System BV without the Equalities for Unit

Ozan Kahramanoğulları

Computer Science Institute, University of Leipzig
International Center for Computational Logic, TU Dresden
ozan@informatik.uni-leipzig.de

Abstract. System BV is an extension of multiplicative linear logic with a non-commutative self-dual operator. In this paper we present systems equivalent to system BV where equalities for unit are oriented from left to right and new structural rules are introduced to preserve completeness. While the first system allows units to appear in the structures, the second system makes it possible to completely remove the units from the language of BV by proving the normal forms of the structures that are provable in BV. The resulting systems provide a better performance in automated proof search by disabling redundant applications of inference rules due to the unit. As evidence, we provide a comparison of the performance of these systems in a Maude implementation.

1 Introduction

The calculus of structures is a proof theoretical formalism, like natural deduction, the sequent calculus and proof nets, for specifying logical systems syntactically. It was conceived in [6] to introduce the logical system BV, which extends multiplicative linear logic by a non-commutative self-dual logical operator. Then it turned out to yield systems with interesting and exciting properties for existing logics and new insights to proof theory [12, 1]. In [14], Tiu showed that BV is not definable in any sequent calculus system. Bruscoli showed in [2] that the non-commutative operator of BV captures precisely the sequentiality notion of process algebra, in particular CCS.

In contrast to sequent calculus, the calculus of structures does not rely on the notion of main connective and, like in term rewriting, it permits the application of the inference rules deep inside a formula (structure) which are considered equivalent modulo different equational theories (associativity, commutativity, unit, etc.). This resemblance allows us to express systems in the calculus of structures as term rewriting systems modulo equational theories [8].

In [9], we presented a Maude [3, 4] implementation of system BV. The language Maude allows implementing term rewriting systems modulo equational theories due to the built in very fast matching algorithm that supports different combinations of associative, commutative equational theories, also with the presence of units. However, we observed that, often, units cause redundant matchings of the inference rules where the premise and conclusion at the application of the inference rule are equivalent structures.

In this paper we present systems equivalent to BV where rule applications with respect to the equalities for unit are made explicit. By orienting the equalities for unit, we disallow redundant applications of inference rules. Then, in order to preserve completeness, we add structural rules that are instances of the rules of system BV. This way, resulting systems, depending on the length of the derivations, perform much better in automated proof search in our Maude implementation.

The rest of the paper is organized as follows: we first summarize the notions and notations of the calculus of structures and system BV. We then present the systems that result from removing the equalities for unit from system BV. After comparing the performance of these systems in our Maude implementation, we conclude with discussions and future work.

2 The Calculus of Structures and System BV

In this section, we shortly present the calculus of structures and the 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. Structures are considered equivalent modulo the relation $\approx$, which is the smallest congruence relation induced by the equations shown in Figure 1.[1] There $\boldsymbol{R}$, $\boldsymbol{T}$ and $\boldsymbol{U}$ stand for finite, non-empty sequence of structures. 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, no unit $\circ$ appears in it.

There is a straightforward correspondence between structures not involving seq and formulae of multiplicative linear logic (MLL). For example $[(a, b), \bar{c}, \bar{d}]$ corresponds to $((a \otimes b) \parr c^{\perp} \parr d^{\perp})$, and vice versa. Units 1 and $\perp$ are mapped into $\circ$, since $1 \equiv \perp$, when the rules mix and $\mathsf{mix0}$ are added to MLL. 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].

In the calculus of structures, an *inference rule* is a scheme of the kind $\rho \dfrac{T}{R}$, where ρ is the *name* of the rule, T is its *premise* and R is its *conclusion*. A

[1] In [6] axioms for context closure are added. However, because each equational system includes the axioms of equality context closure follows from the substitutivity axioms.

Associativity

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

Singleton

$$\langle R \rangle \approx [R] \approx (R) \approx R$$

Commutativity

$$[\boldsymbol{R}, \boldsymbol{T}] \approx [\boldsymbol{T}, \boldsymbol{R}]$$
$$(\boldsymbol{R}, \boldsymbol{T}) \approx (\boldsymbol{T}, \boldsymbol{R})$$

Units

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

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. The equational system underlying BV.

typical (deep) inference rule has the shape $\rho \, \frac{S\{T\}}{S\{R\}}$ and specifies the implication $T \Rightarrow R$ inside a generic context $S\{\ \}$, which is the implication being modeled in the system[2]. When premise and conclusion in an instance of an inference rule are equivalent, that instance is *trivial*, otherwise it is *non-trivial.* 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 $\begin{array}{c} T \\ \Delta \| \mathscr{S} \\ R \end{array}$. Similarly, $\begin{array}{c} \Pi \| \mathscr{S} \\ R \end{array}$ 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.

