Reducing Misses by Compiler Optimizations

- Instructions
  - Reorder procedures in memory so as to reduce misses.
  - Profiling to look at conflicts.
  - McFarling [1989] reduced cache misses by 75% on 8KB direct-mapped cache with 4 byte blocks.

- Data
  - Merging Arrays: improve spatial locality by single array of compound elements vs. 2 arrays.
  - Loop Interchange: change nesting of loops to access data in order stored in memory.
  - Loop Fusion: Combine 2 independent loops that have same looping and some variables overlap.
  - Blocking: Improve temporal locality by accessing "blocks" of data repeatedly vs. going down whole columns or rows.

Merging Arrays Example

/* Before */
int val[SIZEx];
int key[SIZex];

/* After */
struct merge {
  int val;
  int key;
};
struct merge merged_array[SIZex];

Reducing conflicts between val & key (spatial locality)

Array Storage (Column order)

- Store first column, then second column, then next column, ...
- Example
  - [6x6] array A
  - each element 1 word (32 bits)
  - Block size = 4 words
  - For i = 1 to 6; X += a[1,i]: How many blocks are loaded into cache?
    - Misses: Compulsory? Conflict? Capacity?

Array Storage (Row order)

- Store first row, then second row, then next row, ...
- Example
  - [6x6] array A
  - each element 1 word (32 bits)
  - Block size = 4 words
  - For i = 1 to 6; X += a[1,i]: How many blocks are loaded into cache?
    - Misses: Compulsory? Conflict? Capacity?

Loop Interchange Example

/* Before */
for (j = 0; j < 100; j = j+1)
  for (i = 0; i < 5000; i = i+1)
    x[i][j] = 2 * x[i][j];

/* After */
for (i = 0; i < 5000; i = i+1)
  for (j = 0; j < 100; j = j+1)
    x[i][j] = 2 * x[i][j];

Sequential accesses (spatial locality)
  - instead of striding through memory every 100 words

Loop Fusion Example

/* Before */
for (i = 0; i < N; i = i+1)
  for (j = 0; j < N; j = j+1)
    a[i][j] = 1/b[i][j] * c[i][j];
  for (i = 0; i < N; i = i+1)
    for (j = 0; j < N; j = j+1)
      d[i][j] = a[i][j] + c[i][j];

/* After */
for (i = 0; i < N; i = i+1)
  for (j = 0; j < N; j = j+1)
    { a[i][j] = 1/b[i][j] * c[i][j];
      d[i][j] = a[i][j] + c[i][j];}

Temporal locality: 2 misses per access to a & c vs. one miss per access.
Example: Matrix Multiply

\begin{align*}
X &= Y \cdot Z \\
\text{Matrix } X: i \times j \\
\text{Matrix } Y: i \times k \\
\text{Matrix } Z: k \times j
\end{align*}

\begin{align*}
x[i][j] &= r; \\
\text{for } (k = 0; k < N; k = k+1) & \\
& \quad r = r + y[i][k] \cdot z[k][j];
\end{align*}

Blocking Example: Matrix Multiply

\begin{align*}
X &= Y \cdot Z \\
\text{Matrix } X: i \times j \\
\text{Matrix } Y: i \times k \\
\text{Matrix } Z: k \times j
\end{align*}

Reducing Conflict Misses by Blocking

- Conflict misses in caches not FA vs. Blocking size
- Lam et al. [1991] blocking factor of 24 had a fifth the misses vs. 48 despite both fit in cache

Blocking

- Problem:
  - Worst case capacity misses: 2N + N²
  - Working set of matrix elements is too large to fit in cache
- Solution:
  - Sub-divide matrices into smaller groups of working sets that will fit in cache, iterate through all subgroups (improves temporal locality)
  - Maximize accesses to the data loaded into the cache before data is replaced
- Can be done at block or register levels

Blocking Example

- Capacity Misses from 2N + N² to 2N²/B + N²
- B called Blocking Factor
- Block traditionally aimed at capacity misses
  - Assumes conflict misses insignificant or addressed by associativity
  - Blocking \ \Rightarrow \ \text{smaller working set of blocks, therefore may reduce conflicts}
Summary of Compiler Optimizations to Reduce Cache Misses

- 3 Cs: Compulsory, Capacity, Conflict Misses
- Reducing Miss Rate
  - Reduce Misses via Larger Block Size
  - Reduce Misses via Larger Cache Size
  - Reduce Misses via Higher Associativity
  - Reduce Misses via Pseudo-Associativity
- Reducing Misses by Compiler Optimizations
- Remember danger of concentrating on just one parameter when evaluating performance (tradeoffs)

CPUtime = \( X \times (\text{Miss rate} \times \text{Miss penalty}) \times \text{Clock cycle time} \)

Next: Miss penalty…