Consider relations R(a, b, c), S(a, d), and T(a, e, f) to be joined on the common attribute a.

Assume:

- There are B = 445 pages in the buffer
- Table R spans M = 1.500 pages with 80 tuples per page
- Table S spans N = 4.500 pages with 150 tuples per page
- Table T spans O = 200 pages with 250 tuples per page

Answer the following questions on computing the I/O costs for the joins. You can assume the simplest cost model where pages are read and written one at a time. You can also assume that you will need one buffer block to hold the evolving output block and one input block to hold the current input block of the inner relation where needed. You may ignore the cost of the writing of the final results.

1 Join Cost

Assume that there are no indexes available on the tables to speed up the join algorithms.

a) Compute the cost of a Block Nested loop join with S as the outer relation and R as the inner relation.

b) Then compute the cost of a block nested loop join with R as the outer relation and S as the inner relation.

Solution. ***a)*** *for each group of blocks of S iterate over each block of R*

1. *We reserve 1 block for output and 1 block for the inner table.*
2. *Given 443 blocks then we read all S (cost* 4.500*) in* $\lceil 4.500/443 \rceil = 11$ *iterations*
3. *For each iteration we read all pages of R (cost* 1.500*).*
4. *Final cost is* $4.500 + 11 * 1.500 = 21.000$ *I/O*

Solution. ***b)*** *for each group of blocks of S iterate over each block of R*

1. *We reserve 1 block for output and 1 block for the inner table.*
2. *Given 443 blocks then we read all R (cost* 1.500*) in* $\lceil 1.500/443 \rceil = 4$ *iterations*
3. *For each iteration we read all pages of R (cost* 4.500*).*
4. *Final cost is* $1.500 + 4 * 4.500 = 19.500$ *I/O*

2 Join Optimization

Assume we join S with T. Assume there is B+Tree index on S and assume the estimate cost of access to the index is Q=0.25 I/O. Which option is more convenient between the Hash Join and the Index Nested Loop Join.

Solution. *Considering Hash Join*

1. *All of T (cost* 200*) can fit in the* 443 *blocks in main memory*
2. *Then we build a hash index*
3. *Then we iterate over all pages of S (cost* 4.500*)*
4. *The final cost is* $200 + 4.500 = 4.700$

Considering Index Nested Loop Join

1. *since the index is on S we need to iterate over T.*
2. *Given 200 pages with 250 tuples per page, we have a total of* $200 * 250 = 50.000$ *tuples*
3. *Then we read all pages of T (cost* 200*) and search the index for each tuple.*
4. *A cost of 0.25 means that on average 3 out of 4 times all the pages of the index we need are in memory, and 1 out of 4 times we need to read a single new page.*
5. *The final *estimated* cost is* $200 + 50.000 * 0.25 = 12.700$.