CS314 Fall 2018

Assignment 8

Problem 1 - Dependencies

Suppose we have the following program (a sequence of instructions), with each instruction labeled as S<num>:

```
S1:  a := 4;
S2:  b := 2;
S3:  c := 5;
S4:  read(d);
S5:  a := b + 3;
S6:  b := a - 3;
S7:  c := d * b;
S8:  e := a + 6;
S9:  print(c);
S10: print(e);
```

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

(b) 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 (i.e. the critical path)? You may assume that I/O operations (read and print) can be done in parallel.

Problem 2 - Dependence Analysis

Give the source and sink references, the type (whether a dependence is **true**, **anti**, or **output**), and the distance vectors for all dependences in the following loops.

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

(b)
```
do i = 2, 100
    a(3 * i) = a(3 * i - 3) + a(3 * i + 3)
enddo
```

(c)
```
do i = 1, 10
    a(i) = a(5) + a(i)
enddo
```

Use $a_W(i)$ and $a_R(i)$ to annotate the write access to a(i) and the read access to a(i) respectively.

Problem 3 - Loop Parallelization

Given the following nested loop:

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

(a) Give the statement-level dependence graph. Show the dependence graph for statement instances in a part of the iteration space: $i = 2 \ldots 5$, $j = 2 \ldots 5$.

(b) In its current form, can any loop be parallelized? If so, which loop(s)? If not, justify your answer.

(c) Provide one valid affine schedule for statements S_1 and S_2 such that $p(S_1)= C_{11}$*i + C_{12}*j + d_1 and $p(S_2)= C_{21}$*i + C_{22}*j + d_2 in order to achieve synchronization-free parallelism. There could be many possible solutions for $\{C_{11}, C_{12}, C_{21}, C_{22}, d_1, d_2\}$. You only need to provide one feasible solution. (Hint: You can let $d_1 = d_2 = 0$.)

(d) Generate two-level loop code for the affine schedule you provided. Please use p as outermost loop and i as innermost loop in the transformed loop. Calculate the loop bounds for p and i using Fourier-Motzkin elimination. You might need to calculate the overlapping polyhedron for S_1 and S_2 in order to eliminate the j loop. Please refer to the techniques for code generation in lecture 20 and lecture 21.