1 Data Localization

Consider that the relation Reviewers is horizontally fragmented as follows:

\[ \text{Reviewers}_1 = \sigma_{\text{reviewer.id} \leq 20000}(\text{Reviewers}) \]
\[ \text{Reviewers}_2 = \sigma_{\text{reviewer.id} > 20000}(\text{Reviewers}) \]

Now, consider a derived horizontal fragmentation of relation Movie_Reviews:

\[ \text{Movie_Reviews}_1 = \text{Movie_Reviews} \bowtie_{\text{reviewer.id}} \text{Reviewers}_1 \]
\[ \text{Movie_Reviews}_2 = \text{Movie_Reviews} \bowtie_{\text{reviewer.id}} \text{Reviewers}_2 \]

Furthermore, the relation Movies is vertically fragmented as:

\[ \text{Movies}_1 = \Pi_{\text{movie.id}, \text{title}, \text{release.year}}(\text{Movies}) \]
\[ \text{Movies}_2 = \Pi_{\text{movie.id}, \text{star.rating}, \text{era.id}}(\text{Movies}) \]

Transform the following query into a reduced query on fragments.

\[
\text{SELECT m.title FROM Movies m, Reviewers r, Movie_Reviews mr WHERE m.movie_id = mr.movie_id AND r.reviewer_id = mr.reviewer_id AND r.name = 'Cagri'}
\]

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.

\[ (\text{Movies} \bowtie_{\text{movie.id}} \text{Movie_Reviews} \bowtie_{\text{reviewer.id}} \text{Reviewers}) \]

(i) Given the following information: size(Movies)=100, size(Movie_Reviews)=200, size(Reviewers)=300, size(Movies \bowtie Movie_Reviews)=300, size(Movie_Reviews \bowtie 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