18	26
MLSdli	1	3	4	3	6	8	10	7

Figure 4: The number of immediate redexes at the beginning of proof search for the systems MLS, MLSu, MLSl, MLSi, MLSd, MLSli, MLSdl, MLSdi and MLSdli for the structures given in Example 2.9. Unlike other systems, MLS is implemented with equalities for unit, shown in Example 1.20, as in [37].

restricting the sequent calculus in a similar manner would not reveal a complete system. For example, requiring in the instances of the sequent calculus context management rule $\otimes$ that R and ϕ have common dual atoms as in the rule dlis would result in an incomplete system. Thus, a straight-forward mapping from the sequent calculus to the system MLSdli cannot be used to prove the completeness of MLSdli. The example below illustrates the reason for this.

Example 2.8. There are cases similar to the ones in the proofs of the structure

$$[(\bot \otimes \bot) \,\wp\, a \,\wp\, \bar{a} \,\wp\, b \,\wp\, \bar{b}]$$

below, which admit sequent calculus proofs that cannot be directly mapped to MLSdli without a transformation of these proofs. Both of the proofs below are in MLSu, however the proof on the left simulates the only shallow inference proof, whereas the proof in the right is one of the many proofs in MLSdli.

$$
\mathsf{ai}{\downarrow};\ \mathsf{u_2}{\downarrow}\ \cfrac{\mathsf{ai}{\downarrow}\ \cfrac{1}{[b \,\wp\, \bar{b}]}}{\cfrac{([a \,\wp\, \bar{a}] \otimes [b \,\wp\, \bar{b}])}{\mathsf{u_1}{\downarrow}\ \cfrac{([a \,\wp\, \bar{a}] \otimes [\bot \,\wp\, b \,\wp\, \bar{b}])}{\mathsf{u_1}{\downarrow}\ \cfrac{([\bot \,\wp\, a \,\wp\, \bar{a}] \otimes [\bot \,\wp\, b \,\wp\, \bar{b}])}{\mathsf{s}\ \cfrac{[([\bot \,\wp\, a \,\wp\, \bar{a}] \otimes \bot) \,\wp\, b \,\wp\, \bar{b}]}{\mathsf{s}\ \ [(\bot \otimes \bot) \,\wp\, a \,\wp\, \bar{a} \,\wp\, b \,\wp\, \bar{b}]}}}}}
$$

$$
\mathsf{ai}{\downarrow};\ \mathsf{u_2}{\downarrow}\ \cfrac{\mathsf{ai}{\downarrow}\ \cfrac{1}{[\bot \,\wp\, 1]}}{\cfrac{[([\bot \,\wp\, 1] \otimes \bot) \,\wp\, 1]}{\mathsf{dlis}\ \cfrac{[(\bot \otimes \bot) \,\wp\, 1 \,\wp\, 1]}{\mathsf{ai}{\downarrow}\ \cfrac{[(\bot \otimes \bot) \,\wp\, 1 \,\wp\, b \,\wp\, \bar{b}]}{\mathsf{ai}{\downarrow}\ \ [(\bot \otimes \bot) \,\wp\, a \,\wp\, \bar{a} \,\wp\, b \,\wp\, \bar{b}]}}}}
$$

In the proof on the right, we exploit the duality between 1 and $\bot$, which remains hidden in the rigidity of the sequent-calculus-like shallow inference proof on the left, where the instances of the rule $\mathsf{ai}{\downarrow}$ simulate the sequent calculus axioms at the leafs of the proof tree.

Example 2.9. The following structures are provable in MLSu.

1. $[1 \otimes 1 \otimes 1 \otimes (\bot \otimes \bot \otimes \bot)]$
2. $[(a \otimes b) \otimes (a \otimes b) \otimes ([\bar{a} \otimes \bar{b}] \otimes [\bar{a} \otimes \bar{b}])]$
3. $[\bar{a} \otimes \bar{b} \otimes (a \otimes b \otimes [\bar{c} \otimes \bar{d} \otimes (c \otimes d)])]$
4. $[a \otimes b \otimes c \otimes (\bar{a} \otimes \bar{b} \otimes \bar{c})]$
5. $[\bar{a} \otimes \bar{b} \otimes (a \otimes b \otimes [\bar{c} \otimes \bar{d} \otimes (c \otimes d \otimes [\bar{e} \otimes \bar{f} \otimes (e \otimes f)])])]$
6. $[\bar{a} \otimes \bar{b} \otimes (a \otimes b \otimes [\bar{c} \otimes \bar{d} \otimes (c \otimes d \otimes [\bar{a} \otimes \bar{b} \otimes (a \otimes b)])])]$
7. $[\bar{a} \otimes \bar{b} \otimes (a \otimes b \otimes [\bar{a} \otimes \bar{b} \otimes (a \otimes b \otimes [\bar{a} \otimes \bar{b} \otimes (a \otimes b)])])]$
8. $[\bar{c} \otimes (\bar{d} \otimes \bar{e}) \otimes (e \otimes \bar{a}) \otimes (c \otimes [a \otimes d])]$

Figure 4 depicts the number of all the immediate rule instances on these structures in the systems MLS, MLSu, MLSl, MLSi, MLSd, MLSli, MLSdl, MLSdi and MLSdli. While controlling the nondeterminism in context management, the rule dlis provides a more immediate access to the shorter proofs which are available in nested structures as in (3.), (5.), (6.) and (7.) above. This is because the proof can be constructed by starting from the substructures as illustrated below on structure (3.).

$$
\mathsf{ai}{\downarrow};\ \mathsf{u_2}{\downarrow}\ \cfrac{\mathsf{ai}{\downarrow};\ \mathsf{u_2}{\downarrow}\ \cfrac{\mathsf{ai}{\downarrow}\ \cfrac{\mathsf{dlis}\ \cfrac{\mathsf{ai}{\downarrow}\ \cfrac{1}{[a \otimes \bar{a}]}}{[\bar{a} \otimes (a \otimes [b \otimes \bar{b}])]}}{[\bar{a} \otimes \bar{b} \otimes (a \otimes b)]}}{[\bar{a} \otimes \bar{b} \otimes (a \otimes b \otimes [\bar{c} \otimes c])]}}{\mathsf{dlis}\ \cfrac{[\bar{a} \otimes \bar{b} \otimes (a \otimes b \otimes [\bar{c} \otimes (c \otimes [d \otimes \bar{d}])])]}{[\bar{a} \otimes \bar{b} \otimes (a \otimes b \otimes [\bar{c} \otimes \bar{d} \otimes (c \otimes d)])]}}
$$

An alternative proof of this structure in MLSdli is the following one. This proof simulates a sequent calculus proof, where the inference rules are applied only at the top level connectives.

$$
\mathsf{ai}{\downarrow};\ \mathsf{u_2}{\downarrow}\ \cfrac{\mathsf{ai}{\downarrow};\ \mathsf{u_2}{\downarrow}\ \cfrac{\mathsf{ai}{\downarrow};\ \mathsf{u_2}{\downarrow}\ \cfrac{\mathsf{ai}{\downarrow}\ \cfrac{1}{[d \otimes \bar{d}]}}{([c \otimes \bar{c}] \otimes [d \otimes \bar{d}])}}{([a \otimes \bar{a}] \otimes [c \otimes \bar{c}] \otimes [d \otimes \bar{d}])}}{([b \otimes \bar{b}] \otimes [a \otimes \bar{a}] \otimes [c \otimes \bar{c}] \otimes [d \otimes \bar{d}])}
$$

with the lower portion:

$$
\mathsf{dlis}\ \cfrac{([b \otimes \bar{b}] \otimes [a \otimes \bar{a}] \otimes [c \otimes \bar{c}] \otimes [d \otimes \bar{d}])}{\mathsf{dlis}\ \cfrac{([b \otimes \bar{b}] \otimes [a \otimes \bar{a}] \otimes [\bar{c} \otimes (c \otimes [d \otimes \bar{d}])])}{\mathsf{dlis}\ \cfrac{([b \otimes \bar{b}] \otimes [a \otimes \bar{a}] \otimes [\bar{c} \otimes \bar{d} \otimes (c \otimes d)])}{\mathsf{dlis}\ \cfrac{[\bar{a} \otimes ([b \otimes \bar{b}] \otimes a \otimes [\bar{c} \otimes \bar{d} \otimes (c \otimes d)])]}{[\bar{a} \otimes \bar{b} \otimes (a \otimes b \otimes [\bar{c} \otimes \bar{d} \otimes (c \otimes d)])]}}}}
$$

2.2. **Correctness: Exploring the Switch Lattice.** The restrictions on the switch rule, given in Definition 2.4, control the nondeterminism in context management during proof construction. By exploring the lattice in Figure 3, we now show that these different versions of the switch rule preserve completeness. Showing the completeness of the systems in the direction of the arrows in Figure 3 is straightforward. This is because all instances of a system at the source of an arrow are also instances of the system at the target of that arrow. For example, every instance of the rule dlis is an instance of the rules lis, dls or dis. Similarly, every instance of dls is an instance of ds or ls, which are all instances of the switch rule. However, exploring this lattice from the bottom to the top requires a global analysis of proofs. This is because the deep applicability of the inference rules in these systems brings about the possibility for the substructures to interact with their contexts in nested structures. The example below illustrates this idea.

Example 2.10. Consider the following proof on the left in MLSu and a proof of the same structure in MLSdli on the right. In the proof transformation below, we remove the bottom-most instance of the switch rule in the proof on the left, and thereby replace this proof with the proof on the right, where all the instances of switch are instances of the rule dlis.

$$
\cfrac{
\mathsf{ai{\downarrow};\ u_2{\downarrow}}\ \cfrac{
\mathsf{dlis}\ \cfrac{
\mathsf{ai{\downarrow};\ u_2{\downarrow}}\ \cfrac{
\mathsf{dlis}\ \cfrac{
\mathsf{ai{\downarrow}}\ \cfrac{1}{[c \mathbin{\bindnasrepma} \bar{c}]}
}{[c \mathbin{\bindnasrepma} ([a \mathbin{\bindnasrepma} \bar{a}] \otimes \bar{c})]}
}{[a \mathbin{\bindnasrepma} c \mathbin{\bindnasrepma} (\bar{a} \otimes \bar{c})]}
}{[([b \mathbin{\bindnasrepma} \bar{b}] \otimes [a \mathbin{\bindnasrepma} c]) \mathbin{\bindnasrepma} (\bar{a} \otimes \bar{c})]}
}{[(b \otimes [a \mathbin{\bindnasrepma} c]) \mathbin{\bindnasrepma} \bar{b} \mathbin{\bindnasrepma} (\bar{a} \otimes \bar{c})]}
}{\mathsf{s}\ \ [a \mathbin{\bindnasrepma} (b \otimes c) \mathbin{\bindnasrepma} \bar{b} \mathbin{\bindnasrepma} (\bar{a} \otimes \bar{c})]}
\qquad \rightsquigarrow \qquad
\cfrac{
\mathsf{ai{\downarrow};\ u_2{\downarrow}}\ \cfrac{
\mathsf{dlis}\ \cfrac{
\mathsf{ai{\downarrow};\ u_2{\downarrow}}\ \cfrac{
\mathsf{dlis}\ \cfrac{
\mathsf{ai{\downarrow}}\ \cfrac{1}{[c \mathbin{\bindnasrepma} \bar{c}]}
}{[([b \mathbin{\bindnasrepma} \bar{b}] \otimes c) \mathbin{\bindnasrepma} \bar{c}]}
}{[(b \otimes c) \mathbin{\bindnasrepma} \bar{b} \mathbin{\bindnasrepma} \bar{c}]}
}{[(b \otimes c) \mathbin{\bindnasrepma} \bar{b} \mathbin{\bindnasrepma} ([a \mathbin{\bindnasrepma} \bar{a}] \otimes \bar{c})]}
}{[a \mathbin{\bindnasrepma} (b \otimes c) \mathbin{\bindnasrepma} \bar{b} \mathbin{\bindnasrepma} (\bar{a} \otimes \bar{c})]}
$$

We explore the lattice in Figure 3 step by step to consider all the systems obtained by replacing the switch rule with its restricted versions given in Definition 2.4. Thereby, we show that proofs in MLSu can be transformed into proofs in MLSdli and other intermediate systems, and show that these systems are complete for multiplicative linear logic.

Let us first give some definitions, which we use in the proofs below.

Definition 2.11. [18] Given a structure R, its *atom occurrences* are obtained by considering all the atoms appearing in R as distinct (for example, by indexing them so that two atoms, which are equal, get different indices). Then, $\mathsf{occ}\,R$ is the set of all the atom occurrences appearing in R. The *size* of R is the cardinality of the set $\mathsf{occ}\,R$.

Example 2.12. For R in Example 2.3, $\mathsf{occ}\,R = \{a_1, \bar{a}_1, b, \bar{b}, a_2, \bar{a}_2, \bot_1\}$.

Definition 2.13. [18] Let R be a structure. The *structural relation* $\downarrow_R$ is the minimal set such that $\downarrow_R \subset (\mathsf{occ}\,R)^2$ and, for every $S\{\ \}$, U and V and for every a in U and b in V, the following holds: if $R = S[U \mathbin{\bindnasrepma} V]$ then $a \downarrow_R b$. The notation $\mid \downarrow_R \mid$ denotes the cardinality of the set $\downarrow_R$.

Example 2.14. For R in Example 2.3, in $\downarrow_R$ we have $(a_1, \bar{a}_1)$, (a_1, b), $(a_1, \bar{a}_2)$, $(a_1, \bot_1)$, (a_1, a_2), $(a_1, \bar{b})$, $(\bar{a}_1, b)$, $(\bar{a}_1, \bar{a}_2)$, $(\bar{a}_1, \bot_1)$, $(\bar{a}_1, a_2)$, $(\bar{a}_1, \bar{b})$, $(b, \bar{a}_2)$, $(b, \bot_1)$, (b, a_2), $(b, \bar{b})$, $(\bar{a}_2, a_2)$, $(\bar{a}_2, \bar{b})$, $(\bot_1, a_2)$, $(\bot_1, \bar{b})$ and their symmetric instances.

$$\mathsf{ai}{\downarrow}\,\frac{S\{1\}}{S\left[\;a\cdots\cdots\cdots\bar{a}\;\right]} \qquad \mathsf{s}\,\frac{S\left[\begin{array}{cc} R\cdots & T \\ & \;\;U \end{array}\right]}{S\left[\begin{array}{cc} R\cdots\cdots T \\ & \;\;U \end{array}\right]} \qquad \begin{array}{l} \mathsf{u_1}{\downarrow}\,\dfrac{S\{R\}}{S\left[\;R\cdots\cdots\cdots\bot\;\right]} \\[2em] \mathsf{u_2}{\downarrow}\,\dfrac{S\{R\}}{S\left(\;R\qquad 1\;\right)} \end{array}$$

Figure 5: The effect of the inference rules of MLSu on structures with respect to Definition 2.13, where the dashed lines indicate the the relation ↓.

When considered from the point of view of Definition 2.13, the role of the inference rules on a structure R can be seen as modifying the relation $\downarrow_R$. The Figure 5 displays the inference rules of MLSu from such a point of view. The following remarks explicitly identify these modifications, and demonstrate how the relation ↓ is used as an induction measure.

