SSDBM 2010, June 30 – July 2, Heidelberg, Germany

Efficient and Adaptive Distributed Skyline Computation

George Valkanas
MaDgIK
Dept. of Informatics & Telecommunications
University of Athens, Athens, Greece

Apostolos N. Papadopoulos
Data Engineering Lab.,
Department of Informatics
Aristotle University, Thessaloniki, Greece

Motivation

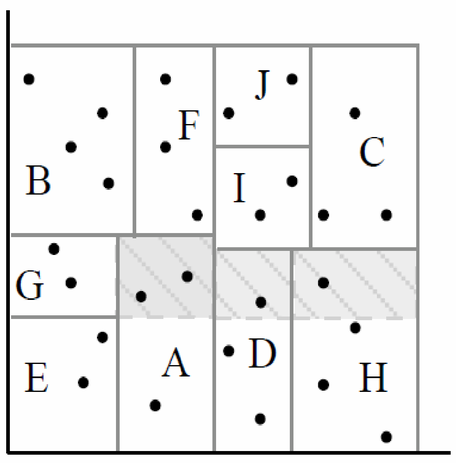
- Skyline Computation is an interesting analysis technique for multi-dimensional data
- Distributed *State-of-the-art* techniques
  - Assume **a-priori** knowledge of the criteria used
  - Lack in progressiveness
  - Perform badly for anticorrelated data distributions
  - Seem to fail for other query types (top-*K*, NN, etc)
- *Different users have different preferences*
  - Multimedia databases, e.g. "color VS shape"
- Different criteria allow for better analysis (OLAP application)
  - e.g. "Profit vs Time", "Humidity vs Temperature"
- We propose *i)* **Adaptive Distributed Skyline Computation** (ADISC) algorithm, *ii)* **Marginal Points** as representative points, *iii)* **Data Propagation** technique

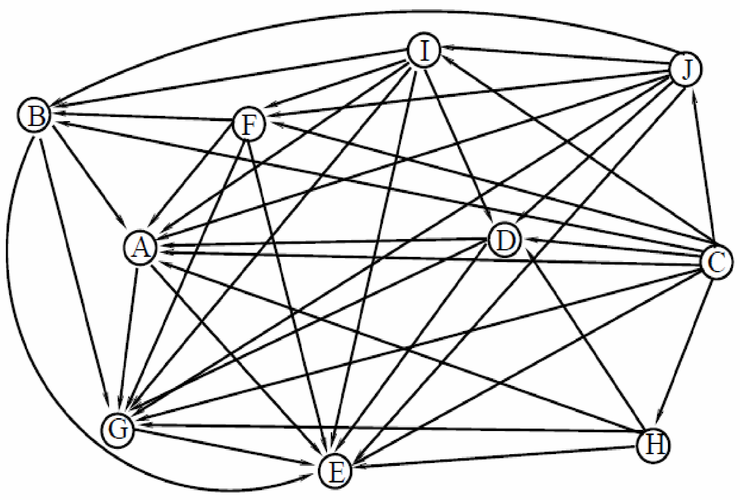
The ADISC algorithm

- Runs both in **parallel** and **cascading** mode
- Uses single internal structure, i.e. ***Dependency graph***, over a grid-based partitioning scheme
- Integrates several optimizations

The Dependency Graph

- Directed Acyclic Graph (DAG)
- Acts like a *can-dominate* index of the partitions
- Dependencies are created at query-time, according to the preferences imposed





Improving the Dependency Graph

- Prune non-contributing nodes (delete nodes – Fig. A)
- ***Improve Parallelism*** by removing dependencies (delete edges – Fig. B)

