Architectures for Cellular Processing

Rolf Hoffmann and Klaus-Peter Völkmann
Microprogramming and Computer Architecture
Technical University of Darmstadt
Alexanderstr. 10, D-64283 Darmstadt
{hoffmann, voelkmann}@informatik.th-darmstadt.de

1 Motivation

With *Cellular Processing (CP)* we mean the mapping of applications onto the CA model, the implementation and the execution/simulation on a system with an appropriate environment (compiler, simulator, control, vizualisation, analysis). Especially we want to support applications, where the cells are complex (a data record) and the array consists of a large number of cells or the CA field is 3-dimensional.

Software simulations take a long time, e.g. if one cell operation takes 10 microseconds and the field is consists of 512^3 cells, the time to compute one generation will be 22 minutes. 100 generations will then need 40 hours. For the broader use and acceptance of the CA model for practical applications, the rate of generations per second must be much higher, to allow online vizualisation and direct user interaction.

At present, the following requirements can only be fulfilled if multiprocessor systems or hardware support are available.

- large number of cells
- large or complex cell states
- complex rules
- many neighbours (neighborhood distance ≤ 1, 3 dimensions)
- many generations per experiment
- many similar experiments with parameter variations
- online vizualization and analysis
- fast user interaction rates.

In the project "Hardware supported simulation system for the graph based and three dimensional cellular processing" supported by the DFG, we are developing a special coprocessor which shall accelerate the simulation by two orders of magnitude compared to software simulations on standard single processor systems. The design is based on the experience we gained through the realizations of the CEPRA-8 [1,2] and CEPRA-1L[3] designed for the two dimensional support.

2 Some Architectural Principles

Memory Table. The state bits of the center cell and the neighborhood cells are concatenated to an address for a memory. The memory holds the rule as a truth table with indexed access through the address. This technique is limited to a few bits per cell because of the limited memory capacity. In the 3-dimensional case and Von-Neumann neighbourhood the number of memory inputs is 7 multiplied by the number b of bits per cell. Thus the needed memory capacity is

$C = 3Dn * 2^{7n}$. (e.g. $n = 3D4 : C = 3D128$ Megabytes)

Local Processing. For simplification we consider an array of n^2 cells. Each processor computes a subarray of size n^2/p. In order to reduce the cost of the architecure, the processors shall be connected to a common communication bus.

The computation inside the processor is sequential and takes t_F for one cell. The total *computation time* for all subarrays will be

$T_F = 3D(n^2/p)t_F$.

At the boarders of the subarrays cell state information must be interchanged. In the case of the Von-Neumann-Neighbourhood (without wrap around the field) the number of such cells is $4n(\sqrt{p} - 1)$.

The *communication time* will be

$T_{COMM} = 3D4n(\sqrt{p} - 1) \cdot t_{COMM}$,

where t_{COMM} is the time for one state transfer between processors.

The following characteristics are generally of interest, which depend on the ratio t_F/t_{COMM}:

- time for a new generation $T = 3DT_F + T_{COMM}$
- percentage of communication T_{COMM}/T
- speed up $T(p = 3D1)/T(p)$
- efficiency S/p.

Example. $p = 3D256$ and $n = 3D512$ and $t_{COMM} = 3D1\mu s$:

$t_F = 3D50\mu s \Rightarrow$ T =3D 51 ms + 31 ms =3D 82 ms

communication=3D37%, speed-up=3D160, efficiency=3D0.62

Valuation. This architecture has good performance as long as communication time is relatively short and not too many processors are used.

Most applications need frequent output/input of the whole generation to/from video store, global data store or disk. The main disadvantage of this architecture is the long time for output/input due to the distribution of the cell data.

Central Store and Pipelining. Two central stores $MEMORY_{old}$ and $MEMORY_{new}$ hold the generations. The cell states from the old generation are read sequentially via shift registers into a 3×3 neigbourhood-field (window) with direct acces to 9 cells. The shift registers hold two complete lines of the field (NORTH and CENTER cells). With this shifting technique a new neigbourhood-field can be set up in every store cycle. The neigbourhood-field is the input to a pipeline which computes the rule stepwise through the pipeline stages. The output of the pipeline is stored in $MEMORY_{new}$. Before the next generation is computed, the two memories are interchanged.

