Ferhat Elmas 214805

Advanced Databases Hw-6

1-)

- A = load ‘ids’ as (name, id_nr);
- B = load ’grades’ as (name, grade);
- C = group B by name;
- D = foreach C generate group, AVG(B.grade);
- E = join D by name, A by name;

2-) First, I will explain the query then I will try to write because here formatting is a bit difficult. Firstly I get the nest of Taken relation (nest is constructed in slides and I use directly) and deeply compare it to Course relation and finally apply the selection on the result of the deep. If the comparison returns false, name is discarded, otherwise name is put into result set.

flatmap(nest_ids=(course_id)(Taken) o =_deep(ids)(Course) o pairwith_2 o map(π_1))

3-)

1-) |R| = 8, |S| = 8, r = 4

- Lower bound of max-reducer-output is |R| x |S| / r = 8 x 8 / 4 = 16
- Lower bound of max-reducer-input is 2 x sqrt(|R| x |S| / r) = 2 x sqrt(16) = 8

2-)

- |R| = cR x sqrt(|R| x |S| / r) where cR = 2
- |S| = cS x sqrt(|R| x |S| / r) where cS = 2
- According to theorem 1 in the paper, we can divide the matrix into cR x cS (2 x 2 = 4) squares which are in the size of 2 x sqrt(|R| x |S| / r) = 2 x sqrt(8 x 8 / 4) = 16.
- max-reducer-output becomes c = 16.
- max-reducer-input becomes 2 * sqrt(c) = 8.

3-)

If we divide the big square into 4 parts as above: For input, it always will be 8.

|R.A - S.C| > 2 case:

	Input	Output
a.	8	8
b.	8	13
c.	8	16
d.	8	8

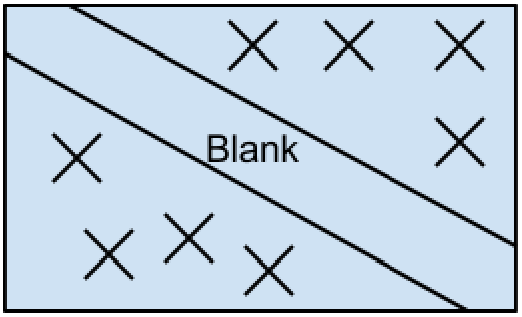
For output, we have 8 + 13 + 16 + 8 = 45 tuples so lower bound will be ceil(45 / r) = ceil(45/4) = 12.

|R.A - S.C| > 7 case:

	Input	Output
a.	8	0
b.	8	6
c.	8	8
d.	8	0

For output, we have 0 + 6 + 8 + 0 = 14 tuples so lower bound will be ceil(14 / r) = ceil (14 / 4) = 4

4-)



As seen in the figure, diagonal will be empty in this type of join predicate. Moreover, the cardinalities of the relations are big, the number of tuples will be generated from each row or column will be nearly equal, so we can divide the matrix by the number of rows or columns to get better output tuple distribution. For example, if matrix has 10000 rows and we have 10 reducers, we assign 1000 rows of the matrix to each reducer.