Remark 2.15. Let $R = S[P \,\invamp\, \bot]$ and $R' = S\{P\}$ be structures such that R consists of pairwise distinct atoms and $\bot$ is uniquely marked. If

$$\mathsf{u_1}{\downarrow}\,\frac{R'}{R} \quad then \quad \downarrow_{R'} = \downarrow_R \setminus \{\,(x,y) \mid (x,y) \in\downarrow_R \wedge (\,x = \bot \vee y = \bot\,)\,\}\,.$$

Remark 2.16. Let $R = S(P \otimes 1)$ and $R' = S\{P\}$ be structures such that R consists of pairwise distinct atoms and 1 is uniquely marked. If

$$\mathsf{u_2}{\downarrow}\,\frac{R'}{R} \quad then \quad \downarrow_{R'} = \downarrow_R \setminus \{\,(x,y) \mid (x,y) \in\downarrow_R \wedge (\,x = 1 \vee y = 1\,)\,\}\,.$$

Remark 2.17. Let $R = S[a \,\invamp\, \bar{a}]$ and $R' = S\{1\}$ be structures such that R consists of pairwise distinct atoms and the occurrence of 1 is uniquely marked. Let $\Phi = \mathsf{occ}\,R \setminus \{a, \bar{a}\}$. If

$$\mathsf{ai}{\downarrow}\,\frac{R'}{R}\;,$$

given that $\mathsf{occ}\,R' = \Phi \cup \{1\}$, then

$$\begin{aligned} \downarrow_{R'} = \;& (\downarrow_R\; \cup \{\,(1,x), (x,1) \mid x \in \Phi\,\}) \\ & \setminus \{\,(x,y) \mid (x,y) \in\downarrow_R \wedge (\,x = a \vee y = a \vee x = \bar{a} \vee y = \bar{a}\,)\,\}\,. \end{aligned}$$

Remark 2.18. Let $R = S[(P \otimes T) \,\invamp\, U]$ and $R' = S([P \,\invamp\, U] \otimes T)$ be structures such that R consists of pairwise distinct atoms. If

$$\mathsf{s}\,\frac{R'}{R} \quad then \quad \downarrow_{R'} = \downarrow_R \setminus \{\,(x,y), (y,x) \mid x \in \mathsf{occ}\,T \wedge y \in \mathsf{occ}\,U\,\}\,.$$

With these observations of the remarks above, we can now state the proposition below.

$$1 \qquad\qquad 1$$

$$\mathsf{ai}{\downarrow} \; \rule{6cm}{0.4pt}$$

$$b \cdots\cdots\cdots \bar{b} \qquad [\bar{b} \,\bindnasrepma\, b]$$

$$\mathsf{u_2}{\downarrow} \; \rule{6cm}{0.4pt}$$

$$1$$

$$[(1 \otimes \bar{b}) \,\bindnasrepma\, b]$$

$$b \cdots\cdots \bar{b}$$

$$\mathsf{ai}{\downarrow} \; \rule{6cm}{0.4pt}$$

$$a \cdots\cdots \bar{a}$$

$$[([a \,\bindnasrepma\, \bar{a}] \otimes \bar{b}) \,\bindnasrepma\, b]$$

$$b \cdots\cdots \bar{b}$$

$$\mathsf{s} \; \rule{6cm}{0.4pt}$$

$$a \cdots\cdots \bar{a}$$

$$[(\bar{a} \otimes \bar{b}) \,\bindnasrepma\, a \,\bindnasrepma\, b]$$

$$b \cdots\cdots \bar{b}$$

Figure 6: A proof of the structure $[(\bar{a} \otimes \bar{b}) \,\bindnasrepma\, a \,\bindnasrepma\, b]$ and the corresponding relation $\downarrow$ as in Definition 2.13, where the dashed lines denote the relation $\downarrow$.

Proposition 2.19. *For any rule instance* $\rho \, \dfrac{R'}{R}$ *of any rule* $\rho \in \{\mathsf{ai}{\downarrow}, \mathsf{s}, \mathsf{u_1}{\downarrow}\}$ *of* MLSu*, we have that* $|\downarrow_{R'}| \,<\, |\downarrow_R|$*, and for* $\rho = \mathsf{u_2}{\downarrow}$ *we have that* $|\downarrow_{R'}| \,\le\, |\downarrow_R|$*.*

Proof. As depicted in Figure 5, it immediately follows from Remark 2.15, Remark 2.16, Remark 2.17 and Remark 2.18 that in any instance of the rules $\mathsf{u_1}{\downarrow}$, $\mathsf{ai}{\downarrow}$ and s the cardinality

of the relation $\downarrow$ is smaller in the premise than in the conclusion, and in an instance of the rule $\mathsf{u_2}\downarrow$ it is either smaller or remains unchanged. $\square$

Proposition 2.20. *The length of any proof of any structure R in* MLSu *is bounded by* $\mathcal{O}(|\mathsf{occ}\,R|^2)$.

Proof. Follows immediately from Proposition 2.19. $\square$

Example 2.21. Let R be the structure $[(\bar{a} \otimes \bar{b} \otimes \bot) \,\bar{\otimes}\, a \,\bar{\otimes}\, b \,\bar{\otimes}\, 1]$. We consider the rule instance (ii) in Example 2.5. If R' denotes the premise of this rule instance, we have $\downarrow_{R'} = \downarrow_R \setminus \{(a, \bar{b}), (a, \bot), (\bar{b}, a), (\bot, a)\}$.

Example 2.22. Figure 6 displays an example proof of the structure $[(\bar{a} \otimes \bar{b}) \,\bar{\otimes}\, a \,\bar{\otimes}\, b]$ and the corresponding modifications on relation $\downarrow$.

Proposition 2.23. *For any multiplicative linear logic structure R and context S, in* MLSu, $S\{R\}$ *has a proof if and only if*

(i) $S[R \,\bar{\otimes}\, \bot]$ *has a proof;*

(ii) $S(R \otimes 1)$ *has a proof.*

Proof. For the if direction, the required proofs are constructed in MLSu by applying the rules $\mathsf{u_1}\downarrow$ and $\mathsf{u_2}\downarrow$. For the only if direction, we construct the required proof first in MLS by applying the rules $\mathsf{u_1}\uparrow$ and $\mathsf{u_2}\uparrow$, then transform these proofs into proofs in MLSu by applying Theorem 1.19. $\square$

Proposition 2.24. *The systems* MLSu *and* MLSl *are equivalent.*

Proof. Every proof in MLSl is a proof in MLSu as every instance of ls is an instance of s. For the other direction, we transform proofs in MLSu into proofs in MLSl as follows: let us call redundant each occurrence of $\bot$ in a structure Q if for some structure U and context S, we have that $Q = S[U \,\bar{\otimes}\, \bot]$.

Let P be a structure with a proof Π in MLSu. If P has any redundant $\bot$ than we apply to P the rule $\mathsf{u_1}\downarrow$ exhaustively to obtain P' as depicted below such that P' does not have any redundant $\bot$. By Proposition 2.23, if P has a proof Π then P' has a proof Π' in MLSu.

$$
\begin{array}{ccc}
\begin{array}{c} 1 \\ \Pi \,\|\, \mathsf{MLSu} \\ P \end{array}
&
\rightsquigarrow
&
\begin{array}{c} 1 \\ \Pi' \,\|\, \mathsf{MLSu} \\ P' \\ \Delta_1 \,\|\, \{\mathsf{u_1}\downarrow\} \\ P \end{array}
\end{array}
\quad .
$$

In Π' we single out the bottom-most rule instance that is either (i) an instance of ls that introduces a redundant $\bot$ or (ii) an instance of $\mathsf{u_2}\downarrow$ that introduces a redundant $\bot$ or (iii) an instance of s that is not an instance of ls. (If there is no such instance then we are done.) We depict case (i) and (iii) below as case (ii) is similar to case (i).

$$\stackrel{1}{\Pi'_1}\Big\|\mathsf{MLSu}$$
$$\mathsf{ls}\frac{S([\bot\,\mathbin{⅋}\,U]\otimes T)}{S[(\bot\otimes T)\,\mathbin{⅋}\,U]}$$
$$\Delta_2\Big\|\mathsf{MLSl}$$
$$P'$$
$$\Delta_1\Big\|\{\mathsf{u}_1\mathord{\downarrow}\}$$
$$P$$

$$\stackrel{(i.)}{\leftsquigarrow}$$

$$\stackrel{1}{\Pi'}\Big\|\mathsf{MLSu}$$
$$P'$$
$$\Delta_1\Big\|\{\mathsf{u}_1\mathord{\downarrow}\}$$
$$P$$

$$\stackrel{(iii.)}{\rightsquigarrow}$$

$$\stackrel{1}{\Pi'_3}\Big\|\mathsf{MLSu}$$
$$\mathsf{s}\frac{S([R\,\mathbin{⅋}\,U_1\,\mathbin{⅋}\,\ldots\,\mathbin{⅋}\,U_n]\otimes T)}{S[(R\otimes T)\,\mathbin{⅋}\,U_1\,\mathbin{⅋}\,\ldots\,\mathbin{⅋}\,U_n]}$$
$$\Delta_2\Big\|\mathsf{MLSl}$$
$$P'$$
$$\Delta_1\Big\|\{\mathsf{u}_1\mathord{\downarrow}\}$$
$$P$$

.

In these three cases we proceed as follows:

(i) To the premise $S([\bot\,\mathbin{⅋}\,U]\otimes T)$ of the instance of ls, we apply the rule $\mathsf{u}_1\mathord{\downarrow}$. By Proposition 2.23, we obtain the proof Π''_1 below, which we replace with the proof Π'_1.

$$\stackrel{1}{\Pi''_1}\Big\|\mathsf{MLSu}$$
$$\mathsf{u}_1\mathord{\downarrow}\frac{S(U\otimes T)}{S([\bot\,\mathbin{⅋}\,U]\otimes T)}$$
$$\mathsf{ls}\frac{}{S[(\bot\otimes T)\,\mathbin{⅋}\,U]}$$

(ii) To the premise $S[\bot\,\mathbin{⅋}\,U]$ of the instance of $\mathsf{u}_2\mathord{\downarrow}$, we apply the rule $\mathsf{u}_1\mathord{\downarrow}$. By Proposition 2.23, we obtain the proof Π''_2 below, which we replace with the proof Π'_2.

$$\stackrel{1}{\Pi''_2}\Big\|\mathsf{MLSu}$$
$$\mathsf{u}_1\mathord{\downarrow}\frac{S\{U\}}{S[\bot\,\mathbin{⅋}\,U]}$$
$$\mathsf{u}_2\mathord{\downarrow}\frac{}{S[(\bot\otimes\mathbf{1})\,\mathbin{⅋}\,U]}$$

(iii) Because none of the $U_1,\ldots,U_n$ is $\bot$, we can replace the instance of the switch with a derivation that consists of n instances of the rule ls.

$$\mathsf{s}\frac{S([R\,\mathbin{⅋}\,U_1\,\mathbin{⅋}\,\ldots\,\mathbin{⅋}\,U_n]\otimes T)}{S[(R\otimes T)\,\mathbin{⅋}\,U_1\,\mathbin{⅋}\,\ldots\,\mathbin{⅋}\,U_n]}\qquad\rightsquigarrow\qquad\mathsf{ls}\frac{\mathsf{ls}\dfrac{S([R\,\mathbin{⅋}\,U_1\,\mathbin{⅋}\,\ldots\,\mathbin{⅋}\,U_n]\otimes T)}{\vdots}}{\mathsf{ls}\dfrac{S[([R\,\mathbin{⅋}\,U_1]\otimes T)\,\mathbin{⅋}\,U_2\ldots\,\mathbin{⅋}\,U_n]}{S[(R\otimes T)\,\mathbin{⅋}\,U_1\,\mathbin{⅋}\,U_2\,\mathbin{⅋}\,\ldots\,\mathbin{⅋}\,U_n]}}$$

If $R=\bot$, as in case (i), we apply the rule $\mathsf{u}_1\mathord{\downarrow}$ to the premise of the derivation on the right above, and obtain a proof Π''_3 in MLSu, which we replace with Π'_3. If R is different from $\bot$, we proceed with Π'_3.

By Proposition 2.19, for the case (i) we have that $|\downarrow_{S(U \otimes T)}| < |\downarrow_P|$ and for the case (ii) we have that $|\downarrow_{S([R \otimes U_1 \otimes ... \otimes U_n] \otimes T)}| < |\downarrow_P|$ and $|\downarrow_{S([U_1 \otimes ... \otimes U_n] \otimes T)}| < |\downarrow_P|$. We thus repeat the procedure above inductively until all the instances of the rule s that are not instances of the rule ls are removed. □

The transformation in the proof of Proposition 2.24 maps instances of the switch rule to derivations that consist of sequences of the lazy switch rule instances. This transformation is linear in the number of the structures connected by par in U. Although lazy instances of the switch rule appear to increase the length of the proofs in comparison to the instances of the switch rule, because the rule ai↓ is applied to pairs of dual atoms, this restriction becomes beneficial in non-deterministic proof search. This is because the laziness condition reduces the size of the information processed at every proof step.

The restrictions of the rule ls and ds are dual restrictions. In [36], Strassburger proves this proposition by permuting the instances of the switch rule that are not instances of the rule ds up in the proofs. Here, we give a similar proof.

Proposition 2.25. *The systems* MLSu, MLSl, MLSd *and* MLSdl *are equivalent.*

Proof. By Proposition 2.24, we have that MLSu and MLSl are equivalent. Every proof in MLSdl is a proof in MLSd and MLSl, and every proof in MLSd is a proof in MLSu. For the proof of the other direction, given Proposition 2.24, we transform proofs in MLSl into proofs in MLSdl, as this suffices to make the equality commute for all four systems. (Note that the procedure in the proof of Proposition 2.24 can also be used to transform proofs in MLSd into proofs in MLSdl.)

Given a proof Π of MLSl, by induction we transform it into a proof in MLSdl. We associate with each proof Π a pair $\#\Pi = (n, r)$, where n is the number of inferences of the switch rule in Π that are not deep, and r is the number of inferences above the top-most non-deep instance of the switch rule in Π (it is zero if there is no such instance). We prove by induction on $\#\Pi$ under the usual lexicographical order. In the base case, where $n = 0$, we are done as all the instances of the switch rule must be the instances of deep switch. For the inductive cases, we single out the top-most non-deep instance of switch in Π. This cannot be the top-most rule instance of Π, that is, $r > 0$. There is thus an instance of a rule ρ and a structure Q such that

$$
\rho \frac{\substack{1 \\ \| \\ Q} }{S([R \otimes U] \otimes T)} \\
\mathsf{ls} \frac{S([R \otimes U] \otimes T)}{S[(R \otimes T) \otimes U]} \quad .
$$

If the instance of ρ is inside any of S, U, R, or T, we can exchange the instances of ρ and the instance of switch, and obtain Π' such that $\#\Pi' = (n, r - 1)$, where we can apply the induction hypothesis. Otherwise, ρ cannot be ai↓ as this would contradict with the instance of switch being non-deep. Thus, there are three cases, where $\rho = \mathsf{u}_1\downarrow$, $\rho = \mathsf{u}_2\downarrow$ or $\rho = \mathsf{dls}$.

If $\rho = \mathsf{u}_1\downarrow$, the only case to consider is the one where $U = \bot$, which is impossible if the switch instance is an instance of lazy switch.

If $\rho = \mathsf{u}_2\downarrow$, the only case to consider is the one where $T = 1$. In this case, we permute trivially by first applying the rule $\mathsf{u}_2\downarrow$ and then the switch.

