The Tyft/Tyxt Format Reduces to Tree Rules

Willem Jan Fokkink

CWI
P.O. Box 94079, 1090 GB Amsterdam, The Netherlands
e-mail: waj@cwi.nl

Abstract. Groote and Vaandrager [5] introduced the tyft/tyxt format for transition system specifications (TSSs), and established that for each TSS in this format that is well-founded, the strong bisimulation it induces is a congruence. In this paper, we construct for each TSS in tyft/tyxt format an equivalent TSS that consists of tree rules only. As a corollary we can give an affirmative answer to an open question, namely whether the well-foundedness condition in the Congruence Theorem of [5] can be dropped. These results extend to tyft/tyxt with negative premises and predicates.

1 Introduction

A current method to provide programming and specification languages with an operational semantics is based on the use of transition systems, advocated by Plotkin [7]. Given a set of states, the transitions between these states are obtained inductively from a transition system specification (TSS), containing transition rules. Such a rule, together with a number of transitions, may imply the validity of another transition.

We will consider a specific type of transition systems, in which states are the closed terms generated by a single sorted signature, and transitions are supplied with labels. A great deal of the operational semantics of formal languages in Plotkin style that have been defined over the years, are within the scope of this format.

To distinguish such labelled transition systems, many different equivalences have been defined, the finest of which is the strong bisimulation equivalence of Park [6]. In general, this equivalence is not a congruence, i.e. the equivalence class of a term $f(p_1, ..., p_m)$ modulo strong bisimulation is not always determined by the equivalence classes of the terms $p_i$. However, congruence is an essential property, for instance, to fit the equivalence into an axiomatic framework.

Several formats have been developed which ensure that the bisimulation equivalence induced by a TSS in such a format is always a congruence. A first proposal was made by De Simone [8], which was generalised by Bloom, Istrail, and Meyer [1] to the GSOS format. Next, Groote and Vaandrager [5] introduced the tyft/tyxt format, and proved a Congruence Theorem for TSSs in this format that satisfy a well-foundedness criterion.

Up to now, it has been an open question whether or not well-foundedness is an essential ingredient of the Congruence Theorem. The requirement popped
up in the proof, but no counter-example was found to show that the theorem breaks down if well-foundedness were omitted from it. In this paper, we prove that the Congruence Theorem does hold for general TSSs in tyft/tyxt format, i.e. that the requirement of well-foundedness can be omitted.

In fact, we will establish a stronger result, namely that for each TSS in tyft/tyxt format, there is an equivalent TSS consisting of `tree rules' only. A tree rule is a well-founded rule of the form

\[
\begin{align*}
\{z_i \xrightarrow{a_i} y_i \mid i \in I\} \\
\frac{f(x_1, \ldots, x_m) \xrightarrow{a} t}{f(x_1, \ldots, x_m) \xrightarrow{a} t}
\end{align*}
\]

where the \(y_i\) and the \(x_j\) are all different variables and are the only variables that occur in the rule, the \(z_i\) are variables, \(f\) is a function symbol, and \(t\) is any term. Using terminology from [5], we can say that a tree rule is a pure and well-founded xyft rule. Since tree rules are well-founded, the reduction of tyft/tyxt format to tree format will immediately imply that the Congruence Theorem concerning the tyft/tyxt format can do without well-foundedness.

Last summer, Rob van Glabbeek independently deduced the same result, which he announced in [3]. His proof is along the same lines as the one presented in this paper.

The major advantage of our main theorem is that it facilitates reasoning about the tyft/tyxt format. Because often it is much easier to prove a theorem for TSSs in tree format than for TSSs in tyft/tyxt format. For example, this is the case with the Congruence Theorem itself. Another striking example consists of Theorems 8.6.6 and 8.9.1 in [5]. With our result at hand, the complicated proof of the second theorem can be skipped, because now the second theorem follows immediately from the first one.

About all TSSs in Plotkin style that have been defined over the years are well-founded. So in this sense, the practical implication of removing well-foundedness from the Congruence Theorem for tyft/tyxt will probably be quite small. But this removal does increase considerably the convenience of applying the tyft/tyxt format, since the user no longer has to recall and check the complicated well-foundedness criterion.

Groote [4] added negative premises to tyft/tyxt, resulting in the ntyft/ntyxt format, and proved that the Congruence Theorem extends to well-founded TSSs in ntyft/ntyxt format. We will show that the reduction of tyft/tyxt rules to tree rules can be lifted to the positive part of rules in ntyft/ntyxt format, but a simple example will learn that this reduction cannot be applied to the negative premises. Again, we will find that the Congruence Theorem concerning the ntyft/ntyxt format can do without well-foundedness.

Finally, Verhoef [9] has defined the panth format, which adds predicates to ntyft/ntyxt, and proved that the Congruence Theorem holds for well-founded TSSs in panth format. We will show that all our results extend to the panth format too.

Acknowledgements. Chris Verhoef is thanked for useful comments, and special thanks go to Rob van Glabbeek and Frits Vaandrager for suggesting some substantial improvements.