CS186 Discussion Section 7
Aggregation and Query Optimization
10/22/2013

Review: Sorting & Hashing

a) Consider a simple multi-pass external merge sort algorithm that in the first pass, produces disk-based runs of “B” pages where B is the number of pages of memory available to the algorithm. If B = 50 pages, how many passes (including pass 1) of the algorithm will be required to sort a relation of size 5000 pages and how many runs will be produced by each of the passes?

b) Suppose we use the in memory sort optimization that generates runs of (on average) length 2B during pass 1. If B = 50, how many passes are required (including pass 1) and how many runs will be produced by each of the passes?

c) In what scenarios might we prefer to use hashing instead of sorting?

Aggregations/Query Optimization

Consider the following schema:

Car(license, owner_ssn, year, company, model)
Accident(license, accident_date, damage_amount, zipcode)
Owner(ssn, license, name, gender, street, city, zipcode)

NTuples(Car) = 1000 ;	NPages(Car) = 100
NTuples(Accident) = 500 ;	NPages(Accident) = 20
NTuples(Owner) = 800 ;	NPages(Owner) = 50

NDistinct(Car.company) = 50;

For the following questions, assume that we have B=20 pages of available in-memory buffer space.

a) For the query: “SELECT gender, COUNT(*) FROM Owner GROUP BY gender;” What is the IO cost for aggregation by sorting?

b) For the query in part a), what is the IO cost for aggregation by hybrid hashing?

c) For the query: “SELECT * FROM Accident A, Car C WHERE A.license = C.license AND A.damage_amount > X;” For what types of values of X would selection push-down significantly improve the cost of the query (Car is the inner table of the join)?’’

d) For the query: “SELECT O.name FROM Car C, Owner O WHERE C.license = O.license AND C.company = ‘Volvo’;” What is the expected cardinality of the Car relation after the initial selections are applied?