Reducing Nondeterminism in the Calculus of Structures

Ozan Kahramanoğulları
ozank@doc.ic.ac.uk

Department of Computing, Imperial College London, UK

Abstract. The calculus of structures is a proof theoretical formalism which generalizes the sequent calculus with the feature of deep inference: In contrast to the sequent calculus, inference rules can be applied at any depth inside a formula, bringing shorter proofs than any other formalisms supporting analytical proofs. However, deep applicability of the inference rules causes greater nondeterminism than in the sequent calculus regarding proof search. In this paper, we introduce a new technique which reduces nondeterminism without breaking proof theoretical properties and provides a more immediate access to shorter proofs. We present this technique on system BV, the smallest technically non-trivial system in the calculus of structures, extending multiplicative linear logic with the rules mix, nullary mix, and a self-dual non-commutative logical operator. Because our technique exploits a scheme common to all the systems in the calculus of structures, we argue that it generalizes to these systems for classical logic, linear logic, and modal logics.

1 Introduction

Developing new representations of logics, which address properties that are central to computer science applications, has been one of the challenging goals of proof theory. In this regard, a proof theoretical formalism must be able to provide a rich combinatorial analysis of proofs while being able to address properties such as modularity and locality that are important for applications.

The calculus of structures [6, 8] is a proof theoretical formalism, like natural deduction, the sequent calculus and proof nets, for specifying logical systems while keeping the above mentioned computational aspects in focus (see, e.g., [3, 19]). The calculus of structures is a generalization of the sequent calculus. Structures are expressions intermediate between formulae and sequents which unify these two latter entities. This way, they provide a greater control over the mutual dependencies of logical relations. The main feature that distinguishes this formalism is *deep inference*: In contrast to the sequent calculus, the calculus of structures does not rely on the notion of main connective and permits the application of the inference rules at any depth inside a structure. Derivations are not trees like in the sequent calculus, but chains of inferences.

The calculus of structures was originally conceived to introduce the logical system BV which admits a self-dual non-commutative logical operator resembling sequential composition in process algebras: System BV is an extension of

multiplicative linear logic with the rules mix, nullary mix, and a self-dual non-commutative logical operator. Bruscoli showed in [4] that this operator captures precisely the sequential composition of the process algebra CCS. System BV cannot be designed in any standard sequent calculus, as it was shown by Tiu in [23], because deep inference is crucial for deriving the provable structures of system BV. System BV is NP-complete [13].

The calculus of structures also provides systems which bring new insights to proof theory of other logics: In [2], Brünnler presents systems in the calculus of structures for classical logic; in [20], Straßburger presents systems for different fragments of linear logic. In [18], Stewart and Stouppa give systems for a class of modal logics. Tiu presents, in [22], a local system for intuitionistic logic. All these systems follow a scheme in which two of the three rules of system BV, namely atomic interaction and switch rule (i.e., rules $\mathsf{ai}{\downarrow}$ and s in Figure 2), are common to all these systems. For instance, these two rules give the multiplicative linear logic, whereas a system for classical logic is obtained by adding the contraction and weakening rules to these two rules (see Definition 9). Furthermore, the third rule in system BV (i.e., rule $\mathsf{q}{\downarrow}$ in Figure 2), which is responsible for the non-commutative context management, is also common to the Turing-complete [21] extension of system BV, presented in [9].

Availability of deep inference does not only provide a richer combinatorial analysis of the logic being studied, but also provides shorter proofs than in the sequent calculus [7]: Applicability of the inference rules at any depth inside a structure makes it possible to start the construction of a proof by manipulating and annihilating substructures. This provides many more different proofs of a structure, some of which are shorter than in the sequent calculus. However, deep inference causes a greater nondeterminism: Because the inference rules can be applied at many more positions than in the sequent calculus, the breadth of the search space increases rather quickly.

Reducing nondeterminism in proof search without losing the completeness of the subject system requires combinatorial techniques which work in harmony with the proof theoretical formalism. Because the rules of the sequent calculus act on the main connective and the notion of main connective resolves in the systems with deep inference, it is not possible to use the techniques of the sequent calculus, e.g., focusing [1] (see Section 7), in the systems with deep inference.

In this paper, we introduce a new technique in the calculus of structures that reduces nondeterminism in proof search and makes the shorter proofs more immediately accessible. For this purpose, we employ system BV that exposes the core of our problem, and argue that these ideas generalize to other systems in the calculus of structures. By exploiting an interaction schema on the structures, we redesign the inference rules by means of restrictions such that the inference rules act on the structures only in those ways which promote the interactions between dual atoms and reduce the interaction between atoms which are not duals of each other. These restrictions on the inference rules reduce the breadth of the search space drastically while preserving the shorter proofs, that are available due to deep inference.

Although this technique is quite intuitive, the completeness argument turned out to be difficult. In order to prove the completeness of these systems, we exploit the strong relation between cut elimination and completeness: We resort to a technique, called *splitting*, introduced in [6] for proving cut elimination for system BV. This technique was used also in [20] and [2] for proving cut elimination for linear logic and classical logic, respectively. Because splitting is closely related with cut elimination, it also justifies the cleanness of our technique. Because our technique exploits a scheme which is common to all the systems in the calculus of structures, we argue that it generalizes to other systems for other logics such as classical logic and linear logic. As evidence, we demonstrate this technique on system KSg, a system for classical logic in the calculus of structures.

The present paper extends our previous work in [13], where we have shown that system BV is NP-complete, and in [10, 14], where we have have presented implementations of system BV. We applied the technique presented in this paper to these implementations, and observed a performance improvement in various amounts depending on the structure being proved.

The rest of the paper is organized as follows: In Section 2, we re-collect the notions and notations of the calculus of structures and system BV. Then in the sections 3, 4, and 5 we introduce our technique for reducing nondeterminism at different levels and provide experimental results. In Section 6, we show this technique on a calculus of structures system for classical logic, i.e., system KSg. Section 7 concludes the paper. Space restrictions did not allow us to give the complete proofs of the results. We refer to technical report [12].

2 The Calculus of Structures and System BV

In this section, we collect some notions and definitions of the calculus of structures and system BV, following [6].

