Example Loop Optimizations
Loop fusion

- Merge two different loops together into a single loop

- Why is this useful? Improve reuse distance!

- May not always be legal
Loop interchange

• Change the order of a nested loop

• This is not always legal – it changes the order that elements are accessed!

• Why is this useful?

• Consider matrix-vector multiply when A is stored in column-major order (i.e., each column is stored in contiguous memory)

\[
y = Ax
\]

```c
for (i = 0; i < N; i++)
    for (j = 0; j < N; j++)
        y[i] += A[i][j] * x[j]
```
Loop interchange

• Change the order of a nested loop

• This is not always legal – it changes the order that elements are accessed!

• Why is this useful?

• Consider matrix-vector multiply when \( A \) is stored in column-major order (i.e., each column is stored in contiguous memory)

```c
for (j = 0; j < N; j++)
  for (i = 0; i < N; i++)
    y[i] += A[i][j] * x[j]
```
Loop interchange

• What about x and y vectors?

• Both vectors have reuse: each element is used $N$ times

• Elements of vector indexed by outer loop has good reuse distance (same element used for each iteration of the inner loop)

• Elements of vector indexed by inner loop has bad reuse distance (same element is accessed after all other elements in the vector are accessed)

• Either have good reuse on the x vector and bad reuse on the y vector or vice versa

```c
for (j = 0; j < N; j++)
    for (i = 0; i < N; i++)
        y[i] += A[i][j] * x[j]
```
Loop interchange

• Change the order of a nested loop

• This is not always legal – it changes the order that elements are accessed!

• Why is this useful?

• Consider matrix-vector multiply when A is stored in column-major order (i.e., each column is stored in contiguous memory)

```plaintext
for (j = 0; j < N; j++)
  for (i = 0; i < N; i++)
    y[i] += A[i][j] * x[j]
```
Loop tiling

- Also called “loop blocking”
- One of the more complex loop transformations
- Goal: break loop up into smaller pieces to get spatial and temporal locality
  - Create new inner loops so that data accessed in inner loops fit in cache
  - Also changes iteration order, so may not be legal

for (i = 0; i < N; i++)
  for (j = 0; j < N; j++)
    y[i] += A[i][j] * x[j]
Loop tiling

• Also called “loop blocking”
• One of the more complex loop transformations
• Goal: break loop up into smaller pieces to get spatial and temporal locality
  • Create new inner loops so that data accessed in inner loops fit in cache
• Also changes iteration order, so may not be legal

```plaintext
for (i = 0; i < N; i++)
  for (j = 0; j < N; j++)
    y[i] += A[i][j] * x[j]

for (ii = 0; ii < N; ii += B)
  for (jj = 0; jj < N; jj += B)
    for (i = ii; i < ii+B; i++)
      for (j = jj; j < jj+B; j++)
        y[i] += A[i][j] * x[j]
```

```
```