Tiling: A Data Locality Optimizing Algorithm

Announcements
- Wednesday doing class surveys

Last Week
- Kelly & Pugh transformation framework
- Loop fusion and fission

Today
- “Unroll and Jam” and Tiling
- Review of the paper “A Data Locality Optimizing Algorithm” by Michael E. Wolf and Monica S. Lam

Loop Unrolling

Motivation
- Reduces loop overhead
- Improves effectiveness of other transformations
  - Code scheduling
  - CSE

The Transformation
- Make n copies of the loop; n is the unrolling factor
- Adjust loop bounds accordingly

Loop Unrolling (cont)

Example
```c
do i=1,n
  A(i) = B(i) + C(i)
enddo
```
```c
do i=1,n by 2
  A(i) = B(i) + C(i)
  A(i+1) = B(i+1) + C(i+1)
enddo
```

Details
- When is loop unrolling legal?
- Handle end cases with a cloned copy of the loop
  - Enter this special case if the remaining number of iterations is less than the unrolling factor

Loop Balance

Problem
- We’d like to produce loops with the right balance of memory operations and floating point operations
- The ideal balance is machine-dependent
  - e.g. How many load-store units are connected to the L1 cache?
  - e.g. How many functional units are provided?

Example
```c
do j = 1,2*n
  A(j) = A(j) + B(i)
enddo
```

What can we do?
- The inner loop has 1 memory operation per iteration and 1 floating point operation per iteration
- If our target machine can only support 1 memory operation for every two floating point operations, this loop will be memory bound
**Unroll and Jam**

**Idea**
- Restructure loops so that loaded values are used many times per iteration

**Unroll and Jam**
- Unroll the outer loop some number of times
- Fuse (Jam) the resulting inner loops

**Example**

do $j = 1, 2n$
do $i = 1, m$
    $A(j) = A(j) + B(i)$
enddo
dendo

**Unroll and Jam Example (cont)**

**Unroll the Outer Loop (cont)**
do $j = 1, 2n$ by 2
do $i = 1, m$
    $A(j) = A(j) + B(i)$
enddo
dendo
do $i = 1, m$
    $A(j+1) = A(j+1) + B(i)$
enddo
dendo

**Jam the inner loops**
do $j = 1, 2n$ by 2
do $i = 1, m$
    $A(j) = A(j) + B(i)$
enddo
dendo
do $i = 1, m$
    $A(j+1) = A(j+1) + B(i)$
enddo
dendo

- The inner loop has 1 load per iteration and 2 floating point operations per iteration
- We reuse the loaded value of $B(i)$
- The Loop Balance matches the machine balance

**Unroll and Jam (cont)**

**Legality**
- When is Unroll and Jam legal?

**Disadvantages**
- What limits the degree of unrolling?
**Concepts**

Unroll and Jam is the same as Tiling with the inner loop unrolled

Tiling can improve ...
- loop balance
- spatial locality
- data locality

**Next Time**

Student Surveys

Lecture
- Beyond Optimization