Valuation. Advantages of this architecture are quick access to the = central store for input/output and the computation time of one read/write cycle (at best) for one cell operation (COP). A difficulty is that the stages of the pipeline must be especially designed for the application and must work on high speed.

Interleaved Processing. This architecture consists of a central store and p processors with data buffers and local stores. The components are connected by a common bus. Cell state information is first distributed via the bus to the input buffers. Thereby advantage is taken by the *broadcasting principle* because a cell state is needed in 4 (von Neumann neighbourhood) to 8 (Moore neighbourhood) processors as neighbourhood information. Each processor starts its computation when the neighbourhood information is completely local to the processor. The starting points of the computations are cyclically interleaved. (p time steps =3D one cycle). Each processor should compute his result in less than one cycle (computation time limit =3D one cycle). The results are cyclically interleaved transferred back to the central store and at the same time to those processors, who need this information as new neighbourhood information.

Valuation. Processors have enough time (one cycle) to compute their results, meaning that low cost processors with optimal sequential algorithms may be used. One time step lasts 4 read/write cycles (NORTH; CENTER; SOUTH; write result) if 3×3 buffers are used in the processors. If in the processors larger buffers (complete lines) are available and the write cycle can be performed in parallel with the read cycle, the time step can further be reduced to one read/write cycle like in a pipeline architecture.

3 An Architecture for 3 Dimensions

In our project we have designed a copressor for 3-dimensional cellular processing. It is based on shifting in 3 dimensions and pipelining.

The architecture consists of (1) a *memory* holding one generation, (2) a *plane shifter*, (3) a *line array shifter*, (4) a *computing window* and (5) *p processing = elements.*

The xyz-memory holds n^3 cells. p cells in z-axis shall be processed in parallel. The xy-plane shifter consists of 3 z-planes. Each plane holds p cells. The bus between the xyz-memory and the xy-plane shifter is p cell wide.

The line array shifter is of size n in x-axis, 3 in y-axis and $p + 2$ in z-axis. A xz-slice output from the plane shifter can be loaded in n steps into it. It can perform shifts in x direction, thereby loading the computing window.

The computing window holds p center cells (to be computed) in z-axis an= d all their neighbours. It is of size $p+2$ in z-axis, 3 in x-axis and 3 in y-axis. It is loaded shift by shift from the line array shifter. The p processing elements have parallel access to the window and are internally pipelined with the shift clock.

Succesive planes are loaded from the memory to the plane shifter, which consist of 3 planes (i=3D1,2,3). In the start-up plane-cycle all 3 planes are loaded from memory. In the following plane steps the planes i are shifted to planes $i+1$ and a new memory plane is loaded into the plane 1.

Between each plane step approx. n line steps are performed. In each line step the plane shifter is shifted in y-direction and thereby $p+2$ lines are loaded into the line array shifter.

Between each line step approx. n window steps are performed. In each window step the line array shifter is shifted in x-direction and thereby $3*3*(p+2)$ cells of the computing window are updated.

The total capacity of memory cells is

$C = 3Dn^3 + 3pn^2 + 3(p+2)n + 9(p+2)$.

The computing time is the sum of the times needed for (1) loading the plane shifter, (2) loading the line array shifter and (3) the time to compute a new cell state which is overlapped with the writing back into the memory. The bus between memory and plane shifter is p word wide, and between plane shifter and line array shifter $p+2$ word wide. The total execution time will result in

$T = 3D3n^3/p$.

After adding of buffers and interleaving of the memory the execution time may be reduced to n^3/p.

Valuation. This architecure is easy scalable in p and is not very expensive to realize. Using field programmable logical array for the processing elements the required performance can cost effectively be realized.

References

[1] R. Hoffmann, K.-P. Völkmann, M. Sobolewski:
The Cellular Processing Machine CEPRA-8.
Proceedings of the VIth International Workshop on Parallel Processing by Cellular Automata and Arrays, Potsdam, Sept. 1994.

[2] R. Hoffmann, K.-P. Völkmann:
CEPRA-8L: A Celluar Processing Machine
in: Massiv Parallelism Hardware, Software and Applications, Editor M. Mango Furnari, Capri 1994

[3] Th. Hahn, Chr. Hochberger, R. Hoffmann, K.-P. Völkmann
Der CEPRA-1L Zellularprozessor
submitted for Workshop on Custom Computing Dagstuhl 6/96