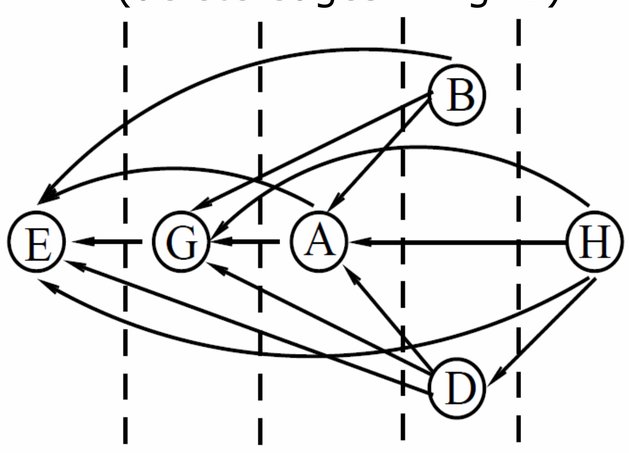


Fig. A

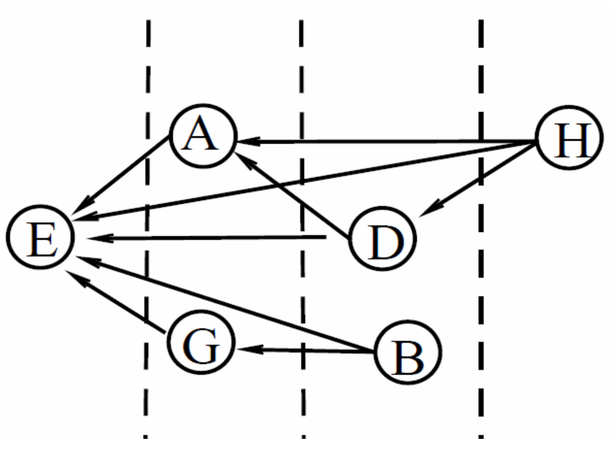


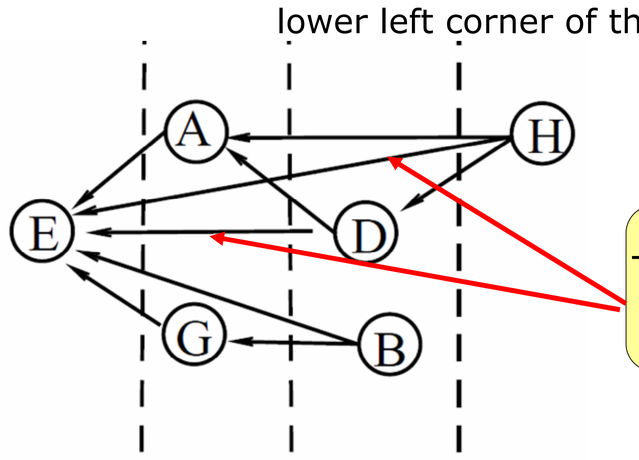
Fig. B

Additional Optimizations

- Point Exclusion
- Eager checking

Data Propagation in cascading mode

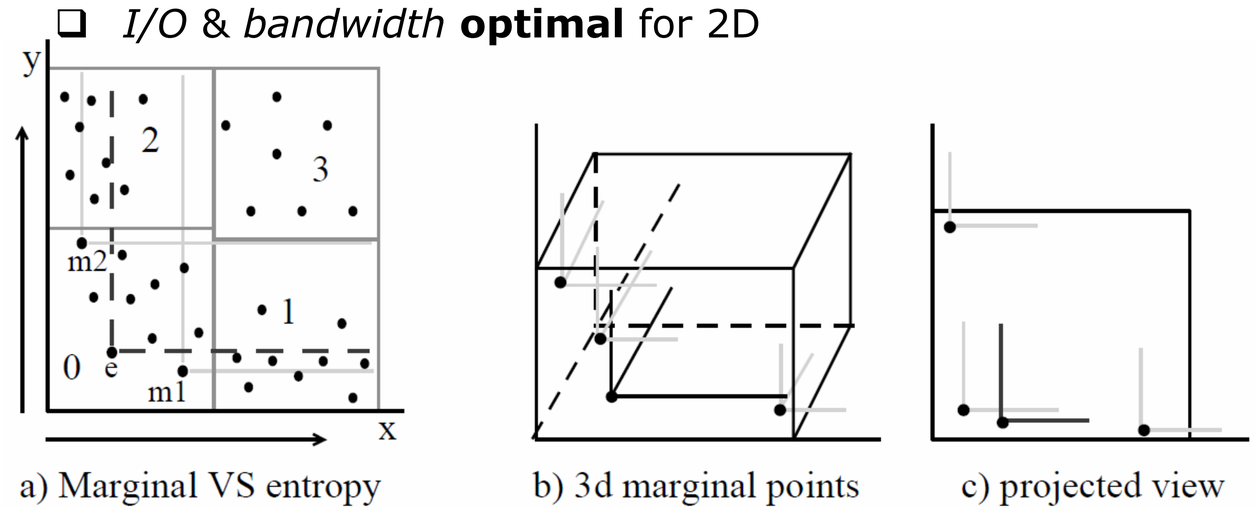
- Dependency graph determines communication
- Apply heuristics to reduce traffic
  - i) Partitions share propagation axis
  - ii) Intermediate partition's lower left corner is above the lower left corner of the 1st partition ($\geq$ 3D)



These edges can be **ignored** during propagation due to 1st heuristic

Marginal Points

- Better representatives than entropy points
- The ones closer to the *(d-1)* coordinates of a partition's lower left corner, according to the L1 distance
  - As if **projecting** on the *(d-1)* subspace
- *I/O* & *bandwidth* **optimal** for 2D

