Multi-join Query Evaluation on Big Data

Lecture 1

Dan Suciu

March, 2015
About Me

- Originally from Romania
- Had fun with Math: 1976 IMO
- PhD from University of Pennsylvania: Parallel Query Languages
- Bell Labs and AT&T Labs: Semistructured Data, XML
- University of Washington: data privacy, probabilistic data, Big Data

Today’s topic: Big Data!
Course Organization

- Four lectures (1.5h): slides available on the course Website

- Two sections (1h): mostly interactive

- A problem set to pass the course: seven problems (simple to challenging); email me your solutions by April 30, 2015.

- I hope you can attend all lectures and sections: you need them in order to solve the problems.
Multi-join Query Evaluation – Outline

Part 1  Optimal Sequential Algorithms. Thursday 14:15-15:45

Part 2  Lower bounds for Parallel Algorithms. Friday 14:15-15:45

Part 3  Optimal Parallel Algorithms. Saturday 9-10:30

Part 3  Data Skew. Saturday 11-12
Multi-join Query Evaluation – Outline

Part 1  Optimal Sequential Algorithms. Thursday 14:15-15:45

Part 2  Lower bounds for Parallel Algorithms. Friday 14:15-15:45

Part 3  Optimal Parallel Algorithms. Saturday 9-10:30

Part 3  Data Skew. Saturday 11-12
Multi-join Query Evaluation – Outline

Part 1 Optimal Sequential Algorithms. Thursday 14:15-15:45

Part 2 Lower bounds for Parallel Algorithms. Friday 14:15-15:45

Part 3 Optimal Parallel Algorithms. Saturday 9-10:30

Part 3 Data Skew. Saturday 11-12
Multi-join Query Evaluation – Outline

**Part 1** Optimal Sequential Algorithms. Thursday 14:15-15:45

**Part 2** Lower bounds for Parallel Algorithms. Friday 14:15-15:45

**Part 3** Optimal Parallel Algorithms. Saturday 9-10:30

**Part 3** Data Skew. Saturday 11-12
Bibliography

- Paul Beame, Paraschos Koutris, Dan Suciu: Skew in parallel query processing. PODS 2014: 212-223
- Paul Beame, Paraschos Koutris, Dan Suciu: Communication steps for parallel query processing. PODS 2013: 273-284
Outline for Lecture 1

- Background: Queries, Databases, Query Evaluation
- The AGM inequality
- Friedgut’s inequality
- Worst-case optimal query evaluation
- Summary
## Relations and Databases

### Person

<table>
<thead>
<tr>
<th>Name</th>
<th>Age</th>
<th>City</th>
<th>Hobby</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>22</td>
<td>Lódtź</td>
<td>knitting</td>
</tr>
<tr>
<td>Bob</td>
<td>33</td>
<td>Lyon</td>
<td>karate</td>
</tr>
<tr>
<td>Carol</td>
<td>44</td>
<td>Lódtź</td>
<td>kayaking</td>
</tr>
<tr>
<td>David</td>
<td>33</td>
<td>Lima</td>
<td>karate</td>
</tr>
<tr>
<td>Eve</td>
<td>22</td>
<td>Lima</td>
<td>knitting</td>
</tr>
</tbody>
</table>

**Schema** Relation/table name Person;
Attribute/column names Name, Age, City, Hobby;
Key Name

**Instance** Set of tuples/rows/records,
e.g. (Alice, 22, Lódtź, knitting)

**Size** Number of tuples $m = 5$; note: relation is a set

**Database** is a set of relations = a finite structure
Relations and Databases

<table>
<thead>
<tr>
<th>Person</th>
<th>Name</th>
<th>Age</th>
<th>City</th>
<th>Hobby</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>22</td>
<td>Lódtź</td>
<td>knitting</td>
<td></td>
</tr>
<tr>
<td>Bob</td>
<td>33</td>
<td>Lyon</td>
<td>karate</td>
<td></td>
</tr>
<tr>
<td>Carol</td>
<td>44</td>
<td>Lódtź</td>
<td>kayaking</td>
<td></td>
</tr>
<tr>
<td>David</td>
<td>33</td>
<td>Lima</td>
<td>karate</td>
<td></td>
</tr>
<tr>
<td>Eve</td>
<td>22</td>
<td>Lima</td>
<td>knitting</td>
<td></td>
</tr>
</tbody>
</table>

