A SYSTOLIC APPROACH
TO
LOOP PARTITIONING AND MAPPING
INTO
FIXED SIZE DISTRIBUTED MEMORY ARCHITECTURES

Ioannis Drositis, Nektarios Koziris, George Papakonstantinou and Panayotis Tsanakas

National Technical University of Athens
Department of Electrical and Computer Engineering
Division of Computer Science

Computing Systems Laboratory
http://www.cslab.ece.ntua.gr
Presentation Overview

- Loop Partitioning and Mapping - *The Systolic Approach*
- Some Terminology
- Communication Cost between Clusters
- The Main Procedure at a Glance
- Analyzing the Main Procedure
- Inductive Definition of h-length
- An Example
- Summarization
- Future Work
Loop Partitioning and Mapping *(the systolic approach)*

Example loop:

```plaintext
for i1 = 1 to 4 do
  for i2 = 1 to 3 do
    for i3 = 1 to 3 do
      (loop body)
    end i3
  end i2
end i1
```

Through a linear transformation $T[n \times n]$:

$$T = \begin{bmatrix} \Pi \\ S \end{bmatrix},$$

where $\Pi[1 \times n]$ and $S[(n-1) \times n]$,

we obtain the array of virtual cells needed to compute the above (initial) index space.

In other words:

$$(i_2', i_3')^T = S \cdot (i_1, i_2, i_3)^T$$

What needed to be done now: cutting the virtual space into *clusters* and assign each cluster to a different processor
The Partitioning Method

Locally Parallel Globally Sequential (LPGS)
where
cardinality of clusters = number of processors

Globally Parallel Locally Sequential (GPLS)
where
number of clusters = number of processors
**Cutting the Virtual Index Space:** The consequences…

Available Processors: 3 → the Virtual (transformed) Space needs to be cut into 3 parts

**FIRST ATTEMPT**

Two horizontal lines, parallel to **horizontal boundary**

**SECOND ATTEMPT**

Two lines, parallel to **side boundary**

**Result statistics:**

- Communication cost = 8 + 8 = 16
- Processor utilization:
  - Processor 1: 5 points
  - Processor 2: 10 points
  - Processor 3: 5 points

✓ Difference in *communication cost* as well as in *processor utilization*
The h-terminology (Part 1/2)

- *h-space*: the *n*-dimensional space that corresponds to loop's indices (and depth)

For **n = 3**, a 3-dimensional (index) space is presented

- *h-plane*: a linear subspace of (n-1)-dimension (a plane in the 3-dimensional space)

For **n = 3**, two 2-dimensional h-planes are presented here, the one perpendicular to the other
The h-terminology (Part 2/2)

- **h-line**: a linear subspace of \((n-2)\)-dimension
  
  *a line in the 3-dimensional space*

  For \(n = 3\), three 1-dimensional h-lines are presented, each one perpendicular to other two

- **h-mesh**: a mesh (of processors usually) in the \((n-1)\)-dimensional space
  
  *an array of cells connected in a mesh topology*

  For \(n = 3\), a 3-dimensional mesh \((3\times2\times3)\) of processors is presented
Communication Costs between Clusters (Introduction)

**CUT**

Cost of a cut = \{ number of transformed dependence vectors that traverse the cut’s h-line \} 
= \{ density of dependence vectors \} \times \{ length of cut \}

**MAPPING**

Cost of a mapping = \sum \{ cost values of its individual cuts \}

So:

Cost of a single cut = \{ length of the cut \} \times \{ overall density (of all dependence vectors) at the direction that is perpendicular to the cut \}

or:

Cost of a single cut = \{ length of the cut \} \times \sum \{ density of each dependence vector on the specified direction \}
Communication Cost between Clusters (continuing…)

Cost of a Single Cut

cost of a single cut = \{ length of the cut \} \times \sum \{ density of each dependence vector on the specified direction \}

\[
\text{cost of a single cut} : c = l \cdot \sum_{i=1}^{m} \left| \frac{p \cdot d'_i}{\|p\|} \right|
\]

where:

- \( m \) is the number of distinct dependence vectors,
- \( p \) is the vector that is perpendicular to the cut,
- \( d'_i \) is a single transformed dependence vector,
- \( \|u\| \) is the Euclidean norm of vector \( u \),
- \( l \) is the \( h \)-length of the segment of the \( h \)-line that corresponds to the cut and is within the bounds of the transformed \( h \)-space.

Cost of a Mapping

cost of a mapping = \{ sum of costs of all cuts that comprise the mapping \}

\[
\text{cost of a mapping} = \sum \{ \text{cost of a single cut} \} = \sum_{k=1}^{m} \left\{ l_k \cdot \sum_{i=1}^{m} \left| \frac{p \cdot d'_i}{\|p\|} \right| \right\}
\]
The Procedure at a Glance

**Algorithm 1**
Calculating the binding h-lines of the transformed index space

**Algorithm 2**
Pre-calculating the cost of any multiple cut (part of a mapping)

**Algorithm 3**
Calculating the cost of any mapping

**Algorithm 4**
Calculating the length cost of any possible cut (parallel to binding h-lines)

**STEP 1**
**STEP 2**
**STEP 3**
**STEP 4**

Finding the mapping with the lower communication cost
Analyzing the Procedure \((Part 1/4)\)

For \(i_1 = 1\) to \(4\) do
  for \(i_2 = 1\) to \(3\) do
    for \(i_3 = 1\) to \(3\) do
      \(\text{(loop body)}\)
    end \(i_3\)
  end \(i_2\)
end \(i_1\)

Boundary points in \(n\)-dimensional index space

Find transformed points and calculate the convex hull of them;

from the convex hull boundaries, calculate virtual space's binding \(h\)-lines.