A rule ρ is *derivable for a system* $\mathscr{S}$ if for every instance of $\rho \, \frac{T}{R}$ there is a derivation $\begin{array}{c} T \\ \| \mathscr{S} \\ R \end{array}$. Two systems $\mathscr{S}$ and $\mathscr{S}'$ are *strongly equivalent* if for every derivation $\begin{array}{c} T \\ \Delta \| \mathscr{S} \\ R \end{array}$ there exists a derivation $\begin{array}{c} T \\ \Delta \| \mathscr{S}' \\ R \end{array}$, 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. They are *strongly (weakly) equivalent with respect to normal forms* if the above statements hold for a normal form of T.

[2] Due to duality between $T \Rightarrow R$ and $\bar{R} \Rightarrow \bar{T}$, rules come in pairs of dual rules: a down-version and an up-version. For instance, the dual of the $\mathsf{ai}{\downarrow}$ rule in Figure 2 is the cut rule. In this paper we only consider the down rules which provide a sound and complete system.

$$\circ\downarrow \frac{}{\circ} \qquad \mathsf{ai}\downarrow \frac{S\{\circ\}}{S[a,\bar{a}]} \qquad \mathsf{s}\frac{S([R,T],U)}{S[(R,U),T]} \qquad \mathsf{q}\downarrow \frac{S\langle [R,U];[T,V]\rangle}{S[\langle R;T\rangle,\langle U;V\rangle]}$$

Fig. 2. System BV

The system $\{\circ\downarrow, \mathsf{ai}\downarrow, \mathsf{s}, \mathsf{q}\downarrow\}$, shown in Figure 2, is denoted BV and called *basic system* V, where V stands for one non-commutative operator[3]. The rules of the system are called *unit* ($\circ\downarrow$), *atomic interaction* ($\mathsf{ai}\downarrow$), *switch* (s) and *seq* ($\mathsf{q}\downarrow$).

3 System BVn

The system shown in Figure 3 is called BVn. Structures on which system BVn is defined are as in the previous section, with the only difference that the equalities for unit do not apply anymore.

Proposition 1. *Every* BV *structure S can be transformed to one of its normal forms S' by applying only the rules $\{\mathsf{u}_1\downarrow, \mathsf{u}_2\downarrow, \mathsf{u}_3\downarrow, \mathsf{u}_4\downarrow\}$ in Figure 3 bottom-up and the equalities for negation in Figure 1 from left to right.*

Proof: Observe that applying the rules $\{\mathsf{u}_1\downarrow, \mathsf{u}_2\downarrow, \mathsf{u}_3\downarrow, \mathsf{u}_4\downarrow\}$ bottom up corresponds to applying the equalities for unit in Figure 1 from left to right. The result follows from the fact that the corresponding term rewriting system is terminating and confluent, and applicability of these rules contradicts with a structure being in normal form. □

Proposition 2. *The rules $\mathsf{q}_1\downarrow$, $\mathsf{q}_2\downarrow$, $\mathsf{q}_3\downarrow$, and $\mathsf{q}_4\downarrow$ are derivable for $\{\mathsf{q}\downarrow\}$. The rules s_1 and s_2 are derivable for $\{\mathsf{s}\}$.*

Proof:

- For the rule $\mathsf{q}_1\downarrow$ take the rule $\mathsf{q}\downarrow$.
- For the rule $\mathsf{q}_2\downarrow$, $\mathsf{q}_2\downarrow$, $\mathsf{q}_4\downarrow$, respectively, take the following derivations, respectively:

$$= \frac{\mathsf{q}\downarrow \dfrac{= \dfrac{\langle R;T\rangle}{\langle [R,\circ];[\circ,T]\rangle}}{[\langle R;\circ\rangle,\langle \circ;T\rangle]}}{[R,T]} \qquad = \frac{\mathsf{q}\downarrow \dfrac{= \dfrac{\langle [R,T];U\rangle}{\langle [R,T];[\circ,U]\rangle}}{[\langle R;\circ\rangle,\langle T;U\rangle]}}{[R,\langle T;U\rangle]} \qquad = \frac{\mathsf{q}\downarrow \dfrac{= \dfrac{\langle T;[R,U]\rangle}{\langle [\circ,T];[R,U]\rangle}}{[\langle \circ;R\rangle,\langle T;U\rangle]}}{[R,\langle T;U\rangle]}$$

