A Compilation Framework for Irregular Memory Accesses on the Cell Broadband Engine

Mainak Chaudhuri

Department of Computer Science and Engineering, IIT Kanpur

Pramod Bhatotia – IBM Research
Sanjeev Aggarwal – IIT Kanpur
Outline

1. Compiling Irregular Accesses
2. Dataflow Analysis
3. Compiler Transformations and Code Generation
4. Compiler-Directed Communication Mechanism
5. Run-time Parallelization
6. Experiments and Results
7. Summary
Cell Broadband Engine Overview

Cell Architecture

The Cell Broadband Engine

Mainak Chaudhuri  Compiling Irregular Accesses for the Cell Processor
Irregular Array Access

An array access is irregular if no closed-form expression, in terms of the loop indices and constants, for the subscripts of the accessed array is available at compile-time.

do t =
  do i = num_edges
    n1 = left[i]
    n2 = right[i]
    force = (x[n1] - x[n2])/4
    y[n1] += force
    y[n2] += force
Irregular Array Access

An array access is irregular if no closed-form expression, in terms of the loop indices and constants, for the subscripts of the accessed array is available at compile-time.

```c
do t =
  do i = num_edges
    n1 = left[i]
    n2 = right[i]
    force = (x[n1] - x[n2]) / 4
    y[n1] += force
    y[n2] += force
```
Compiling Irregular Accesses

Mainak Chaudhuri

Compiling Irregular Accesses for the Cell Processor
Compiling Irregular Accesses on the Cell Processor

Compiler Analysis for Irregular Array Accesses

force = x[right[i]] - x[left[i]];

access pattern @ run-time

static analysis for DMA comm.

MFC-DMA

Dataflow framework based on run-time preprocessing for DMA comm.

Mainak Chaudhuri

Compiling Irregular Accesses for the Cell Processor
Compiling Irregular Accesses on the Cell Processor
Explicit Dynamic Memory Management

Mainak Chaudhuri
Compiling Irregular Accesses for the Cell Processor
Compiling Irregular Accesses on the Cell Processor
Cell Architecture Specific Challenges
Sparse Updates

- Sparse update refers to reduction operations on some elements of an array within a loop where the access pattern of the array in the loop may be irregular.

\[
\text{for } (i = 1 \text{ to } n) \\
\]
Compiling Irregular Accesses on the Cell Processor
Run-time Parallelization

Mainak Chaudhuri  Compiling Irregular Accesses for the Cell Processor
Compiling Irregular Accesses on the Cell Processor

Run-time Parallelization

Mainak Chaudhuri  Compiling Irregular Accesses for the Cell Processor
Compiling Irregular Accesses on the Cell Processor

Run-time Parallelization

Mainak Chaudhuri
Compiling Irregular Accesses for the Cell Processor
Overview of the System

Sequential Input program → Dataflow Variable Analysis → Compiler Transformation → SPMD Cell Code → Runtime System → Memory Communication → Runtime Parallelization

Parallelizing Compiler Framework
Dataflow Analysis for Memory Communication

Figure: Dataflow analysis for memory communication
Compiler Transformations and Code Generation

Compiling Irregular Accesses for the Cell Processor

Mainak Chaudhuri
Figure: Compiler transformations and code generation
Compiler-Directed Communication Mechanism
Compiler-directed Communication Mechanism
Block Access Method for Regular Accesses
Compiler-directed Communication Mechanism
Block Access Method for Regular Accesses

Execution Unit
SPU
Local Store
DMA controller
MFC

Local Store

Tile-Size

Code

Data

Main Memory

EIB
Compiler-directed Communication Mechanism
Block Access Method for Regular Accesses

Mainak Chaudhuri
Compiling Irregular Accesses for the Cell Processor
Compiler-directed Communication Mechanism

Block Access Method for Regular Accesses
Compiler-directed Communication Mechanism
Bounded Method for Irregular Read Accesses

for (i=1 to 4)
{
   sum += a[b[i]];
}

Bounded Access for Irregular read-only Accesses
Compiler-directed Communication Mechanism
Bounded Method for Irregular Read Accesses

for (i=1 to 4) {
    sum += a[b[i]];
}

Bulk Access

Main Memory

EU     SPU     LS
MFC
Compiler-directed Communication Mechanism
Bounded Method for Irregular Read Accesses

for (i=1 to 4)
{
   a[b[i]]= a[b[i]]+x;
}

Mainak Chaudhuri
Compiling Irregular Accesses for the Cell Processor
Compiler-directed Communication Mechanism
Bounded Method for Irregular Read Accesses

Executor prepares list of all the needed elements and initiates the communication with MFC DMA list command.
Compiler-directed Communication Mechanism
Bounded Method for Irregular Read Accesses

*Prepares single bounding box within same-tile for duplicated values of ref-array
*Keeps the size of box as 128 bytes, since DMA transfer unit is 128 bytes
*Does memcopy for values across the tile
Compiling Irregular Accesses
Dataflow Analysis
Compiler Transformations and Code Generation
Compiler-Directed Communication Mechanism
Run-time Parallelization
Experiments and Results
Summary

Compiler-directed Communication Mechanism
Compiler Controlled Cache for Sparse Updates

Mainak Chaudhuri
Compiling Irregular Accesses for the Cell Processor
Compiler-directed Communication Mechanism
Compiler Controlled Cache for Sparse Updates

for (i = 1 to 10)
{
   a[b[i]] += X;
}
Compiler-directed Communication Mechanism

Compiler Controlled Cache for Sparse Updates
Parallelization of Irregular Reduction Loops

Index Values of Array B

<table>
<thead>
<tr>
<th>Iteration No.</th>
<th>SPE-1</th>
<th>SPE-2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>3</td>
</tr>
</tbody>
</table>

Local SPE Buckets

<table>
<thead>
<tr>
<th>SPE Bucket with count stored in the main memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>0(1) 1(0) 2(1) 3(1) 0(0) 1(1) 2(0) 3(2)</td>
</tr>
</tbody>
</table>

Offset address calculation at PPE

Offset for Key-0

Offset for Key-1

Offset for Key-2

Offset for Key-3

Dependence chain [1 -> 4 -> 6]
Parallel Construction of the Dependence Chain

```
for(i=0; i<6; i++)
    A[B[i]] = A[B[i]] + ... ;
```

![Parallel Construction Diagram](attachment:image.png)

Serial execution dependence chain [1->4-> 6]

Mainak Chaudhuri  
Compiling Irregular Accesses for the Cell Processor
Experiments and Results
Iterative Partial Differential Equation (PDE) Solver

**Figure:** Speedup for IRREG

![Graph showing speedup for IRREG](image)
Molecular Dynamics Code

Figure: Speedup for MOLDYN
Nonbonded Force Calculations

**Figure:** Speedup for NBF
Summary
Summary

- Dataflow analysis framework for determining the program points for memory communication.
- Compiler transformation for run-time data processing and actual computation.
- Builds the memory communication schedules.
- Run-time parallelization of loops judiciously partitions the data and computational work.
Thank You