Loop Transformations for Parallelism & Locality

Last time
- Unimodular transformation framework
- Loop permutation
- Loop reversal
- Loop skewing

Today
- Kelly & Pugh transformation framework
- Loop transformations
  - Loop fusion
  - Loop fission

What are the dependences?

\[
\begin{align*}
do \ i = 1, n \\
\text{s}_1: A(i) &= B(i) + 1 \\
\text{s}_2: C(i) &= A(i)/2 \\
\text{s}_3: D(i) &= 1/C(i+1)
\end{align*}
\]

Fusion changes the dependence between \(s_2\) and \(s_3\), so fusion is illegal

Kelly and Pugh Transformation Framework

Specify iteration space as a set of integer tuples
\[
\{[i, j] \mid 1 \leq i, j \leq n\}
\]

Specify data dependences as mappings between integer tuples (i.e., data dependence relations)
\[
\{[i, j] \rightarrow [i', j'] \mid (i = i' - 1) \land (j = j' - 1) \land (1 \leq i, j, i', j' \leq n)\}
\]

Specify transformations as mappings between integer tuples
\[
\{[i, j] \rightarrow [i', j'] \mid (i' = j) \land (j' = i)\}
\]

Execute iterations in transformed iteration space in lexicographic order

Specify Loop Fusion in Kelly and Pugh Framework

Specify iteration space as a set of integer tuples
\[
IS_1 = \{[1, i_1, 1] \mid 1 \leq i_1 \leq n\}
\]

Specify data dependences as mappings between integer tuples (i.e., data dependence relations)
\[
D_{12} = \{[1, i_1, 1] \rightarrow [2, i_2, 1] \mid i_1 = i_2\}
\]

Specify transformations as mappings between integer tuples
\[
T_1 = \{[1, i_1, 1] \rightarrow [1, i_1', 1] \mid i_1' = i_1\}
\]

Fusion changes the dependence between \(s_2\) and \(s_3\), so fusion is illegal
Checking Legality in Kelly & Pugh Framework

For each dependence, \([I] \rightarrow [J]\) the transformed \(I\) iteration must be executed after the transformed \(J\) iteration.

Loop Fusion Example (cont)

Loop reversal is legal for the original loops
- Does not change the direction of any dep in the original code
- Will reverse the direction in the fused loop: \(s_3 s_2 s_1\) will become \(s_2 s_3 s_1\)

Loop Fission (Loop Distribution)

Idea
- Split a loop nest into multiple loop nests (the inverse of fusion)

Example

```plaintext
do i = 1, n
    body1
    body2
enddo
```

Motivation?
- Produces multiple (potentially) less constrained loops
- May improve locality
- Enable other transformations, such as interchange

Legality?
- Fission is legal when the loop body contains no cycles in the dependence graph

```plaintext
Cycles cannot be preserved because after fission all cross-loop dependences flow from body1 to body2
```
Recall our fusion example

\begin{verbatim}
  do i = 1,n
    s_1 A(i) = B(i) + 1
    s_2 C(i) = A(i)/2
    s_3 D(i) = 1/C(i+1)
  enddo
\end{verbatim}

Can we perform fission on this loop?

\begin{verbatim}
  do i = 1,n
    s_1 A(i) = B(i) + 1
    s_2 C(i) = A(i)/2
    s_3 D(i) = 1/C(i+1)
  enddo
\end{verbatim}

\begin{verbatim}
  do i = 1,n
    s_1 A(i) = B(i) + 1
    s_2 C(i) = A(i)/2
    s_3 D(i) = 1/C(i+1)
  enddo
\end{verbatim}

If there are no cycles, we can reorder the loops with a topological sort

\begin{verbatim}
  do i = 1,n
    s_1 A(i) = B(i) + 1
    s_2 C(i) = A(i)/2
    s_3 D(i) = 1/C(i+1)
  enddo
\end{verbatim}

\begin{verbatim}
  do i = 1,n
    s_1 A(i) = B(i) + 1
    s_2 C(i) = A(i)/2
    s_3 D(i) = 1/C(i+1)
  enddo
\end{verbatim}

Next Time

- Lecture
  - Tiling