From the Calculus of Structures to Term Rewriting Systems

Steffen Hölldobler*, Ozan Kahramanoğullari†

Abstract

The calculus of structures is a recently developed proof theoretical formalism that extends one-sided sequent calculus, with the gain of interesting proof theoretical properties. In contrast to sequent calculus, it does not rely on the notion of main connective and, like in term rewriting, permits the application of inference rules anywhere deep inside a formula. In this paper, exploiting this resemblance, we present a procedure turning derivations in the calculus of structures in four steps into rewritings in a term rewriting system modulo equality.

1 Motivation

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 first introduced by Alessio Guglielmi in [6] where it was used to express a logical system, called BV, with two associative and commutative operators and a self-dual, associative, but non-commutative operator, which is shown to be not definable in any sequent calculus system [13]. Then it turned out to yield systems with interesting and exciting properties for existing logics and new insights to proof theory (see e.g.[12, 2, 4]).

The calculus of structures can be considered as a generalization of the one-sided sequent calculus that provides a more refined analysis of proofs: structures are expressions which share properties of formulae and sequents, allowing more control over the mutual dependencies of logical relations. The two main aspects that distinguish this formalism from sequent calculus are *deep inference* and *symmetry*: inferences are symmetric between premises and conclusions and, like in term rewriting, inference rules are deeply applicable inside expressions. Whereas the

*Technische Universität Dresden, International Center for Computational Logic, sh@iccl.tu-dresden.de

†Universität Leipzig, visiting research associate at the ICCL, ozan@informatik.uni-leipzig.de

main work in the calculus of structures was developed with respect to the sequent calculus, we have always suspected that it is strongly related to rewriting with respect to a term rewriting and an equational system. The main purpose of this paper is to investigate into this relationship.

But there are additional questions. The notion of a structure is a non-standard one and people may raise concerns because a "structure" is a well-defined notion in mathematics and logic, which differs from its use in the calculus of structure. Hence, we would like to answer the question of what exactly is a structure. Last but not least the definition of a derivation in the calculus of structures is not operational and lacks precision. For example, notions like a context, a hole in a structure, an instance of an inference rule etc. are not formally defined although their meanings are intuitively obvious. Nevertheless, we wanted to give a concise definition thereof.

The paper is organized as follows: After recollecting some basic definitions in Section 2 we present system BV, a typical system in the calculus of structures, taken from [7]. We then define a four step transformation: reasoning within an equivalence class are replaced by explicit equality steps in Section 4, n-ary operators are replaced by binary ones in Section 5, the notion of a structure is replaced by the notion of a term in Section 6 and inference rules are replaced by rewriting rules in Section 8. Finally, in Section 9 we discuss possible future work.

2 Basic Definitions

> *In this section we collect basic definitions for terms, positions, replacements, substitutions, equations and rewrite rules as can be found in e.g. [1] or [10]. The reader familiar with these notions may skip this section.*

Given a set Σ of graded function symbols and a set $\mathcal{V}$ of variables with $\Sigma \cap \mathcal{V} = \varnothing$, the set $T(\Sigma, \mathcal{V})$ of all Σ-terms over $\mathcal{V}$ is defined as usual. The set of *positions* pos(s) of a term s is inductively defined as follows:

- If $s = X \in \mathcal{V}$, then $pos(s) = \{\Lambda\}$.
- If $s = f(s_1, \ldots, s_n)$ then $pos(s) = \{\Lambda\} \cup \bigcup_{i=1}^{n} \{i\pi \mid \pi \in pos(s_i)\}$.

For $\pi \in pos(s)$ the *sub-term of* s *at position* π, denoted by $s|_\pi$, is inductively defined as follows:

- $s|_\Lambda = s$.
- $f(s_1, \ldots, s_n)|_{i\pi} = s_i|_\pi$.

$s, t, \ldots$	$X, Y, \ldots$	π	$\sigma, \theta, \ldots$	$R, S, \ldots$
terms	*variables*	*positions*	*substitutions*	*structures*

Table 1: Notational Conventions.

For $\pi \in pos(s)$ the term obtained from s by *replacing the sub-term at position* π *by* t, denoted by $s|t|_\pi$, is inductively defined as follows:

- $s|t|_\Lambda = t$.
- $f(s_1, \ldots, s_n)|t|_{i\pi} = f(s_1, \ldots, s_i|t|_\pi, \ldots, s_n)$.