In the language of BV atoms are denoted by $a, b, c, \ldots$ Structures are denoted by $R, S, T, \ldots$ and generated by

$$S ::= \circ \mid a \mid \langle \underbrace{S; \ldots; S}_{>0} \rangle \mid [\underbrace{S, \ldots, S}_{>0}] \mid (\underbrace{S, \ldots, S}_{>0}) \mid \overline{S} \quad ,$$

where $\circ$, the *unit*, is not an atom. $\langle S; \ldots; S \rangle$ is called a *seq structure*, $[S, \ldots, S]$ is called a *par structure*, and $(S, \ldots, S)$ is called a *copar structure*, $\overline{S}$ is the *negation* of the structure S. A structure R is called a *proper par structure* if $R = [R_1, R_2]$ where $R_1 \neq \circ$ and $R_2 \neq \circ$. Structures are considered equivalent modulo the relation $\approx$, which is the smallest congruence relation induced by the equations shown in Figure 1. 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*. Context braces are omitted if no ambiguity is possible: For instance $S[R, T]$ stands for $S\{[R, T]\}$. A structure, or a structure context, is in *normal form* when the only negated structures appearing in it are atoms and no unit $\circ$ appears in it. The BV structures whose normal forms do not contain seq structures are called *flat*.

Associativity

$$\langle\langle R;T\rangle;U\rangle \approx \langle R;\langle T;U\rangle\rangle$$
$$[[R,T],U] \approx [R,[T,U]]$$
$$((R,T),U) \approx (R,(T,U))$$

Context Closure

$$\text{if } R \approx T \text{ then } S\{R\} \approx S\{T\} \text{ and } \bar{R} \approx \bar{T}$$

Commutativity

$$[R,T] \approx [T,R]$$
$$(R,T) \approx (T,R)$$

Units

$$\langle\circ;R\rangle \approx \langle R;\circ\rangle \approx \langle R\rangle$$
$$[\circ,R] \approx [R]$$
$$(\circ,R) \approx (R)$$

Negation

$$\overline{\circ} \approx \circ$$
$$\overline{\langle R;T\rangle} \approx \langle\overline{R};\overline{T}\rangle$$
$$\overline{[R,T]} \approx (\overline{R},\overline{T})$$
$$\overline{(R,T)} \approx [\overline{R},\overline{T}]$$
$$\overline{\overline{R}} \approx R$$

Fig. 1. Equivalence relations underlying BV.

In the calculus of structures, an *inference rule* is a scheme of the kind $\rho\,\dfrac{S\{T\}}{S\{R\}}$ where ρ is the *name* of the rule, $S\{T\}$ is its *premise* and $S\{R\}$ is its *conclusion*. Such an inference rules specifies the implication $T \Rightarrow R$ inside a generic context $S\{\ \}$, which is the implication being modeled in the system. An inference rule is called an *axiom* if its premise is empty. Rules with empty contexts correspond to the case of the sequent calculus.

A (formal) *system* $\mathscr{S}$ is a set of inference rules. A derivation Δ in a certain formal system is a finite chain of instances of inference rules in the system. A derivation can consist of just one structure. The topmost structure in a derivation, if present, is called the *premise* of the derivation, and the bottommost 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 $\Delta\dfrac{\|}{R}^{T}\!\!\mathscr{S}$. Similarly, $\Pi\dfrac{\|}{R}\!\!\mathscr{S}$ will denote a *proof* Π which is a finite derivation whose topmost inference rule is an axiom. The *length* of a derivation (proof) is the number of instances of inference rules appearing in it.

We say that two systems $\mathscr{S}$ and $\mathscr{S}'$ are *strongly equivalent* if for every derivation $\Delta\dfrac{\|}{R}^{T}\!\!\mathscr{S}$ there exists a derivation $\Delta\dfrac{\|}{R}^{T}\!\!\mathscr{S}'$ and vice versa. Two systems $\mathscr{S}$ and $\mathscr{S}'$ are *(weakly) equivalent* if for every proof of a structure T in system $\mathscr{S}$, there exists a proof of T in system $\mathscr{S}'$ and vice versa.

The system $\{\circ\downarrow, \mathsf{ai}\downarrow, \mathsf{s}, \mathsf{q}\downarrow\}$, shown in Figure 2, is denoted by BV, and called *basic system* V. The rules of the system are called *unit* ($\circ\downarrow$), *atomic interaction* ($\mathsf{ai}\downarrow$), *switch* (s), and *seq* ($\mathsf{q}\downarrow$). The system $\{\circ\downarrow, \mathsf{ai}\downarrow, \mathsf{s}\}$ is called *flat system* BV, and denoted by FBV.

Guglielmi proves the following result in [6].

Proposition 1. *System* BV *is a conservative extension of system* FBV*, that is, if a flat structure* R *is provable in* BV*, then it is also provable in* FBV*.*

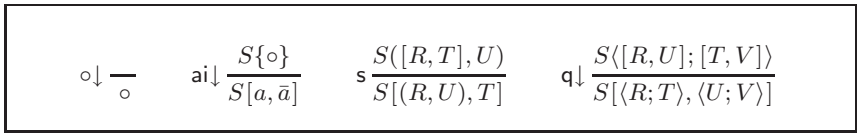

Fig. 2. System BV

There is a straightforward correspondence between flat BV structures and formulae of multiplicative linear logic (MLL) which do not contain the units 1 and $\bot$. For example $[(a,b),\bar{c},\bar{d}]$ corresponds to $((a \otimes b) \parr c^{\bot} \parr d^{\bot})$, and vice versa. Units 1 and $\bot$ are mapped into $\circ$, since $1 \equiv \bot$, when the rules mix and mix0 are added to MLL (see, e.g., [6]). In fact, system FBV proves those structures, which are syntactic variations of the formulae that are provable in MLL+mix+mix0. However, as Tiu showed in [23], system BV cannot be designed in a standard sequent calculus, because a notion of deep rewriting is necessary in order to derive all the provable structures of system BV. For a more detailed discussion on the proof theory of BV and the precise relation between BV and MLL, the reader is referred to [6].

