COMP207 Tutorial Exercise Solutions
Week 7 (16^{th}/18^{th} November)

The exercises below provide the opportunity to practice the concepts and methods discussed during previous week's videos/reading material. If you haven't done so, it is worthwhile to spend some time on making yourself familiar with these concepts and methods. Don't worry if you cannot solve all the exercises during the tutorial session, but try to tackle at least one or two of them. If at some point you do not know how to proceed, you could review the relevant material from the videos/reading material and return to the exercise later.

1 Distributed databases

The first question is not meant to be technical, but is more meant for understanding why we have these forms of partitioning.

Exercise 1 (Adapted from exercise 23.28 in [1])**.** Consider the following database for a chain of book stores in the USA: Books(Book_ no,Primary_ author,Topic,Title,Total_ stock,Price) Bookstore(Store_ no,City,State,Zip,Inventory_ value) Stock(Store_ no,Book_ no,Qty)

Total_ stock is the total number of books in stock and Inventory_ value is the total value of all books in that store.

(a) Give an example of two simple predicates that would be meaningful for the Bookstore relation for horizontal partitioning (consider perhaps warehouses that sends to a number of nearby stores)

(b) Give an example of a meaningful vertical partitioning for the Books relation (consider perhaps a sales register compared to headquarters compared to customers searching for books)

Solutions:

(a) There is a number of meaningful choices, e.g. City (and state), if there are very many shops, state if there are a few in each state and Zip (divided in some sensible way). E.g. you might say that all information about a range of Zip numbers are stored in the same database in some werehouse from which you deliver to those zip codes. Same with the others.

(b) In a sales register, you mainly need Book_ no and price (maybe Primary_ author and Title if you want to show the information on the bill). In headquaters, you might care more about book_ no and total_ stock and perhaps Topic and maybe some of the rest. If a customer is searching for books, most except total_ stock seems useful. So, you could have a server with each of these sets of information.

Exercise 2 (Exercise 20.4.1 in [2])**.** Suppose we want to take the natural join of $R(A,B)$ and $S(B,C)$, where R and S are at different sites, and the size of the data communicated is the dominant cost of the join. Suppose the sizes of R and S are s_R and s_S, respectively. Suppose that the size of $\pi_B(R)$ is fraction p_R of the size of R and $\pi_B(S)$ is fraction p_S of the size of S. Finally, suppose that fractions d_R and d_S of relations R and S, respectively, are dangling (i.e. $(1-d_R)|R| = |R \ltimes S|$ and $(1-d_S)|S| = |S \ltimes R|$). Write expressions, in terms of these six parameters, for the costs of the four strategies for evaluating $R \bowtie S$, and determine the conditions under which each is the best strategy. The four strategies are:

(a) Ship R to the site of S.

(b) Ship 5 to the site of R.

(c) Ship $\pi_B(S)$ to the site of R, and then $R \ltimes S$ to the site of S.

(d) Ship $\pi_B(R)$ to the site of S, and then $S \ltimes R$ to the site of R.

Solutions:

We need to come with expressions for when the different options are best. First, observe that the communication done by option (a) is s_R, by option (b) is s_S, by option (c) is $p_S s_S + (1-d_R)s_R$ and by option (d) is $p_R s_R + (1-d_S)s_S$. Let f be $\frac{s_R}{s_S}$.

For option (a) to be the best option, we need

$$s_R = min(s_R, s_S, p_S s_S + (1-d_R)s_R, p_R s_R + (1-d_S)s_S)$$

That is:

(a) $s_R \leq s_S \Rightarrow f = \frac{s_R}{s_S} \leq 1$

(b) $s_R \leq p_S s_S + (1-d_R)s_R \Rightarrow f = \frac{s_R}{s_S} \leq \frac{p_S}{d_R}$

(c) $s_R \leq p_R s_R + (1-d_S)s_S \Rightarrow f = \frac{s_R}{s_S} \leq \frac{1-d_S}{1-p_R}$.

In other words, we need $f \leq \min(1, \frac{p_S}{d_R}, \frac{1-d_S}{1-p_R})$. Option (b) is similarly best if $1/f \leq \min(1, \frac{p_R}{d_S}, \frac{1-d_R}{1-p_S}) \Rightarrow$ $f \geq \max(1, \frac{d_S}{p_R}, \frac{1-p_S}{1-d_R})$.

For option (c) to be the best, we need that

$$p_S s_S + (1-d_R)s_R = \min(s_R, s_S, p_S s_S + (1-d_R)s_R, p_R s_R + (1-d_S)s_S)$$

That is:

(a) $p_S s_S + (1-d_R)s_R \leq s_R \Rightarrow f = \frac{s_R}{s_S} \geq \frac{p_S}{d_R}$

(b) $p_S s_S + (1-d_R)s_R \leq s_S \Rightarrow f = \frac{s_R}{s_S} \leq \frac{1-p_S}{1-d_R}$

(c) $p_S s_S + (1-d_R)s_R \leq p_R s_R + (1-d_S)s_S) \Rightarrow f = \frac{s_R}{s_S} \leq \frac{1-d_S-p_S}{1-d_R-p_R}$

In other words, option (c) is best if $f \in [\frac{p_S}{d_R}, \min(\frac{1-p_S}{1-d_R}, \frac{1-d_S-p_S}{1-d_R-p_R})]$. Option (d) is similarly best if

$$1/f \in \left[\frac{p_R}{d_S}, \min\left(\frac{1-p_R}{1-d_S}, \frac{1-d_R-p_R}{1-d_S-p_S}\right)\right] \Rightarrow f \in \left[\max\left(\frac{1-d_S}{1-p_R}, \frac{1-d_S-p_S}{1-d_R-p_R}\right), \frac{d_S}{p_R}\right]$$

Exercise 3 (Exercise 20.5.2 in [2])**.** In this exercise, we need a notation for describing sequences of messages that can take place during a two-phase commit. Let (i, j, M) mean that site i sends the message M to site j, where the value of M and its meaning can be P (prepare), R (ready), D (don't commit), C (commit), or A (abort). We shall discuss a simple situation in which site 0 is the coordinator, but not otherwise part of the transaction, and sites 1 and 2 are the components. For instance, the following is one possible sequence of messages that could take place during a successful commit of the transaction:

$$(0,1,P), (0,2,P), (2,0,R), (1,0,R), (0,2,C), (0,1,C)$$

(a) Give an example of a sequence of messages that could occur if site 1 wants to commit and site 2 wants to abort.