A *substitution* σ is a mapping from the set $\mathcal{V}$ of variables to the set $T(\Sigma, \mathcal{V})$ of Σ-terms, which is equal to the identity except for finitely many variables. Thus, σ can be represented by $\{X \mapsto \sigma(X) \mid \sigma(X) \neq X\}$. ε denotes the *empty substitution*. The *instance* of a term s wrt σ, is defined as usual.

An *equation* is an expression of the form $s \approx t$, where s and t are Σ-terms over a countably infinite set $\mathcal{V}$ of variables. An *equational system* is a set of (conditional) equations. We implicitly assume that the equational axioms, i.e., the axioms for reflexivity, symmetry, transitivity and substitutivity, are added to each equational system. Let $\approx_E$ be the finest congruence relation generated by an equational system E.

A *rewrite rule* is an expression of the form $l \to r$, where l is a non-variable term and r is a term. A *term rewriting system* is a set of rewrite rules. A *redex* is an instance of a left-hand side of a rewrite rule. Given terms s, t and a term rewriting system R, s *rewrites to* t *wrt* R, denoted by $s \to_{R(\rho,\pi,\sigma)} t$ if there is a position $\pi \in pos(s)$ and a substitution σ such that $s|_\pi = \sigma(l)$ and $t = s|\sigma(r)|_\pi$, where ρ is the rewrite rule being applied. *Contracting* a redex means replacing it by the corresponding instance of the right-hand side of the rule.

Given terms s, t, a term rewriting system R and an equational system E, s *rewrites to* t *wrt* R *and* E, denoted by $s \to_{R/E(\rho,\pi,\sigma)} t$ if there are terms s', t', a rewrite rule $\rho = l \to r$, a position $\pi \in pos(s')$ and a substitution σ such that $s \approx_E s'$, $s'|_\pi = \sigma(l)$, $t' = s'|\sigma(r)|_\pi$ and $t' \approx_E t$. In other words, $s \to_{R/E(\rho,\pi,\sigma)} t$ iff $(\exists s', t')\, s \approx_E s' \to_{R(\rho,\pi,\sigma)} t' \approx_E t$.

Throughout this paper we will make use of the notations shown in Table 1, where all symbols may be indexed.

3 System BV in the Calculus of Structures

In this section we present system BV*, the system which started the research into the calculus of structures, following [7].*

Associativity

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

Singleton

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

Units

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

Commutativity

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

Negation

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

Figure 1: The equational system underlying BV.

In BV there are infinitely many *positive atoms* and infinitely many *negative atoms*. Atoms, no matter whether positive or negative, are denoted with a, b, c, ... Structures, denoted with R, S, T, ... are 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 with a hole that do not appear in the scope of a negation are denoted as in $S\{\ \}$, and are called *structure context*.

Negation obeys the usual De Morgan laws for par and copar, it switches them. It is natural to state that $\overline{\langle S_1; \ldots; S_h \rangle} \approx \langle \overline{S_1}; \ldots; \overline{S_h} \rangle$. Negation can always be pushed inward to atoms.

Structures are considered equivalent modulo the relation $\approx$, which is the finest congruence relation defined by the equational system shown in Figure 1.[1] There $\vec{R}$, $\vec{T}$ and $\vec{U}$ stand for finite, non-empty sequence of structures. A structure, or a structure context, is said to be in normal form when the only negated structures appearing in it are atoms, no unit $\circ$ appears in it and no parentheses can be equivalently eliminated.

All structures can equivalently be considered in normal form, because negations

[1] In [7] axioms for context closure are added. However, because each equational system includes the axioms of equality context closure follows from the substitutivity axioms.

can always be pushed in-wards to atoms by using the negation axioms, and units can be removed, as well as extra parentheses (by associativity and singleton laws).

An *inference rule* is a scheme of the kind

$$\frac{V}{U}\ \rho \quad ,$$

where ρ is the name of the rule, V is its *premise* and U is its conclusion; rule names are denoted with ρ. In an inference rule, either the premise or the conclusion can be missing, but not both. A (*formal*) *system* is a set of inference rules, formal systems are denoted with $\mathscr{S}$. A *derivation* in a certain formal system $\mathscr{S}$ is either a structure or a finite chain of instances of inference rules in $\mathscr{S}$; derivations are denoted with Δ. The topmost structure in a derivation, if present, is called the *premise* of the derivation; if present, the bottommost structure is called its *conclusion*. A derivation Δ whose premise is T, conclusion is R, and whose inference rules are in $\mathscr{S}$ is indicated by

$$\begin{array}{c} T \\ \Delta \Vdash \mathscr{S} \\ R \end{array}$$

(the name Δ can be omitted). The *length* of a derivation is the number of instances of inference rules appearing in it.