3 The Switch Rule

In this section, we redesign the switch rule such that this rule can be applied only in those ways which promote a specific mutual relation between dual atoms in the structure to which it is applied.[1] Below definition puts this mutual relation between atoms formally.

Definition 1. *Given a structure S, the notation* $\mathsf{at}\,S$ *indicates the set of all the atoms appearing in S. We talk about* atom occurrences *when considering all the atoms appearing in S as distinct (for example, by indexing them so that two atoms which are equal get different indices). The notation* $\mathsf{occ}\,S$ *indicates the set of all the atom occurrences appearing in S. The* size *of S is the cardinality of the set* $\mathsf{occ}\,S$*. Given a structure S in normal form, we define the* structural relation $\downarrow \subseteq (\mathsf{occ}\,S)^2$ *as follows: for every $S'\{\ \}$, U, and V and for every a in U and b in V, if $S = S'[U,V]$ then $a \downarrow_S b$. To a structure that is not in normal form we associate the structural relation obtained from any of its normal forms, since they yield the same relation $\downarrow_S$.*

In order to see the above definition at work, consider the following structure: $S = [a, b, (\bar{b}, [\langle \bar{a}; c\rangle, \bar{c}])]$. We have $\mathsf{at}\,S = \mathsf{occ}\,S = \{a, \bar{a}, b, \bar{b}, c, \bar{c}\}$. Then, we have $a \downarrow b$, $a \downarrow \bar{b}$, $a \downarrow \bar{a}$, $a \downarrow c$, $a \downarrow \bar{c}$, $b \downarrow \bar{b}$, $b \downarrow \bar{a}$, $b \downarrow c$, $b \downarrow \bar{c}$, $\bar{a} \downarrow \bar{c}$, $c \downarrow \bar{c}$ (we omit the symmetric relations, e.g., $b \downarrow a$).

[1] These relations emerge from a graphic representation of structures, called *relation webs*, justified by the equivalence relations in Figure 1. However, in this paper we give a partial exposure to relation webs, referring the reader to [6].

Intuitively, one can consider the relation $\downarrow_S$ as a notion of interaction: The atoms which are related by $\downarrow_S$ are interacting atoms, whereas others are non-interacting. Proofs are constructed by isolating the atoms, by breaking the interaction between some atoms, and this way promoting the interaction between dual atoms, till dual atoms establish a closer interaction in which they can annihilate each other at an application of the atomic interaction rule. During a bottom-up proof search episode, while acting on structures, inference rules perform such an isolation of atoms: In an instance of an inference rule with the conclusion S, a subset of $\downarrow_S$ holds in the premise. For example, consider the following three instances of the switch rule with the same structure at the conclusion:

$$(i.)\quad \mathsf{s}\frac{([\bar{a},a,b],\bar{b})}{[(\bar{a},\bar{b}),a,b]} \qquad (ii.)\quad \mathsf{s}\frac{[([\bar{a},b],\bar{b}),a]}{[(\bar{a},\bar{b}),a,b]} \qquad (iii.)\quad \mathsf{s}\frac{[(\bar{a},\bar{b},a),b]}{[(\bar{a},\bar{b}),a,b]}$$

While going up, from conclusion to premise, in $(i.)$ $a \downarrow b$ and $b \downarrow \bar{b}$; in $(ii.)$ $b \downarrow \bar{b}$; in $(iii.)$ $a \downarrow \bar{a}$ and $a \downarrow \bar{b}$ cease to hold. However, none of these derivations can lead to a proof. Following proposition expresses the intuition behind this.

Proposition 2. *If a structure R has a proof in* BV *then, for all the atoms a that appear in R, there is an atom $\bar{a}$ in R such that $a \downarrow_R \bar{a}$.*

Often, inference rules can be applied to a structure in many different ways, however only few of these applications can lead to a proof. For example, to the structure $[(\bar{a},\bar{b}),a,b]$ switch rule can be applied bottom-up in twelve different ways, three of them which are given above, but only two of these twelve instances can lead to a proof. With the below definition, we will redesign the switch rule such that only these applications will be possible.

Definition 2. *Let* interaction switch *be the rule*

$$\mathsf{is}\frac{S([R,W],T)}{S[(R,T),W]} \quad ,$$

where $\mathsf{at}\,\overline{W} \cap \mathsf{at}\,R \neq \emptyset$.

Definition 3. *Let* system BV with interaction switch*, or system* BVs *be the system* $\{\circ\downarrow, \mathsf{ai}\downarrow, \mathsf{is}, \mathsf{q}\downarrow\}$. *Let* system BV with lazy interaction switch*, or system* BVsl *be the system resulting from replacing the rule* is *in* BVs *with its instance, called* lazy interaction switch*, or* lis*, where the structure W is not a proper par structure.*

The switch rule can be safely replaced with the lazy interaction switch rule in system BV without losing completeness. In the following, we will collect some definitions and lemmas which are necessary to prove this result.

Definition 4. *Let R,T be* BV *structures such that $R \neq \circ \neq T$. R and T are* independent *iff, for $\mathscr{S} \in \{\mathsf{BV}, \mathsf{BVs}, \mathsf{BVsl}\}$,*

$$\prod\nolimits^{\mathscr{S}}_{[R,T]} \qquad \textit{implies} \qquad \prod\nolimits^{\mathscr{S}}_{R} \quad \textit{and} \quad \prod\nolimits^{\mathscr{S}}_{T} \quad .$$

Otherwise, they are dependent.

