CS 243 Assignment 6

Parallelization and Locality Optimizations

Due: March 12, 2008, 11:00 am

This is a written assignment, every student must hand in his or her homework.

1. Consider the following program:

```
int A[1000];

for (i = 0; i < 1000; i++) {
    for (j = i; j < 1000; j++) {
       A[j-i] = 0;
    }
}
```

   a. Draw the iteration space for this loop, and show the dependences.

   b. Write the access function in the above code as an affine function, in the form of Fi + f.

   c. What is the rank of F?

   d. What is the nullity of F?

   e. What is the basis of the nullspace of F?

   f. Write down the integer linear programs to be solved to find the data dependences in this loop.

   g. Is there data dependence in this code?

   h. Find an affine partitioning that maximizes the degree of parallelism without the communication.

   i. Going beyond parallelization, suggest some technique that can speed up this program.

2. Apply affine partitioning to find the maximum degrees of communication-free parallelism in this code.

```
int A[1000];
int B[1000,1000];

for (i = 0; i < 1000; i++) {
    A[i] = i;          /* s1 */
}
for (j = 0; j < 1000; j++) {
    for (k = 0; k < 1000; k++) {
       B[j,k] = A[k] * B[j,k];   /* s2 */
    }
}
```

a. How many degrees of communication-free parallelism are there in this program?

b. What is the affine partition mapping for statement s1?

c. What is the affine partition mapping for statement s2?

d. Write the SPMD code for this parallelized loop.

3. Consider a programming language that lays out the data in row-major order. That is, consecutive elements in a row are contiguous in memory. Consider a machine with a tiny cache with a capacity of 1024 words, and cache lines are 4 words in length.

```
int A[10000][10000];
for (i = 0; i < 10000; i++) {
    for (j = 0; j < 10000; j++) {
        A[j,i] = A[j,i] + A[j,i];
    }
}
```

a. If we execute the code as it is written, how many cache misses are there?

b. How would you transform the code to improve its cache behavior.

c. How many cache misses are there in the optimized program.

4. Let us look at a real-life example. Below implements LU-decomposition without pivoting.

```
for (i= 1; i < 10000; i++) {             /* Loop1 */
    for (j = i+1; j < 10000; j++) {      /* Loop2 */
        A[j,i] = A[j,i] / A[i,i];        /* s1    */
        for (k = i+1, k < 10000; k++) {  /* Loop3 */
           A[j,k] = A[j,k] - A[j,i]*A[i,k];    /* s2    */
        }
     }
}
```

Question: Is there any synchronization-free parallelism. If so, write the SPMD code and specify the affine partitioning mappings of statements s1 and s2.

a. Answer the question for Loop1.

b. If no synchronization-free parallelism has been found, answer the question for Loop2.

c. If no synchronization-free parallelism has been found, answer the question for the body of Loop2.

d. If no synchronization-free parallelism has been found, answer the question for Loop3.