In this paper we consider only analytical systems, where the conclusion is the starting point of a derivation and inference rules are used to reach the desired premises. This analytical way of looking at systems is called the *bottom-up view* in [7].

The following (logical axiom) rule $\circ\downarrow$ is called *unit*:

$$\frac{}{\circ}\ \circ\downarrow \qquad .$$

A *proof* is a derivation whose topmost inference rule is the unit rule. Proofs are denoted with Π. A formal system $\mathscr{S}$ *proves* R if there is in $\mathscr{S}$ a proof Π whose conclusion is R, written

$$\begin{array}{c} \Pi \\ \Vdash \mathscr{S} \\ R \end{array}$$

(the name Π can be omitted).

The system $\{\circ\downarrow, \mathrm{a\!\downarrow}, \mathrm{q\!\downarrow}, \mathrm{s}\}$, shown in Figure 2, is denoted BV and called *basic system* V, where V stands for one non-commutative operator[2]. We consider $\mathrm{a\!\downarrow}$ to be a schema for all positive atoms a.

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

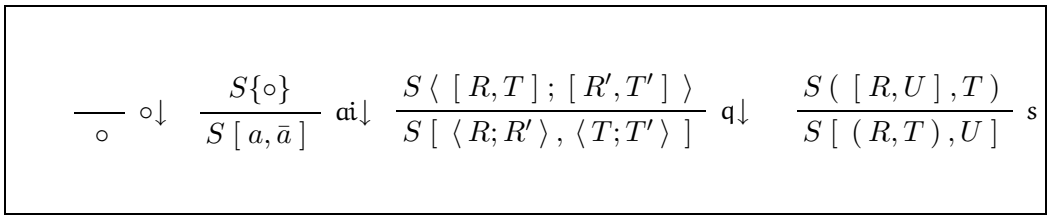

Figure 2: System BV.

As an example consider the following proof in BV:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\circ}{[c,\bar{c}]}\,\mathrm{ai}\downarrow}{[(c,[\bar{b},b]),\bar{c}]}\,\mathrm{ai}\downarrow}{[[b,(c,\bar{b})],\bar{c}]}\,\mathrm{s}}{[\langle[\bar{a},a];[b,(\bar{b},c)]\rangle,\bar{c}]}\,\mathrm{ai}\downarrow}{[\langle\bar{a};b\rangle,\langle a;(\bar{b},c)\rangle,\bar{c}]}\,\mathrm{q}\downarrow$$

We have stated here the definitions as they are given in [7], where the only modifications were linguistic or graphical ones with the exception of the replacement of the axioms for context closure by the axioms of susbstitutivity. Some remarks:

1. In the literature the notion of positive and negative atoms is often replaced by the notion of (positive and negative) literals.

2. Inference rules consist of structures over atoms and variables, however, these structures are not formally defined in [7]. Likewise, the notion of an instance of an inference rule is undefined; this is a non-standard notion because the context variable S appearing in the inference rules of system BV needs also to be instantiated, and this cannot be done by a standard substitution.

3. The notion of a derivation is also not operational because it does not specify a procedure for computing a possible extension of a derivation. One should also observe that the definition of a derivation does not require that structures are in normal form.

4. One should also keep in mind, that holes in structure are not within the scope of negation.

5. The notion of a topmost/bottommost structure of a derivation is undefined.

4 Replacing Equivalence Classes by Equality Steps

> *Each derivation step between two equivalence classes in* BV *is split into (1) an equality step leading to a new representative of the first equivalence class, (2) an application of an inference rule to this representative and (3) an equality step leading to a new representative of the second equivalence class.*

In a first step to-wards an operational definition of a derivation and, in particular, in the search for an operational definition for the application of an inference rule, we have to make precise the role played by the syntactic equalities in a derivation. A second motivation is the fact that deduction are difficult to read because structures represent equivalence classes.

The step taken in this section is to separate the notion of a structure from the equivalence class defined by the equations shown in Figure 1. Hence, from this point on, a structure is a structure and no longer an equivalence class of structures. Let $\approx$ be the finest congruence relation generated by the equations shown in Figure 1. We can now reformulate derivations as being sequences of equality and inference steps:

A structure R is a *derivation from* R *to* R. If Δ is a derivation from structure R to structure T, $T \approx T'$, there is an instance of an inference rule ρ with conclusion T' and premise Q', and $Q' \approx Q$ then

$$\begin{array}{l} Q \\ \hline Q' \\ \hline T' \\ \hline T \\ \| \Delta \\ R \end{array} \begin{array}{l} \approx \\ \rho \\ \approx \end{array}$$

is a *derivation from* R *to* Q. For notational convenience we combine two subsequent equality steps occurring in a derivation into a single equality step.