Proposition 3. *For any* BV *structures* R *and* T*, if* $\mathsf{at}\,\bar{R} \cap \mathsf{at}\,T = \emptyset$ *then* R *and* T *are independent.*

Lemma 1. *For any* BV *structures* R*,* P*, and* U*,*

$$\textit{if} \quad \begin{array}{c} \pi \top\!\!\!\top_{\mathsf{BVsl}} \\ [P, U] \end{array} \quad \textit{then there is a derivation} \quad \begin{array}{c} R \\ \|_{\mathsf{BVsl}} \\ [(R, P), U] \end{array} .$$

Sketch of Proof: If U is not a proper par structure Lemma is proved. Otherwise, by consequent application of the rule lis bring the partition of the structure U which is dependent with P into the same par context as P. □

Proposition 4. *In* BV *(*BVs*,* BVsl*),* $\langle R; T\rangle$ *is provable if and only if* R *and* T *are provable and* (R, T) *is provable if and only if* R *and* T *are provable.*

The following theorem is a specialization of the shallow splitting theorem which was introduced in [6] for proving cut elimination for system BV. Exploiting the fact that systems in the calculus of structures follow a scheme, in which the rules atomic interaction and switch are common to all these systems, this technique was used also to prove cut elimination for classical logic [2], linear logic [20], and system NEL [9, 21] (Turing-complete extension of BV with the exponentials of linear logic). As the name suggests, this theorem splits the context of a structure so that the proof of the structure can be partitioned into smaller pieces in a systematic way. Below we show that splitting theorem can be specialized to system BVsl where the switch rule in system BV is replaced with the lazy interaction switch rule.

Theorem 1. *(Shallow Splitting for* BVsl*) For all structures* R*,* T *and* P*:*

1. *if* $[\langle R; T\rangle, P]$ *is provable in* BVsl *then there exists* P_1*,* P_2 *and* $\begin{array}{c} \langle P_1; P_2\rangle \\ \Delta \|_{\mathsf{BVsl}} \\ P \end{array}$ *such that* $[R, P_1]$ *and* $[T, P_2]$ *are provable in* BVsl*.*
2. *if* $[(R, T), P]$ *is provable in* BVsl *then there exists* P_1*,* P_2 *and* $\begin{array}{c} [P_1, P_2] \\ \Delta \|_{\mathsf{BVsl}} \\ P \end{array}$ *such that* $[R, P_1]$ *and* $[T, P_2]$ *are provable in* BVsl*.*

Sketch of Proof: Proof by induction, with Lemma 1, similar to the proof of shallow splitting for system BV in [6]: Single out the bottom-most rule instance ρ in the given proof, and do case analysis on ρ. □

Because inference rules can be applied at any depth inside a structure, we need the following theorem for accessing the deeper structures.

Theorem 2. (Context Reduction for BVsl) *For all structures* R *and for all contexts* $S\{\ \}$ *such that* $S\{R\}$ *is provable in* BVsl*, there exists a structure* U

such that for all structures X there exist derivations:

$$\begin{array}{c}[X,U]\\ \Vdash_{\mathsf{BVsl}}\\ S\{X\}\end{array} \qquad \textit{and} \qquad \begin{array}{c}\Vdash_{\mathsf{BVsl}}\\ [R,U]\end{array} \quad .$$

Sketch of Proof: Proof by induction, with Proposition 4 and Lemma 1, similar to the proof of context reduction for system BV in [6]: Do case analysis on the context $S\{\ \}$. □

We can now prove the following two results:

Theorem 3. *Systems* BV*,* BVsl*, and* BVs *are equivalent.*

Sketch of Proof: Observe that every proof in BVsl is also a proof in BVs and every proof in BVs is a proof in BV. For the other direction, single out the upper-most instance of the switch rule in the BV proof which is not an instance of the lazy interaction switch rule. Apply Theorem 2 to reduce the context of the premise. Construct a proof in BVsl with Lemma 1 by partitioning the resulting proof by Theorem 1. Repeat the above procedure inductively until all the instances of the switch rule that are not instances of lazy interaction switch rule are removed. □

Let us now consider the rule lis on some examples: In the proof search space of $[(\bar{a},\bar{b}),a,b]$ there are 12 instances of the switch rule. In system FBV, these instances result in 358 different derivations. However, only 6 of these derivations are proofs. Let FBVi denote the system obtained from system FBV by replacing the switch rule with the rule lis. In system FBVi, we observe that we have only the following instances, which lead to 6 proofs mentioned above.

$$\mathsf{lis}\,\frac{[([\bar{a},a],\bar{b}),b]}{[(\bar{a},\bar{b}),a,b]} \qquad\qquad \mathsf{lis}\,\frac{[([\bar{b},b],\bar{a}),a]}{[(\bar{a},\bar{b}),a,b]}$$

When we consider deeply nested structures, we observe that the switch rule can be applied in many more ways due to the deep inference feature. For instance, consider the structure $[([\bar{a}_1,(\bar{a}_2,\bar{b}_2),a_2,b_2],\bar{b}_1),a_1,b_1]$ which is obtained by nesting the structure $[(\bar{a},\bar{b}),a,b]$ in itself. To this structure switch rule can be applied in 51 different ways, but only 4 of these instances provide a proof. These 4 instances are the only possible instances of the rule lis. In particular, the deeper instances of the rule lis (marked above) provide shorter proofs which are not possible in the sequent calculus.

We have implemented the systems above in Maude [5] as described in [10, 11]. In these implementations, inference rules are expressed as (conditional) term rewriting rules. For proof search, we use the built-in breadth-first search function. Some representative examples of our experiments for comparing the performance of systems FBV and FBVi are as follows: (All the experiments below are performed on an Intel Core Duo 1.83 GHz processor.)

1. $[a,b,(\bar{a},\bar{c}),(\bar{b},c)]$
2. $[a,b,(\bar{a},\bar{b},[a,b,(\bar{a},\bar{b})])]$
3. $[a,b,(\bar{a},\bar{b},[c,d,(\bar{c},\bar{d})])]$
4. $[a,b,(\bar{a},\bar{b},[c,d,(\bar{c},\bar{d},[e,f,(\bar{e},\bar{f})])])]$

Query	System	# states explored	finds a proof in # ms (cpu)	Query	System	# states explored	finds a proof in # ms (cpu)
1.	FBV	342	60	2.	FBV	1041	100
	FBVi	34	10		FBVi	264	0
3.	FBV	1671	310	4.	FBV	(∗)	
	FBVi	140	0		FBVi	6595	1370

(∗) On this query, search halted by running out of memory after having spent approximately 3GB memory and 80 minutes (cpu).

4 The Seq Rule

At a first glance, the rules switch and seq appear to be different in nature due to the different logical operators they work on. However, at a closer inspection of these rules, one can observe that both of these rules manage the context of the structures they are applied at: While the switch rule reduces the interaction in the structures involving a copar structure in a bottom-up application, the seq rule does the same with the structures involving seq structures. In this section, exploiting this observation, we will carry the ideas from the previous section to the seq rule.

