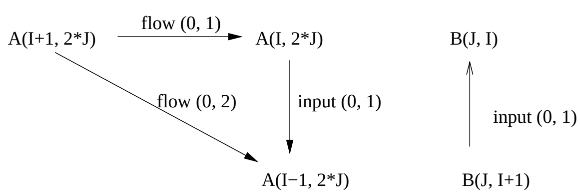
CS5130/CS4130 Homework 4 Solution

1. For the Fortran loop nest (column-major order) below, do the following.

```
DO J = 1, N, 2
  DO I = 1, N
      A(I+1,  2*J) = A(I,  2*J) + B(J,I) + 10.
      A(I,    2*J+1) = A(I-1, 2*J) + B(J,I+1)
  ENDDO
ENDDO
```

   (a) (5 points) Draw the dependence graph and annotate each edge with its distance vector.



   (b) (5 points) What are the reference groups with respect to loop I and loop J?
   Reference groups with respect to loop I.
   Ref group 1: {A(I+1, 2*J), A(I, 2*J), A(I-1, 2*J)}
   Ref group 2: {A(I, 2*J+1)}
   Ref group 3: {B(J, I), B(J, I+1)}

   Reference groups with respect to loop J.
   Ref group 1: {A(I+1, 2*J), A(I, 2*J), A(I-1, 2*J}. All because of spatial reuses.
   Ref group 2: {A(I, 2*J+1)}
   Ref group 3: {B(J, I)}
   Ref group 4: {B(J, I+1)}

   (c) (5 points) What are the loop costs for loop I and J? Is loop interchange legal? Will loop interchange improve the locality of the nest?
   We assume that cache line size is cls.

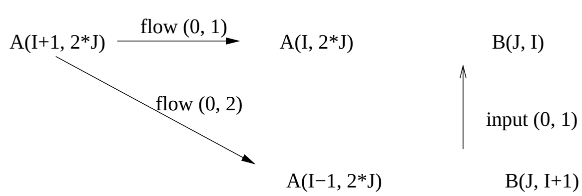
$$
\begin{aligned}
& Loop\ cost\ for\ loop\ I \\
= \; & cost(RefGroup1) + cost(RefGroup2) + cost(RefGroup3) \\
= \; & \frac{N}{2} * \frac{N}{cls} + \frac{N}{2} * \frac{N}{cls} + \frac{N}{2} * N
\end{aligned}
$$

$$
\begin{aligned}
&= \frac{2+cls}{2*cls}N^2 \\
&= \frac{3}{4}N^2(cls = 4words) \\
or \quad & \frac{9}{16}N^2(cls = 16words)
\end{aligned}
$$

$$
\begin{aligned}
& Loop\ cost\ for\ loop\ J \\
= \quad & cost(RefGroup1) + cost(RefGroup2) + cost(RefGroup3) + cost(RefGroup4) \\
= \quad & \frac{N}{2} * N + \frac{N}{2} * N + \frac{N/2}{cls/2} * N + \frac{N/2}{cls/2} * N \\
= \quad & \frac{2+cls}{cls}N^2 \\
= \quad & \frac{3}{2}N^2(cls = 4words) \\
or \quad & \frac{9}{8}N^2(cls = 16words)
\end{aligned}
$$

Loop interchange is legal because no dependence is reversed. Loop integer change will not improve the locality since the inner loop cost is lower.

(d) (20 points) We are going to apply scalar replacement on the loop nest. What is the pruned dependence graph? What are the generators? Show the code after scalar replacement.



A(I+1, 2*J) and B(J, I+1) are generators.

The code after scalar replacement is as shown in the next page which also lists the code removed during the process as comments. Note that you can also use just one register for B.

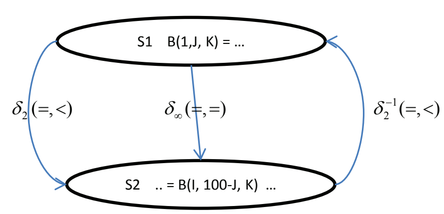
```
DO J = 1,N,2
C peel of 2 iterations
C peeled iteration 1
       A$0$0 = A(1, 2*J) + B(J, 1) + 10.
       A(2, 2*J) = A$0$0
       B$1$0 =  B(J,2)
       A(1,     2*J+1) = A(0, 2*J) + B$1$0
C      A$0$2 = A$0$1
       A$0$1 = A$0$0
       B$1$1 = B$1$0

C peeled iteration 2
       A$0$0 = A$0$1 + B$1$1 + 10.
       A(3, 2*J) = A$0$0
       B$1$0 =  B(J,3)
       A(2, 2*J+1) = A(1, 2*J) + B$1$0
       A$0$2 = A$0$1
       A$0$1 = A$0$0
       B$1$1 = B$1$0

  DO I = 3, N, 3
C iteration 0
       A$0$0 = A$0$1 + B$1$1 + 10.
       A(I+1, 2*J) = A$0$0
       B$1$0 =  B(J,I+1)
       A(I, 2*J+1) = A$0$2 + B$1$0
C      A$0$2 = A$0$1
C      A$0$1 = A$0$0
C      B$1$1 = B$1$0

C unrolled iteration 1
       A$0$2 = A$0$0 + B$1$0 + 10.
       A(I+2, 2*J) = A$0$2
       B$1$1 =  B(J,I+2)
       A(I+1, 2*J+1) = A$0$1 + B$1$1

C unrolled iteration 2
       A$0$1 = A$0$2 + B$1$1 + 10.
       A(I+3, 2*J) = A$0$1
       B$1$0 =  B(J,I+3)
       A(I+2, 2*J+1) = A$0$0 + B$1$0

       B$1$1 = B$1$0
  ENDDO
ENDDO
```

2. (15 points) Consider the following Fortran loop nest:

```
      DO K = 1, 100
        DO J = 1, 100
S1        B(1, J, K) = A(1, J-1, K)
          DO I = 1, 100
S2          A(I+1, J, K) = B(I, 100-J, K) + C
          ENDDO
        ENDDO
      ENDDO
```

Does statement S_2 depend on statement S_1? Does statement S_1 depend on statement S_2? Given the dependence type, direction vector, and array variable involved for each dependence that exists. Vectorize this loop based on the *AdvancedVectorization* algorithm in the notes. (Please note that the first subscripts in both references of S_1 are one not I).

There is a J-loop-carried true dependence from S_1 to S_2 and a J-loop-carried anti-dependence from S_2 to S_1, both of which are from the references to array B. There is also a loop-independence true dependence from S_1 to S_2 when J=50.



Since the dependencies carried by loop J form a cycle, only the innermost loop, loop I, can vectorized.

```
      DO K = 1, 100
        DO J = 1, 100
S1        B(1, J, K) = A(1, J-1, K)
          A(2:101, J, K) = B(1:100, 100-J, K) + C
        ENDDO
      ENDDO
```