University of Edinburgh
School of Informatics
INFR11199 - Advanced Database Systems (Spring 2024)

Tutorial Sheet 5

1. (Transactions & Concurrency) Consider a database with objects X and Y and two transactions. Transaction 1 reads X and Y and then writes X and Y. Transaction 2 reads and writes X then reads and writes Y.

   (a) Create a schedule for these transactions that is *not* serializable. Explain why your schedule is not serializable.

   (b) Would your schedule be allowed under strict two-phase locking? Why or why not?

   Now consider the following schedule:

	1	2	3	4	5	6	7	8
T1				X(B)	X(A)			S(D)
T2		X(D)	S(E)					
T3	X(E)						S(B)	
T4						X(A)		

   (c) Draw the waits-for graph for this schedule.

   (d) Are there any transactions involved in deadlock?

   (e) Next, assume that T1 priority > T2 > T3 > T4. You are the database administrator and one of your users purchases an exclusive plan to ensure that their transaction, T2, runs to completion. Assuming the same schedule, what deadlock avoidance policy would you choose to make sure T2 commits?

2. (Parallel Query Processing) For each of the following scenarios, state whether it is an example of:

   - Inter-query parallelism
   - Intra-query, inter-operator parallelism
   - Intra-query, intra-operator parallelism

- No parallelism

(a) A query with a selection, followed by a projection, followed by a join, runs on a single machine with one thread.

(b) Same as before, but there is a second machine and a second query, running independently of the first machine and the first query.

(c) A query with a selection, followed by a projection, runs on a single machine with multiple threads; one thread is given to the selection and one thread is given to the projection.

(d) We have a single machine, and it runs recursive hash partitioning (for external hashing) with one thread.

(e) We have a multi-machine database, and we are running a join over it. For the join, we are running parallel sort-merge join.

3. (Parallel Query Processing) Suppose we have 4 machines, each with 10 buffer pages. Machine 1 has a `Students` table which consists of 100 pages. Each page is 1 KB, and it takes 1 second to send 1 KB of data across the network to another machine.

   (a) How long would it take to send the data over the network after we uniformly range partition the 100 pages? Assume that we can send data to multiple machines at the same time.

   (b) Next, imagine that there is another table, `Classes`, which is 10 pages. Using just one machine, how long would a BNLJ take if each disk access (read or write) takes 0.5 seconds?

   (c) Now assume that the `Students` table has already been uniformly range partitioned across the four machines, but `Classes` is only on Machine 1. How long would a broadcast join take if we perform BNLJ on each machine? Do not worry about the cost of combining the output of the machines.

   (d) Which algorithm performs better?

   (e) Knowing that the `Students` table was range partitioned, how can we improve the performance of the join even further?

4. (Two-Phase Commit with Logging) Suppose we have one coordinator and three participants. It takes 30ms for a coordinator to send messages to all participants; 5, 10, and 15ms for participant 1, 2, and 3 to send a message to the coordinator respectively; and 10ms for each machine to generate and flush a record. Assume that for the same message, each participant receives it from the coordinator at the same time.

   (a) Under 2PC with logging, how long does the whole 2PC process (from the beginning to the coordinator's final log flush) take for a successful commit in the best case?

(b) Now in the 2PC protocol, describe what happens if a participant receives a PREPARE message, replies with a YES vote, crashes, and restarts (All other participants also voted YES and didn't crash).

(c) In the 2PC protocol, suppose the coordinator sends PREPARE message to Participants 1 and 2. Participant 1 sends a "VOTE YES" message, and Participant 2 sends a "VOTE NO" message back to the coordinator.

   i. Before receiving the result of the commit/abort vote from the coordinator, Participant 1 crashes. Upon recovery, what actions does Participant 1 take (1) if we were not using presumed abort, and (2) if we were using presumed abort?

   ii. Before receiving the result of the commit/abort vote from the coordinator, Participant 2 crashes. Upon recovery, what actions does Participant 2 take (1) if we were not using presumed abort, and (2) if we were using presumed abort?

(d) In the 2PC protocol, suppose that the coordinator sends PREPARE messages to the participants and crashes before receiving any votes from the participants. Assuming that we are running 2PC with presumed abort, answer the following questions.

   i. What sequence of operations does the coordinator take after it recovers?

   ii. What sequence of operations does a participant who received the message and replied NO before the coordinator crashed take?

   iii. What sequence of operations does a participant who received the message and replied YES before the coordinator crashed take?

   iv. Let's say that the coordinator instead crashes after successfully receiving votes from all participants, with all participants voting YES except for one NO vote. Assuming the coordinator sees no records for this transaction in its log after coming back online, how does this affect the answers to parts (d).i-(d).iii.?