Definition 5. *Let the system consisting of the rules*

$$\mathsf{q}_1{\downarrow}\,\frac{S\langle [R,T];[U,V]\rangle}{S[\langle R;U\rangle,\langle T;V\rangle]} \qquad \mathsf{q}_2{\downarrow}\,\frac{S\langle R;T\rangle}{S[R,T]} \qquad \mathsf{lq}_3{\downarrow}\,\frac{S\langle [R,W];T\rangle}{S[W,\langle R;T\rangle]} \qquad \mathsf{lq}_4{\downarrow}\,\frac{S\langle R;[T,W]\rangle}{S[W,\langle R;T\rangle]}$$

where W is not a proper par structure, and none of the structures R, T, U, V, W is the unit $\circ$, be the lazy seq system V*, or* QVI*.*

In the above definition, we partition the seq rule, making its instances with respect to the unit specific. This way, one can also observe the similarity between the switch rule and seq rule, in particular the rules $\mathsf{lq}_3{\downarrow}$ and $\mathsf{lq}_4{\downarrow}$. In fact, Retoré gives similar rules for Pomset Logic in [17], which is conjectured to be equivalent to BV in [6]. However he does not provide a cut-elimination proof. The following proposition, that we proved in [11], shows that in any system the seq rule can be safely replaced with the system QVI.

Proposition 5. *System* QVI *and system* $\{\mathsf{q}{\downarrow}\}$ *are strongly equivalent.*

Proposition 6. *Let* $\mathscr{S} \in \{\mathsf{BV}, \mathsf{BVs}, \mathsf{BVsl}\}$*. The system resulting from replacing the rule* $\mathsf{q}{\downarrow}$ *in* $\mathscr{S}$ *with system* QVI *and system* BV *are equivalent.*

Below, we will carry the ideas of the previous section to the seq rule.

Definition 6. *The following rules are called* interaction seq rule 1*,* lazy interaction seq rule 3*, and* lazy interaction seq rule 4*, respectively,*

$$\mathsf{iq}_1{\downarrow}\,\frac{S\langle [R,T];[U,V]\rangle}{S[\langle R;U\rangle,\langle T;V\rangle]} \qquad \mathsf{liq}_3{\downarrow}\,\frac{S\langle [R,W];T\rangle}{S[W,\langle R;T\rangle]} \qquad \mathsf{liq}_4{\downarrow}\,\frac{S\langle T;[R,W]\rangle}{S[W,\langle T;R\rangle]}$$

where in $\mathsf{iq}_1{\downarrow}$ *we have* $\mathsf{at}\,\overline{R}\cap\mathsf{at}\,T\neq\emptyset$ *and* $\mathsf{at}\,\overline{U}\cap\mathsf{at}\,V\neq\emptyset$*; in* $\mathsf{liq}_3{\downarrow}$ *and in* $\mathsf{liq}_4{\downarrow}$ *we have* $\mathsf{at}\,\overline{R}\cap\mathsf{at}\,W\neq\emptyset$ *and* W *is not a proper par structure. The system resulting from replacing the seq rule in system* BVsl *with the rules* $\mathsf{iq}_1{\downarrow}$*,* $\mathsf{q}_2{\downarrow}$*,* $\mathsf{liq}_3{\downarrow}$*, and* $\mathsf{liq}_4{\downarrow}$ *is called* interaction system BV*, or* BVi*.*

Definition 7. *The following rules are called* non-interaction seq rule 1*,* non-interaction seq rule 3 *and* non-interaction seq rule 4*, respectively,*

$$\mathsf{niq}_1{\downarrow}\,\frac{S\langle [R,T];[U,V]\rangle}{S[\langle R;U\rangle,\langle T;V\rangle]} \qquad \mathsf{niq}_3{\downarrow}\,\frac{S\langle [R,W];T\rangle}{S[W,\langle R;T\rangle]} \qquad \mathsf{niq}_4{\downarrow}\,\frac{S\langle T;[R,W]\rangle}{S[W,\langle T;R\rangle]}$$

where in $\mathsf{niq}_1{\downarrow}$ *we have* $\mathsf{at}\,\overline{R}\cap\mathsf{at}\,T=\emptyset$ *or* $\mathsf{at}\,\overline{U}\cap\mathsf{at}\,V=\emptyset$*; in* $\mathsf{niq}_3{\downarrow}$ *and in* $\mathsf{niq}_4{\downarrow}$ *we have* $\mathsf{at}\,\overline{R}\cap\mathsf{at}\,W=\emptyset$*.*

Remark 1. Every instance of the rule $\mathsf{q}{\downarrow}$ is an instance of one of the rules $\mathsf{iq}_1{\downarrow}$, $\mathsf{niq}_1{\downarrow}$, $\mathsf{q}_2{\downarrow}$, $\mathsf{liq}_3{\downarrow}$, $\mathsf{niq}_3{\downarrow}$, $\mathsf{liq}_4{\downarrow}$, $\mathsf{niq}_4{\downarrow}$.

Below, we will see that system BV and BVi are equivalent. However, using the splitting technique, in the form it was used in the previous section, will not be possible for proving this argument. In order to see the reason for this consider the structure $[\langle [a,b,c];[d,e]\rangle,\bar{a},\langle\bar{b};\bar{d}\rangle,\langle\bar{c};\bar{e}\rangle]$ which is provable in BVsl (and also in system BVi). By applying Theorem 1, we can obtain the derivation

$$\begin{array}{l}\mathsf{q}_3{\downarrow}\\ \mathsf{q}_1{\downarrow}\end{array}\frac{\dfrac{\langle[\bar{a},\bar{b},\bar{c}];[\bar{d},\bar{e}]\rangle}{[\bar{a},\langle[\bar{b},\bar{c}];[\bar{d},\bar{e}]\rangle]}}{[\bar{a},\langle\bar{b};\bar{d}\rangle,\langle\bar{c};\bar{e}\rangle]} \quad \text{such that} \quad \begin{array}{c}\Pi\\ \prod\!\!\!\!\prod^{\mathsf{BVsl}}\\ [\bar{a},\bar{b},\bar{c},a,b,c]\end{array} \quad \text{and} \quad \begin{array}{c}\Pi\\ \prod\!\!\!\!\prod^{\mathsf{BVsl}}\\ [\bar{d},\bar{e},d,e]\end{array}.$$

However, the derivation on the left-hand side above is not possible in system BVi. For this reason, in the following, we will introduce a generalization of the splitting Theorem for system BVi.

