Parallel Elementwise Processable Functions in Concurrent Clean [*]

Zoltán Horváth, Viktória Zsók
Department of General Computer Science
Eötvös Loránd University, Budapest
Pascal Serrarens, Rinus Plasmeijer
Faculty of Mathematics and Computer Science
University of Nijmegen
e-mail: hz@ludens.elte.hu, zsv@inf.elte.hu,
pascalrs@cs.kun.nl, rinus@cs.kun.nl

Abstract

The behaviour of concurrent and parallel programs can be specified in a functional style. Functional programming style has some inherent concurrent features. However, for a higher degree of expressing parallelism there is a need for new language constructs. In this paper we introduce Concurrent Clean moduls for evaluation strategies in order to control the evaluation degree, the dynamic behaviour and the parallelism. The usage of the strategies will be illustrated by the parallel elementwise processing method. The implementation of the method in the lazy functional programming language Concurrent Clean operates with a two arguments two values elementwise processable function. In order to obtain abstract type specification for the generalised manipulation of the linear data structures like lists, arrays, strict arrays a linear data structure class module is defined. The programming style is skeleton based. Skeletons in functional languages are higher order functions. The skeleton given in this paper represents a generalisation of the `map` function. It is triply parameterized: by a parallel elementwise processable function, by type specification and by the strategy parameter that defines the dynamic behaviour of the program.

References

[1] Chandy, K.M., Misra, J.: *Parallel program design: a foundation*, Addison-Wesley, 1989.

[2] Cole, M.: Algorithmic Skeletons, In: Hammond, K., Michaelson, G., eds., *Research Directions in Parallel Functional Programming*, pp. 289-303, Springer-Verlag, 1999.

[3] Cole, M: Algorithmic Skeletons: Structured Management of Parallel Computation, *Research Monographs in Parallel and Distributed Computing*, The MIT Press, Cambridge, Massachusetts, 1989.

[*] Supported by the Hungarian State Eötvös Fellowship, by the Hungarian National Science Research Grant (OTKA), Grant No. T032178 and by FKFP 0206/1997.

[4] Darlington, J., Field, A.J., Harrison, P.G., Kelly, P.H.J., Sharp, D.W.N., Wu, Q., While, R.L., Parallel Programming Using Skeleton Functions, In: *Proc. PARLE '93 - Parallel Architectures and Languages Europe*, LNCS, Vol. 694, pp. 146-160, Springer-Verlag, 1993.

[5] Fóthi Á., Horváth Z., Kozsik T.: Parallel Elementwise Processing – A Novel Version, In: Varga L., ed., *Proceedings of the Fourth Symposium on Programming Languages and Software Tools*, Visegrád, Hungary, June 9-10, 1995, pp. 180-194. and in *Annales Uni. Sci. Budapest de R. Eötvös Nom. Sectio Computatorica*, Vol. 17, pp. 105-124, 1998.

[6] Fóthi Á. et al., Workgroup on Relational Models of Programming - Some concepts of a Relational Model of Programming, In: Varga, L., ed., *Proc. of the Fourth Symp. on Programming Languages and Software Tools*, Visegrád, Hungary, June 8-14, 1995.

[7] 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, LNCS, Vol. 1140, pages 289-303, Springer-Verlag, 1996.

[8] Hammond, K., Rebón Portillo, Á.J.: HaskSkel: Algorithmic Skeletons in Haskell, In: Koopman, P. et al., eds., *Implementation of Functional Languages, 11th International Workshop, IFL'99*, Lochem, The Netherlands, September 1999, LNCS, Vol. 1868, pp. 181-198, Springer-Verlag, 2000.

[9] Horváth Z.: Parallel asynchronous computation of the values of an associative function, *Acta Cybernetica*, Vol. 12, No. 1, pp. 83-94, Szeged, 1995.

[10] Kesseler, M.H.G.: *The Implementation of Functional Languages on Parallel Machines with Distributed Memory*, PhD Thesis, Catholic University of Nijmegen, 1996.

[11] Kozma L., Frohner Á., Kozsik T., Porkoláb Z.: Beyond 2000, beyond object-orientation, to appear in *Proceedings of the 5th International Conference on Applied Informatics*.

[12] Okasaki, C.: *Edison User's Guide (Haskell version)*, Department of Computer Science, Columbia University, Online Document, May, 1999.
http://www.cs.columbia.edu/˜cdo/edison/.

[13] Serrarens, P.R.: *Communication Issues in Distributed Functional Computing*, Ph.D. Thesis, University of Nijmegen, January 2001.

[14] Serrarens, P.R: Distributed arrays in the functional language Concurrent Clean, In: *Proceedings of the 3rd International Euro-Par Conference*, Passau, Germany, August 1997, LNCS, Vol. 1300, pp. 1201-1208, Springer-Verlag, 1997.

[15] Serrarens, P.R.: Explicit Message Passing for Concurrent Clean, In: Hammond, K. et al., eds., *Implementation of Functional Languages, 10th International Workshop, IFL'98*, London, UK, September 1998, LNCS, Vol. 1595, pp. 229-245, Springer-Verlag, 1999.

[16] Trinder, P.W., Hammond, K., Loidl, H.W., Peyton Jones, S.J.: Algorithm + Strategy = Parallelism, *Journal of Functional Programming*, Vol. 8, No. 1, pp. 23-60, Cambridge University Press, January, 1998.