Confluence of curried term-rewriting systems. (English. English summary)


An “abstract reduction system” is a structure $R = (A, \rightarrow)$ which consists of a set $A$ and a binary relation $\rightarrow$ on $A$. A “term rewriting system (TRS)” is an abstract reduction system $R$ where $A$ is the set $T(\Sigma, V)$ of terms over a ranked alphabet $\Sigma$ of function symbols and a set $V$ of variables, and $\rightarrow$ is a set of rewrite rules $(l, r)$, such that $l \notin V$ and any variable in $r$ also occurs in $l$.

Let $\Rightarrow$ denote the rewrite relation of $R$ and $\Rightarrow^*$ its reflexive and transitive closure. A TRS $R$ is called “confluent” if for all $u, v, w \in \Sigma^*$, $u \Rightarrow^* v$ and $u \Rightarrow^* w$ imply that $v$ and $w$ have a common descendent.

Currying a function $f$ of type $f : X_1 \times \cdots \times X_n \rightarrow X$ means reformulating $f$ to a mapping of type $f : X_1 \rightarrow X_2 \rightarrow \cdots \rightarrow X_n \rightarrow X$. Presenting first-order terms—as encountered in TRSs—in curried form is usually regarded as a trivial change of notation. The author argues that in the absence of a type-discipline, or in the presence of a more powerful type-discipline than simply typed $\lambda$-calculus, the change is not as trivial as one might first think.

The author’s main result shows that currying preserves confluence of arbitrary term rewriting systems. The structure of his proof is similar to Toyama’s proof that confluence is a modular property of TRSs [Y. Toyama, J. Assoc. Comput. Mach. 34 (1987), no. 1, 128–143; MR0882665 (88e:68072)].