**Schema**
Relation/table name Person;
Attribute/column names Name, Age, City, Hobby;
Key Name

**Instance**
Set of tuples/rows/records,
e.g. (Alice, 22, Lódtź, knitting)

**Size**
Number of tuples $m = 5$; note: relation is a set

**Database**
is a set of relations = a finite structure
Relations and Databases

<table>
<thead>
<tr>
<th>Person</th>
<th>Name</th>
<th>Age</th>
<th>City</th>
<th>Hobby</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>22</td>
<td></td>
<td>Lödtz</td>
<td>knitting</td>
</tr>
<tr>
<td>Bob</td>
<td>33</td>
<td></td>
<td>Lyon</td>
<td>karate</td>
</tr>
<tr>
<td>Carol</td>
<td>44</td>
<td></td>
<td>Lödtz</td>
<td>kayaking</td>
</tr>
<tr>
<td>David</td>
<td>33</td>
<td></td>
<td>Lima</td>
<td>karate</td>
</tr>
<tr>
<td>Eve</td>
<td>22</td>
<td></td>
<td>Lima</td>
<td>knitting</td>
</tr>
</tbody>
</table>

Schema
- Relation/table name: Person
- Attribute/column names: Name, Age, City, Hobby
- Key: Name

Instance
- Set of tuples/rows/records,
  - e.g. (Alice, 22, Lödtz, knitting)

Size
- Number of tuples $m = 5$; note: relation is a set

Database
- is a set of relations = a finite structure
Relations and Databases

<table>
<thead>
<tr>
<th>Person</th>
<th>Name</th>
<th>Age</th>
<th>City</th>
<th>Hobby</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>22</td>
<td>Lódź</td>
<td>knitting</td>
<td></td>
</tr>
<tr>
<td>Bob</td>
<td>33</td>
<td>Lyon</td>
<td>karate</td>
<td></td>
</tr>
<tr>
<td>Carol</td>
<td>44</td>
<td>Lódź</td>
<td>kayaking</td>
<td></td>
</tr>
<tr>
<td>David</td>
<td>33</td>
<td>Lima</td>
<td>karate</td>
<td></td>
</tr>
<tr>
<td>Eve</td>
<td>22</td>
<td>Lima</td>
<td>knitting</td>
<td></td>
</tr>
</tbody>
</table>

Schema
Relation/table name Person;
Attribute/column names Name, Age, City, Hobby;
Key Name

Instance
Set of tuples/rows/records,
e.g. (Alice, 22, Lódź, knitting)

Size
Number of tuples $m = 5$; note: relation is a set

Database is a set of relations = a finite structure
## Relations and Databases

<table>
<thead>
<tr>
<th>Person</th>
<th>Name</th>
<th>Age</th>
<th>City</th>
<th>Hobby</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>22</td>
<td>Lódź</td>
<td>knitting</td>
<td></td>
</tr>
<tr>
<td>Bob</td>
<td>33</td>
<td>Lyon</td>
<td>karate</td>
<td></td>
</tr>
<tr>
<td>Carol</td>
<td>44</td>
<td>Lódź</td>
<td>kayaking</td>
<td></td>
</tr>
<tr>
<td>David</td>
<td>33</td>
<td>Lima</td>
<td>karate</td>
<td></td>
</tr>
<tr>
<td>Eve</td>
<td>22</td>
<td>Lima</td>
<td>knitting</td>
<td></td>
</tr>
</tbody>
</table>

### Schema
- Relation/table name: Person
- Attribute/column names: Name, Age, City, Hobby
- Key: Name

### Instance
- Set of tuples/rows/records,
  - e.g. (Alice, 22, Lódź, knitting)

### Size
- Number of tuples $m = 5$; note: relation is a set

### Database
- Is a set of relations = a finite structure
Basic Stuff that’s Good To Know

- Relational database systems: Oracle, SQL Server, DB2, Postgres, SQLite, Dremel, Scope, Spark SQL

- Relations are flat (atomic values only): 1st normal form.