If $\rho = \mathsf{dls}$, because the instance of ls below it is not an instance of deep switch, we have either $R = 1$ or $R = (R_1 \otimes R_2)$. We discard the case $R = 1$ by resorting to Proposition 2.23 and assuming that the rule $\mathsf{u_2{\downarrow}}$ is applied exhaustively in proof construction, and prove the latter case. We have two possibilities:

• If we have the situation on the left below, we can then replace the two instances of the switch rule with a single instance:

$$
\mathsf{ls}\ \cfrac{\mathsf{dls}\ \cfrac{S([U \mathbin{\bindnasrepma} R_1] \otimes R_2 \otimes T)}{S([(R_1 \otimes R_2) \mathbin{\bindnasrepma} U] \otimes T)}}{S[(R_1 \otimes R_2 \otimes T) \mathbin{\bindnasrepma} U]}
\qquad \rightsquigarrow \qquad
\mathsf{dls}\ \cfrac{S([U \mathbin{\bindnasrepma} R_1] \otimes R_2 \otimes T)}{S[(R_1 \otimes R_2 \otimes T) \mathbin{\bindnasrepma} U]} \quad ,
$$

where R_1 is not a copar. We obtain a proof Π' such that $\#\Pi' = (n-1, r')$ to which we can then apply the induction hypothesis, independent from the value of r'.

• If we have the situation on the left below, we can permute the non-deep instance of switch up in the derivation as follows:

$$
\mathsf{ls}\ \cfrac{\mathsf{dls}\ \cfrac{S([(R_1 \otimes R_2) \mathbin{\bindnasrepma} U_1] \otimes U_2 \otimes T)}{S([(R_1 \otimes R_2) \mathbin{\bindnasrepma} (U_1 \otimes U_2)] \otimes T)}}{S[(R_1 \otimes R_2 \otimes T) \mathbin{\bindnasrepma} (U_1 \otimes U_2)]}
\qquad \rightsquigarrow \qquad
\mathsf{dls}\ \cfrac{\mathsf{ls}\ \cfrac{S([(R_1 \otimes R_2 \otimes T) \mathbin{\bindnasrepma} U_1] \otimes U_2)}{S([(R_1 \otimes R_2 \otimes T) \mathbin{\bindnasrepma} U_1] \otimes U_2)}}{S[(R_1 \otimes R_2 \otimes T) \mathbin{\bindnasrepma} (U_1 \otimes U_2)]} \quad .
$$

We obtain a proof Π' such that $\#\Pi' = (n, r-1)$ to which we can then apply the induction hypothesis. $\qquad \square$

As Example 2.10 demonstrates, a constructive transformation that replaces all the instances of the switch rule that are not interaction switch rule instances requires an observation mechanism on the context of the switch rule instances to be removed. We use below a splitting theorem [18], which provides such a global view of the rule instances. This way, we show that MLSdli is complete for multiplicative linear logic by exploiting the relation between completeness and cut-elimination. We prove this result on MLSdli, because the other systems with interaction switch follow simpler versions of the same argument.

Lemma 2.26. *For any structures P, U and R, if $[P \mathbin{\bindnasrepma} U]$ has a proof in MLSdli, then there is a derivation* $\begin{array}{c} R \\ \| \, \mathsf{MLSdli} \\ {[(R \otimes P) \mathbin{\bindnasrepma} U]} \end{array}$.

Proof. Let Π be the proof of the structure $[P \mathbin{\bindnasrepma} U]$ in MLSdli. We associate with every structure Q a pair $\#Q = (n, r)$, where n is the cardinality of $\downarrow_Q$ and r is the cardinality of $\mathsf{occ}\, Q$. We prove by induction on $\#[P \mathbin{\bindnasrepma} U]$ by Proposition 2.19. We single out the bottom-most rule instance in Π such that for some structure Q and rule $\rho \in \{\, \mathsf{ai{\downarrow}}, \mathsf{dlis}, \mathsf{u_1{\downarrow}}, \mathsf{u_2{\downarrow}} \,\}$

$$
\rho\ \cfrac{\begin{array}{c} R \\ \| \, \mathsf{MLSdli} \\ Q \end{array}}{[(R \otimes P) \mathbin{\bindnasrepma} U]} \qquad .
$$

The base case is given with one of the following: (i) $P = \bot$ and $U = 1$; (ii) $P = 1$ and $U = \bot$; and (iii) $P = a$ and $U = \bar{a}$ or vice versa, which are proved by replacing the derivation above with the following derivations.

$$
\mathsf{dlis} \cfrac{\mathsf{ai\downarrow} \cfrac{\mathsf{u_2\downarrow} \cfrac{R}{(R \otimes 1)}}{(R \otimes [\bot \,\mathord{\otimes}\, 1])}}{[(R \otimes \bot) \,\mathord{\otimes}\, 1]}
\qquad
\mathsf{u_1\downarrow} \cfrac{\mathsf{u_2\downarrow} \cfrac{R}{(R \otimes 1)}}{[(R \otimes 1) \,\mathord{\otimes}\, \bot]}
\qquad
\mathsf{dlis} \cfrac{\mathsf{ai\downarrow} \cfrac{\mathsf{u_2\downarrow} \cfrac{R}{(R \otimes 1)}}{(R \otimes [a \,\mathord{\otimes}\, \bar{a}])}}{[(R \otimes a) \,\mathord{\otimes}\, \bar{a}]}
$$

For the inductive cases, let us reason on the position of the instance of the rule ρ in $[R \,\mathord{\otimes}\, P]$. There are the following possibilities:

(1) $\rho = \mathsf{ai\downarrow}$: There are the following cases:

    (a) If $\mathsf{ai\downarrow}$ is applied inside P

$$
\text{such that} \qquad
\mathsf{ai\downarrow} \cfrac{\begin{matrix} 1 \\ \| \\ [P' \,\mathord{\otimes}\, U] \end{matrix}}{[P \,\mathord{\otimes}\, U]} \quad, \quad \text{then} \quad
\mathsf{ai\downarrow} \cfrac{\begin{matrix} R \\ \Delta\| \\ [(R \otimes P') \,\mathord{\otimes}\, U] \end{matrix}}{[(R \otimes P) \,\mathord{\otimes}\, U]} \quad,
$$

    where Δ is delivered by the induction hypothesis.

    (b) The $\mathsf{ai\downarrow}$ is applied inside U: similar to the previous case.

    (c) If $P = [P' \,\mathord{\otimes}\, a]$, $U = [\bar{a} \,\mathord{\otimes}\, U']$, and

$$
\mathsf{ai\downarrow} \cfrac{\begin{matrix} 1 \\ \Pi'\| \\ [P' \,\mathord{\otimes}\, 1 \,\mathord{\otimes}\, U'] \end{matrix}}{[P' \,\mathord{\otimes}\, a \,\mathord{\otimes}\, \bar{a} \,\mathord{\otimes}\, U']} \quad, \quad \text{then} \quad
\mathsf{dlis} \cfrac{\mathsf{ai\downarrow} \cfrac{\begin{matrix} R \\ \Delta\| \\ [(R \otimes [P' \,\mathord{\otimes}\, 1]) \,\mathord{\otimes}\, U'] \end{matrix}}{[(R \otimes [P' \,\mathord{\otimes}\, a \,\mathord{\otimes}\, \bar{a}]) \,\mathord{\otimes}\, U']}}{[(R \otimes [P' \,\mathord{\otimes}\, a]) \,\mathord{\otimes}\, \bar{a} \,\mathord{\otimes}\, U']} \quad,
$$

    where Δ is obtained by applying the induction hypothesis to Π'.

(2) $\rho = \mathsf{u_1\downarrow}$: if the rule $\mathsf{u_1\downarrow}$ is applied inside P or U, then we have cases that are similar to the case (a) of $\rho = \mathsf{ai\downarrow}$. Otherwise either it is the case that $P \approx \bot$ or $U \approx \bot$. For these cases, we prove as follows.

- If

$$
\begin{array}{cc}
\begin{array}{c}
1 \\
\Pi' \Vert \\
U \\
\mathsf{u_1}{\downarrow}\,\dfrac{}{[\bot \,\rotatebox[origin=c]{180}{\&}\, U]}
\end{array}
&
\text{then}
&
\begin{array}{c}
\mathsf{u_2}{\downarrow}\,\dfrac{R}{(R \otimes 1)} \\
\mathsf{u_1}{\downarrow}\,\dfrac{}{(R \otimes [\bot \,\rotatebox[origin=c]{180}{\&}\, 1])} \\
\mathsf{dlis}\,\dfrac{}{[(R \otimes \bot) \,\rotatebox[origin=c]{180}{\&}\, 1]} \\
\Pi' \Vert \\
{[(R \otimes \bot) \,\rotatebox[origin=c]{180}{\&}\, U]}
\end{array}
&
;
\end{array}
$$

- and if

$$
\begin{array}{cc}
\begin{array}{c}
1 \\
\Pi' \Vert \\
P \\
\mathsf{u_1}{\downarrow}\,\dfrac{}{[P \,\rotatebox[origin=c]{180}{\&}\, \bot]}
\end{array}
&
\text{then}
&
\begin{array}{c}
\mathsf{u_1}{\downarrow}\,\dfrac{R}{(R \otimes 1)} \\
\Pi' \Vert \\
(R \otimes P) \\
\mathsf{u_1}{\downarrow}\,\dfrac{}{[(R \otimes P) \,\rotatebox[origin=c]{180}{\&}\, \bot]}
\end{array}
&
.
\end{array}
$$

(3) $\rho = \mathsf{u_2}{\downarrow}$: the only possible cases are those that are applied inside P or U, which are similar to the case (a) of $\rho = \mathsf{ai}{\downarrow}$. In this case, if n in $\#(n,r)$ does not decrease, r always decreases by 1, thus induction hypothesis can be applied.

(4) $\rho = \mathsf{dlis}$: If the rule is inside P or U, similar to the case (a) of $\rho = \mathsf{ai}{\downarrow}$. Otherwise there are the following cases:

(a) If $P = [P' \,\rotatebox[origin=c]{180}{\&}\, (Q \otimes T)]$, $U = [W \,\rotatebox[origin=c]{180}{\&}\, U']$, and

$$
\begin{array}{c}
1 \\
\Pi' \Vert \\
\mathsf{dlis}\,\dfrac{[P' \,\rotatebox[origin=c]{180}{\&}\, U' \,\rotatebox[origin=c]{180}{\&}\, ([Q \,\rotatebox[origin=c]{180}{\&}\, W] \otimes T)]}{[P' \,\rotatebox[origin=c]{180}{\&}\, U' \,\rotatebox[origin=c]{180}{\&}\, (Q \otimes T) \,\rotatebox[origin=c]{180}{\&}\, W]}
\end{array}
$$

then we construct the derivation

$$
\begin{array}{c}
R \\
\Delta \Vert \\
\mathsf{dlis}\,\dfrac{[(R \otimes [P' \,\rotatebox[origin=c]{180}{\&}\, ([Q \,\rotatebox[origin=c]{180}{\&}\, W] \otimes T)]) \,\rotatebox[origin=c]{180}{\&}\, U']}{[(R \otimes [P' \,\rotatebox[origin=c]{180}{\&}\, (Q \otimes T) \,\rotatebox[origin=c]{180}{\&}\, W]) \,\rotatebox[origin=c]{180}{\&}\, U']} \\
\mathsf{dlis}\,\dfrac{}{[(R \otimes [P' \,\rotatebox[origin=c]{180}{\&}\, (Q \otimes T)]) \,\rotatebox[origin=c]{180}{\&}\, W \,\rotatebox[origin=c]{180}{\&}\, U']}
\end{array}
\,,
$$

where Δ is obtained by applying the induction hypothesis to Π'.

(b) The case where $P = [P' \,\rotatebox[origin=c]{180}{\&}\, W]$, $U = [(Q \otimes T) \,\rotatebox[origin=c]{180}{\&}\, U']$, and the rule dlis is applied as in case (a) is similar. $\qquad\square$

Proposition 2.27. *In* MLSdli, *a structure* $(R \otimes T)$ *has a proof if and only if* R *and* T *have proofs.*

Proof. If direction being trivial, for the only if direction construct the proofs of R and T by induction on the length of the proof of $(R \otimes T)$. □

In the theorem below, we use Lemma 2.26 together with an argument that takes into consideration the role of the units 1 and $\bot$ in MLSdli proofs. The aim of this theorem is enabling the breaking of a formula into smaller pieces, while preserving provability. This way, the obtained pieces can be used to build a constructive induction argument for proving properties of MLSdli such as completeness and cut-elimination.

Theorem 2.28 (Shallow Splitting for MLSdli). *For all structures R, T and P, if $[(R \otimes T) \mathbin{\text{⅋}} P]$ is provable in* MLSdli *then*

(i) *either* $[R \mathbin{\text{⅋}} P]$ *and T are provable in* MLSdli;

(ii) *or R and $[P \mathbin{\text{⅋}} T]$ are provable in* MLSdli;

(iii) *or there exist P_1, P_2 and* $\begin{array}{c} [P_1 \mathbin{\text{⅋}} P_2] \\ \| \text{MLSdli} \\ P \end{array}$ *such that $[R \mathbin{\text{⅋}} P_1]$ and $[T \mathbin{\text{⅋}} P_2]$ are provable in*

MLSdli.

Proof. We associate with every structure Q a pair $\#Q = (n, r)$, where n is the cardinality of $\downarrow_Q$ and r is the cardinality of $\mathsf{occ}\, Q$. We prove by induction on $\#[(R \otimes T) \mathbin{\text{⅋}} P]$ by Proposition 2.19. The base case is given with the structures $[(1 \otimes 1) \mathbin{\text{⅋}} \bot]$ and $[(\bot \otimes 1) \mathbin{\text{⅋}} 1]$, which are trivially covered by the cases (i) or (ii).

For the inductive cases, we single out the bottom most rule application ρ in the proof of $[(R \otimes T) \mathbin{\text{⅋}} P]$. We reason on the position where ρ is applied in $[(R \otimes T) \mathbin{\text{⅋}} P]$. There are the following possibilities:

(1) $\rho = \mathsf{ai}{\downarrow}$: There are the following cases:

(a) If $\mathsf{ai}{\downarrow}$ is applied inside R such that

$$
\mathsf{ai}{\downarrow} \frac{\begin{array}{c} 1 \\ \| \\ [(R' \otimes T) \mathbin{\text{⅋}} P] \end{array}}{[(R \otimes T) \mathbin{\text{⅋}} P]} \quad \text{then} \quad \mathsf{ai}{\downarrow} \frac{\begin{array}{c} 1 \\ \Pi_1 \| \\ [R' \mathbin{\text{⅋}} P] \end{array}}{[R \mathbin{\text{⅋}} P]} \;, \quad \mathsf{ai}{\downarrow} \frac{\begin{array}{c} 1 \\ \Pi_2 \| \\ R' \end{array}}{R} \quad \text{and} \quad \mathsf{ai}{\downarrow} \frac{\begin{array}{c} 1 \\ \Pi_3 \| \\ [R' \mathbin{\text{⅋}} P_1] \end{array}}{[R \mathbin{\text{⅋}} P_1]} \;,
$$

where Π_1, Π_2 and Π_3, respectively, are delivered by induction hypothesis for the cases (i), (ii) and (iii).

(b) If the rule $\mathsf{ai}{\downarrow}$ is applied inside T, we have a case similar to the previous one.

