419 Compilers II

Tutorial exercise 3

You may find the paper "A Data Locality Optimising Algorithm" by Wolf and Lam useful. You can find a postscript copy in the directory `~phjk/CompilersII`.

Consider the matrix multiply loop

```
for i = 0 to N-1
  for j = 0 to N-1
    for k = 0 to N-1
      C[i,j] := C[i,j] + A[i,k] * B[k,j]
```

Note that in this loop there are no dependences apart from the summation into C[i,j]. However for optimisation on machines with caches we are interested in how data is reused. We are interested in pairs of iterations which *read* the same location. Detecting such reuses involves precisely the same subscript analysis as finding dependences.

1. Draw the three-dimensional iteration space for $N = 4$. On it, show an arc for each reuse (don't try to show every single point and arc, just draw a diagram which shows the structure as clearly as possible).

2. For each of the three matrices, explain what reuse distance vectors are present.

3. For this question only, suppose that words are loaded into the cache in pairs (i.e. the cache line size is two words). Two statement instances reuse a cache line if they refer to either of its constituent words.

   (a) What reuse distance vectors are now present?

   (b) Write down the 3×3 unimodular matrix which represents the interchange of the inner two loops which yields the following:

   ```
   for i = 0 to N-1
     for k = 0 to N-1
       for j = 0 to N-1
         C[i,j] := C[i,j] + A[i,k] * B[k,j]
   ```

   (c) Explain the cache use pattern of this formulation by reference to the reuse pattern as shown in the iteration space.

4. Consider the strip-mined version of the loop (note that strip-mining is always valid):

```
for i = 0 to N-1
 for kk = 0 to N-1 step B
  for k = kk to kk+B-1
   for jj = 0 to N-1 step B
    for j = jj to jj+B-1
      C[i,j] := C[i,j] + A[i,k] * B[k,j]
```

   (a) Write down the reuse distance vectors for this loop nest.

   (b) Write down the 5×5 unimodular matrix which represents the two interchanges which lead to the following:

```
for kk = 0 to N-1 step B
 for jj = 0 to N-1 step B
  for i = 0 to N-1
   for k = kk to kk+B-1
    for j = jj to jj+B-1
      C[i,j] := C[i,j] + A[i,k] * B[k,j]
```

   (c) Explain the cache use pattern of this formulation by reference to the reuse distance vectors after transformation by the matrix.

Paul Kelly, Imperial College, February 1996.