- Relations are persistent: stored in file systems, or in distributed file systems like Hadoop

- Physical data independence: system is allowed to organize the relation how it wishes. E.g. indexes, column-oriented DBs, partition on distributed servers, replicated.
Relational Algebra

- Cartesian product / Join: \( \bowtie \)

- Projection: \( \Pi_A \)

- Selection: \( \sigma_C \)

- Union: \( \cup \)

- Set difference: \( - \)

This course: select-project-join
Join

\[ R \bowtie_{X=Y} S \]

The set of pairs \((t_1, t_2)\), with \(t_1 \in R\) and \(t_2 \in S\), s.t. \(t_1.X = t_2.Y\)

\[
\begin{array}{cc}
R & S \\
X & U & Y & V \\
--- & --- & --- & --- \\
a_1 & b_1 & a_1 & c_1 \\
a_1 & b_2 & a_1 & c_2 \\
a_2 & b_3 & a_3 & c_3 \\
a_3 & b_4 & a_4 & c_4 \\
\end{array}
\]

\[
\begin{array}{cc}
T = R \bowtie_{X=Y} S \\
X & U & Y & V \\
--- & --- & --- & --- \\
a_1 & b_1 & a_1 & c_1 \\
a_1 & b_1 & a_1 & c_2 \\
a_1 & b_2 & a_1 & c_1 \\
a_1 & b_2 & a_1 & c_2 \\
a_3 & b_4 & a_3 & c_3 \\
\end{array}
\]

Input schemas: \(R(X, U), S(Y, V)\)
Output schema: \(T(X, U, Y, V)\)
Natural Join

$R \bowtie S$

Joins $R, S$ on all common attributes, removes duplicate attributes

\[
\begin{array}{|c|c|}
\hline
a_1 & b_1 \\
\hline
a_1 & b_2 \\
\hline
a_2 & b_3 \\
\hline
a_3 & b_4 \\
\hline
\end{array}
\quad\quad
\begin{array}{|c|c|}
\hline
a_1 & c_1 \\
\hline
a_1 & c_2 \\
\hline
a_3 & c_3 \\
\hline
a_4 & c_4 \\
\hline
\end{array}
\quad\quad
\begin{array}{|c|c|c|}
\hline
a_1 & b_1 & c_1 \\
\hline
a_1 & b_1 & c_2 \\
\hline
a_1 & b_2 & c_1 \\
\hline
a_1 & b_2 & c_2 \\
\hline
a_3 & b_4 & c_3 \\
\hline
\end{array}
\]

Input schemas: $R(A, B), S(A, C)$
Output schema: $T(A, B, C)$
Natural Join Examples

Question
In each case below: what is the output schema? What does the join do?

- \( R(A, B, E, G) \bowtie S(A, C, D, E, F) \)
Natural Join Examples

**Question**

In each case below: what is the output schema? What does the join do?

- \( R(A, B, E, G) \bowtie S(A, C, D, E, F) \)
  
  Returns Output \((A, B, C, D, E, F, G) = R \bowtie (R.A=S.A) \land (R.E=S.E) S\)
Natural Join Examples

Question
In each case below: what is the output schema? What does the join do?

- \( R(A, B, E, G) \bowtie S(A, C, D, E, F) \)
  Returns Output \( (A, B, C, D, E, F, G) = R \bowtie (R.A=S.A) \land (R.E=S.E) S \)

- \( R(A, B) \bowtie S(C, D, E) \)
Natural Join Examples

**Question**

In each case below: what is the output schema? What does the join do?

- \( R(A, B, E, G) \bowtie S(A, C, D, E, F) \)
  Returns Output \((A, B, C, D, E, F, G) = R \bowtie_{(R.A=S.A) \land (R.E=S.E)} S\)

- \( R(A, B) \bowtie S(C, D, E) \)
  Returns the cartesian product: Output \((A, B) = R \times S\).
Natural Join Examples

Question
In each case below: what is the output schema? What does the join do?

- $R(A, B, E, G) \bowtie S(A, C, D, E, F)$
  Returns $Output(A, B, C, D, E, F, G) = R \bowtie_{(R.A=S.A) \land (R.E=S.E)} S$

- $R(A, B) \bowtie S(C, D, E)$
  Returns the cartesian product: $Output(A, B, C, D, E) = R \times S$.

- $R(A, B) \bowtie S(A, B)$
Natural Join Examples

Question
In each case below: what is the output schema? What does the join do?

