Uni Freiburg, Web Science Group
Prof. Peter Fischer
Systems Infrastructure for Data Science - Winter 2014/15

Exercise Sheet #8: Distributed Query Processing

December 18, 2014

1 Data Localization

Consider that the relation *Reviewers* is horizontally fragmented as follows:

$$Reviewers_1 = \sigma_{reviewer\_id \leq '20000'}(Reviewers)$$
$$Reviewers_2 = \sigma_{reviewer\_id > '20000'}(Reviewers)$$

Now, consider a derived horizontal fragmentation of relation *Movie_Reviews*:

$$Movie\_Reviews_1 = Movie\_Reviews \ltimes_{reviewer\_id} Reviewers_1$$
$$Movie\_Reviews_2 = Movie\_Reviews \ltimes_{reviewer\_id} Reviewers_2$$

Furthermore, the relation *Movies* is vertically fragmented as:

$$Movies_1 = \Pi_{movie\_id,title,release\_year}(Movies)$$
$$Movies_2 = \Pi_{movie\_id,star\_rating,era\_id}(Movies)$$

Transform the following query into a reduced query on fragments.

```
SELECT m.title
FROM Movies m, Reviewers r, Movie_Reviews mr
WHERE m.movie_id = mr.movie_id AND r.reviewer_id = mr.reviewer_id
  AND r.name = 'Cagri'
```

2 Query Optimization

A. Consider a join among tables Reviews, Movie_Reviews and Movies from the previous example. Figure 1 shows both the join graph and the distribution onto three sites.

$$(\text{Movies} \bowtie_{movie\_id} \text{Movie\_Reviews} \bowtie_{reviewer\_id} \text{Reviewers})$$

(i) Given the following information: size(Movies)=100, size(Movie_Reviews)=200, size(Reviewers)=300, size(Movies ⋈ Movie_Reviews)=300, size(Movie_Reviews ⋈ Reviewers) = 200, describe several alternatives for building a join ordering program.

(ii) What is the optimal ordering that minimizes query response time (consider communication only)?

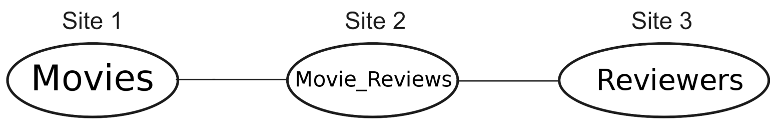

Figure 1: Join graph