Available online at www.sciencedirect.com
SCIENCE DIRECT
Information Processing Letters 93 (2005) 217–223
Information Processing Letters
www.elsevier.com/locate/ipl

A syntactic commutativity format for SOS

MohammadReza Mousavi *, Michel Reniers, Jan Friso Groote

Department of Computer Science, Eindhoven University of Technology, P.O. Box 513, NL-5600 MB Eindhoven, The Netherlands

Received 14 September 2004; received in revised form 21 November 2004
Available online 16 December 2004
Communicated by J.L. Fiadeiro

Abstract

Considering operators defined using Structural Operational Semantics (SOS), commutativity axioms are intuitive properties that hold for many of them. Proving this intuition is usually a laborious task, requiring several pages of boring and standard proof. To save this effort, we propose a syntactic SOS format which guarantees commutativity for a set of composition operators.

Keywords: Formal semantics; Structural Operational Semantics (SOS); Standard SOS formats; Commutativity

1. Introduction

Structural Operational Semantics [1] has become a de facto standard in defining operational semantics for specification and programming languages. Hence, developing meta-theorems for SOS specifications can be beneficial to a large community of researchers in different areas of computer science and can save them a lot of repetitive effort in proving theorems about their theories. Congruence formats for different notions of equality are the best known examples of such meta-theorems (see [2] for an overview) which guarantee a particular notion of equality to be a congruence provided that the SOS rules in the specification conform to a certain syntactic format. Deriving algebraic axioms for SOS rules in [3–6] are other examples in this direction which try to generate a set of sound and complete axioms for a given operational semantics in a syntactic format. Although commutativity axioms are derivable from the set of axioms generated by [3–6], none of the approaches generate commutativity axioms explicitly and furthermore, they assume the existence of a number of standard constants and operators in the signature.

In this paper, we aim at developing a meta-theorem for deriving commutativity axioms for certain operators in an SOS specification. Our format does not assume the presence of any special operator and builds upon a general congruence format, namely tyft [7]. The ultimate goal of this line of research is to develop the necessary theoretical background for a toolset that can assist specifiers in developing Structural

* Corresponding author.
E-mail address: m.r.mousavi@tue.nl (M.R. Mousavi).

0020-0190/$ – see front matter
doi:10.1016/j.ipl.2004.11.007

Operational Semantics for their languages, by proving different properties for the developed languages automatically.

The rest of this paper is organized as follows. In Section 2, we start by presenting some preliminary notions about (Structural) Operational Semantics, congruence, standard congruence formats and commutativity. Then, in Section 3, we give our proposal for a syntactic format for commutativity called comm-tyft (for commutative tyft). Section 4 addresses possible extensions of this format by adding tyxt rules, predicates and negative premises to the format (thus, achieving the expressivity of PANTH format [8]). Finally, Section 5 summarizes the results and presents concluding remarks.

2. Preliminaries

2.1. Transition system specification

Definition 1 (*Signature and terms*). We assume an infinite set of variables V (with typical members $x, y, x', y', x_i, y_i, \ldots$). A *signature* Σ is a set of function symbols (operators) with fixed arities. Functions with zero arity are called constants. A term $t \in T(\Sigma)$ is defined inductively as follows: a variable $x \in V$ is a term, if $t_0, \ldots, t_{n-1}$ are terms then for all $f \in \Sigma$ with arity n, $f(t_0, \ldots, t_{n-1})$ is a term, as well (i.e., constants are indeed terms). Terms are typically denoted by $t, t', t_i, t'_i, \ldots$. All terms are considered open terms. Closed terms $C(\Sigma)$ are terms that do not contain any variable and are typically denoted by $p, q, p', q', p_i, q_i, p'_i, q'_i, \ldots$. A substitution σ replaces variables in a term with other terms. The set of variables appearing in term t is denoted by $vars(t)$.

Definition 2 (*Transition System Specification* (*TSS*)). A *transition system specification* is a tuple (Σ, L, Rel, D) where Σ is a signature, L is a set of labels (with typical members $l, l', l_i, \ldots$), Rel is a set of transition relation symbols, and D is a set of deduction rules, where for all $r \in Rel$, $l \in L$ and $t, t' \in T(\Sigma)$ we define that $(t, t') \in \xrightarrow{l}_r$ is a formula. A deduction rule $dr \in D$, is defined as a tuple (H, c) where H is a set of formulae and c is a formula. The formula c is called the *conclusion* and the formulae from H are called *premises*.

Notions of open and closed extend to formulae as expected. Also, the concept of substitution is lifted to formulae and sets of formulae in the natural way (i.e., a substitution applied to a formula, applies to both terms). A formula $(t, t') \in \xrightarrow{l}_r$ is denoted by the more intuitive notation $t \xrightarrow{l}_r t'$, as well. We refer to t as the source and to t' as the target of the transition.

A deduction rule (H, c) is mostly denoted by $\frac{H}{c}$.

Definition 3. A *proof* of a closed formula ϕ is a well-founded upwardly branching tree whose nodes are labeled by closed formulae such that

- the root node is labeled by ϕ, and
- if ψ is the label of a node and $\{\psi_i \mid i \in I\}$ is the set of labels of the nodes directly above this node, then there are a deduction rule $\frac{\{\chi_i \mid i \in I\}}{\chi}$ and a substitution σ such that $\sigma(\chi) = \psi$, and for all $i \in I$, $\sigma(\chi_i) = \psi_i$.

2.2. Bisimulation, congruence and standard formats

Definition 4 (*Bisimulation* [9]). A relation $R \subseteq C(\Sigma) \times C(\Sigma)$ is a *bisimulation* relation if and only if $\forall_{p,q,l,r}$ $(p, q) \in R \Rightarrow$

(1) $\forall_{p'}\ p \xrightarrow{l}_r p' \Rightarrow \exists_{q'}\ q \xrightarrow{l}_r q' \wedge (p', q') \in R$;
(2) $\forall_{q'}\ q \xrightarrow{l}_r q' \Rightarrow \exists_{p'}\ p \xrightarrow{l}_r p' \wedge (p', q') \in R$.

Two closed terms p and q are *bisimilar*, denoted by $p \underline{\leftrightarrow} q$, if and only if there exists a bisimulation relation R such that $(p, q) \in R$.

Definition 5 (*Congruence*). A relation $R \subseteq T(\Sigma) \times T(\Sigma)$ is a congruence relation with respect to an n-ary function symbol $f \in \Sigma$ if and only if it is an equivalence relation and for all terms $p_i, q_i \in T(\Sigma)$, if $(p_i, q_i) \in R$ $(0 \leqslant i < n)$ then $(f(p_0, \ldots, p_{n-1}), f(q_0, \ldots, q_{n-1})) \in R$. Furthermore, R is called a *congruence* for a transition system specification if and only if it is a congruence with respect to all function symbols of the signature.

Definition 6 (Tyft *format* [7]). A deduction rule is in tyft format if and only if it has the following form

$$\frac{\{t_i \xrightarrow{l_i}_{r_i} y_i \mid i \in I\}}{f(x_0, \ldots, x_{n-1}) \xrightarrow{l}_r t},$$