- \( R(A, B, E, G) \bowtie S(A, C, D, E, F) \)
  
  Returns Output\((A, B, C, D, E, F, G) = R \bowtie_{(R.A=S.A) \land (R.E=S.E)} S \)

- \( R(A, B) \bowtie S(C, D, E) \)
  
  Returns the cartesian product: Output\((A, B, C, D, E) = R \times S. \)

- \( R(A, B) \bowtie S(A, B) \)
  
  Returns the intersection: Output\((A, B) = R \cap S \)
Very Quick Review of Basic Join Algorithms

Compute $R \bowtie_{A=B} S$

- Nested-loop join
- Hash-join
- Merge-join

(To describe in class.)

Complexity: $O((|R| + |S| + |R \bowtie_{A=B} S|) \log(|R| + |S|))$

Ignoring log factors, Complexity: $O(|\text{Input}| + |\text{Output}|)$
Projection

\[ \Pi_{AC}(T) \]

Projects \( T \) on the attributes \( A \) and \( C \).

\[
\begin{array}{|c|c|c|}
\hline
A & B & C \\
\hline
a_1 & b_1 & c_1 \\
a_1 & b_1 & c_2 \\
a_1 & b_2 & c_1 \\
a_1 & b_2 & c_2 \\
a_3 & b_4 & c_3 \\
\hline
\end{array}
\]

\[
\begin{array}{|c|c|}
\hline
A & C \\
\hline
a_1 & c_1 \\
a_1 & c_2 \\
a_3 & c_3 \\
\hline
\end{array}
\]

Note: projection does duplicate elimination.
Selection

\[ \sigma_{A=a}(R) \]

Returns all rows where attribute \( A \) has value \( a \).

\[
\begin{array}{|c|c|c|}
\hline
A & B & C \\
\hline
a_1 & b_1 & c_1 \\
a_1 & b_1 & c_2 \\
a_1 & b_2 & c_1 \\
a_1 & b_2 & c_2 \\
a_3 & b_4 & c_3 \\
\hline
\end{array}
\]

\[ \sigma_{C=c_2}(R) \]

\[
\begin{array}{|c|c|c|}
\hline
A & B & C \\
\hline
a_1 & b_1 & c_2 \\
a_1 & b_2 & c_2 \\
\hline
\end{array}
\]
Queries

Relational Algebra

Defined alternatively as:
- Relational Algebra: \( \bowtie, \sigma, \Pi, \cup, \neg \), or
- Relational Calculus, or First Order Logic: \( \land, \lor, \exists, \forall, \neg, = \)
- Non-recursively datalog with negation, or
- A certain well-behaved fragment of SQL

Conjunctive queries

Defined as:
- \( \bowtie, \sigma, \Pi \), or
- \( \land, \exists, = \), or
- A single datalog rule, or
- `select-from-where` SQL queries

This course: **full** conjunctive queries, meaning without \( \Pi \).
Conjunctive Queries

Example

\[ Q_1(x, y, z, u) = R(x, y), S(y, z), T(z, u) \]

- **Relational Algebra:** \( (R(x, y) \bowtie S(y, z)) \bowtie T(z, u) \)
- **First Order Logic:**
  \[ Q_1 = \{(x, y, z, u) \mid (x, y) \in R \land (y, z) \in S \land (z, u) \in T\} \]
Conjunctive Queries

Example

$Q_1(x, y, z, u) = R(x, y), S(y, z), T(z, u)$

- Relational Algebra: $(R(x, y) \bowtie S(y, z)) \bowtie T(z, u)$
- First Order Logic:
  $Q_1 = \{(x, y, z, u) \mid (x, y) \in R \land (y, z) \in S \land (z, u) \in T\}$

Example

$Q_2(x, u) = R(x, y), S(y, z), T(z, u)$

- Relational Algebra: $\Pi_{x,u}((R(x, y) \bowtie S(y, z)) \bowtie T(z, u))$
- First Order Logic:
  $Q_1 = \{(x, u) \mid \exists y \exists z ((x, y) \in R \land (y, z) \in S \land (z, u) \in T)\}$
Traditional Approach to Computing Conjunctive Queries

\[ Q(x, y, z) = R(x, y), S(y, z), T(z, x) \]

Optimizer generates a *query plan*:

\[
\text{Temp}(x, y, z) = R(x, y) \bowtie S(y, z) \\
Q(x, y, z) = \text{Temp}(x, y, z) \bowtie T(z, x)
\]

Optimizers examines many possible plans, evaluates the cheapest plan.