The notion of a proof can be analogously redefined. If Δ is a derivation from R to T and $T \approx \circ$, then

$$\begin{array}{l} \\ \hline \circ \\ \hline T \\ \| \Delta \\ R \end{array} \begin{array}{l} \circ\downarrow \\ \approx \\ \end{array}$$

is a *proof* of R. As an example consider the following proof which corresponds

to the proof depicted in Section 3:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\circ}\circ\downarrow}{\circ}\approx}{\circ}}{[c,\bar{c}]}\mathrm{ai}\downarrow}{[(\circ,c),\bar{c}]}\approx}{[([b,\bar{b}],c),\bar{c}]}\mathrm{ai}\downarrow}{[([\bar{b},b],c),\bar{c}]}\approx}{[[(\bar{b},c),b],\bar{c}]}\mathrm{s}}{[\langle\circ;[b,(\bar{b},c)]\rangle,\bar{c}]}\approx}{[\langle[a,\bar{a}];[b,(\bar{b},c)]\rangle,\bar{c}]}\mathrm{ai}\downarrow}{[\langle[\bar{a},a];[b,(\bar{b},c)]\rangle,\bar{c}]}\approx}{[[\langle\bar{a};b\rangle,\langle a;(\bar{b},c)\rangle],\bar{c}]}\mathrm{q}\downarrow}{[\langle\bar{a};b\rangle,[\langle a;(\bar{b},c)\rangle,\bar{c}]]}\approx$$

One should observe that not all structures occurring in the previous proof are in normal form. Moreover, we still have not defined the notion of an instance of an inference rule, but rely on the readers intuition. As before we assume that holes in a structure are not within a negation.

Because $\approx$ is the finest congruence relation generated by the equational system shown in Figure 1, each derivation and each proof as defined in Section 3 can be transformed into a derivation and a proof as defined in this section, respectively. We have thus clarified the role of the equational theory underlying derivations in BV. The same kind of changes to BV have already been considered in [2].

5 Replacing n-ary Operators by binary Ones

> *The n-ary operators par, copar and seq are replaced by their binary counterparts.*

The use of a family of operators, one for each arity, might be helpful in presenting structures, derivations and proofs. However, from a technical point of view binary operators suffice and restricting ourselves to those may simplify proofs. The following recursive transformation turns each structure into a structure, where only

the binary operators $\langle \_ ; \_ \rangle$, $(\_, \_)$ and $[\_, \_]$ are used:

$$n22(S) = \begin{cases} \circ & \text{if } S = \circ, \\ S & \text{if } S \text{ is a positive atom}, \\ \overline{n22(R)} & \text{if } S = \overline{R}, \\ \langle n22(R); n22(\vec{T}) \rangle & \text{if } S = \langle R; \vec{T} \rangle, \\ (n22(R), n22(\vec{T})) & \text{if } S = (R, \vec{T}), \\ [n22(R), n22(\vec{T})] & \text{if } S = [R, \vec{T}]. \end{cases}$$

As an example consider:

$$\begin{array}{ll} & n22(\,[\langle \bar{a}; b\rangle, \langle a; (\bar{b}, c)\rangle, \bar{c}]\,) \\ = & [\,n22(\,\langle \bar{a}; b\rangle\,),\, n22(\,[\langle a; (\bar{b}, c)\rangle, \bar{c}]\,)\,] \\ = & [\,\langle n22(\,\bar{a}\,); n22(\,b\,)\rangle,\, [\,n22(\,\langle a; (\bar{b}, c)\rangle\,)\,,\, n22(\,\bar{c}\,)\,]\,] \\ = & [\,\langle\, \overline{n22(\,a\,)}\,; b\rangle,\, [\langle\, n22(\,a\,); n22(\,(\bar{b}, c)\,)\,\rangle,\, n22(\,\bar{c}\,)\,]\,] \\ = & [\,\langle \bar{a}\,; b\rangle,\, [\langle a; (n22(\,\bar{b}\,)\,,\, n22(\,c\,)\,)\,\rangle, \overline{n22(\,c\,)}\,]\,] \\ = & [\,\langle \bar{a}\,; b\rangle,\, [\langle a; (\overline{n22(\,b\,)}\,, c)\,\rangle, \bar{c}]\,] \\ = & [\,\langle \bar{a}\,; b\rangle,\, [\langle a; (\bar{b}\,, c)\,\rangle, \bar{c}]\,]. \end{array}$$

As a consequence, we can also simplify the equations defining the syntactic equivalence leading to a refined set of equations as shown in Figure 3. Let $\approx_{\mathsf{EBV}}$ denote the finest congruence relation generated by this equational theory. Because

$$\begin{array}{rcll} n22(\langle R\rangle) & = & \langle R; \circ\rangle & \approx_{\mathsf{EBV}} \; R, \\ n22([R]) & = & [R, \circ] & \approx_{\mathsf{EBV}} \; R, \\ n22((R)) & = & (R, \circ) & \approx_{\mathsf{EBV}} \; R \end{array}$$

we do not need the equations for singletons (see Figure 1) any more.