(c) If the rule $\mathsf{ai}{\downarrow}$ is applied inside P, such that

$$
\mathsf{ai}\downarrow \frac{\overset{1}{\overset{\|}{[(R \otimes T) \mathbin{⅋} P']}}}{[(R \otimes T) \mathbin{⅋} P]}
\qquad \text{then} \qquad
\mathsf{ai}\downarrow \frac{\overset{1}{\overset{\Pi_1\|}{[R \mathbin{⅋} P']}}}{[R \mathbin{⅋} P]}
\qquad \text{and} \qquad
\mathsf{ai}\downarrow \frac{\overset{1}{\overset{\Pi_2\|}{[P' \mathbin{⅋} T]}}}{[P \mathbin{⅋} T]} \quad ,
$$

for (i) and (ii), where Π_1 and Π_2 are delivered by induction hypothesis. For (iii), we have

$$
\mathsf{ai}\downarrow \frac{\overset{[P_1 \mathbin{⅋} P_2]}{\overset{\Delta\|}{P'}}}{P}
$$

where Δ is delivered by induction hypothesis.

(2) $\rho = \mathsf{u}_1\!\downarrow$: if the rule is applied inside R, T or P, the proof is similar to the cases for $\rho = \mathsf{ai}\!\downarrow$. Otherwise, it must be that $P = \bot$ such that

$$
\mathsf{u}_1\downarrow \frac{\overset{1}{\overset{\|}{(R \otimes T)}}}{[(R \otimes T) \mathbin{⅋} \bot]}
\qquad \text{then} \qquad
\mathsf{u}_1\downarrow \frac{\overset{1}{\overset{\|}{R}}}{[R \mathbin{⅋} \bot]}
$$

as R and T must have proofs by Proposition 2.27.

(3) $\rho = \mathsf{u}_2\!\downarrow$: if the rule is applied inside R, T or P, the proof is similar to the cases for $\rho = \mathsf{ai}\!\downarrow$. Otherwise it must be that $T = 1$ such that

$$
\mathsf{u}_2\downarrow \frac{\overset{1}{\overset{\Pi\|}{[R \mathbin{⅋} P]}}}{[(R \otimes 1) \mathbin{⅋} P]} \quad .
$$

Then $[R \mathbin{⅋} P]$ has the proof Π.

(4) $\rho = \mathsf{dlis}$: if the rule is applied inside R, T or P, the proof is similar to the cases for $\rho = \mathsf{ai}\!\downarrow$. Otherwise, there are the following possibilities: (Note that we dismiss the possibility that $T \approx (T' \otimes T'')$ as this would not introduce a new case for the instances of the rule dlis.)

   (a) $R \approx (R' \otimes R'')$ and $P \approx [P' \mathbin{⅋} P'']$ such that the bottom most rule instance is of the following form:

$$\mathsf{dlis}\,\frac{[([R'\,\Bbb{P}\, P']\otimes R''\otimes T)\,\Bbb{P}\, P'']}{[(R'\otimes R''\otimes T)\,\Bbb{P}\, P'\,\Bbb{P}\, P'']}\quad,$$

where the structure R' is not a copar and P' is not a par: we apply the induction hypothesis to the premise, by Proposition 2.19, and get

(i) either

$$
\begin{array}{ccc}
1 & & 1 \\
\Pi_1\,\| & \text{and} & \Pi_2\,\| \\
{[R'\,\Bbb{P}\, P'\,\Bbb{P}\, P'']} & & (R''\otimes T)
\end{array}\quad;
$$

(ii) or

$$
\begin{array}{ccc}
1 & & 1 \\
\Pi_3\,\| & \text{and} & \Pi_4\,\| \\
{[(R''\otimes T)\,\Bbb{P}\, P'']} & & {[R'\,\Bbb{P}\, P']}
\end{array}\quad;
$$

(iii) or

$$
\begin{array}{ccccc}
{[P_1'\,\Bbb{P}\, P_2']} & & 1 & & 1 \\
\Delta_1\,\| & , & \Pi_5\,\| & \text{and} & \Pi_6\,\| \\
P'' & & {[R'\,\Bbb{P}\, P'\,\Bbb{P}\, P_1']} & & {[(R''\otimes T)\,\Bbb{P}\, P_2']}
\end{array}\quad.
$$

- If (i) is the case, we have all the required derivations to complete the proof. By Proposition 2.27, from Π_2 we can construct the proof Π' of R'' and the proof Π'' of T. With Π' and Π_1, we then construct

$$
\begin{array}{c}
1 \\
\Pi_1\,\| \\
{[R'\,\Bbb{P}\, P'\,\Bbb{P}\, P'']} \\
\mathsf{u_2}{\downarrow}\,\dfrac{}{[(R'\otimes 1)\,\Bbb{P}\, P'\,\Bbb{P}\, P'']} \\
\Pi'\,\| \\
{[(R'\otimes R'')\,\Bbb{P}\, P'\,\Bbb{P}\, P'']}
\end{array}\quad.
$$

- If (ii) is the case, we can apply the induction hypothesis to Π_3, because we have that $|\,\downarrow_{[(R'\otimes R''\otimes T)\,\Bbb{P}\, P'\,\Bbb{P}\, P'']}\,| > |\,\downarrow_{[(R''\otimes T)\,\Bbb{P}\, P'']}\,|$.

  This way, from Π_3 we obtain, by Proposition 2.19, the following three cases:

  (ii.i) We have the proof Π_7 of $[R''\,\Bbb{P}\, P'']$ and the proof Π_8 of T: together with Π_8, we construct

$$\begin{array}{c} 1 \\ \Pi_7 \| \\ \mathsf{u}_2 \downarrow \dfrac{[R'' \mathbin{\rotatebox[origin=c]{180}{\&}} P'']}{[(R'' \otimes 1) \mathbin{\rotatebox[origin=c]{180}{\&}} P'']} \\ \Delta' \| \\ [(R' \otimes R'') \mathbin{\rotatebox[origin=c]{180}{\&}} P' \mathbin{\rotatebox[origin=c]{180}{\&}} P''] \end{array} \qquad .$$

where Δ' is delivered by Lemma 2.26 with proof Π_4.

(ii.ii) We have the proof Π_9 of R'' and the proof Π_{10} of $[T \mathbin{\rotatebox[origin=c]{180}{\&}} P'']$: together with Π_{10}, we then construct

$$\begin{array}{c} 1 \\ \Pi_4 \| \\ \mathsf{u}_2 \downarrow \dfrac{[R' \mathbin{\rotatebox[origin=c]{180}{\&}} P']}{[(R' \otimes 1) \mathbin{\rotatebox[origin=c]{180}{\&}} P']} \\ \Pi_9 \| \\ [(R' \otimes R'') \mathbin{\rotatebox[origin=c]{180}{\&}} P'] \end{array} \qquad .$$

(ii.iii) We have:

$$\begin{array}{ccc} \begin{array}{c} [P_1'' \mathbin{\rotatebox[origin=c]{180}{\&}} P_2''] \\ \Delta_2 \| \\ P'' \end{array} \qquad , & \qquad \begin{array}{c} 1 \\ \Pi_{11} \| \\ [R'' \mathbin{\rotatebox[origin=c]{180}{\&}} P_1''] \end{array} \quad \text{and} \quad & \begin{array}{c} 1 \\ \Pi_{12} \| \\ [T \mathbin{\rotatebox[origin=c]{180}{\&}} P_2''] \end{array} \qquad . \end{array}$$

We then construct the derivation and the proofs

$$\begin{array}{ccc} \begin{array}{c} [P' \mathbin{\rotatebox[origin=c]{180}{\&}} P_1'' \mathbin{\rotatebox[origin=c]{180}{\&}} P_2''] \\ \Delta_2 \| \\ [P' \mathbin{\rotatebox[origin=c]{180}{\&}} P''] \end{array} \quad , & \quad \begin{array}{c} 1 \\ \Pi_4 \| \\ \mathsf{u}_2 \downarrow \dfrac{[R' \mathbin{\rotatebox[origin=c]{180}{\&}} P']}{[(R' \otimes 1) \mathbin{\rotatebox[origin=c]{180}{\&}} P']} \\ \Delta' \| \\ [(R' \otimes R'') \mathbin{\rotatebox[origin=c]{180}{\&}} P' \mathbin{\rotatebox[origin=c]{180}{\&}} P_1''] \end{array} \quad \text{and} \quad & \begin{array}{c} 1 \\ \Pi_{12} \| \\ [T \mathbin{\rotatebox[origin=c]{180}{\&}} P_2''] \end{array} \quad , \end{array}$$

where Δ' is delivered by Lemma 2.26 with proof Π_{11}.

- If (iii) is the case, we can apply the induction hypothesis to Π_6, because we have that $|\downarrow_{[(R' \otimes R'' \otimes T) \mathbin{\rotatebox[origin=c]{180}{\&}} P' \mathbin{\rotatebox[origin=c]{180}{\&}} P'']}| > |\downarrow_{[(R'' \otimes T) \mathbin{\rotatebox[origin=c]{180}{\&}} P_2']}|$.

  This way, from Π_6 we obtain, by Proposition 2.19, the following three cases:

(iii.i) We have the proof Π'_7 of $[R'' \mathbin{\bindnasrepma} P'_2]$ and the proof Π'_8 of T: we then construct the derivation and the proofs

$$
\begin{array}{c}
[P' \mathbin{\bindnasrepma} P'_1 \mathbin{\bindnasrepma} P'_2] \\
\Delta_1 \| \\
[P' \mathbin{\bindnasrepma} P'']
\end{array}
\qquad , \qquad
\begin{array}{c}
1 \\
\Pi_5 \| \\
\mathsf{u_2}{\downarrow}\dfrac{[R' \mathbin{\bindnasrepma} P' \mathbin{\bindnasrepma} P'_1]}{[(R' \otimes 1) \mathbin{\bindnasrepma} P' \mathbin{\bindnasrepma} P'_1]} \\
\Delta' \| \\
[(R' \otimes R'') \mathbin{\bindnasrepma} P' \mathbin{\bindnasrepma} P'_1 \mathbin{\bindnasrepma} P'_2]
\end{array}
\qquad \text{and} \qquad
\begin{array}{c}
1 \\
\Pi'_8 \| \\
T
\end{array} \quad ,
$$

where Δ' is delivered by Lemma 2.26 with proof Π'_7.

(iii.ii) We have the proof Π'_9 of R'' and the proof Π'_{10} of $[T \mathbin{\bindnasrepma} P'_2]$: We then construct the derivation and the proofs

$$
\begin{array}{c}
[P' \mathbin{\bindnasrepma} P'_1 \mathbin{\bindnasrepma} P'_2] \\
\Delta_1 \| \\
[P' \mathbin{\bindnasrepma} P'']
\end{array}
\qquad , \qquad
\begin{array}{c}
1 \\
\Pi_5 \| \\
\mathsf{u_2}{\downarrow}\dfrac{[R' \mathbin{\bindnasrepma} P' \mathbin{\bindnasrepma} P'_1]}{[(R' \otimes 1) \mathbin{\bindnasrepma} P' \mathbin{\bindnasrepma} P'_1]} \\
\Pi'_9 \| \\
[(R' \otimes R'') \mathbin{\bindnasrepma} P' \mathbin{\bindnasrepma} P'_1]
\end{array}
\qquad \text{and} \qquad
\begin{array}{c}
1 \\
\Pi'_{10} \| \\
[T \mathbin{\bindnasrepma} P'_2]
\end{array} \quad .
$$

(iii.iii) We have:

$$
\begin{array}{c}
[P''_1 \mathbin{\bindnasrepma} P''_2] \\
\Delta'_2 \| \\
P'_2
\end{array}
\qquad , \qquad
\begin{array}{c}
1 \\
\Pi'_{11} \| \\
[R'' \mathbin{\bindnasrepma} P''_1]
\end{array}
\qquad \text{and} \qquad
\begin{array}{c}
1 \\
\Pi'_{12} \| \\
[T \mathbin{\bindnasrepma} P''_2]
\end{array} \quad .
$$

We construct the derivation and the proofs

$$
\begin{array}{c}
[P' \mathbin{\bindnasrepma} P'_1 \mathbin{\bindnasrepma} P''_1 \mathbin{\bindnasrepma} P''_2] \\
\Delta'_2 \| \\
[P' \mathbin{\bindnasrepma} P'_1 \mathbin{\bindnasrepma} P'_2] \\
\Delta_1 \| \\
[P' \mathbin{\bindnasrepma} P'']
\end{array}
\qquad , \qquad
\begin{array}{c}
1 \\
\Pi_5 \| \\
\mathsf{u_2}{\downarrow}\dfrac{[R' \mathbin{\bindnasrepma} P' \mathbin{\bindnasrepma} P'_1]}{[(R' \otimes 1) \mathbin{\bindnasrepma} P' \mathbin{\bindnasrepma} P'_1]} \\
\Delta' \| \\
[(R' \otimes R'') \mathbin{\bindnasrepma} P' \mathbin{\bindnasrepma} P'_1 \mathbin{\bindnasrepma} P''_1]
\end{array}
\qquad \text{and} \qquad
\begin{array}{c}
1 \\
\Pi'_{12} \| \\
[T \mathbin{\bindnasrepma} P''_2]
\end{array} \quad ,
$$

where Δ' is delivered by Lemma 2.26 with proof Π'_{11}.

(b) $P \approx [P' \mathbin{\bindnasrepma} P'']$ such that the bottom most rule instance is given with:

$$\mathsf{dlis}\, \frac{[([R \mathbin{\bindnasrepma} P'] \otimes T) \mathbin{\bindnasrepma} P'']}{[(R \otimes T) \mathbin{\bindnasrepma} P' \mathbin{\bindnasrepma} P'']} \quad .$$

We prove as in case (a) where we replace R' with R and R'' with 1.

(c) $R \approx (R' \otimes R'')$ such that the bottom-most rule instance is given with:

$$\mathsf{dlis}\, \frac{([R' \mathbin{\bindnasrepma} P] \otimes R'' \otimes T)}{[(R' \otimes R'' \otimes T) \mathbin{\bindnasrepma} P]} \quad .$$

By Proposition 2.27, because the structure $([R' \mathbin{\bindnasrepma} P] \otimes R'' \otimes T)$ has a proof, we have that the structures $[R' \mathbin{\bindnasrepma} P]$, R'' and T have proofs, with which we can easily construct a proof of $[(R' \otimes R'') \mathbin{\bindnasrepma} P]$.

(d) The bottom-most rule instance is given with:

$$\mathsf{dlis}\, \frac{([R \mathbin{\bindnasrepma} P] \otimes T)}{[(R \otimes T) \mathbin{\bindnasrepma} P]} \quad .$$

By Proposition 2.27, we have that $[R \mathbin{\bindnasrepma} P]$ and T have proofs.