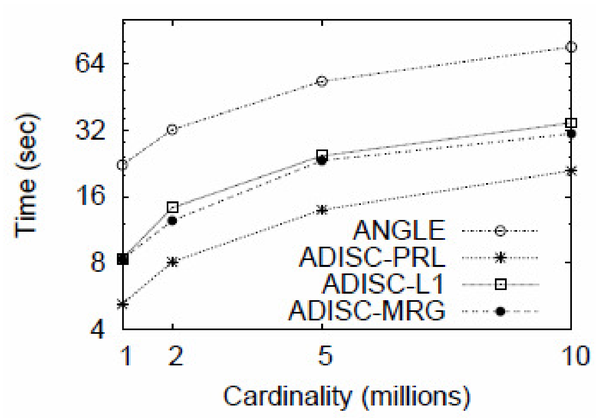


a) Marginal VS entropy

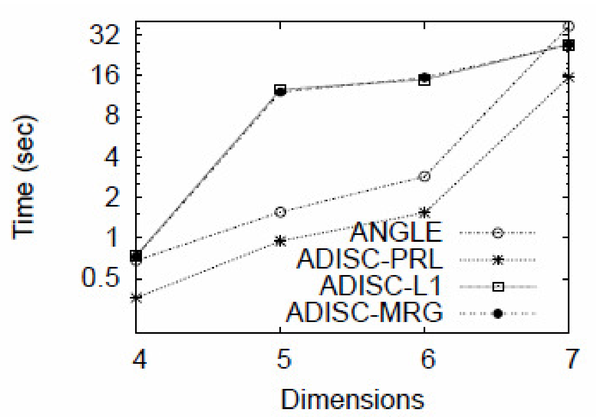
b) 3d marginal points

c) projected view

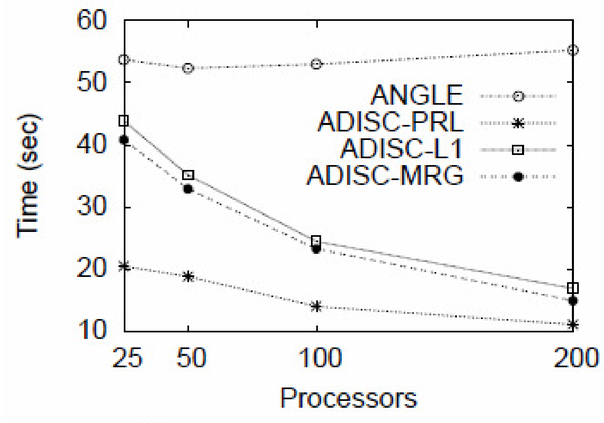
Performance Results



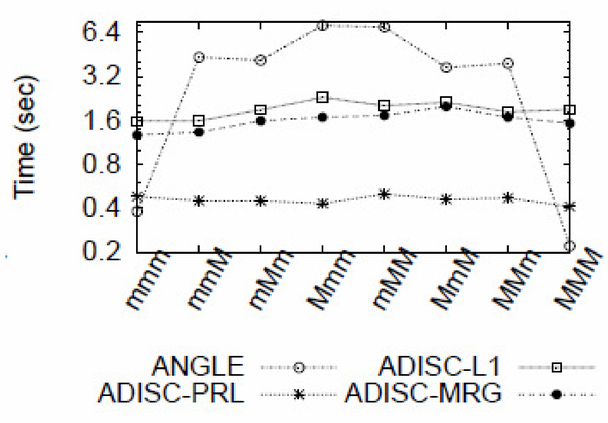
(a) ANT, min preferences



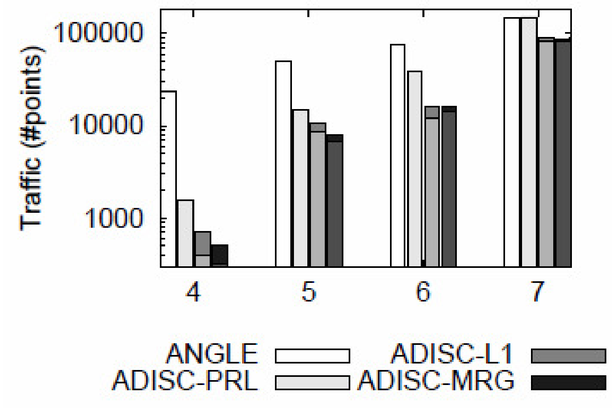
(d) FC, mixed preferences



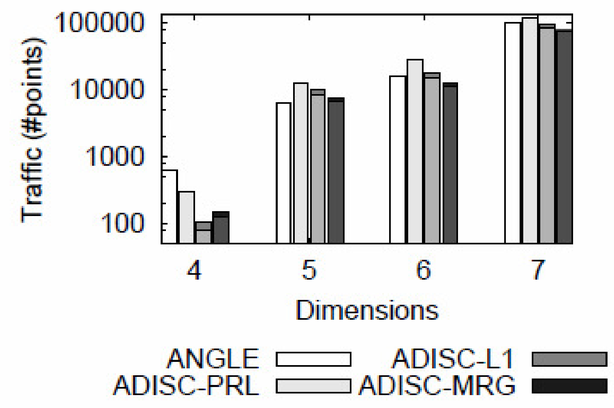
(f) ANT, min preferences



(h) IND, diverse query criteria



(d) FC, mixed preferences



(b) FC, min preferences