Proving Liveness in Ring Protocols by Termination

Hans Zantema[1] and Jürgen Giesl[2]

[1] Department of Computer Science, TU Eindhoven, P.O. Box 513,
5600 MB Eindhoven, The Netherlands, h.zantema@tue.nl
[2] LuFG Informatik II, RWTH Aachen, Ahornstr. 55,
52074 Aachen, Germany, giesl@informatik.rwth-aachen.de

We consider the following family of protocols on a ring of processes (similar to a token ring protocol). Every process is in one of finitely many states. Depending on the state of a particular process and the states of a number of its neighbors a step can be done, by which the states of the process and of its neighbors may change. For a protocol of this shape we want to prove the following property:

> Starting from an initial configuration satisfying some property, in every infinite run of the protocol, after finitely many steps one will reach a configuration in which none of the processes is in state no.

Here no is a particular state of a process, typically meaning 'not received'. Thus, the above property means that eventually some message will be received by every process.

This property to be proved is an example of a liveness property: we have to prove that in every ongoing computation eventually 'something good' will happen. In this case the 'something good' is the receipt of the message by all processes. Proving liveness properties is closely related to proving termination as we pointed out in [3].

As an instance of such a protocol we consider the case in which apart from no a process can be in state sent or in state rec (received). Initially at least one of the processes is in state rec which means that it has received a message (token). Now the protocol is defined as follows:

> If a process is in state rec then it may send its message to its right neighbor which then will be in state rec, while the process itself then will be in state sent.

Clearly, at least one process will always be in state rec, and this procedure can go on forever. In [3] we introduced an approach to use termination techniques for term rewriting in order to prove liveness properties of protocols. For example, our approach in [3] can verify the desired property for the particular ring protocol above. However, for other instances in our family of ring protocols the technique from [3] may fail. This is partly due to the sound but incomplete automation of that method and partly due to the fact that the approach in [3] works for networks of processes of arbitrary shape. Hence there, the behavior of the ring topology must also be encoded in the protocol description. But then it can be difficult to state the desired property in the form of a liveness property as permitted in [3].

Therefore, in the present paper we introduce an alternative technique to the one in [3] which is especially designed for ring protocols. It states that for proving the desired liveness property it suffices to prove termination of the string rewriting system (SRS) describing the steps that can be done in the protocol.

First we explain how the protocol is described by such an SRS. We assumed that every process can be in one of finitely many states. Let the alphabet for the SRS correspond to these states. Now the states of a consecutive number of processes in the ring can be described by a string over this alphabet. Depending on this string, by the protocol rule it may be replaced by a string of the same length. For instance, in the above example the corresponding SRS consists of the three rules

$$\begin{aligned} \mathsf{rec\ rec} &\to \mathsf{sent\ rec} \\ \mathsf{rec\ sent} &\to \mathsf{sent\ rec} \\ \mathsf{rec\ no} &\to \mathsf{sent\ rec}. \end{aligned}$$

Theorem 1. *Let R be the SRS corresponding to a ring protocol of the given shape. Assume that the symbol* no *does not occur in any right-hand side of R. Then the corresponding liveness property holds on arbitrary ring size if and only if R is terminating.*

The condition that no does not occur in any right-hand side is essential, for instance if R consists of the single rule $\mathsf{rec\ no} \to \mathsf{no\ rec}$ then R is terminating but the corresponding liveness property does not hold.

By using this theorem, the liveness property for the above example can be proved by the simple observation that the corresponding SRS is terminating by the recursive path order [2].

Now we consider a more complicated example for which the methods from [3] require a solution of a TRS termination problem for which our attempts to use standard techniques all failed.

> If a process is in state rec then it may send its message to its two right neighbors which then will be both in state rec, while the process itself then will be in state sent.