(e) $P \approx [(P' \otimes P'') \mathbin{\bindnasrepma} U]$ such that the bottom-most rule instance is

$$\mathsf{dlis}\, \frac{[([(R \otimes T) \mathbin{\bindnasrepma} P'] \otimes P'') \mathbin{\bindnasrepma} U]}{[(R \otimes T) \mathbin{\bindnasrepma} (P' \otimes P'') \mathbin{\bindnasrepma} U]} \quad .$$

We apply the induction hypothesis, by Proposition 2.19, and get

(i) either

$$\begin{array}{ccc} 1 & & 1 \\ \Pi_1 \| & \text{and} & \Pi_2 \| \quad ; \\ [(R \otimes T) \mathbin{\bindnasrepma} P' \mathbin{\bindnasrepma} U] & & P'' \end{array}$$

(ii) or

$$\begin{array}{ccc} 1 & & 1 \\ \Pi_3 \| & \text{and} & \Pi_4 \| \quad ; \\ [(R \otimes T) \mathbin{\bindnasrepma} P'] & & [P'' \mathbin{\bindnasrepma} U] \end{array}$$

(iii) or

$$\begin{array}{cccc} [U_1 \mathbin{\bindnasrepma} U_2] & & 1 & 1 \\ \Delta_1 \| \quad, & & \Pi_5 \| & \text{and} \quad \Pi_6 \| \quad . \\ U & & [(R \otimes T) \mathbin{\bindnasrepma} P' \mathbin{\bindnasrepma} U_1] & [P'' \mathbin{\bindnasrepma} U_2] \end{array}$$

Other cases being similar, we consider (iii): by Proposition 2.19, given

$$| \downarrow_{[(R \otimes T) \mathbin{\bindnasrepma} (P' \otimes P'') \mathbin{\bindnasrepma} U]} | > | \downarrow_{[(R \otimes T) \mathbin{\bindnasrepma} P' \mathbin{\bindnasrepma} U_1]} |$$

we can apply the induction hypothesis on Π_5, and get

(iii.i) either

$$
\begin{array}{ccc}
1 & & 1 \\
\Pi_7 \| & \text{and} & \Pi_8 \| \\
[R \mathbin{\bindnasrepma} P' \mathbin{\bindnasrepma} U_1] & & T
\end{array}
$$

together with the derivation

$$
\begin{array}{c}
\mathsf{u}_2 \downarrow \dfrac{[P' \mathbin{\bindnasrepma} U_1]}{[(P' \otimes 1) \mathbin{\bindnasrepma} U_1]} \\
\Delta' \| \\
[(P' \otimes P'') \mathbin{\bindnasrepma} U_1 \mathbin{\bindnasrepma} U_2] \\
\Delta_1 \| \\
[(P' \otimes P'') \mathbin{\bindnasrepma} U]
\end{array} \quad ,
$$

where Δ' is delivered by Lemma 2.26 with proof Π_6;

(iii.ii) or

$$
\begin{array}{ccc}
1 & & 1 \\
\Pi_9 \| & \text{and} & \Pi_{10} \| \\
[T \mathbin{\bindnasrepma} P' \mathbin{\bindnasrepma} U_1] & & R
\end{array}
$$

together with the derivation in case $(iii.i.)$ above;

(iii.iii) or

$$
\begin{array}{ccccc}
[P_1 \mathbin{\bindnasrepma} P_2] & & 1 & & 1 \\
\Delta_2 \| & , & \Pi_{11} \| & \text{and} & \Pi_{12} \| \\
[P' \mathbin{\bindnasrepma} U_1] & & [R \mathbin{\bindnasrepma} P_1] & & [T \mathbin{\bindnasrepma} P_2]
\end{array} \quad .
$$

In this case, by using Δ_2, we can construct the derivation

$$[P_1 \otimes P_2]$$
$$\Delta_2 \|$$
$$\mathsf{u}_2{\downarrow} \frac{[P' \otimes U_1]}{[(P' \otimes 1) \otimes U_1]}$$
$$\Delta' \|$$
$$[(P' \otimes P'') \otimes U_1 \otimes U_2]$$
$$\Delta_1 \|$$
$$[(P' \otimes P'') \otimes U]$$

,

where Δ' is delivered by Lemma 2.26 with proof Π_6.

(f) $P \approx (P' \otimes P'')$ such that the bottom most rule instance is given with:

$$\mathsf{lis} \frac{([(R \otimes T) \otimes P'] \otimes P'')}{[(R \otimes T) \otimes (P' \otimes P'')]} \quad .$$

By Proposition 2.27, we have the proofs

$$\Pi_1 \| \frac{1}{P''} \qquad \text{and} \qquad \Pi_2 \| \frac{1}{[(R \otimes T) \otimes P']} \quad .$$

We apply the induction hypothesis to Π_2 and obtain

(i) either

$$\Pi_3 \| \frac{1}{[R \otimes P']} \qquad \text{and} \qquad \Pi_4 \| \frac{1}{T} \quad ;$$

(ii) or

$$\Pi_5 \| \frac{1}{[T \otimes P']} \qquad \text{and} \qquad \Pi_6 \| \frac{1}{R} \quad ;$$

(iii) or

$$\frac{[P_1' \otimes P_2']}{\Delta_1 \|} \qquad , \qquad \Pi_7 \| \frac{1}{[R \otimes P_1']} \qquad \text{and} \qquad \Pi_8 \| \frac{1}{[T \otimes P_2']} \quad ,$$
$$P'$$

which, together with Π_1, suffice to construct the required proofs and derivations as in the cases above. $\square$

Theorem 2.29 (Context Reduction for MLSdli)**.** *For all structures R and for all contexts $S\{\ \}$ if $S\{R\}$ is provable in* MLSdli *then*

(i) *either for all structures X there exist derivations*

$$
\begin{array}{ccc}
X & & 1 \\
\|\,\text{MLSdli} & and & \|\,\text{MLSdli} \;\; ; \\
S\{X\} & & R
\end{array}
$$

(ii) *or there exists a U such that for all structures X there exist derivations*

$$
\begin{array}{ccc}
[X \mathbin{\bindnasrepma} U] & & 1 \\
\|\,\text{MLSdli} & and & \|\,\text{MLSdli} \;\; . \\
S\{X\} & & [R \mathbin{\bindnasrepma} U]
\end{array}
$$

Proof. By induction on the size of $S\{\ \}$. The base case is given with empty context, which is covered by (i). There are two inductive cases as these give the only possibilities for $S\{\ \}$ with respect to its structure:

(1) $S\{\ \} \approx (S'\{\ \} \otimes P)$, for some P. From Proposition 2.27, it follows that there must be proofs of $S'\{R\}$ and P, thus we have

$$
\begin{array}{c}
S\{X\} \\
\Pi \| \\
(S'\{X\} \otimes P)
\end{array} \quad ,
$$

where Π is the proof of P. By applying the induction hypothesis on $S'\{R\}$, we can construct the desired derivations.

(2) $S\{\ \} \approx [S'\{\ \} \mathbin{\bindnasrepma} P]$, for some P such that $S'\{\ \}$ is not a par: if $S'\{\ \}$ is the empty context, that is, $S'\{\bot\} = \bot$, then the theorem is proved. Otherwise, it must be that, for some T, $S'\{\ \} \approx (S''\{\ \} \otimes T)$. By Theorem 2.28 there exist

(a) either

$$
\begin{array}{ccc}
1 & & 1 \\
\Pi_1 \| & and & \Pi_2 \| \;\; ; \\
[S''\{R\}, P] & & T
\end{array}
$$

(b) or

$$
\begin{array}{ccc}
1 & & 1 \\
\Pi_3 \| & and & \Pi_4 \| \;\; ; \\
[T, P] & & S''\{R\}
\end{array}
$$

(c) or

$$
\begin{array}{ccccc}
[P_1 \mathbin{\bindnasrepma} P_2] & & 1 & & 1 \\
\Delta \| & , & \Pi_5 \| & and & \Pi_6 \| \;\; . \\
P & & [S''\{R\} \mathbin{\bindnasrepma} P_1] & & [T \mathbin{\bindnasrepma} P_2]
\end{array}
$$

If (a) is the case, then we have

$$
\mathsf{u_2}\!\downarrow \frac{[S''\{X\} \mathbin{\bindnasrepma} P]}{[(S''\{X\} \otimes 1) \mathbin{\bindnasrepma} P]}
$$
$$
\Pi_2 \Big\|
$$
$$
[(S''\{X\} \otimes T) \mathbin{\bindnasrepma} P]
$$
,

which is proved by applying the induction hypothesis on proof Π_1.

If (b) is the case, then we have

$$
S''\{X\}
$$
$$
\Delta' \Big\|
$$
$$
[(S''\{X\} \otimes T) \mathbin{\bindnasrepma} P]
$$
,

where Δ' is the derivation delivered by Lemma 2.26 with proof Π_3. We prove by applying the induction hypothesis to the proof Π_4 of $S''\{R\}$.

If (c) is the case, then by applying the induction hypothesis to Π_5, we obtain a derivation Δ_1, and we can then construct the derivation

$$
[X \mathbin{\bindnasrepma} U]
$$
$$
\Delta_1 \Big\|
$$
$$
\mathsf{u_2}\!\downarrow \frac{[S''\{X\} \mathbin{\bindnasrepma} P_1]}{[(S''\{X\} \otimes 1) \mathbin{\bindnasrepma} P_1]}
$$
$$
\Delta'' \Big\|
$$
$$
[(S'\{X\} \otimes T) \mathbin{\bindnasrepma} P_1 \mathbin{\bindnasrepma} P_2]
$$
$$
\Delta \Big\|
$$
$$
[(S''\{X\} \otimes T) \mathbin{\bindnasrepma} P]
$$
$$
\text{and} \qquad
\begin{array}{c} 1 \\ \| \\ [R \mathbin{\bindnasrepma} U] \end{array}
$$
,

where Δ'' is the derivation delivered by Lemma 2.26 with proof Π_6. □

We are now ready to show that the systems MLS and MLSdli prove the same structures.

Theorem 2.30. *The systems* MLSdl *and* MLSdli *are equivalent.*

Proof. Observe that every proof in MLSdli is also a proof in MLSdl. For the other direction, single out the upper-most instance of the switch rule in the MLSdl proof, which is not an instance of the rule dlis:

$$
\begin{array}{c} 1 \\ \| \text{ MLSdli} \end{array}
$$
$$
\mathsf{dls} \frac{S([R \mathbin{\bindnasrepma} U] \otimes T)}{S[(R \otimes T) \mathbin{\bindnasrepma} U]}
$$

From Theorem 2.29, for any structure X, we have

(i) either

$$\begin{array}{c} X \\ \Delta \, \| \, \mathsf{MLSdli} \\ S\{X\} \end{array} \qquad \text{and} \qquad \begin{array}{c} 1 \\ \| \, \mathsf{MLSdli} \\ ([R \,\invamp\, U] \otimes T) \end{array} \quad ;$$

(ii) or

$$\begin{array}{c} [X \,\invamp\, V] \\ \Delta \, \| \, \mathsf{MLSdli} \\ S\{X\} \end{array} \qquad \text{such that} \qquad \begin{array}{c} 1 \\ \| \, \mathsf{MLSdli} \\ [([R \,\invamp\, U] \otimes T) \,\invamp\, V] \end{array} \quad .$$

We prove case (ii) as the proof of case (i) is similar. We apply Theorem 2.28.

(ii.i) We get either

$$\begin{array}{c} 1 \\ \Pi_1 \, \| \, \mathsf{MLSdli} \\ [R \,\invamp\, U \,\invamp\, V] \end{array} \qquad \text{and} \qquad \begin{array}{c} 1 \\ \Pi_2 \, \| \, \mathsf{MLSdli} \\ T \end{array} \quad ;$$

(ii.ii) or

$$\begin{array}{c} 1 \\ \Pi_3 \, \| \, \mathsf{MLSdli} \\ [R \,\invamp\, U] \end{array} \qquad \text{and} \qquad \begin{array}{c} 1 \\ \Pi_4 \, \| \, \mathsf{MLSdli} \\ [T \,\invamp\, V] \end{array} \quad ;$$

(ii.iii) or

$$\begin{array}{c} [K_1 \,\invamp\, K_2] \\ \Delta_1 \, \| \, \mathsf{MLSdli} \\ V \end{array} \quad , \qquad \begin{array}{c} 1 \\ \Pi_5 \, \| \, \mathsf{MLSdli} \\ [R \,\invamp\, U \,\invamp\, K_1] \end{array} \qquad \text{and} \qquad \begin{array}{c} 1 \\ \Pi_6 \, \| \, \mathsf{MLSdli} \\ [K_2 \,\invamp\, T] \end{array} \quad .$$

We can construct the following proofs for the cases (ii.i), (ii.ii) and (ii.iii), respectively, where Δ_2 is the derivation delivered by Lemma 2.26 with the proof Π_4, and Δ_3 is the derivation delivered by Lemma 2.26 with the proof Π_6.

$$
\begin{array}{c}
1 \\
\Pi_1 \big\| \; \mathsf{MLSdli} \\[4pt]
\mathsf{u_2}{\downarrow}\dfrac{[R \,\rotatebox[origin=c]{180}{\&}\, U \,\rotatebox[origin=c]{180}{\&}\, V]}{[(R \otimes 1) \,\rotatebox[origin=c]{180}{\&}\, U \,\rotatebox[origin=c]{180}{\&}\, V]} \\[10pt]
\Pi_2 \big\| \; \mathsf{MLSdli} \\[4pt]
[(R \otimes T) \,\rotatebox[origin=c]{180}{\&}\, U \,\rotatebox[origin=c]{180}{\&}\, V] \\[4pt]
\Delta \big\| \; \mathsf{MLSdli} \\[4pt]
S[(R \otimes T) \,\rotatebox[origin=c]{180}{\&}\, U]
\end{array}
\qquad
\begin{array}{c}
1 \\
\Pi_3 \big\| \; \mathsf{MLSdli} \\[4pt]
\mathsf{u_2}{\downarrow}\dfrac{[R \,\rotatebox[origin=c]{180}{\&}\, U]}{[(R \otimes 1) \,\rotatebox[origin=c]{180}{\&}\, U]} \\[10pt]
\Delta_2 \big\| \; \mathsf{MLSdli} \\[4pt]
[(R \otimes T) \,\rotatebox[origin=c]{180}{\&}\, U \,\rotatebox[origin=c]{180}{\&}\, V] \\[4pt]
\Delta \big\| \; \mathsf{MLSdli} \\[4pt]
S[(R \otimes T) \,\rotatebox[origin=c]{180}{\&}\, U]
\end{array}
\qquad
\begin{array}{c}
1 \\
\Pi_5 \big\| \; \mathsf{MLSdli} \\[4pt]
\mathsf{u_2}{\downarrow}\dfrac{[R \,\rotatebox[origin=c]{180}{\&}\, U \,\rotatebox[origin=c]{180}{\&}\, K_1]}{[(R \otimes 1) \,\rotatebox[origin=c]{180}{\&}\, U \,\rotatebox[origin=c]{180}{\&}\, K_1]} \\[10pt]
\Delta_3 \big\| \; \mathsf{MLSdli} \\[4pt]
[(R \otimes T) \,\rotatebox[origin=c]{180}{\&}\, U \,\rotatebox[origin=c]{180}{\&}\, K_1 \,\rotatebox[origin=c]{180}{\&}\, K_2] \\[4pt]
\Delta_1 \big\| \; \mathsf{MLSdli} \\[4pt]
[(R \otimes T) \,\rotatebox[origin=c]{180}{\&}\, U \,\rotatebox[origin=c]{180}{\&}\, V] \\[4pt]
\Delta \big\| \; \mathsf{MLSdli} \\[4pt]
S[(R \otimes T) \,\rotatebox[origin=c]{180}{\&}\, U]
\end{array}
$$

We repeat the procedure above inductively until all the instances of the rule dls, which are not instances of the rule dlis, are removed from the proof. □

Corollary 2.31. *The systems* MLS, MLSu, MLSl, MLSd, MLSdl *and* MLSdli *are equivalent.*

Proof. First apply Proposition 2.25, and then Theorem 2.30. □

Corollary 2.32. *The systems* MLSu, MLSl, MLSd, MLSi, MLSdl, MLSli, MLSdi *and* MLSdli *are equivalent.*

Proof. Given that every instance of the rule dlis is an instance of the rules dis, lis and is, the conditions (2) and (3) in Definition 2.4 can be removed from MLSdli without losing completeness. □

Remark 2.33. Lemma 2.26, Theorem 2.28, Theorem 2.29 and Theorem 2.30 can be proved by using the same argument also for the systems MLSi, MLSdi and MLSli.

