Compiling Image Processing Applications for Many-Core Accelerators
Pierre Guillou

To cite this version:
Pierre Guillou. Compiling Image Processing Applications for Many-Core Accelerators. Journées de seconde année de l’Ecole Doctorale, Jun 2015, Paris, France. hal-01178938

HAL Id: hal-01178938
https://hal-mines-paristech.archives-ouvertes.fr/hal-01178938
Submitted on 21 Jul 2015

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Compiling Image Processing Applications for Many-Core Accelerators

Pierre Guillou – CRI MINES ParisTech, PSL Research University

Image Processing

image analysis: detect geometrical structures in an image
mathematical morphology: image analysis theory and technique based on lattice theory

Mathematical Morphology Base Operators

- arithmetic operators
  - unary (input parameter, 1 input image)
  - binary (input parameter, 2 input images)
- morphological operators
- threshold
- neighbour selection: min/max/avg
- reduction operators: global max/min/sum
- other operators
  - threshold, mask, log2, ...

Implement more complex algorithms: watershed, arrow, labelling, minima, ...

Improve data-parallelism to take better advantage of the current architecture

Other programming models:

- OpenCL via local memory pagination
- Pthreads/OpenMP on compute clusters, communication library between clusters

Optimisations

- unrolling of converging loops
- arithmetic operators aggregation
- generation of kernel-specific convolutions
- data parallelization for compute-intensive operators

Results: Execution Times and Energy Consumption (MPPA-256 = 1, lower is better)

Future Work

- Other programming models:
  - Pthreads/OpenMP on compute clusters, communication library between clusters
  - OpenCL via local memory pagination
- Improve data-parallelism to take better advantage of the current architecture
- Implement more complex algorithms: watershed, arrow, labelling, minima, ...

Runtime Environment

- Host runtime
- Compute clusters
- Accelerator runtime on I/O clusters
- Host binary
- Compute binaries
- Accelerator control code
- Host runtime
- Compute clusters
- Optimisations
  - unrolling of converging loops
  - arithmetic operators aggregation
  - generation of kernel-specific convolutions
  - data parallelization for compute-intensive operators

References

Pierre Guillou, Fabien Coelho, and François Irigoin.
Automatic Streamization of Image Processing Applications.
The 27th International Workshop on Languages and Compilers for Parallel Computing (LCPC), 2014.
Available at http://www.cri.ensmp.fr/classement/doc/A-570.pdf

Compilation Chain

- original application
- source-to-executable compiler
- call graph optimisations
- target-specific compiler
- compute binaries
- host binary
- Host runtime
- Compute clusters
- Optimisations
- unrolling of converging loops
- arithmetic operators aggregation
- generation of kernel-specific convolutions
- data parallelization for compute-intensive operators

Example: Licence Plate Extraction

Sigma-C, a Dataflow Programming Language

agent foo () {
  // describe agent interface
  interface {
    input 0 inp0 [2], input 1 inp1, output outp[2];
    // define the state machine
    spec { outp[0] = inp0[0]; outp[1] = inp0[1];
    // loop over the state
    void start () exchange (inp0 inp0[2], inp1, outp outp[2]);
    // describe subgraph interface
    map {
      // instanciate agents
      agent a1 = new Agent1();
      connect (a3.output , a5.input1); // ...
      connect (a1.output0 , a2.input);
      connect (a5.output , output1); // ...
      connect (input0 , a1.input0);
    }
  }

Sigma-C agent library

The MPPA-256 Chip

I/O cluster

noC Interface

Local memory (3 GB)

DDR

Compute clusters

NoC Interface

Shared memory (2 MB)

I/O cluster

PCI-E

Host machine