- For the rule s_1 take the rule s.
- For the rule s_2 take the following derivation:

$$= \frac{\mathsf{s} \dfrac{= \dfrac{(R,T)}{([\circ,T],R)}}{[(\circ,R),T]}}{[R,T]} \quad .$$

□

[3] This name is due to the intuition that W stands for two non-commutative operators.

$$\circ\!\downarrow \frac{}{\circ} \qquad \mathsf{ai}\!\downarrow \frac{S\{\circ\}}{S[a,\bar{a}]} \qquad \mathsf{s}_1 \frac{S([R,T],U)}{S[(R,U),T]} \qquad \mathsf{s}_2 \frac{S(R,T)}{S[R,T]}$$

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

$$\mathsf{u}_1\!\downarrow \frac{S\{R\}}{S[R,\circ]} \qquad \mathsf{u}_2\!\downarrow \frac{S\{R\}}{S(R,\circ)} \qquad \mathsf{u}_3\!\downarrow \frac{S\{R\}}{S\langle R;\circ\rangle} \qquad \mathsf{u}_4\!\downarrow \frac{S\{R\}}{S\langle \circ;R\rangle}$$

Fig. 3. System BVn

Theorem 1. *For every derivation* $\begin{matrix} W \\ \Delta \| \mathsf{BV} \\ Q \end{matrix}$ *there exists a derivation* $\begin{matrix} W' \\ \Delta' \| \mathsf{BVn} \\ Q \end{matrix}$ *where* W' *is a normal form of the structure* W.

Proof: Observe that every derivation Δ in BV can be equivalently written as a derivation where all the structures are in normal form. Let us denote with Δ these derivations where there are only occurrences of structures in normal form. From Proposition 1 we get a normal form Q' of Q going up in a derivation. With structural induction on Δ we will construct the derivation Δ'

– If Δ is $\circ\!\downarrow \frac{}{\circ}$ then take $\Delta' = \Delta$.

– If, for an atom a, $\mathsf{ai}\!\downarrow \frac{S\{\circ\}}{S[a,\bar{a}]}$ is the last rule applied in Δ, then by Proposition 1 and by the induction hypothesis there is a derivation $\begin{matrix} W' \\ \| \mathsf{BVn} \\ T \end{matrix}$ where T is a normal form of $S\{\circ\}$. The following cases exhaust the possibilities.

- If $S[a,\bar{a}] = S'[P,[a,\bar{a}]]$ then take the following derivation.

$$\mathsf{ai}\!\downarrow \frac{\mathsf{u}_1\!\downarrow \frac{S'\{P\}}{S'[\,P\,,\,\circ\,]}}{S'[P\,,\,[a,\bar{a}]]} \qquad .$$

- If $S[a,\bar{a}] = S'(P,[a,\bar{a}])$ then take the following derivation.

$$\mathsf{ai}\!\downarrow \frac{\mathsf{u}_2\!\downarrow \frac{S'\{P\}}{S'(\,P\,,\,\circ\,)}}{S'(P\,,\,[a,\bar{a}])} \qquad .$$

- If $S[a,\bar{a}] = S'\langle P; [a,\bar{a}]\rangle$ then take the following derivation.

$$\mathsf{ai}\!\downarrow \frac{\mathsf{u}_3\!\downarrow \dfrac{S'\{P\}}{S'\langle P ; \circ \rangle}}{S'\langle P ;\, [a,\bar{a}]\rangle} \quad .$$

- If $S[a,\bar{a}] = S'\langle [a,\bar{a}]; P\rangle$ then take the following derivation.

$$\mathsf{ai}\!\downarrow \frac{\mathsf{u}_4\!\downarrow \dfrac{S'\{P\}}{S'\langle \circ ; P \rangle}}{S'\langle [a,\bar{a}]; P \rangle} \quad .$$

– If $\mathsf{s}\,\dfrac{P}{Q}$ is the last rule applied in Δ where $Q = S[(R,T),U]$ for a context S and structures R, T and U, then by induction hypothesis there is a derivation $\begin{array}{c} W' \\ \Vert \mathsf{BVn} \\ P \end{array}$.