Because the inference rules for BV (see Figure 2) use only binary seq-, par- and copar-operators, there is no need to change them.

Because $n22(S) \approx S$ derivations and proofs wrt n-ary seq-, par- and copar-operators can be equivalently turned into derivations and proofs using only binary seq, par- and copar-operators and vice versa. This may lead to less intelligible structures, but the n-ary operators may be reintroduced as abbreviations (see e.g. [5, 9]).

6 Replacing Structures by Terms

We replace the notion of a structure by the notion of a term, and consider terms over variables, thus formalizing the concept of structures

Associativity

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

Commutativity

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

Units

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

Negation

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

Figure 3: The equational system EBV.

with variable occurrences.

The notion of a structure is non-standard one. Let

$$\Sigma_{\mathsf{BV}} = \{\circ,\ \bar{\_},\ [\,\_,\_\,],\ (\,\_,\_\,),\ \langle\,\_;\_\,\rangle\,\} \cup \{a \mid a \text{ is a positive atom}\}.$$

Then, structures as defined in Section 3 are simply Σ_{BV}-terms over the empty set of variables, i.e., ground Σ_{BV}-terms. On the other hand, by considering a non-empty set $\mathcal{V}$ of variables, we obtain Σ_{BV}-terms over $\mathcal{V}$, which correspond to structures with variables.

From now on we use the notions structure and Σ_{BV}-term synonymously.

7 Replacing Context by Positions

We replace the notion of context in derivations within BV *by the notion of a position, thus being precise about which substructure or sub-term is replaced in a derivation step.*

As structures are nothing but terms the notions introduced in Section 2 can be applied. For example, let $s = [\,[\,(\bar{b},c\,),b\,],\bar{c}\,]$ and $t = (\,[\,\bar{b},b\,],c\,)$ then

$$pos(s) = \{\Lambda,\ 1,\ 11,\ 111,\ 1111,\ 112,\ 12,\ 2,\ 21\}$$

and

$$s|t|_1 = [\,(\,[\,\bar{b},b\,],c\,),\bar{c}\,].$$

Thus, the notion of positions, sub-terms and the replacement of a sub-term by another one at a particular position take over the role of a context in BV.

8 Replacing Inference Rules by Rewrite Rules

We define the term rewriting system RBV *and the equational theory* EBV *corresponding to BV such that derivations in BV correspond to rewritings* $\rightarrow_{\mathsf{RBV}/\mathsf{EBV}}$ *.*

In the final step the context occurring in inference rules is eliminated and inference rules are turned into rewrite rules. Each inference rule occurring in BV as shown in Figure 2 except $\circ\downarrow$ is turned into a rewrite rules as shown in Figure 4 by dropping the context S. As before, $\mathfrak{ai}\downarrow$ is a schema for all positive atoms a. We allow the rewrite rules to be applied only to those terms, which are not sub-terms of the function symbol "$^-$". This restriction corresponds to the counterpart in the calculus of structures which states that stucture contexts are not under the scope of negation. We can then compute rewrite sequences as follows:

$$
\begin{array}{ll}
[\langle \bar{a}; b\rangle, [\langle a; (\bar{b}, c)\rangle, \bar{c}]] & \\
\approx_{\mathsf{EBV}} & [[\langle \bar{a}; b\rangle, \langle a; (\bar{b}, c)\rangle], \bar{c}] \\
\rightarrow_{\mathsf{RBV}(\mathfrak{q}\downarrow,1,\{R\mapsto\bar{a},\, R'\mapsto b,\, T\mapsto a,\, T'\mapsto(\bar{b},c)\})} & [\langle [\bar{a}, a]; [b, (\bar{b}, c)]\rangle, \bar{c}] \\
\approx_{\mathsf{EBV}} & [\langle [a, \bar{a}]; [b, (\bar{b}, c)]\rangle, \bar{c}] \\
\rightarrow_{\mathsf{RBV}(\mathfrak{ai}\downarrow,11,\varepsilon)} & [\langle \circ; [b, (\bar{b}, c)]\rangle, \bar{c}] \\
\approx_{\mathsf{EBV}} & [[(\bar{b}, c), b], \bar{c}] \\
\rightarrow_{\mathsf{RBV}(\mathfrak{s},1,\{R\mapsto\bar{b},\, T\mapsto c,\, U\mapsto b\})} & [([\bar{b}, b], c), \bar{c}] \\
\approx_{\mathsf{EBV}} & [([b, \bar{b}], c), \bar{c}] \\
\rightarrow_{\mathsf{RBV}(\mathfrak{ai}\downarrow,11,\varepsilon)} & [(\circ, c), \bar{c}] \\
\approx_{\mathsf{EBV}} & [c, \bar{c}] \\
\rightarrow_{\mathsf{RBV}(\mathfrak{ai}\downarrow,\Lambda,\varepsilon)} & \circ
\end{array}
$$

This rewrite sequence corresponds precisely to the proof given in Section 4.

$$
\begin{array}{lcll}
[a, \bar{a}] & \rightarrow & \circ & \mathfrak{ai}\downarrow \\
[\langle R; R'\rangle, \langle T; T'\rangle] & \rightarrow & \langle [R, T]; [R', T']\rangle & \mathfrak{q}\downarrow \\
[(R, T), U] & \rightarrow & ([R, U], T) & \mathfrak{s}
\end{array}
$$

Figure 4: The rewrite system RBV corresponding to BV.

Proposition 1 *Let* s *and* t *be two* Σ_{BV}*-terms or structures.*

1. *There is a derivation in* BV *from* s *to* t *having length* n *iff there exists a rewriting* $s \overset{n}{\rightarrow}_{\mathsf{RBV/EBV}} t$.

2. *There is a proof of* s *in* BV *having length* n *iff there exists a rewriting* $s \overset{n}{\rightarrow}_{\mathsf{RBV/EBV}} \circ$.

Proof (sketch) The proof of 1. is by induction on the length of the derivation in BV and the number of rewrite steps in $\mathsf{RBV/EBV}$, respectively, for the if part and the only if part, respectively, and follows immediately from the discussion in this an the previous sections. 2. follows immediately from 1. □

One should observe that in the rewritings generated by this proposition correspond one to one to the inference steps in derivations of BV. Because in the latter derivations no holes in structures are within the scope of negation no contracted redex in the former occurs within the scope of a negation sign.

9 Outlook

In this paper we have shown that structures are nothing but Σ_{BV}-terms, where

$$\Sigma_{\mathsf{BV}} = \{\circ,\ \overline{\ \_\ },\ [\ \_,\_\ ],\ (\ \_,\_\ ),\ \langle\ \_;\_\ \rangle\ \} \cup \{a \mid a \text{ is a positive atom}\}.$$

Moreover, we have shown that derivations and proofs in the BV can be replaced by rewritings with respect to RBV and EBV. The fact that in the latter only binary par-, copar- and seq-operators are allowed is not a restriction because the n-ary ones can be reintroduced as abbreviations.

9.1 Other Systems

We claim that the procedure described in this paper does not only apply to BV but to any system in the calculus of structures. Some examples are shown in the following.

System KS A system is local if the application of each inference rule consumes a bounded amount of computational resources, i.e., time and space. System KS is a local system for classical propositional logic in the calculus of structures. It was first introduced in [3]. There, structures are Σ_{KS}-terms, where

$$\Sigma_{\mathsf{KS}} = \{\mathsf{t},\ \mathsf{f},\ \overline{\ \_\ },\ [\ \_,\_\ ],\ (\ \_,\_\ )\ \} \cup \{a \mid a \text{ is a positive atom}\}.$$

Figures 5 and 6 show the equational system EKS and the term rewriting system RKS corresponding to KS, where the rules $\mathsf{a}i{\downarrow}$, $\mathsf{a}w{\downarrow}$ and $\mathsf{a}c{\downarrow}$ are schemas.

System LS System LS is a system for linear logic in the calculus of structures. It was first introduced in [11] where also a local version is presented. There, structures are Σ_{LS}-terms, where

$$\begin{aligned}\Sigma_{\mathsf{LS}} &= \{1,\ 0,\ \top,\ \bot,\ ?\_,\ !\_,\ \overline{\ \ },\ [\_,\_],\ (\_,\_),\ \{\_,\_\},\ \langle\_,\_\rangle\} \\ &\cup \{a \mid a \text{ is a positive atom}\}.\end{aligned}$$

Figures 7 and 8 show the equational system ELS and the term rewriting system RLS corresponding to LS, where the rule $\mathsf{ai}{\downarrow}$ is a schema.