Theorem 4. *(Shallow Splitting for* BVi*) For all structures* R*,* T*, and* P*: if the structure* $[\langle L;R\rangle,U]$ *or the structure* $[(L,R),U]$ *has a proof* Π *in* BVsl*, then there are structures* $L_1,\ldots,L_m$*,* $P_{1,1},\ldots,P_{s,2}$*,* $R_1,\ldots,R_n$ *and there exist a derivation*

$$\begin{array}{c}[L_1,\ldots,L_m,\langle P_{1,1};P_{1,2}\rangle,\ldots,\langle P_{s,1};P_{s,2}\rangle,R_1,\ldots,R_n]\\ \big[\!\big[^{\mathsf{BVi}}\\ U\end{array}$$

and proofs

$$\begin{array}{c}\prod\!\!\!\!\prod^{\mathsf{BVi}}\\ [L,L_1,\ldots,L_m,P_{1,1},\ldots,P_{s,1}]\end{array} \quad \textit{and} \quad \begin{array}{c}\prod\!\!\!\!\prod^{\mathsf{BVi}}\\ [R,P_{1,2},\ldots,P_{s,2},R_1,\ldots,R_n]\end{array}.$$

Sketch of Proof: Proof by induction: Apply Theorem 1 to the proof Π. This delivers a derivation Δ and two proofs Π_1 and Π_2 in BVsl. Take the derivation Δ and permute down all the instances of $\mathsf{niq}_1\downarrow$, $\mathsf{niq}_3\downarrow$, and $\mathsf{niq}_4\downarrow$ in Δ and apply the induction hypothesis to the proofs Π_1 and Π_2. □

Corollary 1. *Systems* BV *and* BVi *are equivalent.*

Sketch of Proof: Observe that every proof in BVi is also a proof in BV. For the other direction, first construct proof in BVsl by Theorem 3, and then construct a proof in BVi by Theorem 4. □

Let us now consider system BV and BVi with respect to our Maude implementations. Some representative examples for comparing the performance of systems BV and BVi are as follows:

1. $[\langle a; [b, c]\rangle, \langle [\bar{a}, \bar{b}]; \bar{c}\rangle]$
2. $[\langle (\langle [d, \bar{d}], \langle a; b\rangle); c\rangle, \langle \bar{a}; (\langle \bar{b}; \bar{c}\rangle, [e, \bar{e}])\rangle]$
3. $[\langle (b, c); [d, e]\rangle, \langle [\bar{b}, \bar{c}]; (\bar{d}, \bar{e})\rangle]$
4. $[\bar{a}, (a, \langle d; \bar{b}\rangle), (b, c), \langle \bar{d}; \bar{c}\rangle]$

Query	System	# states explored	finds a proof in # millisec.	Query	System	# states explored	finds a proof in # millisec.
1.	BV	1263	630	2.	BV	8069	890
	BVi	995	480		BVi	2138	620
3.	BV	11191	1740	4.	BV	123154	5010
	BVi	3696	560		BVi	20371	1050

The restrictions that are imposed on the inference rules of system BVi succeed in eliminating unsuccessful branches in the proof search space of BV structures. However, the rule $\mathsf{q}_2\downarrow$ causes still a huge amount of redundant nondeterminism in proof search: For instance, consider the BV structure $[a, \bar{a}, b, \bar{b}]$ which can be trivially proved by applying the rule $\mathsf{ai}\downarrow$ twice. To this structure, the rule $\mathsf{q}_2\downarrow$ can be applied in 50 different ways, but removing this rule from system BVi results in an incomplete system, because some provable BV structures, e.g., $[\langle a; [b, c]\rangle, \langle [\bar{a}, \bar{b}]; \bar{c}\rangle]$, are not provable without this rule.

5 Cautious Rules

In a bottom-up application of the rules switch and seq in proof construction, besides promoting interactions between some atoms, the interaction between some atoms are broken (for instance, consider the example derivations (*i.*), (*ii.*), and (*iii.*) in Section 3.). However, if the structure being proved consists of pairwise distinct atoms, breaking the interaction between dual atoms, in a bottom-up inference step delivers a structure which cannot be proved. The following definition introduces a further restriction on these inference rules, which exploits this observation and allows only cautious instances of the inference rules which do not break the interaction between dual atoms.

Definition 8. *Let* pruned switch *be the rule* ps *below where* $\mathsf{at}\,\overline{T} \cap \mathsf{at}\,W = \emptyset$*, and let* pruned seq *be the rule* $\mathsf{pq}{\downarrow}$ *below where* $\mathsf{at}\,\overline{T} \cap \mathsf{at}\,U = \emptyset$ *and* $\mathsf{at}\,\overline{R} \cap \mathsf{at}\,V = \emptyset$*:*

$$\mathsf{ps}\,\frac{S([R,W],T)}{S[(R,T),W]} \qquad\qquad \mathsf{pq}{\downarrow}\,\frac{S\langle[R,T];[U,V]\rangle}{S[\langle R;U\rangle,\langle T;V\rangle]} \quad ,$$

Let pruned system BV*, or system* BVp *be the system* $\{\circ{\downarrow}\,,\,\mathsf{ai}{\downarrow}\,,\,\mathsf{ps}\,,\,\mathsf{pq}{\downarrow}\}$.

