CS314 Spring 2014

Assignment 8

Due Friday, April 25, **before** class

Problem 1 – Dependencies

```
S1:   a := 3;
S2:   b := 5;
S3:   c := 7;
S4:   read(d);
S5:   if (d > 0) then
        begin
S6:       b := b + 1;
S7:       c := a + 3;
        end
      else
        begin
S8:      a := b + 2;
S9:      b := a - 3;
        end;
S10:  c := b * d;
S11:  e := a + 2;
S12:  print(e);
S13:  print(c);
end.
```

1. Give the statement-level dependence graph for the above program. A node in the statement-level dependence graph represents a statement, and edges represent dependences between the statements (nodes). Label each edge as a **true** data dependence, an **anti** data dependence, an **output** data dependence, or a **control** dependence.

2. Assume that each statement takes 1 cycle to execute. What is the execution time of the sequential code? What is the fastest parallel execution time of the program? You may assume that I/O operations (read, print) can be done in parallel.

3. Extend the notion of statement dependences to procedure call dependences. In a procedure-level dependence graph nodes represent procedure calls, and edges represent dependences between procedure calls. For each procedure "p", assume the following definitions. The set MUST_WRITE(p) is the set of all memory locations that are written by every call to procedure "p", the set MAY_WRITE(p) is the set of memory locations that may be written by a call to procedure "p". Therefore, the following

relation holds: MUST_WRITE(p) $\subset$ MAY_WRITE(p). The sets MUST_READ(p) and MAY_READ(p) have corresponding definitions.

Redefine true, anti, and output dependence for procedure calls based on sets MUST_WRITE(p), MAY_WRITE(p), MUST_READ(p), MAY_READ(p), where "p" is a procedure. Remember that parallelization is based on the notion of preserving dependences, so if there is no dependence bewteen two procedure calls, the calls may be executed in parallel.

Problem 2 – Dependence Analysis

Give the direction vectors, and if possible the distance vectors for all dependences in the following loop nests. State explicitly whether a dependence is a true, anti, or output dependence.

1. ```
   do i = 3, 100
     a(i) = a (i-1) + a(i+1) + a(i-2)
   enddo
   ```

2. ```
   do i = 1, 100
     a(2*i) = a(2*i-1) + a(2*i+1)
   enddo
   ```

3. ```
   do i = 1, 10
     aL(i) = a(5) + aR(i)
   enddo
   ```

   a_L and a_R are write and read accesses, respectively, to the same variable "a".

Problem 3 – Loop Parallelization and Vectorization

```
do i = 2, 100
  do j = 2, 100
S1:   a(i, j) = b(i-1, j-1) + 1
S2:   b(i, j) = a(i, j-1) - 5
  enddo
enddo
```

1. Give the statement-level dependence graph with distance vectors
2. In its current form, can any loop level be parallelized or vectorized? If so, use the doall parallel construct, or a partically vectorized statement (e.g.: c(k, 1:100) = d(k, 1:100)) to show the resulting (partially) parallelized loop.
3. Can you increase the available parallelism by transforming the loop? If so, show the resulting parallel and vectorized versions.
```