The following cases exhaust the possibilities:

- If $R \neq \circ$, $T \neq \circ$ and $U \neq \circ$, then apply the rule s_1 to Q'.
- If $R = \circ$, $T \neq \circ$ and $U \neq \circ$ then $Q' = S'[T,U]$ where S' is a normal form of context S. Apply the rule s_2 to Q'.
- Other 6 cases are trivial instances of the s rule. Take $P = Q'$.

– If $\mathsf{q}\!\downarrow \dfrac{P}{Q}$ is the last rule applied in Δ where $Q = S[\langle R;T\rangle,\langle U;V\rangle]$ for a context S and structures R, T, U and V, then by induction hypothesis there is a derivation $\begin{array}{c} W' \\ \Vert \mathsf{BVn} \\ P \end{array}$. The following cases exhaust the possibilities:

- If $R \neq \circ$, $T \neq \circ$, $U \neq \circ$ and $V \neq \circ$, then apply the rule $\mathsf{q}_1\!\downarrow$ to Q'.
- If $R = \circ$, $T \neq \circ$, $U \neq \circ$ and $V \neq \circ$ then $Q' = S'[T,\langle U;V\rangle]$ where S' is a normal form of context S. Apply the rule $\mathsf{q}_4\!\downarrow$ to Q'.
- If $R \neq \circ$, $T = \circ$, $U \neq \circ$ and $V \neq \circ$ then $Q' = S'[R,\langle U;V\rangle]$ where S' is a normal form of context S. Apply the rule $\mathsf{q}_3\!\downarrow$ to Q'.
- If $R \neq \circ$, $T \neq \circ$, $U = \circ$ and $V \neq \circ$ then $Q' = S'[[R;T],V]$ where S' is a normal form of context S. Apply the rule $\mathsf{q}_4\!\downarrow$ to Q'.
- If $R \neq \circ$, $T \neq \circ$, $U \neq \circ$ and $V = \circ$ then $Q' = S'[\langle R;T\rangle,U]$ where S' is a normal form of context S. Apply the rule $\mathsf{q}_3\!\downarrow$ to Q'.
- If $R \neq \circ$, $T = \circ$, $U = \circ$ and $V \neq \circ$ then $Q' = S'[R,V]$ where S' is a normal form of context S. Apply the rule $\mathsf{q}_2\!\downarrow$ to Q'.
- Other 10 cases are trivial instances of the $\mathsf{q}\!\downarrow$ rule. Take $P = Q'$. □

Corollary 1. *System* BV *and system* BVn *are strongly equivalent with respect to normal forms.*

Proof: From Proposition 2 it follows that the derivations in BVn are also derivations in BV. Derivations in BV are translated to derivations in BVn by Theorem 1. □

Remark 1. From the view point of bottom-up proof search, rule s_2 is a redundant rule since the structures in a copar structure can not interact with each other. Hence, it does not make any sense to disable the interaction between two structures by applying this rule in proof search. However, in order to preserve completeness for arbitrary derivations this rule is added to the system.

4 System BVu

With the light of the above remark and observations that we made while proving Theorem 1, it is possible to improve further on the rules of system BVn: the system BVu in Figure 4, like system BVn, does not allow the application of the equalities for unit. Furthermore, in this system, we merge each one of the rules for unit $\{\mathsf{u}_1\!\downarrow, \mathsf{u}_2\!\downarrow, \mathsf{u}_3\!\downarrow, \mathsf{u}_4\!\downarrow\}$ in Figure 3 with the rule $\mathsf{ai}\!\downarrow$ since the rules for unit are used only after rule $\mathsf{ai}\!\downarrow$ is applied in a bottom-up proof search. This way we get the rules $\{\mathsf{ai}_1\!\downarrow, \mathsf{ai}_2\!\downarrow, \mathsf{ai}_3\!\downarrow, \mathsf{ai}_4\!\downarrow\}$.

$$ax\,\frac{}{[a,\bar{a}]} \qquad \mathsf{s}_1\,\frac{S([R,T],U)}{S[(R,U),T]}$$

$$\mathsf{ai}_1\!\downarrow\frac{S\{R\}}{S[R,[a,\bar{a}]]} \quad \mathsf{ai}_2\!\downarrow\frac{S\{R\}}{S(R,[a,\bar{a}])} \quad \mathsf{ai}_3\!\downarrow\frac{S\{R\}}{S\langle R;[a,\bar{a}]\rangle} \quad \mathsf{ai}_4\!\downarrow\frac{S\{R\}}{S\langle [a,\bar{a}];R\rangle}$$