System NEL System NEL is a non-commutative extension of MELL in the calculus of structures. It was first introduced in [8]. There, structures are Σ_{NEL}-terms, where

$$\Sigma_{\mathsf{NEL}} = \{\circ,\ ?\_,\ !\_,\ \overline{\ \ },\ [\,\_,\_\,],\ (\,\_,\_\,),\ \langle\,\_;\_\,\rangle\,\} \cup \{a \mid a \text{ is a positive atom}\}.$$

Figures 9 and 10 show the equational system ENEL and the term rewriting system RNEL corresponding to NEL, where the rule $\mathsf{ai}{\downarrow}$ is a schema.

9.2 Open Problems

The procedure we presented may be the starting point for further investigations: Is it possible to replace $\rightarrow_{\mathsf{RBV}/EBV}$ by $\rightarrow_{RBV,EBV}$? We expect that this can be achieved by considering critical pairs between RBV and EBV and introducing corresponding additional rewriting rules.

In this paper, we have only considered the deep rules, that is, the rules that can be applied in an arbitrary context. However, in the calculus of structures, it is possible to define rules which are not deep, i.e., shallow rules. As an example consider the shallow instance of the s rule:

$$\frac{(\ [\,R,U\,]\,,T\,)}{[\ (\,R,T\,)\,,U\,]}\ s'$$

Such a rule corresponds to a rewrite rule which can only be applied at the root position. i.e., position Λ. We can express this restriction by introducing a unary function symbol, say "$str\_$", which is the outer most function symbol of a structure. Then the above rule can be expressed as the following rewrite rule.

$$str\,[(R,T),U] \quad \rightarrow \quad str\,([R,U],T) \qquad\qquad s'$$

In the definition of normal form of a structure, the requirement that the only negated structures appearing in it are atoms, is a standard one corresponding to the negation normal form of formulas. The advantage of considering the negation normal form of a formula is that the syntactic equivalences concerning negation

can be removed provided that each application of an inference rule yields again a formula in negation normal form. An inspection of system BV shows that in an application of an inference rule in an analytic way, i.e., from the conclusion to the premise, no new negation signs are introduced. Moreover, because holes in a structure do not appear in the scope of a negation sign, the property that negated structures appear only in atoms is preserved by the application of an inference rule.

This observation points out the possibility of orienting the equalities for negation as rewrite rules from left to right, aiming at the negation normal form of the structures at the very beginning, which will be preserved all through the proof search. Furthermore, we conjecture that by orienting the equalities for unit from left to right as rewrite rules the completeness of the system will not be lost, and this will make it possible to remove the units that appear after the application of the rule $\mathsf{ai}{\downarrow}$.

References

[1] Franz Baader and Tobias Nipkow. *Term Rewriting and All That*, volume 1. Cambridge University Press, 1998.

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

[3] Kai Brünnler and Alwen Fernanto Tiu. A local system for classical logic. In R. Nieuwenhuis and A. Voronkov, editors, *LPAR 2001*, volume 2250 of *Lecture Notes in Artificial Intelligence*, pages 347–361. Springer-Verlag, 2001.

[4] 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.

[5] Melvin C. Fitting. *First–Order Logic and Automated Theorem Proving*. Springer, Berlin, 2nd edition, 1996.

[6] Alessio Guglielmi. A calculus of order and interaction. Technical Report WV-99-04, TU Dresden, 1999.

[7] Alessio Guglielmi. A system of interaction and structure. Technical Report WV-02-10, TU Dresden, 2002. to appear in ACM Transactions on Computational Logic.

[8] 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.

[9] Steffen Hölldobler. *Logik und Logikprogrammierung.* Synchron Publishers GmbH, second, extended edition, 2001.

[10] David A. Plaisted. Equational reasoning and term rewriting systems. In Dov Gabbay, Christopher Hogger, and J. A. Robinson, editors, *The Handbook of Logic in Artificial Intelligence and Logic Programming, Volume 1: Deductive Methodologies*, pages 274–367. Oxford University Press, Oxford, 1993.

[11] Lutz Straßburger. A local system for linear logic. In Matthias Baaz and Andrei Voronkov, editors, *Logic for Programming, Artificial Intelligence, and Reasoning, LPAR 2002*, volume 2514 of *LNAI*, pages 388–402. Springer-Verlag, 2002.