Proposition 7. *Let* P *be a* BV *structure that consists of pairwise distinct atoms and* Π *be a proof of* P *in* BV *(*BVs*,* BVsl*, respectively). In* Π*, all the instances of the rule* s *(*is*,* lis*, respectively) are instances of the rule* ps*; and all the instances of the rule* $\mathsf{q}{\downarrow}$ *are instances of the rule* $\mathsf{pq}{\downarrow}$*.*

Sketch of Proof: It suffices to show that, by Proposition 2, a bottom-up application of the inference rules without respecting the above restrictions result in a structure which is not provable in BV. □

Proposition 8. *Let* P *be a* BV *structure that consists of pairwise distinct atoms and* Π *be a proof of* P *in* BVi*. In* Π*, all the instances of the rule* s *are instances of the rule* ps*; and all the instances of the rule* $\mathsf{iq}_1{\downarrow}$*,*$\mathsf{q}_2{\downarrow}$*,* $\mathsf{liq}_3{\downarrow}$*, and* $\mathsf{liq}_4{\downarrow}$ *are instances of the rule* $\mathsf{pq}{\downarrow}$*.*

Sketch of Proof: Follows immediately from Remark 1 and Proposition 7. □

6 Nondeterminism in Classical Logic

Systems in the calculus of structures follow a common scheme where the context management of the commutative operators is performed by the switch rule. System KSg for classical logic [2] is no exception to this. In this section, we will see that, similar to system BV, the switch rule of system KSg can be safely replaced with the lazy interaction switch rule in order to reduce nondeterminism in proof search.

Definition 9. *The system* KSg *is the system consisting of the rules*

$$\mathsf{t}{\downarrow}\,\frac{}{\mathtt{t}} \quad , \quad \mathsf{ai}{\downarrow}\,\frac{S\{\mathtt{t}\}}{S[a,\bar{a}]} \quad , \quad \mathsf{s}\,\frac{S([R,U],T)}{S[(R,T),U]} \quad , \quad \mathsf{w}{\downarrow}\,\frac{S\{\mathtt{f}\}}{S\{R\}} \quad , \textit{and} \quad \mathsf{c}{\downarrow}\,\frac{S[R,R]}{S\{R\}} \quad .$$

The rules of the system KSg *are called* axiom*,* atomic interaction*,* switch*,* weakening*, and* contratction*, respectively.* KSg *structures are defined as* FBV *structures with the difference that* $\mathtt{f}$ *is the unit for the* disjunction $[\_,\_]$ *and* $\mathtt{t}$ *is the unit for the* conjunction $(\_,\_)$ *and we also impose the equalities* $[\mathtt{t},\mathtt{t}] \approx \mathtt{t}$ *and* $(\mathtt{f},\mathtt{f}) \approx \mathtt{f}$*. The system* KSgi *is the system obtained from system* KSg *by replacing the rule* s *with the rule* lis*.*

Theorem 5. *A structure* R *has a proof in* KSg *if and only if there is a structure* R' *and there is a proof of the form*

$$\begin{array}{l} \top \\ \Vert\,\{\mathsf{s},\mathsf{ai}{\downarrow}\} \\ R' \\ \Vert\,\{\mathsf{w}{\downarrow},\mathsf{c}{\downarrow}\} \\ R \end{array} \quad .$$

Sketch of Proof: If R is provable in KSg, then we can construct the conjunctive normal form of R while going up in the derivation by first applying only the rule $\mathsf{c}{\downarrow}$ and then only the rule s. Then a proof of conjunctive normal form of R can be constructed by applying first only the rule $\mathsf{w}{\downarrow}$ and then the rule $\mathsf{ai}{\downarrow}$. By permuting all the instances of $\mathsf{w}{\downarrow}$ under the instances of s, we get the desired proof. □

The reader might realize that there is a significant similarity between the systems $\{\mathsf{ai}{\downarrow}, \mathsf{lis}\}$ and the system FBVi (FBV) (the system for multiplicative linear logic extended by the rules mix and nullary mix). Indeed, these two systems are the same up to the inference rules. However, the treatment of the units in these systems is quite different: In system FBV there is a single unit, which is shared by all the connectives. On the other hand, in system $\{\mathsf{ai}{\downarrow}, \mathsf{lis}\}$, there are two different units, tt and ff, which are units for different operators. We can now state the main result of this section:

Theorem 6. *System* KSg *and* KSgi *are equivalent.*

Sketch of Proof: Observe that every proof in KSgi is a proof in system KSg. For the other direction, replace the proof Π in $\{\mathsf{s}, \mathsf{ai}{\downarrow}\}$, delivered from Theorem 5 with a proof Π' in $\{\mathsf{lis}, \mathsf{ai}{\downarrow}\}$ similar to the proof of Theorem 3. □

7 Discussion

We presented a novel technique for reducing nondeterminism in proof search by restricting the application of the inference rules. This resulted in a class of equivalent systems to system BV where nondeterminism is reduced at different levels. We have also seen that this technique generalizes to system KSg for classical logic. In these systems, inference rules can be applied only in certain ways that promote the interaction, in the sense of a specific mutual relation, between dual atoms. Because of the splitting argument that we use in our completeness proof, which is strongly related to cut elimination, our rules remain clean from a proof theoretic point of view. Because proofs are constructed by annihilating dual atoms, these restrictions reduce the breadth of the search space drastically and preserve the shorter proofs that are available due to deep inference.

We have implemented the proof search for the systems BV and BVi in the lines of [10]. These implementations makes use of the simple high level language, the term rewriting features, and the built-in breadth-first function of the language Maude [5]. In [14], we have presented another implementation of system BV in Java, where different search strategies can be easily employed. This implementation uses the pattern matching preprocessor TOM [16] that makes it possible to integrate term rewriting features into Java. The Maude modules[2] together with representative proof search queries, the source code of the Java implementation[3], and a proof search applet[4] are available online.

[2] `http://www.iccl.tu-dresden.de/~ozan/maude_cos.html`
[3] `http://tom.loria.fr`
[4] `http://tom.loria.fr/examples/structures/BV.html`

In our approach, in order to prove the completeness of the restricted systems, we use the *splitting* technique which was introduced and used by Guglielmi in [6] for proving cut elimination in system BV. In [20], Straßburger used the splitting technique to prove cut elimination in the calculus of structures systems for different fragments of linear logic. All the systems in the calculus of structures follow a scheme where the context management is performed by the *switch* rule. Because splitting technique is common to these other systems, our technique should generalize to other systems for linear logic. In Section 6, we have seen that for the case of classical logic, switch rule can be replaced with the lazy interaction switch rule in system KSg. In the light of this result, we conjecture that this technique generalizes to the calculus of structures systems for a class of modal logics [18] that extend system KSg with the modal rules.

