Examination Queries & Retrieval

December 8, 2014
9:00 – 10:45 BBL 061

- You may use a cheat sheet (A4)
- Mention your name and student number on every answer sheet
- Write and formulate clearly
- Always explain your answers
- There are **three** exercises
- The maximum amount of points you can earn is 40.
- Good luck

[1] Map Reduce (10 points)

Our task is to take a very large file of integers as input and to produce as output the count of the number of distinct integers in the input. Write pseudo code for Map and Reduce. Performance is critical.

[2] SSD (8 points)

Use at most 200 words (~= half a page) to explain the Flash Join: how does it work, which are the claimed advantages and why are these advantages specifically useful for SSD. You may add figures.

please turn the page

[3] Division: rewriting and methods (22 points: 6+4+4+4+4)

A somewhat notorious algebraic operator is the *division* (%). We give a definition of this binary operator based on a left operand *T* with two attributes *A,B* and a right operand *U* with one attribute *B*. This definition can be generalized to more attributes. We suppose *set* oriented semantics in this exercise: no duplicates in the operands.

The result of *T[A,B] % U[B]* has schema *[A]*. It contains the set of *a*-values from the left column of *T* such that for every *b* in *U*, there exists a tuple *<a,b>* in *T*. In other words, it contains the *a*-values from *T[A]* such that the corresponding *b*-values cover *U*.

division $T[A,B] \div U[B]$

T	A	B
	1	1
	1	3
	1	4
	2	2
	2	4
	6	1
	6	3
	8	1
	8	3
	8	4
	8	7

U	B
	1
	3
	4

$T \div U$	A
	1
	8

Note that in the figure, *T* is sorted on *A*. In general, this does not need to be the case.
This operator typically represents queries that involve universal quantification when expressed in logic, like "give the students who passed all first year courses".

(i) Give a algebraic equivalence rule for the expression $\sigma_p(T \,\%\, U)$.
Give a short explanation how this rule could help query processing.

We are going to investigate which join-like algorithms could be used to calculate the result of a division. Note that this binary operator is not commutative. You should take the asymmetry in account for a method when necessary. For algorithmic analysis, use B1 = B(*T*) and B2 = B(*U*) for the number of blocks and T1 = T(*T*) and T2 = T(*U*) for the number of tuples. Consider also the (likely) possibility that one of the operands might fit in main memory. Apply modifications to the join-based algorithm where necessary.

(ii) Could the division be calculated in a way that resembles the sort-merge join? Explain. If so, give an algorithmic analysis.
(iii) Could the division be calculated in a way that resembles nested block loop? Explain.
(iv) Could the division be calculated in a way that exploits hashing? Explain.
(v) Could the division be calculated by making use of available indexes? Explain.

Finish