Verification of Erlang Processes*

Thomas Arts[1] and Jürgen Giesl[2]

[1] Computer Science Laboratory, Ericsson Telecom AB, 126 25 Stockholm, Sweden, E-mail: `thomas@cslab.ericsson.se`

[2] FB Informatik, Darmstadt University of Technology, Alexanderstr. 10, 64283 Darmstadt, Germany, E-mail: `giesl@informatik.tu-darmstadt.de`

Erlang is a functional language, developed and extensively used within Ericsson Telecom, for implementing concurrent systems [AVWW96]. In the application area of these programs there are high demands on the correctness of the software. For that reason, there is great interest for a formal verification of critical and complicated components in order to isolate potential bugs that are not or cannot be found by testing.

Techniques for verification are in general based on model-checking and theorem proving. There even exists a special purpose theorem prover for Erlang programs [Dam95,ADFG98]. The advantage of model-checking is that it can be performed completely automatically, but its use is restricted to an abstraction of the communicating behaviour of the program. The use of a theorem prover demands a lot of human interaction and understanding of the problem, but this allows to prove properties of the outcome and behaviour of the functions as well.

Termination is a property that often has to be proved during verification, e.g. for function calls of which the result is irrelevant for the final outcome, termination might still be important for progress. Moreover, even properties of non-terminating programs (which often occur in the application area) might sometimes be expressed in terms of a termination property. Since proving this kind of termination properties goes far beyond the possibilities of model-checking, we have been looking for other techniques to automatically perform this task.

The semantics of Erlang differs from those of term rewriting systems, i.e. Erlang is not a pure functional language and the notion of concurrency is lacking in the standard term rewriting formalisms. Nevertheless, sequential parts of Erlang (i.e. programs without communication) that are written like pure functions are so similar to TRSs that treating the program like a TRS is almost natural. Doing so, techniques for proving termination of TRSs automatically, in particular the dependency pair approach [AG97,AG98a,AG99], become available. Additionally, techniques for innermost termination come into scope, since the TRSs corresponding to Erlang functions are non-overlapping.

In a small but realistic case-study we demonstrated the usefulness of this approach [AG98b]. We proved from a certain Erlang process that after constantly

* This work was partially supported by the Deutsche Forschungsgemeinschaft under grants no. Wa 652/2-2 as part of the focus program 'Deduktion'.

receiving a certain message, it eventually would send this message to another process. The proof is divided in several steps. First, we translate the process and the property to a termination problem of a TRS. This step is performed by hand, but care has been taken to stay close to the original problem and to make the transformation commonly applicable and understandable. Second, we extend the dependency pair approach by introducing *rewriting of dependency pairs*. Lastly, we can automatically prove termination of the obtained TRS by using our refined techniques.

References

[AVWW96] J. Armstrong, R. Verding, C. Wikström and M. Wiliams, *Concurrent Programming in Erlang.* 2:nd edition, Pretence Hall, 1996.

[ADFG98] T. Arts, M. Dam, L-å. Fredlund, and D. Gurov, System Description: Verification of Distributed Erlang Programs. In *Proc. CADE'98*, LNAI 1421, pp. 38–42, Lindau, Germany, July 1999.

[AG97] T. Arts & J. Giesl, Proving innermost normalisation automatically. In *Proc. RTA-97*, LNCS 1232, pp. 157–172, Sitges, Spain, 1997.

[AG98a] T. Arts & J. Giesl, Modularity of termination using dependency pairs. In *Proc. RTA-98*, LNCS 1232, pp. 226–240, Tsukuba, Japan, 1998.

[AG98b] T. Arts & J. Giesl, Applying Rewrite Techniques for Verification of Erlang Processes. *Tech. Rep. IBN 98/50*, Technical University Darmstadt, Germany, December 1998

[AG99] T. Arts & J. Giesl, Termination of term rewriting using dependency pairs. *Theoretical Computer Science.* To appear.

[Dam95] M. Dam, Compositional proof systems for model checking infinite state processes. In *Proc. CONCUR'95*, LNCS 962, pp. 12–26, 1995.