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

Fig. 4. System BVu

Corollary 2. *System* BV *and system* BVu *are equivalent with respect to normal forms.*

Proof: It is immediate that the rules $\mathsf{ai}_1\!\downarrow, \mathsf{ai}_2\!\downarrow, \mathsf{ai}_3\!\downarrow, \mathsf{ai}_4\!\downarrow$ and ax are derivable (sound) for system BVn. Completeness follows from the proof of Theorem 1 and Remark 1. □

The following proposition helps to understand why BVu provides shorter proofs than BVn.

Proposition 3. *Let R be a* BV *structure in normal form with n number of positive atoms. If R has a proof in* BVn *with length k, then R has a proof in* BVu *with length $k-n$.*

Proof: (Sketch) By induction on the number of positive atoms in R, together with the observation that while going up in the proof of R in BVn, each positive atom must be annihilated with its negation by an application of the rule $\mathsf{ai}\!\downarrow$ and then the resulting structure must be transformed to a normal form by equivalently removing the unit $\circ$ with an application of one of the rules $\mathsf{u}_1\!\downarrow, \mathsf{u}_2\!\downarrow, \mathsf{u}_3\!\downarrow$ and $\mathsf{u}_4\!\downarrow$. In BVn these two steps are replaced by a single application of one of the rules $\mathsf{ai}_1\!\downarrow, \mathsf{ai}_2\!\downarrow, \mathsf{ai}_3\!\downarrow$ and $\mathsf{ai}_4\!\downarrow$. □

5 Implementation and Performance Comparison

In an implementation of the above systems, the structures must be matched modulo an equational theory. In the case of system BV this equational theory is the union of the AC1 theory for par, the AC1 theory for copar and A1 theory for seq structures, where 1 denotes the unit ∘ shared by these structures. However, in the case of BVn the equalities for unit become redundant, since their role in the rules is made explicit. This way, in contrast to the BV structures, the equivalence class of BVn structures become finite and redundant matchings of structures with rules are disabled. This results in a significant gain in the performance in automated proof search and derivation search.

In [8], we showed that systems in the calculus of structures can be expressed as term rewriting systems modulo equational theories. Exploiting the fact that the Maude System [3, 4] allows implementing term rewriting systems modulo equational theories, in [9], we presented a Maude implementation of system BV. There we also provided a general recipe for implementing systems in the calculus of structures and described the use of the relevant Maude commands. Then, we implemented the systems BVn and BVu. All these modules are available for download at `http://www.informatik.uni-leipzig.de/~ozan/maude_cos.html` .

Below is a comparison of these systems in our implementation of these systems on some examples of proof search and derivation search queries. (All the experiments below are performed on an Intel Pentium 1400 MHz Processor.)

Consider the following example taken from [2] where we search for a proof of a *process structure*.

```
search in BV : [a,[< a ; [c,- a] >,< - a ; - c >]] =>+ o .
search in BVn : [a,[< a ; [c,- a] >,< - a ; - c >]] =>+ o .
search in BVu : [a,[< a ; [c,- a] >,< - a ; - c >]] =>+ [A,- A] .
```

	finds a proof		search terminates	
	in # millisec.	after # rewrites	in # millisec.	after # rewrites
BV	1370	281669	5530	1100629
BVn	500	59734	560	65273
BVu	0	581	140	15244

When we search for the proof of a similar query which involves also copar structures we get the following results.

```
search [- c,[< a ; {c,- b} >,< - a ; b >]] => o .
```

	finds a proof		search terminates	
	in # millisec.	after # rewrites	in # millisec.	after # rewrites
BV	950	196866	1490	306179
BVn	120	12610	120	12720
BVu	10	1416	60	4691

It is also possible to search for arbitrary derivations. For instance, consider the derivation

$$\begin{array}{c}\langle d; e\rangle \\ \| \mathsf{BVn} \\ [\bar{a}, \langle a; d; \bar{b}\rangle, \langle b; e; \bar{c}\rangle, c]\end{array}$$

with the query below, which results in the table below.

```
search [ - a , [ < a ; < d ; - b > > , [ < b ; < e ; - c > >, c ]]]
                                                    =>+ < d ; e > .
```

	finds a proof		search terminates	
	in # millisec.	after # rewrites	in # millisec.	after # rewrites
BV	494030	66865734	721530	91997452
BVn	51410	4103138	51410	4103152
BVu	10090	806417	10440	822161

In all the above experiments it is important to observe that, besides the increase in the speed of search, number of rewrites performed differ dramatically between the runs of the same search query on systems BV, BVn and BVu.

