

CS 347 Transaction Processing and Distributed Database Systems Spring 2008

Assignment 1

Due by 11:59pm on Friday, April 18, 2008.

- State all assumptions.
 - Assignment policies are listed on the Course Information page of the course website (<http://cs347.stanford.edu>).
 - Post questions to the newsgroup `su.class.cs347`.
-

Problem 1 (10 points)

The following paper discusses an interesting fragmentation problem, where we use fragmentation to enhance the privacy of data:

Two Can Keep a Secret: A Distributed Architecture for Secure Database Services

Available at: <http://dbpubs.stanford.edu/pub/2004-42>

(OK, I am biased regarding the “interesting” aspect since I am one of the authors :-))

- (1) Read the paper, EXCEPT Sections 5.1, 5.2, 6.
- (2) Consider a relation R with attributes $X, A, B, C, D, E, F, G, H, I, J, K$. Attribute X is the primary key of the relation. We know the following are privacy constraints:
 - $\{A, B, C, D\}$
 - $\{E, F\}$
 - $\{E, G\}$
 - $\{E, H\}$
 - $\{F, G, H\}$
 - $\{I, J\}$
 - $\{I, K\}$
 - $\{J, K\}$

We also know that in the affinity matrix all entries have a value of 1, EXCEPT the entries:

- $M[A, B], M[B, A]$
- $M[C, D], M[D, C]$

- $M[A, F], M[F, A]$
- $M[C, H], M[H, C]$
- $M[A, I], M[I, A]$

which all have a value of 10.

Describe a good, privacy-preserving fragmentation across 2 databases. You should encrypt as few attributes as possible. Briefly explain why your design is good. (Note there may be more than one good solution.)

Solution

From the affinity matrix, we can quickly determine that a minimum cost decomposition must keep the groups $\{A, B, F, I\}$ and $\{C, D, H\}$ intact on some node, and must not encrypt any of the attributes in either group. Luckily, if we split these two groups onto two different nodes, then no privacy constraints are violated, so encryption of these attributes is not necessary.

Four attributes: E, G, J, and K remain to be assigned to nodes. Privacy constraints force us to encrypt some of these attributes. Attribute E cannot be stored unencrypted on either node, due to the privacy constraints $\{E, F\}$ and $\{E, H\}$. Also, neither J nor K can be stored unencrypted on the node holding I, but the privacy constraint $\{J, K\}$ prevents them from being unencrypted together. Therefore either J or K must be encrypted as well.

The following is one minimal cost, privacy preserving decomposition:

$$R_1 = \{A, B, E, F, G, I, J\}, R_2 = \{C, D, E, H, J, K\}, E = \{E, J\}$$

Grading note: The problem description erroneously implied that the diagonals of the affinity matrix (i.e. $M[i, i]$) have value 1. A matrix of that sort makes no sense, as it would suggest that more queries use both A and B together than use A with or without B! Using a matrix with diagonals set to 1, combined with relying on the equations in the paper could result in some strange decompositions, such as encrypting every attribute. If it is clear from your work that the mistake in the problem description is to blame for an incorrect answer, then full credit will be given.