**Problem:** intermediate results may be large, and very hard to estimate.
Upper Bound on the Size of the Answer

Consider the join of two relations:

\[ Q(x, y, z) = R(x, y), S(y, z) \]

**Question**

If \( |R| = m_1, |S| = m_2 \), how large can \( |Q| \) be?

Answer: \( 0 \leq |Q| \leq m_1 m_2 \).
Upper Bound on the Size of the Answer

Consider the join of two relations:

\[ Q(x, y, z) = R(x, y), S(y, z) \]

**Question**

If \( |R| = m_1, |S| = m_2 \), how large can \( |Q| \) be?

- Can be 0
Upper Bound on the Size of the Answer

Consider the join of two relations:

\[ Q(x, y, z) = R(x, y), S(y, z) \]

**Question**

If \(|R| = m_1, |S| = m_2\), how large can \(|Q|\) be?

- Can be 0
- Can be \(m_1 m_2\)
Upper Bound on the Size of the Answer

Consider the join of two relations:

\[ Q(x, y, z) = R(x, y), S(y, z) \]

**Question**

If \(|R| = m_1, |S| = m_2\), how large can \(|Q|\) be?

- Can be 0
- Can be \(m_1 m_2\)
- **Answer**: \(0 \leq |Q| \leq m_1 m_2\).
Upper Bound on the Size of the Answer

\[ Q(x, y, z) = R(x, y), S(y, z), T(z, x) \]

**Question**

If \(|R| = m_1, |S| = m_2, |T| = m_3\), how large can the result be?
Upper Bound on the Size of the Answer

\[ Q(x, y, z) = R(x, y), S(y, z), T(z, x) \]

**Question**

If \(|R| = m_1, |S| = m_2, |T| = m_3\), how large can the result be?

- Naive answer: \( \leq m_1 m_2 m_3 \) (why?)
Upper Bound on the Size of the Answer

\[ Q(x, y, z) = R(x, y), S(y, z), T(z, x) \]

**Question**

If \(|R| = m_1, |S| = m_2, |T| = m_3\), how large can the result be?

- Naive answer: \(\leq m_1 m_2 m_3\) (why?)
- Better answer: \(\leq m_1 m_2\) (why?)
Upper Bound on the Size of the Answer

\[ Q(x, y, z) = R(x, y), S(y, z), T(z, x) \]

**Question**

If \(|R| = m_1, |S| = m_2, |T| = m_3\), how large can the result be?

- Naive answer: \(\leq m_1 m_2 m_3\) (why?)
- Better answer: \(\leq m_1 m_2\) (why?)
- But also: \(\leq m_1 m_3, \leq m_2 m_3\)
The Hypergraph of a Query

Definition

Let $Q$ be a full conjunctive query without self-joins. The hypergraph $G$ of $Q$ consists of:

- $\text{Nodes}(G) = \text{Vars}(Q)$ the set of variables of $Q$
- $\text{HyperEdges}(G) = \text{Atoms}(Q)$ the set of atoms of $Q$. 
The Hypergraph of a Query

**Definition**

Let $Q$ be a full conjunctive query without self-joins. The hypergraph $G$ of $Q$ consists of:

- $\text{Nodes}(G) = Vars(Q)$ the set of variables of $Q$
- $\text{HyperEdges}(G) = Atoms(Q)$ the set of atoms of $Q$.

Examples:

- $Q(x,y,z) = R(x,y), S(y,z), T(z,x)$

- $Q(x,y,z) = R(x,y,z), S(x), T(y), K(z), M(x,u)$

Diagram:

```
  x -- y
   |   |
   |   |
   z --
```

```
  u -- x
   |   |
   |   |
  z - y
```
Fractional Edge Cover / Vertex Packing of a Hypergraph $G$

$G = \text{nodes } x_1, \ldots, x_k \text{ and hyperedges } R_1, \ldots, R_\ell$.

An edge cover = subset of edges that contain all nodes.