2.3. **Interaction and Cut Elimination.** The restrictions that we have imposed on the switch rule, which is the rule responsible for context management, succeed in controlling the nondeterminism in proof search. In the following, we show that these restrictions do not sacrifice a clean proof theory, as they permit an independent cut elimination theorem, which does not rely on the unrestricted system MLS. Splitting theorem (Theorem 2.28) provides a general procedure for proving cut-elimination in many deep inference systems [18]. However, the restrictions that we impose on the switch rule require special care. In order to obtain a constructive proof of cut elimination for MLSdli, we need the following lemma.

Lemma 2.34. *For any contexts $S_1\{\ \}$ and $S_2\{\ \}$, if $S_1\{1\}$ and $S_2\{1\}$ are provable in* MLSdli *then $[S_1\{a\} \,\rotatebox[origin=c]{180}{\&}\, S_2\{\bar{a}\}]$ is provable in* MLSdli*.*

Proof. We prove with induction on the size of $\mathsf{S}_1\{1\}$. There are the following possibilities for the structure of $S_1\{\ \}$.

- If, for some structure P, $S_1\{\ \} \approx (\{\ \} \otimes P)$, by Proposition 2.27, we have that P must have a proof Π in MLSdli. We construct the following proof:

$$
\begin{array}{c}
1 \\
\| \\
\mathsf{ai}\!\downarrow \dfrac{S_2\{1\}}{S_2[a \,\bindnasrepma\, \bar{a}]} \\
\| \,\{\mathsf{dlis}\} \\
\mathsf{u}_2\!\downarrow \dfrac{[a \,\bindnasrepma\, S_2\{\bar{a}\}]}{[(a \otimes 1) \,\bindnasrepma\, S_2\{\bar{a}\}]} \\
\Pi \| \\
[(a \otimes P) \,\bindnasrepma\, S_2\{\bar{a}\}]
\end{array}
$$

- If, for some structure P and context S_1', $S_1\{\ \} \approx (P \otimes S_1'\{\ \})$, by Proposition 2.27, we have that P has a proof Π in MLSdli. We construct

$$
\begin{array}{c}
1 \\
\Pi' \| \\
\mathsf{u}_2\!\downarrow \dfrac{[S_1'\{a\} \,\bindnasrepma\, S_2\{\bar{a}\}]}{[(1 \otimes S_1'\{a\}) \,\bindnasrepma\, S_2\{\bar{a}\}]} \\
\Pi \| \\
[(P \otimes S_1'\{a\}) \,\bindnasrepma\, S_2\{\bar{a}\}]
\end{array}
$$

  where proof Π' is delivered by the induction hypothesis.

- If, for some structure P, $S_1\{\ \} \approx [\{\ \} \,\bindnasrepma\, P]$ then we can then construct the proof

$$
\begin{array}{c}
1 \\
\Pi_1 \| \\
[P \,\bindnasrepma\, 1] \\
\Pi_2 \| \\
\mathsf{ai}\!\downarrow \dfrac{[P \,\bindnasrepma\, S_2\{1\}]}{[P \,\bindnasrepma\, S_2[a \,\bindnasrepma\, \bar{a}]]} \\
\| \{\mathsf{dlis}\} \\
[a \,\bindnasrepma\, P \,\bindnasrepma\, S_2\{\bar{a}\}]
\end{array}
$$

  where proofs Π_1 and Π_2 are the proofs of $S_1\{1\}$ and $S_2\{1\}$.

- If $S_1\{\ \} \approx [P \,\bindnasrepma\, S_1'\{\ \}]$, for some structure P, we can assume that $S_1'\{\ \} \approx (Q \otimes S_1''\{\ \})$, because otherwise we have one of the previous cases. We apply Theorem 2.28 to the proof of $[P \,\bindnasrepma\, (Q \otimes S_1''\{1\})]$ and we obtain

(1) either

$$
\Pi_1 \Big\| \begin{array}{c} 1 \\ \\ [Q \parr P] \end{array} \qquad \text{and} \qquad \Pi_2 \Big\| \begin{array}{c} 1 \\ \\ S_1''\{1\} \end{array} \quad ;
$$

(2) or

$$
\Pi_3 \Big\| \begin{array}{c} 1 \\ \\ Q \end{array} \qquad \text{and} \qquad \Pi_4 \Big\| \begin{array}{c} 1 \\ \\ [S_1''\{1\} \parr P] \end{array} \quad ;
$$

(3) or

$$
\begin{array}{c} [P_1 \parr P_2] \\ \Delta \Big\| \\ P \end{array} \quad , \qquad \Pi_5 \Big\| \begin{array}{c} 1 \\ \\ [Q \parr P_1] \end{array} \qquad \text{and} \qquad \Pi_6 \Big\| \begin{array}{c} 1 \\ \\ [S_1''\{1\} \parr P_2] \end{array} \quad .
$$

We can then construct the following derivations for the cases (1) and (2)

$$
\Pi_7 \Big\| \begin{array}{c} 1 \\ \end{array}
$$
$$
\mathsf{u_2}{\downarrow} \frac{[S_1''\{a\} \parr S_2\{\bar{a}\}]}{[(S_1''\{a\} \otimes 1) \parr S_2\{\bar{a}\}]}
$$
$$
\Delta' \Big\| \\ [P \parr (Q \otimes S_1''\{a\}) \parr S_2\{\bar{a}\}]
$$

$$
\Pi_8 \Big\| \begin{array}{c} 1 \\ \end{array}
$$
$$
\mathsf{u_2}{\downarrow} \frac{[P \parr S_1''\{a\} \parr S_2\{\bar{a}\}]}{[P \parr (1 \otimes S_1''\{a\}) \parr S_2\{\bar{a}\}]}
$$
$$
\Pi_3 \Big\| \\ [P \parr (Q \otimes S_1''\{a\}) \parr S_2\{\bar{a}\}]
$$

where Δ' is the derivation delivered by Lemma 2.26 with proof Π_1 and the proofs Π_7 and Π_8 are delivered by the induction hypothesis. For the case (3), we can construct the following proof

$$
\Pi_9 \Big\| \begin{array}{c} 1 \\ \end{array}
$$
$$
\mathsf{u_2}{\downarrow} \frac{[P_2 \parr S_1''\{a\} \parr S_2\{\bar{a}\}]}{[P_2 \parr (1 \otimes S_1''\{a\}) \parr S_2\{\bar{a}\}]}
$$
$$
\Delta'' \Big\| \\ [P_1 \parr P_2 \parr (Q \otimes S_1''\{a\}) \parr S_2\{\bar{a}\}]
$$
$$
\Delta \Big\| \\ [P \parr (Q \otimes S_1''\{a\}) \parr S_2\{\bar{a}\}]
$$

where Δ'' is the derivation delivered by Lemma 2.26 with proof Π_5 and the proof Π_9 is delivered by the induction hypothesis. $\square$

Lemma 2.35. *Let $S[a \bindnasrepma a]$ be a structure such that atoms a and $\bar{a}$ do not occur in $S\{\ \}$. $S[a \bindnasrepma \bar{a}]$ has a proof in* MLSdli *if and only if $[a \bindnasrepma S\{\bar{a}\}]$ has a proof.*

Proof. For the proof of the if part, we have the derivation

$$
S[a \bindnasrepma \bar{a}] \\
\big\| \{\text{dlis}\} \quad . \\
[a \bindnasrepma S\{\bar{a}\}]
$$

For the only if part, we construct a proof of $S[a \bindnasrepma \bar{a}]$ from the proof Π of $[a \bindnasrepma S\{\bar{a}\}]$ by inductive case analysis, which results in removing the instances of the rule dlis. □

Theorem 2.36 (Cut Elimination). *The rule* ai↑ *is admissible for* MLSdli.

Proof. We single out the top-most instance of the rule ai↑.

$$
\begin{array}{c}
1 \\
\Pi_1 \big\| \text{MLSdli} \\
\text{ai↑} \dfrac{S[R \bindnasrepma (a \otimes \bar{a})]}{S\{R\}}
\end{array}
$$

We first apply Theorem 2.29 and then Theorem 2.28 and obtain

$$
\begin{array}{ccccc}
[S_1 \bindnasrepma S_2] & & 1 & & 1 \\
\Delta \big\| & , & \Pi_1 \big\| & \text{and} & \Pi_2 \big\| & . \\
S\{R\} & & [a \bindnasrepma S_1] & & [\bar{a} \bindnasrepma S_2]
\end{array}
$$

In order for a structure to have a proof in MLSdli, that structure must contain pairs of dual atoms. Because there are the proofs Π_1 and Π_2, it follows that we find $S_1'\{\ \}$ and $S_2'\{\ \}$, such that $S_1 = S_1'\{\bar{a}\}$ and $S_2 = S_2'\{a\}$, where we can uniquely mark a and $\bar{a}$. From Lemma 2.35, it follows that there are proofs

$$
\begin{array}{ccc}
1 & & 1 \\
\Pi_3 \big\| & \text{and} & \Pi_4 \big\| & . \\
S_1'\{1\} & & S_2'\{1\}
\end{array}
$$

We can then construct

$$
\begin{array}{c}
1 \\
\Pi_3 \big\| \\
[S_1'\{\bar{a}\} \bindnasrepma S_2'\{a\}] \\
\Delta \big\| \\
S\{R\}
\end{array}
$$

where Π_3 is delivered by Lemma 2.34 from the proofs Π_1 and Π_2. By repeating this procedure inductively, starting from the top, we remove one by one all the instances of the rule ai↑. □

Remark 2.37. For the case of the systems MLSi, MLSdi, MLSli and MLSdli, it is not possible to give a decomposition of proofs [36, 39], where the instances of the rule ai↓ are permuted above the instances of the switch rule as the structure in Example 2.8 demonstrates. In order to prove this structure, in bottom-up proof construction, we need an instance of rule ai↓ prior to any instance of the rule is. This is because the unit ⊥ requires its dual structure 1, with which it can interact. This can be delivered only by an instance of the rule ai↓, and only after this, an instance of is can be applied. However, in BV, which extends multiplicative linear logic with the rules mix and nullary mix [18, 24], such a decomposition result can be given, because the units can be omitted in BV, and this kind of behavior cannot emerge.

2.4. **Interaction and Proof Complexity.** We have shown that the conditions we impose on the switch rule provide a more immediate access to the shorter proofs that are made available by deep inference. We now show that these conditions do not introduce any additional complexity as they succeed in polynomially simulating proofs in other formalisms such as the sequent calculus or proof nets. For the case of multiplicative linear logic this is not surprising as this logic is known to be NP-complete [30, 31], thus in any reasonable formalism, providing polynomial time proofs of multiplicative linear logic formulae boils down to correctly guessing the right pairing of the atoms.

Definition 2.38. A system $\mathscr{S}$ polynomially simulates proof system $\mathscr{S}'$ if there is a polynomial p such that for every proof Π in $\mathscr{S}$ there is a proof Π' in $\mathscr{S}'$ of the same formula such that $|\Pi'| \leq p(|\Pi|)$.

Corollary 2.39. *The system* MLSdli *polynomially simulates any proof system for multiplicative linear logic.*

Proof. From Corollary 2.32 we have that system MLSdli is complete for multiplicative linear logic. From Proposition 2.20 it follows that the size of any proof in system MLSdli is bounded by $\mathcal{O}(|\mathsf{occ}\,R|^2)$. ∎

Remark 2.40. Corollary 2.39 sets a polynomial bound on the size of the MLSdli proofs, however it does not provide a constructive procedure for translating proofs in other formalisms. This can be achieved by using the existing proof as an oracle for the construction of the deep inference proof. For this, it suffices to label the formula such that this labeling results in pairwise distinctness of the atoms with respect to the pairing of the dual atoms at the axioms. On these structures, we can use the restriction of the rule dlis that we call cdlis such that

$$\mathsf{dlis}\ \frac{S([U \mathbin{⅋} R] \otimes T)}{S[U \mathbin{⅋} (R \otimes T)]}$$

is an instance of cdlis if $\mathsf{at}\,\overline{T} \cap \mathsf{at}\,U = \emptyset$. By applying the rules $\mathsf{u_1}{\downarrow}$, $\mathsf{u_2}{\downarrow}$, ai↓ and cdlis exhaustively on the labeled structure, we can construct the desired proof as all these rules preserve the relation ↓ between dual atoms. However, it is important to note that the rule cdlis is not complete for generic structures as illustrated by the structure $[a \mathbin{⅋} a \mathbin{⅋} (\bar{a} \otimes \bar{a})]$.

2.5. **Relation with Focusing.** Focusing as a technique was introduced by Andreoli [1, 2] to structure the sequent calculus proofs for controlling the nondeterminism in proof search. Focusing is based on the intuition that the sequent rules have certain permutative affinities, which can be exploited in proof search by separating the inference rules into groups of synchronous and asynchronous rules. Asynchronous rules, which are invertible, are then applied first whenever they are applicable, as they do not require back-tracking in proof search. By assigning polarities to the inference rules that manage different logical operators, inference rules of the same polarity are applied always in groups. In the context of multiplicative linear logic, this machinery has the implication that $\wp$ connective is mapped to meta-level sequents prior to handling of any $\otimes$ connective, and any atom with a positive polarity is paired immediately with a negative atom at an instance of an identity rule that closes the sequent calculus proof branch. As a result of this, proof construction is separated into alternating deterministic and non-deterministic phases.

Synthetic connectives [11] as a method, which operates within focusing technique, uses an idea similar to deep and lazy conditions in Definition 2.4, however within the sequent calculus setting. Given that focusing technique aims at processing connectives of the same polarity until only subformulae of the opposite polarities are left, this idea is based on considering n-ary connectives, whose immediate subformulae are necessarily of the opposite polarity. For example, in a synthetic tensor $A_1 \otimes ... \otimes A_n$, each A_i cannot itself be a tensor; it must be either a par, or an atom. The deep and lazy conditions of the rule dlis impose similar alternations, however within the deep inference setting. While focusing in the sequent calculus must process connectives at the top level, deep inference may operate at any depth. In particular, deep lazy switch can be applied at the highest possible depth, directly where the subformulae of opposite polarity are to be found.

Focusing technique has been adapted to the deep inference proofs in linear logic [12]. In the deep inference setting, because there is no distinction between meta-level and object level, the invertible inference rules that map the logical operators to sequents become redundant. Apart from this, in the focused deep inference system, a mechanism that treats the units explicitly by means of inference rules as in this paper is used. This makes it possible to remove the units 1 and $\perp$ whenever it is possible to apply the corresponding inference rules that are invertible. The rest of the nondeterminism is annotated with syntactic expressions that makes the considered synchronous connectives explicit, so that backtracking can be managed to enter another focused choice in case of a bad decision.

Remark 2.41. In Definition 2.4, the restrictions (2) and (3) implement a mechanism that is similar to the focusing technique. This is because the restriction (2) does not allow the structure U to be $\perp$ in an instance of dlis, and this leaves the rule $\mathsf{u}_1{\downarrow}$ as the only applicable rule to the $\perp$ structures that are connected with par. Similarly, the restriction (3) does not allow the structure R to be 1 in an instance of dlis, which leaves the rule $\mathsf{u}_2{\downarrow}$ as the only applicable rule to the 1 structures that are connected with copar. This introduces a mechanism that results in the removal of the units 1 and $\perp$ during proof construction whenever it is possible, since the rules $\mathsf{u}_1{\downarrow}$ and $\mathsf{u}_2{\downarrow}$, are the only applicable rules on these structures.

