Parallel Functional Reactive Skeletons in Concurrent Clean

Zoltán Horváth, Viktória Zsók
Department of General Computer Science, University of Eötvös Loránd, Budapest
hz@inf.elte.hu, zsv@inf.elte.hu

Pascal Serrarens, Rinus Plasmeijer
Faculty of Mathematics and Computer Science, University of Nijmegen
pascalrs@cs.kun.nl, rinus@cs.kun.nl

The skeletons are parameterised algorithmic schemes. We can use them to control parallel execution of the programs. There are two possibilities for integration of the skeletons in concurrent programming: one is to nest the skeletons in compiler and so the users are no aware of parallelism, and the other one is to write implementation modules that can be used and changed by users.

Skeletons in functional programming languages are expressed as higher order functions and they allow to implement the well-known parallel programming paradigm in portable, efficient programs [1]. The skeletons can be parameterised triply: by a function that computes the value of the result, by a strategy determining the dynamic behaviour and by types. Reactive skeletons are models of the reactive systems, which have some interactions with their environment, but doesn't give a final result. Usually they are non-deterministic.

In this paper we present a set of algorithmic skeletons in the parallel lazy functional language Concurrent Clean [2, 3]. Concurrent Clean provides process annotations for explicit thread creation and data-driven message passing system for the communication between processes. We would like to present the reactive skeletons with typical examples of concurrent systems.

References

[1] Galán, L.A.; Pareja, C.; Peña, R.: Functional Skeletons Generate Process Topologies in Eden. In *Int. Symp. on Programming Languages: Implementations, Logics and Programs, PLILP'96.* Aachen, Germany, pp. 289-303. Springer-Verlag LNCS 1140, 1996.

[2] Serrarens, P.R.: Explicit Message Passing for Concurrent Clean. In K. Hammond et al.: *Proc. of Implementation of Functional Languages, IFL'98*, pp. 298-308, Sept. 1998, London, to appear in LNCS.

[3] Kesseler, M. H. G.: *The Implementation of Functional Languages on Parallel Machines with Distributed Memory.* PhD thesis, University of Nijmegen, Constructing skeletons, pp. 153-173, 1996.