**Algorithm 1**

Calculate the *binding \(h\)-lines* of the transformed index space

**Determining possible cut directions**
Analyzing the Procedure (Part 2/4)

Implemented by function \textit{cutArea()}:

\begin{algorithm}
\caption{Calculate the length cost of any possible cut (parallel to binding \textit{h-lines})}
\label{alg:cutArea}
\text{cutArea}(i, p_1, p_2, \ldots, p_b, k, \gamma_b, \beta_i, \psi_j)
\end{algorithm}
Analyzing the Procedure (Part 3a/4)

A. Evaluate $\text{depCost}_i$, which is the overall dependence vector density along direction of binding h-line pair $i$.

B. Call several times $\text{cutArea()}$ function with properly specified parameters:

- for all pairs of binding h-lines
- for all combinations of processor-grid arrangement

Algorithm 2

Pre-calculate the cost of any multiple cut (part of a mapping)

What we do in this step

We computes multiple-cut cost for every multiple-cut possible (by lines parallel to binding h-lines)
Analyzing the Procedure (Part 3b/4)

Clustering #1

Cutting lines: a. parallel to binding h-line pairs 3 (lines $\varepsilon_5$ and $\varepsilon_6$) and 1 (lines $\varepsilon_1$ and $\varepsilon_2$) and

b. using three processors along first pair (grid 1st dimension) and four processors along second pair.

Algorithm 2

Pre-calculate the cost of any multiple cut (part of a mapping)
Analyzing the Procedure (Part 3c/4)

Clustering #2

Cutting lines: 

a. parallel to binding h-line pairs 3 (lines $\varepsilon_5$ and $\varepsilon_6$) and 1 (lines $\varepsilon_1$ and $\varepsilon_2$) and 
b. using four processors along first pair (grid 2\textsuperscript{nd} dimension) and three processors along second pair.

Algorithm 2

Pre-calculate the cost of any multiple cut (part of a mapping)
Analyzing the Procedure (Part 4/4)

For any valid mapping, find the mapping cost, by summing all multiple-cut costs that comprise the mapping and keep track of the lower cost.

**Mapping #1**

\[
\text{cost} = \text{mcCost}_{3,2} + \text{mcCost}_{1,1} \\
\text{cost} = \text{depCost}_3 \times \{ \text{cutArea}(3, \ldots, 1, \ldots, \psi_2) + \text{cutArea}(3, \ldots, 2, \ldots, \psi_2) \} + \text{depCost}_1 \times \{ \text{cutArea}(1, \ldots, 1, \ldots, \psi_1) + \text{cutArea}(1, \ldots, 2, \ldots, \psi_1) \}
\]

**Mapping #2**

\[
\text{cost} = \text{mcCost}_{3,1} + \text{mcCost}_{1,2} \\
\text{cost} = \text{depCost}_3 \times \{ \text{cutArea}(3, \ldots, 1, \ldots, \psi_1) + \text{cutArea}(3, \ldots, 2, \ldots, \psi_1) \} + \text{depCost}_1 \times \{ \text{cutArea}(1, \ldots, 1, \ldots, \psi_2) + \text{cutArea}(1, \ldots, 2, \ldots, \psi_2) \}
\]
Inductive Definition of h-length

**Algorithm 5**

*Polygon triangulation to calculate its area*

For $n = 3$, use Euclidean distance

For $n > 3$:
- exclude one point $u$ arbitrarily
- use the same algorithm to calculate the h-length $l'$ of the h-line segment that is defined by the remaining $n-1$ points, in an h-space of dimension $n-2$
- find the projection $u'$ of $u$ on the h-plane defined by the remaining $n-1$ points
- calculate the Euclidean distance $d$ between $u$ and $u'$; the result is the product of $l$ and $d$. 
An Example

for $i_1 = 1$ to $6$ do
  for $i_2 = 1$ to $4$ do
    for $i_3 = 1$ to $3$ do
      $a(i_1,i_2,i_3) = a(i_1,i_2-1,i_3) + a(i_1-1,i_2,i_3) + a(i_1,i_2,i_3-1)$
    end $i_3$
  end $i_2$
end $i_1$

These matrices result in systolic arrays of 42, 24, 12 and 12 cells respectively.

For this problem, optimal transformation methods for systolic arrays produce matrices:

$$T_1 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}, \quad T_2 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \quad T_3 = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix}, \quad T_4 = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

These matrices result in systolic arrays of 42, 24, 12 and 12 cells respectively.
Summarization

A FOR loop

for \( i_1 = 1 \) to 4 do
  for \( i_2 = 1 \) to 3 do
    for \( i_3 = 1 \) to 3 do
      (loop body)
      end \( i_3 \)
    end \( i_2 \)
  end \( i_1 \)
end

The method presented:
finds the lower cost mapping for a given processor grid, using cuts that are parallel to virtual space boundaries.
Future Work

- Intra-processor scheduling
  
  *Mapping that different points correspond to the same time instance and same processor.*

  *How they are executed?*