Besides the similarity described above and the similarity addressed in Remark 2.41, the focusing approach presented in [12] is orthogonal to the approach of this paper, thus focusing technique and our technique should be applicable in a complementary manner. It

$$\mathsf{ai}\!\downarrow\frac{S\{\mathsf{tt}\}}{S[a\vee\bar{a}]}\qquad\mathsf{s}\,\frac{S([U\vee R]\wedge T)}{S[U\vee(R\wedge T)]}\qquad\mathsf{w}\!\downarrow\frac{S\{\mathsf{ff}\}}{S\{R\}}\qquad\mathsf{c}\!\downarrow\frac{S[R\vee R]}{S\{R\}}$$

$$\mathsf{u}_1\!\downarrow\frac{S\{R\}}{S[R\vee\mathsf{ff}]}\qquad\mathsf{u}_2\!\downarrow\frac{S\{R\}}{S(R\wedge\mathsf{tt})}\qquad\mathsf{u}_3\!\downarrow\frac{S\{\mathsf{tt}\}}{S[\mathsf{tt}\vee\mathsf{tt}]}\qquad\mathsf{u}_4\!\downarrow\frac{S\{\mathsf{ff}\}}{S(\mathsf{ff}\wedge\mathsf{ff})}$$

Figure 7: Deep Inference System KSu

should therefore be possible to benefit from both approaches for obtaining the shorter proofs that are made available by deep inference.

3. Interaction and Depth in Classical Logic

Deep inference systems for different logics follow a common scheme where context management in proof construction is performed by the switch rule. In this section, we show that the ideas above can be carried over to classical logic, that is, the switch rule can be replaced with the deep lazy interaction switch rule without loosing completeness in a deep inference system [4] for classical logic.

Definition 3.1. There are countably many *atomic propositions* which are denoted by a, b, c,... Classical logic *structures* are generated by

$$R ::= \mathsf{ff} \mid \mathsf{tt} \mid a \mid [\,R \vee R\,] \mid (\,R \wedge R\,) \mid \bar{R}$$

where ff and tt are the units false and true, respectively. $[R \vee R]$ is a *disjunction* and $(R \wedge R)$ is a *conjunction*. $\bar{R}$ is the *negation* of the structure R. Classical logic structures are considered equivalent modulo relation $\approx$, which is the smallest congruence relation induced by the equational system consisting of the equations for associativity and commutativity for disjunction and conjunction and De Morgan equations for negation. While writing the classical logic structures we apply the same conventions as those for multiplicative linear logic structures: we denote the structures in the same equivalence class by picking a structure from the equivalence class. If there is no ambiguity, when writing the structures, we drop the superfluous brackets by resorting to the equations for associativity.

Remark 3.2. As for multiplicative linear logic structures, we consider classical logic structures in negation normal form by applying the De Morgan equations for negation to push the negation symbol to the atoms. Because the bottom-up instances of the inference rules, defined below, do not introduce new negation symbols, we consider the classical logic structures to be in negation normal form, and we remove the De Morgan equations from the relation $\approx$.

Definition 3.3. The system KSu for classical logic is the system depicted in Figure 7. The rules in the upper row, which we borrow from [4], are called *atomic interaction*, *switch*, *weakening* and *contraction*. The rules in the bottom row are the invertible rules corresponding to the equations for unit given in [4]. We call these rules unit one, unit two, unit three and unit four.

Definition 3.4. A *proof* Π in KSu is a finite derivation whose premise is the unit tt. The size $|\Pi|$ of a proof Π is the number of unit and atom occurrences appearing in it.

Definition 3.5. *System* KSu *with deep lazy interaction switch*, or KSdli is the system obtained by replacing the switch rule with the rule dlis as defined in Definition 2.4, however here copar is replaced with conjunction and par is replaced with disjunction. We define the other systems obtained by replacing in KSu the switch rule with one of the rules given in Figure 3 similarly by adding the prefix of the switch rule as a suffix to the name of the system. For example, the *system* KSu *with deep interaction switch*, or KSdi is the system that replaces the rule s in KSu with the rule dis.

In the following, we show that proofs in KSu can be constructed as proofs consisting of separate phases such that in each phase only certain inference rules are used. For this purpose, we use a semantic argument, similar to the one given in [4]. Due to the conjunctive normal forms that we obtain, this also provides a completeness argument for KSu. The proofs are given by first simulating the construction of the sequent calculus proofs, as in the left-most derivation below, and then obtaining their permutations that are available due to deep inference. The decomposed structure of the proofs obtained this way becomes instrumental in showing that KSdli is complete.

Theorem 3.6. *If a structure R has a proof in* KSu*, then there exist structures R_1, R_2, R_3, R_4, R_1', R_2', and R_3' and proofs of the following forms. In other words, every structure that has a proof in* KSu *has also proofs consisting of different phases with distinct inference rules as depicted below, where a proof phase can be empty, that is, there can be proofs with no instance of a particular rule.*

$$
\begin{array}{cccc}
\begin{array}{c}
\mathsf{tt} \\
\Delta_4 \,\big\|\, \{\mathsf{u_1\downarrow, u_2\downarrow}\} \\
R_4 \\
\Delta_3 \,\big\|\, \{\mathsf{w\downarrow}\} \\
R_3 \\
\Delta_2 \,\big\|\, \{\mathsf{ai\downarrow}\} \\
R_2 \\
\Delta_1 \,\big\|\, \{\mathsf{s, c\downarrow}\} \\
R
\end{array}
&
\begin{array}{c}
\mathsf{tt} \\
\Delta_4 \,\big\|\, \{\mathsf{u_1\downarrow, u_2\downarrow}\} \\
R_4 \\
\Delta_3 \,\big\|\, \{\mathsf{w\downarrow}\} \\
R_3 \\
\Delta_2 \,\big\|\, \{\mathsf{ai\downarrow}\} \\
R_2 \\
\Delta_{1,b} \,\big\|\, \{\mathsf{s}\} \\
R_1 \\
\Delta_{1,a} \,\big\|\, \{\mathsf{c\downarrow}\} \\
R
\end{array}
&
\begin{array}{c}
\mathsf{tt} \\
\Delta_4 \,\big\|\, \{\mathsf{u_1\downarrow, u_2\downarrow}\} \\
R_4 \\
\Delta_2 \,\big\|\, \{\mathsf{ai\downarrow}\} \\
R_3' \\
\Delta_3 \,\big\|\, \{\mathsf{w\downarrow}\} \\
R_2 \\
\Delta_{1,b} \,\big\|\, \{\mathsf{s}\} \\
R_1 \\
\Delta_{1,a} \,\big\|\, \{\mathsf{c\downarrow}\} \\
R
\end{array}
&
\begin{array}{c}
\mathsf{tt} \\
\Delta_4 \,\big\|\, \{\mathsf{u_1\downarrow, u_2\downarrow}\} \\
R_4 \\
\Delta_2 \,\big\|\, \{\mathsf{ai\downarrow}\} \\
R_3' \\
\Delta_{1,b}' \,\big\|\, \{\mathsf{s}\} \\
R_2' \\
\Delta_3' \,\big\|\, \{\mathsf{w\downarrow}\} \\
R_1' \\
\Delta_{1,a} \,\big\|\, \{\mathsf{c\downarrow}\} \\
R
\end{array}
\end{array}
$$

$$\overset{i.}{\rightsquigarrow} \qquad \overset{ii.}{\rightsquigarrow} \qquad \overset{iii.}{\rightsquigarrow}$$

Proof. We can derive the rule, that we call *distributive* (d), as follows:

$$
\cfrac{S([R \vee U] \wedge [T \vee U])}{\cfrac{S[([R \vee U] \wedge T) \vee U]}{\cfrac{S[(R \wedge T) \vee U \vee U]}{S[(R \wedge T) \vee U]}\, \mathsf{c\downarrow}}\, \mathsf{s}}\, \mathsf{s}
$$

By applying this rule exhaustively to the structure R bottom up, we obtain the derivation Δ_1 with the premise R_2, which is in conjunctive normal form. Because R_2 is provable, each disjunction in R_2 must have an atom a and its dual $\bar{a}$. By applying the rule $\mathsf{ai\downarrow}$ bottom up

to each one of these pairs of dual atoms, we obtain the derivation Δ_2 with the premise R_3, where each disjunction has an instance of the unit tt. By applying the rule $\mathsf{w}{\downarrow}$ exhaustively to all the remaining structures in each disjunction which are different from the unit tt, we obtain the derivation Δ_3. By applying the rules $\mathsf{u}_1{\downarrow}$ and $\mathsf{u}_2{\downarrow}$ exhaustively we obtain Δ_4.

(i) With structural induction on R, we obtain the derivations $\Delta_{1,a}$ and $\Delta_{1,b}$ from the derivation Δ_1. If R is an atom or the unit tt or ff, then it is already in conjunctive normal form. If $R = (T \wedge U)$ or $R = [T \vee U]$ then we have the derivations (1.) and (2.) below by induction hypothesis where T_2 and U_2 are in conjunctive normal form. Let n be the number of disjunctions in U_2. We assume that n is greater than one. Otherwise, we can exchange T_2 with U_2, or if in both T_2 and U_2, there are less than 2 disjunctions, then they are already in conjunctive normal form. We construct the derivations for $R = (T \wedge U)$ and $R = [T \vee U]$, respectively, as in (3.) and (4.) below:

(1.)   (2.)   (3.)   (4.)

$$
\begin{array}{cccc}
T_2 & U_2 & (T_2 \wedge U_2) & R_2 \\
\Delta'_T \big\|\{\mathsf{s}\} & \Delta'_U \big\|\{\mathsf{s}\} & [\Delta'_T,\Delta'_U]\big\|\{\mathsf{s}\} & \big\|\{\mathsf{s}\} \\
T_1 & U_1 & (T_1 \wedge U_1) & [T_2 \vee \ldots \vee T_2 \vee U_2] \\
\Delta_T \big\|\{\mathsf{c}\downarrow\} & \Delta_U \big\|\{\mathsf{c}\downarrow\} & [\Delta_T,\Delta_U]\big\|\{\mathsf{c}\downarrow\} & [\Delta'_T,...,\Delta'_T,\Delta'_U]\big\|\{\mathsf{s}\} \\
T & U & (T \wedge U) & [T_1 \vee \ldots \vee T_1 \vee U_1] \\
& & & [\Delta_T,...,\Delta_T,\Delta_U]\big\|\{\mathsf{c}\downarrow\} \\
& & & [T \vee \ldots \vee T,U] \\
& & & \big\|\{\mathsf{c}\downarrow\} \\
& & & [T \vee U]
\end{array}
$$

(ii) We trivially permute each instance of $\mathsf{w}{\downarrow}$ under the instances of $\mathsf{ai}{\downarrow}$.

(iii) We permute the instances of the rule s over the rule $\mathsf{w}{\downarrow}$: Other cases being trivial, we consider the following:

(a) The redex is of $\mathsf{w}{\downarrow}$ is inside the contractum of s.

$$
\mathsf{s}\cfrac{\mathsf{w}{\downarrow}\cfrac{S(\mathsf{ff} \wedge T)}{S([R \vee U] \wedge T)}}{S[(R \wedge T) \vee U]} \quad \xrightarrow{a.} \quad \mathsf{w}{\downarrow}\cfrac{\mathsf{w}{\downarrow}\cfrac{S(\mathsf{ff} \wedge T)}{S(R \wedge T)}}{S[(R \wedge T) \vee U]}
$$

(b) The contractum of s is inside the redex of $\mathsf{w}{\downarrow}$.

$$
\mathsf{s}\cfrac{\mathsf{w}{\downarrow}\cfrac{S([R \vee U] \wedge \mathsf{ff})}{S([R \vee U] \wedge T \wedge P)}}{S([(R \wedge T) \vee U] \wedge P)} \quad \xrightarrow{b.} \quad \mathsf{w}{\downarrow}\cfrac{\mathsf{w}{\downarrow}\cfrac{\mathsf{s}\cfrac{S([R \vee U] \wedge \mathsf{ff})}{S([(R \wedge \mathsf{ff}) \vee U] \wedge \mathsf{ff})}}{S([(R \wedge T) \vee U] \wedge \mathsf{ff})}}{S([(R \wedge T) \vee U] \wedge P)} \qquad \square
$$

The proof scheme given by the theorem above simulates the construction of the sequent calculus proofs, where in the worst case, an exponentially branching proof tree is generated. The transformation into conjunctive normal form provides a strategy for proof construction, and this results in an exponential cost for some classes of formulae. However, besides these proofs that simulate sequent calculus, deep inference provides proofs that are exponentially shorter than those that are available in the sequent calculus [9]. Below we use Theorem 3.6 as a tool for proving that KSdli is complete by exploiting the intermediate stages between proof phases.

Theorem 3.7. *The systems* KSu *and* KSdli *are equivalent.*

Proof. Every proof in KSdli is a proof in KSu. For the other direction, we have by Theorem 3.6 that if a structure R has a proof in KSu, it has proof of the form on the left below. In proof Π, we apply the map

$$\{\, \wedge \leftrightarrow \otimes \,,\; \vee \leftrightarrow \bindnasrepma \,,\; \mathtt{tt} \leftrightarrow 1 \,,\; \mathtt{ff} \leftrightarrow \bot \,\}$$

and we transform Π into the proof Π' on the right by applying Theorem 2.30.

$$
\begin{array}{ccc}
\begin{array}{c}
\mathtt{tt} \\
\Pi \big\| \{\, \mathsf{ai}{\downarrow}\,,\mathsf{s}\,,\mathsf{u}_1{\downarrow}\,,\mathsf{u}_2{\downarrow}\,\} \\
R_1 \\
\Delta \big\| \{\,\mathsf{c}{\downarrow}\,,\mathsf{w}{\downarrow}\,\} \\
R
\end{array}
&
\rightsquigarrow
&
\begin{array}{c}
\mathtt{tt} \\
\Pi' \big\| \{\, \mathsf{ai}{\downarrow}\,,\mathsf{dlis}\,,\mathsf{u}_1{\downarrow}\,,\mathsf{u}_2{\downarrow}\,\} \\
R_1 \\
\Delta \big\| \{\,\mathsf{c}{\downarrow}\,,\mathsf{w}{\downarrow}\,\} \\
R
\end{array}
\end{array}
\qquad \square
$$

Corollary 3.8. *The systems* KSu*,* KSd*,* KSl*,* KSi*,* KSdi*,* KSli*,* KSdl *and* KSdli *are equivalent.*

We have shown that KSdli is complete for classical logic. We conclude the discussion on classical logic by showing that KSdli polynomially simulates the sequent calculus and also provides exponentially shorter proofs for certain formulae.

Definition 3.9. The sequent-calculus proof system **Analytic Gentzen** is defined by the inference rules in Figure 8, where ϕ and ψ stand for multisets of formulae and the symbol ',' represents the multiset union. We interpret multisets of formulae as their disjunction. Derivations in **Analytic Gentzen**, denoted by Φ, are trees obtained by composing instances of these inference rules. The leaves of a derivation are its premises and the root is its conclusion. A derivation with no premises is a proof. The size $|\Phi|$ of a derivation Φ is the number of unit and atom occurrences appearing in it.