This protocol rule may have to be applied an exponential number of times (exponential in the size of the ring) before all processes receive the message. For this protocol, the corresponding SRS consists of the nine rules

$$\mathsf{rec}\ p\ q \to \mathsf{sent\ rec\ rec}$$

where p, q run over the three symbols rec, sent, and no. Now a direct application of the recursive path order fails, but termination can easily be proved by the dependency pair technique [1] in combination with the recursive path order (by choosing all symbols to be equal in the precedence). As an alternative termination proof, note that a string rewriting system is terminating iff its reversed variant (where each string in the rules is reversed) is terminating. For the reversed system $q\ p\ \mathsf{rec} \to \mathsf{rec\ rec\ sent}$, the polynomial interpretation $[\mathsf{sent}](x) = 2x+1, [\mathsf{rec}](x) = 2x$ suffices for proving termination. Hence, the desired liveness property is verified.

The kind of string rewriting involved in Theorem 1 is in fact *ring rewriting* rather than string rewriting: the objects that are rewritten are not strings having a begin and an end, but they are rings, being strings in which the begin and the end are connected and in which one abstracts from the position representing this connection. More precisely,

for any alphabet Σ we define the set $\mathsf{Ring}(\Sigma)$ of rings over Σ by $\mathsf{Ring}(\Sigma) = \Sigma^*/\sim$ where $\sim$ is the equivalence relation on Σ^* defined by

$$u \sim v \iff \exists u_1, u_2 \in \Sigma^* : u = u_1 u_2 \wedge v = u_2 u_1.$$

Writing $[u]$ for the equivalence class of u w.r.t $\sim$, for an SRS R we define the corresponding ring rewrite relation $\circ\!\!\rightarrow_R$ on $\mathsf{Ring}(\Sigma)$ by

$$[u] \circ\!\!\rightarrow_R [v] \iff \exists u', v' : u \sim u' \wedge v \sim v' \wedge u' \rightarrow_R v'.$$

Now the liveness property from Theorem 1 can be stated as follows: in every infinite $\circ\!\!\rightarrow_R$-reduction after finitely many steps a ring is obtained not containing the symbol no. Surprisingly, for the SRSs given in the two examples the relation $\circ\!\!\rightarrow_R$ is not terminating since they contain the rules $\mathsf{rec\ sent} \rightarrow \mathsf{sent\ rec}$ and $\mathsf{rec\ sent\ rec} \rightarrow \mathsf{sent\ rec\ rec}$, respectively, both giving rise to the cyclic reduction $[\mathsf{rec\ sent\ rec}] \circ\!\!\rightarrow_R [\mathsf{sent\ rec\ rec}] = [\mathsf{rec\ sent\ rec}]$. The proof of Theorem 1 can be given using the following observation. Let R be an SRS in which the symbol no does not occur and let $[\mathsf{no}\ u] \circ\!\!\rightarrow_R [v]$. Then there exists a string w such that $u \rightarrow_R w$ and $[\mathsf{no}\ w] = [v]$. This shows that for a terminating SRS R, rules without no can only be applied finitely many times, and rules with no on the left- but not on the right-hand side decrease the number of occurring no-symbols.

Apart from liveness, one can wonder how termination of $\circ\!\!\rightarrow_R$ relates to termination of $\rightarrow_R$ for arbitrary SRSs R. It is easily seen that termination of $\circ\!\!\rightarrow_R$ implies termination of $\rightarrow_R$, but the above example shows that the converse does not hold. This leads to following question: given an SRS R, how to prove termination of $\circ\!\!\rightarrow_R$? It turns out that arguments using decrease of weight, and a version of semantic labelling can be given for this kind of ring rewriting, but for techniques like recursive path order or dependency pairs which strongly rely on the consideration of term structure, we did not (yet) succeed in finding a variant capable of proving termination of $\circ\!\!\rightarrow_R$.

References

1. T. Arts and J. Giesl. Termination of term rewriting using dependency pairs. *Theoretical Computer Science*, 236:133–178, 2000.
2. N. Dershowitz. Termination of rewriting. *Journal of Symbolic Computation*, 3:69–116, 1987.
3. J. Giesl and H. Zantema. Liveness in rewriting. In *Proc. 14th RTA*, LNCS, 2003. To appear. Extended version appeared as Technical Report AIB-2002-11, RWTH Aachen, Germany. Available from `http://aib.informatik.rwth-aachen.de`.