CS632 Endsem, Spring 2016, Apr 22, 2016, 9.30 AM - 12.30 PM

Total Marks: 100

NOTE: Answer all subparts of a question together, do not split them up.

1. Short answers:
   (a) Spanner requires commit wait before releasing locks. Suppose that the average error in time ϵ is 5 msec, and suppose there is an application where every transaction needs to access/update a single shared tuple. What is the maximum throughput possible in this setting. ...3
   (b) Given the above, it would be a bad idea to update shared items as part of a transaction. Consider the case where the update simply adds a value to one of multiple accumulators, and the value is not read by the query. What would be a good way to architect the system to improve throughput. ...4
   (c) Describe a few key differences between the normal partitioned hash join, and the hash join technique in the Morsel driven evaluation paper. Also explain why some of these difference in the Morsel papers hash join would be a bad idea, use unless you have a shared memory system. ...4

2. Declarative Networking and Datalog
   (a) Give a recursive Datalog program to compute a view $pathk(X, Y)$ containing all pairs (ni, nj) such that nj can be reached by a path of up to length k edges, where k is a constant. ...4
   (b) Now consider the "doubly-recursive" path program
   $path(X, Y) :- \ edge(X, Y)$.
   $path(X, Y) :- \ path(X, Z),\ path(Z, Y)$
   The standard "linear" form of the program has the first occurrence of *path* above replaced by *edge*.

   Suppose the edges in a particular graph form a chain from $n1, n2, ..., n128$. Compare the doubly-recursive and linear path program, in terms of:
      i. Number of steps of derivation needed to derive $(n1, n128)$. ...4
      ii. Number of times the above fact is derived. ...4
   (c) Give pseudocode for a Pregel program corresponding to the linear version of path Datalog program. You can assume that each node has an InEdge iterator, and a set of nodes reachableNodes containing nodes that have been inferred to be reachable from the current node. ...10

3. Asynchronous and Batched Query Submission.
   (a) Give a small example program where rule A cannot be applied to a query, due to an external anti dependency arising from a database update statement. ...6
   (b) Suppose you have an application with a query inside a loop where rule A is applicable, but where the application may exit after some number of loop iterations if the user is satisfied with that many answers. The application does not know how many iterations are required.

   Explain which of the extensions/optimizations in Section 5 are the most important in this context. Explain your answer. ...6
   (c) For the above case, what is the risk of having the producer in a separate thread? ...3

4. BANKS

   (a) Consider the node weights computed using PageRank (assume for simplicity that all edge weights are equal to 1). And consider each of the following applications in turn:

      i. A movie database with relations
      `movie(movieID, title)`, `actor(actorID, firstname, lastname)`, and
      `actedIn (movieID, actorID)`.
      The keyword queries can be assumed to provide part of the name of an actor or movie title, which can match multiple actors/movies.

      ii. A sales database with relations
      `item(itemID, name, description)`, `customer(custId, name)`, and
      `order(orderId, custId, itemId)`
      The keyword queries can be asumed to provide words that appear in the name or description of multiple items.

   For each application above, discuss whether BANKS' node weighting scheme is likely to lead to a good ranking of nodes, and why. ...4+4

   (b) The Backwards Expanding search algorithm in the paper (which we now call the MultipleIterator, or MI algorithm) creates an iterator starting from every node containing a given keyword. In constrast the Single Iterator (or SI) algorithm in a subsequent paper creates only a single iterator for a given keyword.

   Give a small example showing an answer which the MI algorithm can generate, which the SI algorithm will not generate. ...7

5. Streaming

   (a) In Millwheel, consider the following steps which are part of exactly once delivery:
   1) Duplicate checking is done; 2) user code is run; 3) Pending changes are committed to the backing store; 4) Senders are ACKed; and 5) Pending downstream productions are sent

      i. How does Millwheel ensure that step 5 will certainly happen, once the previous steps are completed? ...5

      ii. What happens if an ACK is lost? ...2

   (b) Aurora places a lot of emphasis on load shedding, which is not even mentioned in MillWheel. What does MillWheel do if there is an increase in the load? Explain briefly. ...4

   (c) Give an example of a windowing operator that Aurora and Millwheel do not support, which is supported (and mentioned as important in) in the DataFlow paper. ...3

   (d) The appropriate cost metric for a streaming query plan is how many input tuples it can process per second.

   Consider a streaming query plan which is just a simple sequence of n selects $\sigma_1, \ldots \sigma_n$, each with selectivity s_i and cost c_i seconds to process one tuple. Assume that select predicates are independent of each other. If the whole plan runs on a single one-core system, give a formula to estimate many tuples it can process per second.

   Now if each select runs on a separate core/process, ignoring synchronization and communication overheads, give a formula for estimating how many tuples can be processed per second. ...4+4

contd.

6. RDF-3X

   (a) Suppose you have a triple pattern ?x TeacherOf ?y which is part of a query. Which triple indices are potentially useful for a query containing this pattern? Consider the case where ?y is used elsewhere in the query, as well as the case where it is not used anywhere else. ...4

   (b) RDF-3X uses index lookups for selections, but not for indexed-nested-loops joins. Instead, an index scan is used to generate tuples in sorted order, and merge join is done using the sort ordering. Merge join is inefficient if one relation is very small while another is very large. The scan interface can easily be extended to support a function jumpTo(val), which uses the index to skip to the smallest value $\geq$ val. Explain how this may be used to optimize merge joins. (This was used in a follow up paper.) ...6

   (c) RDF-3X maintains statistics of frequent paths. What is their definition of frequent? If you have a workload of queries, would you define the notion of frequent differently? Suggest what notion of frequent you may use in his situation. ...5

Total Marks = 100