Notation 3.10. For convenience, we denote the structures and the formulae with the same notation. However, the congruence relation in Definition 3.1 applies only to structures, and they do not hold within the proofs in **Analytic Gentzen** system.

$$
\mathsf{id}\,\frac{}{\vdash A, \bar{A}} \qquad \mathtt{tt}\,\frac{}{\vdash \mathtt{tt}} \qquad \mathsf{w}\,\frac{\vdash \phi}{\vdash \phi, A} \qquad \mathsf{c}\,\frac{\vdash \phi, A, A}{\vdash \phi, A} \qquad \vee\,\frac{\vdash \phi, A, B}{\vdash \phi, [A \vee B]} \qquad \wedge\,\frac{\vdash \phi, A \quad \vdash B, \psi}{\vdash \phi, (A \wedge B), \psi}
$$

Figure 8: **Analytic Gentzen** system in the sequent calculus.

Theorem 3.11. *For every* **Analytic Gentzen** *proof Φ with conclusion P, there is a proof Π in* KSu *such that*

(i) *if n is the size of Φ, the size of Π is $\mathcal{O}(n^2)$;*
(ii) *and Π is of the following form:*

$$
\begin{array}{c}
\mathsf{tt} \\
\Pi \big\| \ \{\, \mathsf{ai}{\downarrow}, \mathsf{s}, \mathsf{u}_1{\downarrow}, \mathsf{u}_2{\downarrow} \,\} \\
R_1 \\
\Delta \big\| \ \{\, \mathsf{c}{\downarrow}, \mathsf{w}{\downarrow} \,\} \\
R
\end{array}
$$

Proof. (i) is as in the proof of a more a general result in [9], and it proceeds by induction on the tree structure of Φ. The base case is given with the translation of the rule id as in the proof of Theorem 1.19 and the translation of the instances of rule tt in Figure 8 to the instances of the rule $\mathsf{u}_2{\downarrow}$. The inductive cases are translated into derivations in KSu as in [9], where the rules $\mathsf{u}_1{\downarrow}$ and $\mathsf{u}_2{\downarrow}$ are explicitly applied when required, and the size of the resulting proof is given by the larger cases as $\mathcal{O}(n^2)$. For the proof of (ii), we permute up all the instances of the rules s, $\mathsf{ai}{\downarrow}$, $\mathsf{u}_1{\downarrow}$ and $\mathsf{u}_2{\downarrow}$ as in Theorem 3.6, whereby the size of the proof in (i) is preserved by these permutations. $\qquad\square$

Corollary 3.12. *The system* KSdli *polynomially simulates* **Analytic Gentzen** *system.*

Proof. We first apply Theorem 3.11 and then Corollary 2.39 as in the proof of Theorem 3.7. $\qquad\square$

Remark 3.13. In [9], Bruscoli and Guglielmi show that Statman tautologies have quadratic size proofs in the size of the proved tautologies in KSu, in contrast to their exponential size proofs in the sequent calculus. It is straight-forward to see that KSdli preserves these quadratic size proofs of Statman tautologies.

Remark 3.14. By replacing the rule $\mathsf{c}{\downarrow}$ and dlis in KSdli with the rule derived by combining these rules as in

$$
\mathsf{c}{\downarrow} \frac{\mathsf{dlis} \dfrac{S[([U \vee R] \wedge T) \vee U]}{S[U \vee U \vee (R \wedge T)]}}{S[U \vee (R \wedge T)]}
\qquad \rightsquigarrow \qquad
\mathsf{c}{\downarrow}; \mathsf{dlis} \frac{S[([U \vee R] \wedge T) \vee U]}{S[U \vee (R \wedge T)]}
$$

and integrating the rule $\mathsf{w}{\downarrow}$ to the rule $\mathsf{ai}{\downarrow}$ as in

$$
\mathsf{ai}{\downarrow} \frac{\mathsf{w}{\downarrow} \dfrac{\mathsf{u}_1{\downarrow} \dfrac{S\{\mathsf{tt}\}}{S[\mathsf{tt} \vee \mathsf{ff}]}}{S[\mathsf{tt} \vee R]}}{S[a \vee \bar{a} \vee R]}
\qquad \rightsquigarrow \qquad
\mathsf{ai}{\downarrow}; \mathsf{w}{\downarrow}; \mathsf{u}_1{\downarrow} \frac{S\{\mathsf{tt}\}}{S[a \vee \bar{a} \vee R]} \quad ,
$$

we obtain a complete deep inference system for classical logic, which consists of only invertible rules.

4. Discussion

We have introduced a technique on deep inference systems for reducing nondeterminism in proof search. The restrictions that we impose on the switch rule provide a reduction in non-determinism in proof search without sacrificing proof theoretic cleanliness as these restrictions do not break the cut elimination property. Our technique permits the construction of proofs with the capability of polynomially simulating shallow inference of the sequent calculus, while preserving shorter proofs that are available due to deep inference. For example, polynomial size proofs of Statman tautologies [9] can be built in KSdli in contrast to their exponential size sequent calculus proofs.

Our technique exploits an interaction pattern on the switch rule, which is the rule responsible for context management. This interaction pattern is realized by a condition given in Definition 2.4. In this respect, further refining the condition of the interaction switch rule without sacrificing proof theoretic cleanliness is a topic of future investigation, which is however difficult (see, for example, Remark 2.6).

The switch rule is the rule responsible for context management in deep inference systems for different fragments of linear logic [37, 35], modal logics [15, 34], intuitionistic logic [40], the logic BV [18] and its extension with the exponentials of linear logic, that is, the logic NEL [22, 23, 39]. We believe that the technique here generalizes to these other systems. For instance, in [28, 27], we have shown that in BV the switch rule can be replaced with the rule lis without losing completeness. However, in BV and its extensions, the non-trivial interactions between commutative and non-commutative contexts make it difficult to extend these ideas to non-commutative context management rule of this logic. Some preliminary ideas along these lines can be found in [27].

The key to extending our technique to the deep inference systems for other logics is provided by the splitting theorem (Theorem 2.28). Given that splitting theorems for linear logic and NEL, given in [36, 23], can be modified to corporate the restrictions that we have introduced on the switch rule, the technique introduced here can be carried over also to the setting of these logics. Another related topic that we leave for future work is extending the scope of the interaction pattern to the other inference rules of these other logics. For example, for the case of linear logic [37, 35] and modal logics [15, 34], the interaction pattern that we exploit in the rule dlis can be used also for the other rules of these logics. For example, when we consider the promotion rule of linear logic ($\mathsf{p}\downarrow$) and the rule $\mathsf{k}\downarrow$ of modal logic K below, we observe that these rules are subject to restrictions with respect to the interaction patterns between the structures R and T.

$$\mathsf{p}\downarrow \frac{S\{![R,T]\}}{S[!R,?T]} \qquad\qquad \mathsf{k}\downarrow \frac{S\{\Box[R,T]\}}{S[\Box R,\Diamond T]}$$

Introducing an interaction condition for the instances of these rules, that is, $\mathsf{at}\,\bar{R}\cap\mathsf{at}\,T \neq \emptyset$, should result in a reduced nondeterminism for proof search in these systems.

Our long term goal is developing analytic deep inference [8] theorem provers for different logics. Multiplicative linear logic provides an excellent theoretical framework for these investigations due to its complexity [30, 31] and simplicity [16] under the same hood. In this respect, the ideas presented in this paper in combination with the focusing technique should provide effective means to benefit from short proofs that deep inference makes available for different logics.

References

[1] Jean-Marc Andreoli. Logic programming with focusing proofs in linear logic. *J. Log. Comput.*, 2(3):297–347, 1992.

[2] Jean-Marc Andreoli. Focussing and proof construction. *Ann. Pure Appl. Logic*, 107(1-3):131–163, 2001.

[3] Kai Brünnler. Atomic cut elimination for classical logic. In M. Baaz and J. A. Makowsky, editors, *CSL 2003*, volume 2803 of *LNCS*, pages 86–97. Springer, 2003.

[4] Kai Brünnler. *Deep Inference and Symmetry in Classical Proofs*. PhD thesis, Technische Universität Dresden, 2003.

[5] Kai Brünnler. Deep inference and its normal form of derivations. In Arnold Beckmann, Ulrich Berger, Benedikt Löwe, and John V. Tucker, editors, *Computability in Europe 2006*, volume 3988 of *LNCS*, pages 65–74. Springer, July 2006.

[6] Kai Brünnler. Locality for classical logic. *Notre Dame Journal of Formal Logic*, 47(4):557–580, 2006.

[7] Paola Bruscoli. A purely logical account of sequentiality in proof search. In Peter J. Stuckey, editor, *Logic Programming, 18th International Conference*, volume 2401 of *Lecture Notes in Computer Science*, pages 302–316. Springer, 2002.

[8] Paola Bruscoli and Alessio Guglielmi. On analyticity in deep inference. Available on the web at `http://cs.bath.ac.uk/ag/p/ADI.pdf`, 2009.

[9] Paola Bruscoli and Alessio Guglielmi. On the proof complexity of deep inference. *ACM Transactions on Computational Logic*, 2(14):1–34, 2009.

[10] Iliano Cervesato, Joshua S. Hodas, and Frank Pfenning. Efficient resource management for linear logic proof search. *Theoretical Computer Science*, 232:133–163, 2000.

[11] Kaustuv Chaudhuri. Focusing strategies in the sequent calculus of synthetic connectives. In *Proceedings of LPAR'08, Logic for Programming, Artificial Intelligence, and Reasoning, 15th International Conference*, volume 5330 of *LNCS*, pages 467–481. Springer, 2008.

[12] Kaustuv Chaudhuri, Nicolas Guenot, and Lutz Straßburger. The focused calculus of structures. In Marc Bezem, editor, *Computer Science Logic (CSL'11) - 25th International Workshop/20th Annual Conference of the EACSL*, volume 12, pages 159–173. LIPICS, 2011.

[13] Anupam Das. On the proof complexity of cut-free bounded deep inference. In *Proceedings of Tableaux'11*, volume 6793 of *LNCS*, pages 134–148. Springer, 2011.

[14] Anupam Das. Complexity of deep inference via atomic flows. In *Proceedings of CiE'12*, volume 7318 of *LNCS*, pages 139–150. Springer, 2012.

[15] Rajeev Goré and Alwen Tiu. Classical modal display logic in the calculus of structures and minimal cut-free deep inference calculi for S5. *Journal of Logic and Computation*, 17(4):767–794, 2007.

[16] Stefano Guerrini. Correctness of multiplicative proof nets is linear. In *In Fourteenth Annual IEEE Symposium on Logic in Computer Science*, pages 454–463, 1999.

[17] Alessio Guglielmi. Mismatch. Available on the web at `http://cs.bath.ac.uk/ag/p/AG9.pdf`, 2003.

[18] Alessio Guglielmi. A system of interaction and structure. *ACM Transactions on Computational Logic*, 8(1):1–64, 2007.

[19] Alessio Guglielmi and Tom Gundersen. Normalisation control in deep inference via atomic flows. *Logical Methods in Computer Science*, 2008. In press.

[20] Alessio Guglielmi, Tom Gundersen, and Michel Parigot. A proof calculus which reduces syntactic bureaucracy. In *Proceedings of the International Conference on Rewriting Techniques and Applications 2010 (Edinburgh)*, pages 135–150. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik 2010 LIPIcs, 2010.

[21] Alessio Guglielmi, Tom Gundersen, and Lutz Straßburger. Breaking paths in atomic flows for classical logic. In *Proceedings of LICS'10*, pages 284–293. IEEE, 2010.

[22] Alessio Guglielmi and Lutz Straßburger. A non-commutative extension of MELL. In M. Baaz and A. Voronkov, editors, *LPAR 2002*, volume 2514 of *LNAI*, pages 231–246. Springer, 2002.

[23] Alessio Guglielmi and Lutz Strassburger. A system of interaction and structure V: The exponentials and splitting. *Mathematical Structures in Computer Science*, 21(3):563–584, 2010.

[24] Ozan Kahramanoğulları. System BV is NP-complete. *Annals of Pure and Applied Logic*, 152(1–3):107–121, 2008.

[25] Ozan Kahramanoğulları. On linear logic planning and concurrency. *Information and Computation*, 207(11):1229–1258, 2009.

[26] Ozan Kahramanoğulları. System BV without the equalities for unit. In *Proceedings of the 19th International Symposium on Computer and Information Sciences, ISCIS'04*, volume 3280 of *LNCS*, pages 986–995, Antalya, Turkey, 2004. Springer.

[27] Ozan Kahramanoğulları. *Nondeterminism and Language Design in Deep Inference*. PhD thesis, TU Dresden, 2006.

[28] Ozan Kahramanoğulları. Reducing nondeterminism in the calculus of structures. In Miki Hermann and Andrei Voronkov, editors, *Logic for Programming, Artificial Intelligence, and Reasoning, Proceedings of the 13th International Conference, LPAR 2006, Phnom Penh, Cambodia*, volume 4246 of *LNCS*, pages 272–286. Springer, 2006.

[29] Ozan Kahramanoğulları. Maude as a platform for designing and implementing deep inference systems. In *Proceedings of the Eighth International Workshop on Rule-Based Programming, RULE'07*, ENTCS. Elsevier, 2008. In press.

[30] Max Kanovich. The multiplicative fragment of linear logic is NP-complete. Technical Report X-91-13, Institute for Language, Logic, and Information, 1991.

[31] Patrick Lincoln and Timothy C. Winkler. Constant-only multiplicative linear logic is NP-complete. In *Theoretical Computer Science*, volume 135(1), pages 155–169. 1994.

[32] Narciso Martí-Oliet and Josè Meseguer. Rewriting logic as a logical and semantic framework. In J. Meseguer, editor, *Proc. 1st Internat Workshop on Rewriting Logic and its Application, WRLA' 96*, volume 4 of *Electronic Notes in Theoretical Computer Science*, pages 189–224. Elsevier, 1996.

[33] Richard Statman. Bounds for proof-search and speed-up in the predicate calculus. *Annals of Mathematical Logic*, 15:225–287, 1978.

[34] Phiniki Stouppa. A deep inference system for the modal logic S5. *Studia Logica*, 85(2):199–214, 2007.

[35] Lutz Straßburger. A local system for linear logic. In M. Baaz and A. Voronkov, editors, *LPAR 2002*, volume 2514 of *LNAI*, pages 388–402. Springer, 2002.

[36] Lutz Straßburger. *Linear Logic and Noncommutativity in the Calculus of Structures*. PhD thesis, TU Dresden, 2003.

[37] Lutz Straßburger. MELL in the calculus of structures. *Theoretical Computer Science*, 309:213–285, 2003.

[38] Lutz Straßburger. System NEL is undecidable. In Ruy De Queiroz, Elaine Pimentel, and Lucília Figueiredo, editors, *10th Workshop on Logic, Language, Information and Computation (WoLLIC)*, volume 84 of *Electronic Notes in Theoretical Computer Science*, 2003.

[39] Lutz Strassburger and Alessio Guglielmi. A system of interaction and structure IV: The exponentials and decomposition. *ACM Transactions on Computational Logic*, 12(4), 2011.

[40] Alwen Fernanto Tiu. A local system for intuitionistic logic. In Miki Hermann and Andrei Voronkov, editors, *Logic for Programming, Artificial Intelligence, and Reasoning, Proceedings of the 13th International Conference, LPAR 2006, Phnom Penh, Cambodia*, volume 4246 of *LNCS*, pages 242–256. Springer, 2006.

[41] Alwen Fernanto Tiu. A system of interaction and structure II: the need for deep inference. *Logical Methods in Computer Science*, 2 (2:4):1–24, April 2006.