**Definition**

A fractional edge cover = sequence of positive numbers $u_1, \ldots, u_\ell$ s.t.:

$$\forall i : \sum_{j : x_i \in R_j} u_j \geq 1$$

Note: every edge cover is also a fractional edge cover (why?)

**Definition**

A fractional vertex packing = sequence of positive numbers $v_1, \ldots, v_k$ s.t.

$$\forall j : \sum_{i : x_i \in R_j} v_i \leq 1$$

Duality: $\min_u \sum_j u_j = \max_v \sum_i v_i = \rho^* = \text{fractional edge covering number}$
Fractional Edge Cover / Vertex Packing of a Hypergraph $G$

$G = \text{nodes } x_1, \ldots, x_k \text{ and hyperedges } R_1, \ldots, R_\ell$.

An edge cover = subset of edges that contain all nodes.

Definition

A fractional edge cover = sequence of positive numbers $u_1, \ldots, u_\ell$ s.t.:

$$\forall i : \sum_{j: x_i \in R_j} u_j \geq 1$$

Note: every edge cover is also a fractional edge cover (why?)

Definition

A fractional vertex packing = sequence of positive numbers $v_1, \ldots, v_k$ s.t.

$$\forall j : \sum_{i: x_i \in R_j} v_i \leq 1$$

Duality: $\min_u \sum_j u_j = \max_v \sum_i v_i = \rho^* = \text{fractional edge covering number}$
Fractional Edge Cover / Vertex Packing of a Hypergraph $G$

$G =$ nodes $x_1, \ldots, x_k$ and hyperedges $R_1, \ldots, R_\ell$.
An *edge cover* = subset of edges that contain all nodes.

**Definition**

A *fractional edge cover* = sequence of positive numbers $u_1, \ldots, u_\ell$ s.t.:

$$\forall i : \sum_{j : x_i \in R_j} u_j \geq 1$$

Note: every edge cover is also a fractional edge cover (why?)

**Definition**

A *fractional vertex packing* = sequence of positive numbers $v_1, \ldots, v_k$ s.t.:

$$\forall j : \sum_{i : x_i \in R_j} v_i \leq 1$$

Duality: $\min_u \sum_j u_j = \max_v \sum_i v_i = \rho^* = $ fractional edge covering number
Fractional Edge Cover / Vertex Packing of a Hypergraph $G$

$G = \text{nodes } x_1, \ldots, x_k \text{ and hyperedges } R_1, \ldots, R_\ell$.

An edge cover = subset of edges that contain all nodes.

**Definition**

A fractional edge cover = sequence of positive numbers $u_1, \ldots, u_\ell$ s.t.:

$$\forall i : \sum_{j : x_i \in R_j} u_j \geq 1$$

Note: every edge cover is also a fractional edge cover (why?)

**Definition**

A fractional vertex packing = sequence of positive numbers $v_1, \ldots, v_k$ s.t.:

$$\forall j : \sum_{i : x_i \in R_j} v_i \leq 1$$

Duality: $\min_u \sum_j u_j = \max_v \sum_i v_i = \rho^* = \text{fractional edge covering number}$
Fractional Edge Cover / Vertex Packing of a Hypergraph $G$

$G = \text{nodes } x_1, \ldots, x_k \text{ and hyperedges } R_1, \ldots, R_\ell$.

An edge cover = subset of edges that contain all nodes.

**Definition**

A fractional edge cover = sequence of positive numbers $u_1, \ldots, u_\ell$ s.t.:

$$\forall i : \sum_{j: x_i \in R_j} u_j \geq 1$$

Note: every edge cover is also a fractional edge cover (why?)

**Definition**

A fractional vertex packing = sequence of positive numbers $v_1, \ldots, v_k$ s.t.

$$\forall j : \sum_{i: x_i \in R_j} v_i \leq 1$$

Duality: $\min_u \sum_j u_j = \max_v \sum_i v_i = \rho^* = \text{fractional edge covering number}$
Fractional Edge Cover / Vertex Packing of a Hypergraph $G$

$G = \text{nodes } x_1, \ldots, x_k \text{ and hyperedges } R_1, \ldots, R_\ell.$

An edge cover = subset of edges that contain all nodes.

