CMPUT 391: Database Management Systems

Assignment 5

Due date: March 26, 2004 **Due in class**

Question 1:

Assume you have the following tables in your database:

Courses(CourseID, Name, Description)
StudentRegistration(StudID, CID, Year)

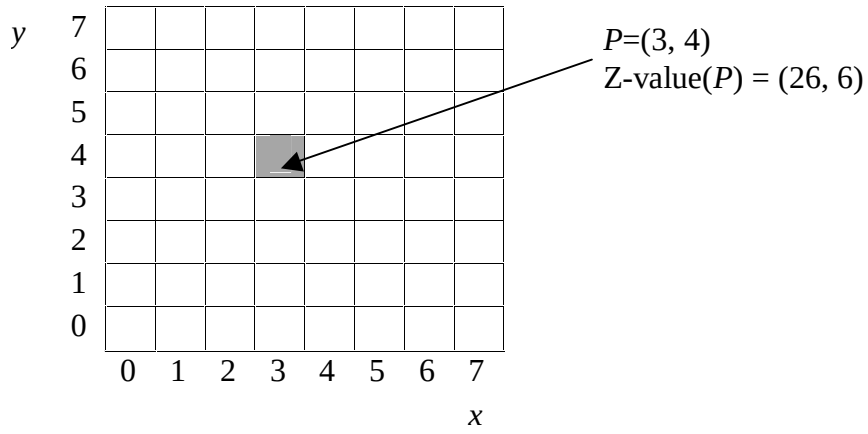
Courses contains the names, ID, and a description of all courses. There are 10,000 courses, 100 per page. StudentRegistration contains the courses in which each student was registered and the year of registration. The relation covers 5 years from 1999 to 2003 and has 500,000 tuples, 50 tuples per page.
You would like to get pairs of student ID and course name for all students that registered for a course after the year 2000.

1) Write the query in SQL, then write it in relational algebra pushing the selection as early as possible.
2) Draw a possible query tree for the query you provided in 1).
3) Estimate the cost, in number of I/Os, of the execution of the query if we use Block Nested Loop Join (BNLJ), Sort Merge Join (SMJ), and Index Nested Loop Join (INLJ), knowing that we have a non clustered hash index for the relation Courses on CourseID. Avoid whenever possible the creation of intermediary tables. Be careful which relation should be outer or inner in each case.

Assume that you have 10 buffer pages in main memory, and assume a uniform distribution of values whenever necessary.

Question 2

1) Assume a $2^L \times 2^L$ grid where L is an integer. Give a pseudo-code algorithm that computes the decimal c of the Z-value(P)=(c, l) for any given cell P=(x, y) in the grid. E.g.:

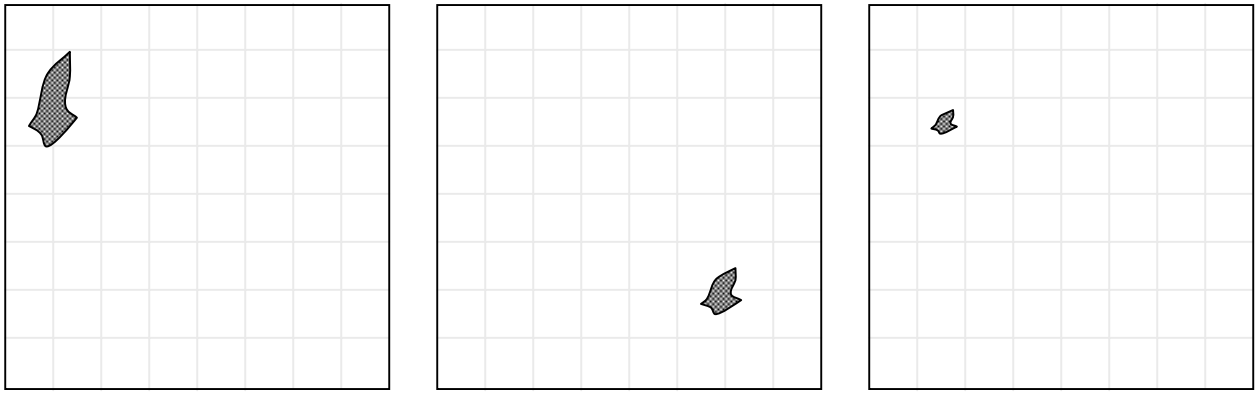


The algorithm should not be based on a recursive procedure that partitions the space, but on the following two procedures, which you can call in your pseudo-code:

- *decimal_to_binary*(n, b): computes the binary representation b of an integer n.
- *binary_to_decimal*(b, n): computes the decimal value n of a bit-array b.

where $0 \leq n < 2^{2L}$ is an integer and b is a bit-array of length $2L$, which contains the binary representation of n – including leading zeros if necessary.
Tip: Think about the binary representation of the x and y coordinates of a cell and their relation to a recursive partitioning of the space into halves.

2) Compute the Z-Values of the minimum enclosing cell (as explained in class) for the following polygons. Draw a line on the grid whenever a division of the space is necessary.



3) Give a simple pseudo-code algorithm that computes the spatial join of objects stored in two R-trees R_1 and R_2. You can assume that both R-trees R_1 and R_2 have the same height.

Question 3:

Assume you have the following database objects storing the given values:

A = 10
B = 20
C = 30
D = 40
E = 50
F = 60

You can assume that each object is stored on a different disk page in the database.

Consider also the following transactions:

T1: R(A), W(A:77); R(B), Commit
T2: R(B), W(B:25), W(E:55), Commit
T3: R(A), W(D:45), R(F), Commit
T4: R(C), W(C:35), R(F), Commit

Where an operation W(X:V) means that the transaction want to write the value V onto object X.

Create a schedule according to Strict 2PL with wait-die rule as the deadlock resolution mechanisms, and in addition, in each step of the schedule, describe the content of the log buffer and the database cache in main memory, as well as the content of the log file and the database pages on disk. You can assume that the cache can hold 10 pages, and the log buffer can hold a maximum of 50 entries. Your schedule should be elaborated until T2 commits. Assume that immediately after T2 commits, the system crashes, and describe what has to be done in the recovery process of the system.

The transactions should be scheduled in a *round-robin fashion*, i.e., first, transaction T1 is allowed to execute its first action, then T2, then T3, then T4, and after that again T1, and so on – if that is possible (a transaction may have to wait, in which case the next transaction can try to execute its next action).

The schedules should include not only the read and write operations of the transactions, but also the lock requests (e.g. “request Xlock(C), granted”, “request SLock(A), denied”, “Wait”, “Abort/Rollback”, or “request lockUpgrade”). Assume that if a lock request is granted the corresponding read or write operation is executed immediately, i.e., lock request and the read or write are counted as one action for the scheduling.

After 7 log entries, the system should create a sharp checkpoint in the logfile (i.e., every 8^{th} entry in you log file must be a checkpoint entry).