Although our technique attacks the same problem as Miller's Forum [15] where Andreoli's focusing technique [1] is used for reducing nondeterminism in linear logic proofs, our approach is different, in essence, than uniform proofs: Focusing technique is based on permuting different phases of a proof by distinguishing between asynchronous (deterministic) and synchronous (nondeterministic) parts of a proof. This approach depends on the fact that in the sequent calculus asynchronous connectives, e.g., par, and synchronous connectives, e.g., copar, can be treated in isolation. However, in the calculus of structures connectives are never in isolation: Asynchronous connectives are always matched to a synchronous connective at each inference step. Furthermore, asynchronous parts of a proof normally spread the object level, given by the logical operators, onto the meta-level. For instance, par operators are mapped to commas. In the systems with deep inference, because what is meta-level in the sequent calculus is brought to the object level, thus there is no meta-level, this is a superfluous operation.

Acknowledgements: The author would like to thank Alessio Guglielmi, Lutz Straßburger, Kai Brünnler, Alwen Tiu, and the anonymous referees for valuable remarks and improvements. This work has been supported by the DFG Graduiertenkolleg 446 at the University of Leipzig, and accomplished during author's stay at the ICCL–TU Dresden as a visiting researcher.

References

1. J.-M. Andreoli. Logic programming with focussing proofs in linear logic. *Journal of Logic and Compututation*, 2(3):297–347, 1992.
2. K. Brünnler. *Deep Inference and Symmetry in Classical Proofs*. PhD thesis, TU Dresden, 2003.
3. K. Brünnler and A. F. Tiu. A local system for classical logic. In R. Nieuwenhuis and A. Voronkov, editors, *LPAR 2001*, volume 2250 of *LNAI*, pages 347–361. Springer, 2001.
4. P. Bruscoli. A purely logical account of sequentiality in proof search. In P. J. Stuckey, editor, *Logic Prog., 18th Int. Conf.*, volume 2401 of *LNCS*, pages 302–316. Springer, 2002.

5. M. Clavel, F. Durán, S. Eker, P. Lincoln, N. Martí-Oliet, J. Meseguer, and C. Talcott. The Maude 2.0 system. In Robert Nieuwenhuis, editor, *Rewriting Techniques and Applications, Proc. of the 14th Int. Conf.*, volume 2706. Springer, 2003.
6. A. Guglielmi. A system of interaction and structure. Technical Report WV-02-10, TU Dresden, 2002. Accepted by ACM Transactions on Computational Logic.
7. A. Guglielmi. Polynomial size deep-inference proofs instead of exponential size shallow-inference proofs. Available on the web at http://cs.bath.ac.uk/ag/p/AG12.pdf, 2004.
8. A. Guglielmi and L. Straßburger. Non-commutativity and MELL in the calculus of structures. In L. Fribourg, editor, *CSL 2001*, volume 2142 of *LNCS*, pages 54–68. Springer, 2001.
9. A. Guglielmi and L. 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.
10. O. Kahramanoğulları. Implementing system BV of the calculus of structures in Maude. In L. Alonso i Alemany and P. Égré, editors, *Proc. of the ESSLLI-2004 Student Session*, pages 117–127, Université Henri Poincaré, Nancy, France, 2004.
11. O. Kahramanoğulları. System BV without the equalities for unit. In C. Aykanat, T. Dayar, and I. Körpeoğlu, editors, *Proc. of the 19th Int. Symp. on Comp. and Inform. Sciences, ISCIS'04*, volume 3280 of *LNCS*. Springer, 2004.
12. O. Kahramanoğulları. Reducing nondeterminism in the calculus of structures. Technical Report WV-06-01, TU Dresden, 2006. Available at http://www.ki.inf.tu-dresden.de/~ozan/redNondet.pdf.
13. O. Kahramanoğulları. System BV is NP-complete. In R. de Queiroz, A. Macintyre, and G. Bittencourt, editors, *WoLLIC 2005*, volume 143 of *ENTCS*, pages 87–99, Florianapolis, Brazil, 2006. Elsevier.
14. O. Kahramanoğulları, P.-E. Moreau, and A. Reilles. Implementing deep inference in TOM. In P. Bruscoli, F. Lamarche, and C. Stewart, editors, *Structures and Deduction'05 (ICALP'05 Workshop)*, pages 158–172, Lisbon, Portugal, 2005.
15. D. Miller. Forum: A multiple-conclusion specification logic. *Theoretical Computer Science*, 165:201–232, 1996.
16. P.-E. Moreau, C. Ringeissen, and M. Vittek. A pattern matching compiler for multiple target languages. In G. Hedin, editor, *12th Conference on Compiler Construction, Warsaw*, volume 2622 of *LNCS*, pages 61–76. Springer, 2003.
17. C. Retoré. Pomset logic: A non-commutative extension of classical linear logic. In Ph. de Groote and J. R. Hindley, editors, *Typed Lambda Calculus and Applications, TLCA'97*, volume 1210 of *LNCS*, pages 300–318. Springer, 1997.
18. C. Stewart and P. Stouppa. A systematic proof theory for several modal logics. In R. Schmidt, I. Pratt-Hartmann, M. Reynolds, and H. Wansing, editors, *Advances in Modal Logic*, volume 5 of *King's College Publications*, pages 309 – 333, 2005.
19. L. 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.
20. L. Straßburger. *Linear Logic and Noncommutativity in the Calculus of Structures*. PhD thesis, TU Dresden, 2003.
21. L. Straßburger. System NEL is undecidable. In R. de Queiroz, E. Pimentel, and L. Figueiredo, editors, *WoLLIC 2003*, volume 84 of *ENTCS*. Elsevier, 2003.
22. A. F. Tiu. A local system for intuitionistic logic. Accepted at LPAR 2006.
23. A. F. Tiu. A system of interaction and structure II: The need for deep inference. to appear on Logical Methods in Computer Science, 2005.