**Definition**

A fractional edge cover = sequence of positive numbers $u_1, \ldots, u_\ell$ s.t.:

\[
\forall i : \sum_{j : x_i \in R_j} u_j \geq 1
\]

Note: every edge cover is also a fractional edge cover (why?)

**Definition**

A fractional vertex packing = sequence of positive numbers $v_1, \ldots, v_k$ s.t.

\[
\forall j : \sum_{i : x_i \in R_j} v_i \leq 1
\]

Duality: $\min_u \sum_j u_j = \max_v \sum_i v_i = \rho^* = \text{fractional edge covering number}$
AGM Inequality

Full conjunctive query: \( Q(x) = R_1(x_1), \ldots, R_\ell(x_\ell) \)
Relation sizes: \( |R_1| = m_1, \ldots, |R_\ell| = m_\ell \)

Proposition (Simple!)

Let \( R_{i_1}, \ldots, R_{i_u} \) be any edge cover. Then \( |Q| \leq m_{i_1} \cdot m_{i_2} \cdots m_{i_u} \)

(proof in class)

Atserias, Grohe and Marx proved:

Theorem (AGM’13)

Let \( u_1, \ldots, u_\ell \) be any fractional edge cover. Then \( |Q| \leq m_1^{u_1} \cdot m_2^{u_2} \cdots m_\ell^{u_\ell} \)

We will prove it today. But first, let’s see examples.
**AGM Inequality**

Full conjunctive query: $Q(x) = R_1(x_1), \ldots, R_\ell(x_\ell)$
Relation sizes: $|R_1| = m_1, \ldots, |R_\ell| = m_\ell$

**Proposition (Simple!)**

Let $R_{i_1}, \ldots, R_{i_u}$ be any edge cover. Then $|Q| \leq m_{i_1} \cdot m_{i_2} \cdots m_{i_u}$

(proof in class)

Atserias, Grohe and Marx proved:

**Theorem (AGM’13)**

Let $u_1, \ldots, u_\ell$ be any fractional edge cover. Then $|Q| \leq m_{u_1}^{u_1} \cdot m_{u_2}^{u_2} \cdots m_{u_\ell}^{u_\ell}$

We will prove it today. But first, let’s see examples.
**AGM Inequality**

Full conjunctive query: \( Q(\mathbf{x}) = R_1(\mathbf{x}_1), \ldots, R_\ell(\mathbf{x}_\ell) \)

Relation sizes: \(|R_1| = m_1, \ldots, |R_\ell| = m_\ell\)

---

**Proposition (Simple!)**

Let \( R_{i_1}, \ldots, R_{i_u} \) be any edge cover. Then \( |Q| \leq m_{i_1} \cdot m_{i_2} \cdots m_{i_u} \)

(proof in class)

Atserias, Grohe and Marx proved:

**Theorem (AGM’13)**

Let \( u_1, \ldots, u_\ell \) be any fractional edge cover. Then \( |Q| \leq m_{1}^{u_1} \cdot m_{2}^{u_2} \cdots m_{\ell}^{u_\ell} \)

We will prove it today. But first, let’s see examples.
AGM Inequality – A Simple Example

\[ AGM_u(Q) = m_1^{u_1} \cdot m_2^{u_2} \cdots m_\ell^{u_\ell} \]

\[ Q(x, y, z) = R(x, y), S(y, z), T(z, x) \]
\[ |R| = |S| = |T| = m \]
AGM Inequality – A Simple Example

\[
AGM_u(Q) = m_1^{u_1} \cdot m_2^{u_2} \cdots m_{\ell}^{u_{\ell}}
\]

\[
Q(x, y, z) = R(x, y), S(y, z), T(z, x)
\]

\[
|R| = |S| = |T| = m
\]

A fractional edge: \(u = (1/2, 1/2, 1/2)\)
AGM Inequality – A Simple Example

\[ \text{AGM}_u(Q) = m_1^{u_1} \cdot m_2^{u_2} \cdots m_\ell^{u_\ell} \]

\[ Q(x, y, z) = R(x, y), S(y, z), T(z, x) \]
\[ |R| = |S| = |T| = m \]

A fractional edge: \( u = (1/2, 1/2, 1/2) \)

It follows that \( |Q| \leq m^{1/2} m^{1/2} m^{1/2} = m^{3/2} \)

With \( m \) (typed) edges you can build at most \( m^{3/2} \) triangles!
AGM Bound

Definition

\[ \text{AGM}(Q) = \min_u m_1^{u_1} \cdot m_2^{u_2} \cdots m_\ell^{u_\ell} \]

Obviously: \( |Q| \leq \text{AGM}(Q) \).

