ACL — Building Cellular Processing Kernels Based On Attributes

Ralf Schneider
Microprogramming and Computer Architecture
Technical University of Darmstadt
Alexanderstr. 10, Darmstadt, D-64283, Germany
schneider@informatik.th-darmstadt.de

Abstract

Cellular processing (CP) is defined as an extended interpretation of Cellular Automata (CA) in the sense of an extension of the internal cell state complexity and a broader variety of the cell topologies. The range from a few bits to complex record-like cell state structures with floating point values are covered by the following approach. Also different regular grids are included to map application models in the fields of mathematics, biology, physics and artificial worlds. Simulation (processing and visualization) of these models is realized by Cellular Processing kernels (CP kernels), which provide special aspects and features of CP. Since different target system architectures are used for CP simulation all hardware and application specific features can not be combined in one single CP kernel without loosing efficiency, transparency and portability. This is a major criticism of universal CP kernels which try to cover all features of cellular processing at once. In this paper a new approach is introduced which automatically generates user specific CP kernels based CP attributes and optimization strategies using the high programming model ACL.

Extended Abstract

Cellular models can be defined by a number of CP attributes concerning their *internal data structure*, *algorithmic behavior*, the *target system architecture* and the *optimization strategies.* Cellular processing is done by cellular processing kernels (CP kernels) which consider CP attributes to map cellular models onto specific target system architectures. Single and multi-processor CP systems differ on various aspects and once manually created programs will have to be modified and adjusted for each target system architecture by experienced programmers. Therefore CP kernels should have following properties:

- Best possible adaption to the target system architecture
- Efficient storing of the cellular space
- Additional application specific optimization strategies on algorithmic level
- Portability on source code and abstract level
- Extendibility and software re-usability
- Simple and fast source code without time consuming condition queries (condition-less run-through)

With the new introduced high level programming model ACL (*Abstract Cellular model Language*) these aims will be reached by considering the model properties (algorithmic behavior, internal data structure, optimization strategies) when mapping cellular models onto different target system architectures. The local rules as part of the algorithmic behavior will be described in CDL (*Cellular Description Language*) which has a "PASCAL/C"-like syntax.

Based on CP attributes, the entire CP kernel is generated by ACL in a special transformation process on source code level. The generated CP kernel will be compilable on the chosen target system architecture. This transformation process will be embedded into a programming environment to simplify the generation of CP kernels for the user. ACL can be divided into transformation processes of selections and shaping using substitution and replacement techniques on a LISP-like data structure. ACL is also a meta language level which describes the CP kernel as an abstract program. It is therefore highly flexible to application or hardware specific requirements. The idea of the high level program model ACL will be introduced using a 2D-traffic simulation as an example.

With special target language skeletons (TL-skeletons) defining the target language structures the next transformation step leads to the final CP kernel. This CP kernel is compilable with the standard compiling techniques. Any language as "C", "C++" or "Pascal" can be included as language level for a target system architecture.

With the flexible use of natural formed ACL-commands and the substitution and replacement technique it is possible to construct complex programs for cellular processing on different target system architectures. The ACL-program skeleton and the TL-skeletons provide a very simple mapping of special CP features onto target system architectures. For adaption of new or extended tasks software re-usability is the main criterion even in this area of interests. Due to the structures of the ACL-transformation process it is simple to extend and modify existing CP kernels for special tasks and applications without re-engineering or re-developing.

References

[1] M. Cannataro, S. D. Gregorio, R. Rongo, W. Spataro, G. Spezzano, and D. Talia. A parallel cellular automata environment on multicomputers for computational science. *Parallel Computing*, 99(21), 1995.

[2] J. Darlington, A. Field, P. Harrison, P. Kelly, D. Sharp, Q. Wu, and R. While. Parallel programming using skeleton functions. *Parallel Architectures And Languages, Europe, PARLE 93*, 1993.

[3] K. M. Decker, J. J. Dvorak, and R. M. Rehmann. A knowledge-based scientific parallel programming environment. *Swiss Scientific Computing Center, CSCS-TR-93-07*, 1993.

[4] M. M. Gutzmann and S. Kindermann. Transformation based development of efficient programs for massively parallel architectures. *Dep. of Computer Science, University of Jena/Erlangen, Germany.*

[5] C. Hochberger, R. Hoffmann, and S. Waldschmidt. Compilation of CDL for different target architectures. *Proceedings of the Third International Conference PaCT-9, St. Petersburg*, 1995.

[6] R. Hoffmann. Cellular processing architectures. *Mitteilungen - Gesellschaft für Informatik e.V., Parallel-Algorithmen und Rechnerstrukturen, PARS Workshop Dresden*, 1993.

[7] J. K. Ousterhout. *Tcl and Tk Toolkit.* Addison-Wesley, Reading, Massachusetts, 1994.

[8] R. Schneider. Ein Verfahren zur Laufzeitoptimierung bei der zellularen Verarbeitung. *Proceedings of the PARS Conference Stuttgart*, 1995.