6 Discussion

We presented two systems equivalent to system BV where equalities for unit become redundant. Within a Maude implementation of these systems, we also showed that, by disabling the redundant applications of the inference rules, these systems provide a better performance in automated proof search.

Our results find an immediate application for a fragment of CCS which was shown to be equivalent to BV in [2]. Furthermore, we believe that the methods presented in this paper can be analogously applied to the existing systems in the calculus of structures for classical logic [1] and linear logic [12], which are readily expressed as Maude modules.

However, termination of proof search in our implementation is a consequence of BV being a multiplicative logic. Although, the new systems presented in this paper improve the performance by making the rule applications explicit and shortening the proofs by merging rule steps, due to the exponential blow up in the search space, an implementation for practical purposes that allows "bigger structures" will require introduction of strategies at the Maude meta-level [5], in the lines of uniform proofs [11] and Guglielmi's *Splitting Theorem* [6].

System NEL [7] is a Turing-complete extension of BV [13] with the exponentials of linear logic. In [10], we employed system NEL for concurrent conjunctive planning problems. Future work includes carrying our results to NEL and linear logic systems in the calculus of structures [12].

Acknowledgments This work has been supported by the DFG Graduiertenkolleg 446. I would like to thank Alessio Guglielmi, Steffen Hölldobler, Roy Dyckhoff and anonymous referees for valuable remarks and improvements.

References

1. Kai Brünnler. *Deep Inference and Symmetry in Classical Proofs.* PhD thesis, Technische Universität Dresden, 2003.
2. 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-Verlag, 2002.
3. 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, Proceedings of the 14th International Conference,*, volume 2706. Springer, 2003.
4. M. Clavel, F. Durán, S. Eker, P. Lincoln, N. Martí-Oliet, J. Meseguer, and C. Talcott. Maude 2.1 manual. Technical report, Computer Science Laboratory, SRI International, 2004. http://maude.cs.uiuc.edu/manual/.
5. Manuel Clavel, Francisco Durán, Steven Eker, José Meseguer, and Mark-Oliver Stehr. Maude as a formal meta-tool. In Jeannette M. Wing, Jim Woodcock, and Jim Davies, editors, *FM'99 — Formal Methods, World Congress on Formal Methods in the Development of Computing Systems, Toulouse, France, September 20–24, 1999 Proceedings, Volume II*, volume 1709 of *Lecture Notes in Computer Science*, pages 1684–1703. Springer, 1999.
6. Alessio Guglielmi. A system of interaction and structure. Technical Report WV-02-10, TU Dresden, 2002. to appear in ACM Transactions on Computational Logic.
7. Alessio Guglielmi and Lutz Straßburger. A non-commutative extension of MELL. In M. Baaz and A. Voronkov, editors, *LPAR 2002*, volume 2514 of *Lecture Notes in Artificial Intelligence*, pages 231–246. Springer-Verlag, 2002.
8. Steffen Hölldobler and Ozan Kahramanoğulları. From the calculus of structures to term rewriting systems. Technical Report WV-04-03, TU Dresden, 2004.
9. Ozan Kahramanoğulları. Implementing system BV of the calculus of structures in Maude. In *Proceedings of the ESSLLI-2004 Student Session, Université Henri Poincaré, Nancy, France*. Kluwer Academic Publishers, 2004.
10. Ozan Kahramanoğulları. Plans as formulae with a non-commutative operator. Technical report, TU Dresden, 2004.
11. Dale Miller, Gopalan Nadathur, Frank Pfenning, and Andre Scedrov. Uniform proofs as a foundation for logic programming. In *Annals of Pure and Applied Logic*, volume 51, pages 125–157. 1991.
12. Lutz Straßburger. *Linear Logic and Noncommutativity in the Calculus of Structures.* PhD thesis, TU Dresden, 2003.
13. 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.
14. Alwen Fernanto Tiu. Properties of a logical system in the calculus of structures. Technical Report WV-01-06, Technische Universität Dresden, 2001.