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

[13] Alwen Fernanto Tiu. Properties of a logical system in the calculus of structures. Technical Report WV-01-06, Technische Universität Dresden, 2001.

Associativity

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

Commutativity

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

Units

$$(\mathsf{f},\mathsf{f}) \approx \mathsf{f} \qquad [\mathsf{f},R] \approx R$$
$$[\mathsf{t},\mathsf{t}] \approx \mathsf{t} \qquad (\mathsf{t},R) \approx R$$

Negation

$$\overline{\mathsf{f}} \approx \mathsf{t}$$
$$\overline{\mathsf{t}} \approx \mathsf{f}$$
$$\overline{[R,T]} \approx (\overline{R},\overline{T})$$
$$\overline{(R,T)} \approx [\overline{R},\overline{T}]$$
$$\overline{\overline{R}} \approx R$$

Figure 5: The equational system EKS .

$[a,\bar{a}]$	$\rightarrow$	t	ai$\downarrow$
$[(R,T),U]$	$\rightarrow$	$([R,U],T)$	s
$([R,T],[U,V])$	$\rightarrow$	$[(R,U),(T,V)]$	m
a	$\rightarrow$	f	aw$\downarrow$
a	$\rightarrow$	$[a,a]$	ac$\downarrow$

Figure 6: The rewrite system RKS corresponding to KS .

Associativity

$$[R,[T,U]] \approx [[R,T],U]$$
$$(R,(T,U)) \approx ((R,T),U)$$
$$⟅R,⟅T,U⟆⟆ \approx ⟅⟅R,T⟆,U⟆$$
$$⟨R,⟨T,U⟩⟩ \approx ⟨⟨R,T⟩,U⟩$$

Commutativity

$$[R,T] \approx [T,R]$$
$$(R,T) \approx (T,R)$$
$$⟅R,T⟆ \approx ⟅T,R⟆$$
$$⟨R,T⟩ \approx ⟨T,R⟩$$

Units

$$[\bot,R] \approx R \qquad (1,R) \approx R$$
$$⟅0,R⟆ \approx R \qquad ⟨\top,R⟩ \approx R$$

$$⟅\bot,\bot⟆ \;\approx\; \bot \;\approx\; ?\bot$$
$$⟨1,1⟩ \;\approx\; 1 \;\approx\; !1$$

Negation

$$\overline{[R,T]} \approx (\overline{R},\overline{T})$$
$$\overline{(R,T)} \approx [\overline{R},\overline{T}]$$
$$\overline{⟅R,T⟆} \approx ⟨\overline{R},\overline{T}⟩$$
$$\overline{⟨R,T⟩} \approx ⟅\overline{R},\overline{T}⟆$$
$$\overline{?R} \;\approx\; !\overline{R}$$
$$\overline{!R} \;\approx\; ?\overline{R}$$
$$\overline{\overline{R}} \approx R$$

Exponentials

$$??R \;\approx\; R$$
$$!!R \;\approx\; R$$

Figure 7: The equational system ELS .

$[a,\bar{a}]$	$\to$	1	ai↓
$[(R,T),U]$	$\to$	$([R,U],T)$	s
$[⟨R,T⟩,⟅U,V⟆]$	$\to$	$⟨[R,U],[T,V]⟩$	d↓
R	$\to$	0	t↓
R	$\to$	$⟅R,R⟆$	c↓
$[!R,?T]$	$\to$	$![R,T]$	p↓
$?R$	$\to$	$\bot$	w↓
$?R$	$\to$	$[?R,R]$	b↓

Figure 8: The rewrite system RLS corresponding to LS .

Associativity

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

Commutativity

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

Units

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

Negation

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

Exponentials

$$?\circ \approx !\circ \approx \circ$$
$$??R \approx ?R$$
$$!!R \approx !R$$

Figure 9: The equational system ENEL .

$[a,\bar{a}]$	$\rightarrow$	$\circ$	$\mathfrak{ai}{\downarrow}$
$[(R,T),U]$	$\rightarrow$	$([R,U],T)$	$\mathfrak{s}$
$[\langle R;T\rangle,\langle U;V\rangle]$	$\rightarrow$	$\langle [R,U];[T,V]\rangle$	$\mathfrak{q}{\downarrow}$
$[!R,?T]$	$\rightarrow$	$![R,T]$	$\mathfrak{p}{\downarrow}$
$?R$	$\rightarrow$	$\circ$	$\mathfrak{w}{\downarrow}$
$?R$	$\rightarrow$	$[?R,R]$	$\mathfrak{b}{\downarrow}$

Figure 10: The rewrite system RNEL corresponding to NEL .