(b) How many possible sequences of messages such as the above are there, if the transaction successfully commits?

(c) If site 1 wants to commit, but site 2 does not, how many sequences of messages are there, assuming no failures occur?

(d) If site 1 wants to commit, but site 2 is down and does not respond to messages, how many sequences are there? Some of the sequences are more reasonable than others in this scenario, try to distinguish which is which.

Solutions:

(a) An example of a sequence of messages that could occur if site 1 wants to commit and site 2 wants to abort, could be (we will later see how many there are)

$$(0,1,P), (0,2,P), (1,0,R), (2,0,D), (0,1,A)$$

(b) We have to count how many sequences there are if the transactions commits. We have that $(0,2,C)$ and $(0,1,C)$ must both be later than any other operation, but can be in any order. Therefore, we can ignore by multiplying the rest by 2 (for each of the two orderings of those two messages). For now, let us call $A = (0,1,P)$, $B = (0,2,P)$, $C = (1,0,R)$ and $D = (2,0,R)$. We have $4! = 4*3*2 = 24$ different ways to order those messages, if we had no requirements on it (i.e. there are 4 options for the first position, when we have picked that there are three for the next, and so on). However, we must have that $(0,i,P)$ is before $(i,0,R)$. Observe that there is a one-to-one correspondence between sequences where $A = (0,1,P)$ is after $C = (1,0,R)$ and ones where A is before C - we just swap the two operations. Similarly, for B and D. This implies that in $1/4$ of the cases, A is before C and B is before D. In other words, there are 6 such sequences. We must remember to multiply by 2 and we get 12 (because of $(O,1,C)$ and $(0,2,C)$ as mentioned earlier).

(c) We want to count how many sequences there are if 1 wants to commit and 2 wants to abort. The following operations are possible: $A' = (0,1,P)$, $B' = (0,2,P)$, $C' = (1,0,R)$, $D' = (2,0,D)$ and $E' = (0,1,A)$.

We must have the following sub-sequence of messages (in that order): $B'D'E'$, but can omit any number of the remaining messages (in particular, $S = B'D'E'$ is a sequence on its own). We have some more restrictions though, E' must be last, A' must be before C' and A' can not come after D' (the coordinator knows we need to abort at that point). Besides S all the other sequences contains A' (since it must be before C'). We can now consider two cases for where we put A': Before or after B' (recall that it can't be after D'). If before, we can either put C' before B' as well, we can put it between B' and D' or between B' and E' or omit it. I.e. there are 4 sequences where we put A' before B'. If we put A' after B' (but before D'), we can put C' before D' as well, between D' and E' or omit it. Hence, there are 3 sequences when we put A' after B'. Thus, in total we have $1+4+3 = 8$ sequences.

(d) We want to count how many sequences there are when 1 wants to commit but 2 does not respond. The following operations are possible: $A'' = (0,1,P)$, $B'' = (0,2,P)$, $C'' = (1,0,C)$, $D'' = (0,1,A)$ and $E'' = (0,2,A)$. We have some logic based requirements, B'' before D'' and E'' and A'' before C''. Also, both A'' and C'' must be before D''.

Formally, from a logic perspective, we have the sequences

(i) $B''E''$

(ii) $A''B''D''E''$

(iii) $A''C''B''D''E''$

(iv) $A''B''C''D''E''$

(v) $B''A''D''E''$

(vi) $B''A''C''D''E''$

(vii) $A''B''E''D''$

(viii) $A''C''B''E''D''$

(ix) $A''B''C''E''D''$

(x) $A''B''E''C''D''$

(xi) $B''A''E''D''$

(xii) $B''A''C''E''D''$

(xiii) $B''A''E''C''D''$

That said, while the other questions in this exercise are based mainly on logic, this question is based more on discussion. Some sequences are reasonable some are not. E.g. the sequence $B''E''$ could be viewed as a valid sequence from a logic perspective (the coordinator decides that 2 has timed out, before sending anything to 1), but in practice, the decision to abort will come a while after the prepare message is send, and it seems

unreasonable that the coordinator has not decided to send prepare to 1 in all that time. Basically, the only sequences that are unarguably reasonable are

(i) $A''C''B''E''D''$

(ii) $A''B''C''E''D''$

(iii) $B''A''C''E''D''$

(iv) $A''C''B''D''E''$

(v) $A''B''C''D''E''$

(vi) $B''A''C''D''E''$

I.e. the first three messages A'', B'' and C'' (note that C'' must come after A'') get send first and a long while later, when the coordinator has decided that 2 has timed out D'' and E'' gets send.

References

[1] Ramez Elmasri and Shamkant B. Navathe. *Fundamentals of Database Systems*. Pearson Education, 7th edition, 2016.

[2] Hector Garcia-Molina, Jeffrey D. Ullman, and Jennifer Widom. *Database Systems - The Complete Book*. Pearson Education, 2nd edition, 2009.