Example

\[ Q(x, y, z) = R(x, y), S(y, z), T(z, x), \quad \begin{array}{c} \text{agm}(Q) = \min \text{ of} \\ m_1 m_2 \\ m_1 m_3 \\ m_2 m_3 \\ \left( m_1 m_2 m_3 \right)^{1/2} \end{array} \]

Example

\[ Q(x, y, z, v, w) = R(x, y), S(y, z), T(z, v), K(v, w) \]

\[ u = \begin{array}{c} \begin{array}{c} (1, 0, 1, 1) \\ (1, 1, 0, 1) \end{array} \\ \begin{array}{c} (1, 1, 0, 1) \\ (1, 1, 0, 1) \end{array} \end{array} \]

\[ \text{agm}(Q) = \begin{array}{c} \min \text{ of} \\ m_1 m_3 m_4 \\ m_1 m_2 m_4 \end{array} \]
AGM Bound

Definition

\[ AGM(Q) = \min_u m_1^{u_1} \cdot m_2^{u_2} \cdots m_\ell^{u_\ell} \]

Obviously: \( |Q| \leq AGM(Q) \).

Example

\[ Q(x, y, z) = R(x, y), S(y, z), T(z, x), \quad |R| = m_1, |S| = m_2, |T| = m_3 \]

\[
\begin{array}{|c|c|c|c|}
\hline
u & (1, 1, 0) & (1, 0, 1) & (0, 1, 1) & (\frac{1}{2}, \frac{1}{2}, \frac{1}{2}) \\
\hline
AGM(Q) = \min \text{ of} & m_1 m_2 & m_1 m_3 & m_2 m_3 & (m_1 m_2 m_3)^{1/2} \\
\hline
\end{array}
\]

Example

\[ Q(x, y, z, v, w) = R(x, y), S(y, z), T(z, v), K(v, w) \]

\[
\begin{array}{|c|c|c|c|}
\hline
u & (1, 0, 1, 1) & (1, 1, 0, 1) \\
\hline
AGM(Q) = \min \text{ of} & m_1 m_3 m_4 & m_1 m_2 m_4 \\
\hline
\end{array}
\]
AGM Bound

**Definition**

\[
AGM(Q) = \min_u m_1^{u_1} \cdot m_2^{u_2} \cdots m_\ell^{u_\ell}
\]

Obviously: \(|Q| \leq AGM(Q)|.

**Example**

\[
Q(x, y, z) = R(x, y), S(y, z), T(z, x), \quad |R| = m_1, |S| = m_2, |T| = m_3
\]

\[
\begin{array}{c|c|c|c}
\text{u} & (1, 1, 0) & (1, 0, 1) & (0, 1, 1) & (\frac{1}{2}, \frac{1}{2}, \frac{1}{2}) \\
\text{AGM}(Q) & m_1 m_2 & m_1 m_3 & m_2 m_3 & (m_1 m_2 m_3)^{1/2}
\end{array}
\]

**Example**

\[
Q(x, y, z, v, w) = R(x, y), S(y, z), T(z, v), K(v, w)
\]

\[
\begin{array}{c|c}
\text{u} & (1, 0, 1, 1) \quad (1, 1, 0, 1) \\
\text{AGM}(Q) & m_1 m_3 m_4 \quad m_1 m_2 m_4
\end{array}
\]
AGM Bound v.s. Fractional Vertex Covering Number

\[ AGM_u(Q) = m_1^{u_1} \cdot m_2^{u_2} \cdots m_\ell^{u_\ell} \]

\[ AGM(Q) = \min_u \ AGM_u(Q) \]

is the optimal solution to:

\[
\begin{align*}
\text{minimize} & \quad \sum_j u_j \log m_j \\
\forall i : & \quad \sum_{j:x_i \in R_j} u_j \geq 1
\end{align*}
\]

Notice: when \(m_1 = \cdots = m_\ell = m\) then \(AGM(Q) = m^{\rho^*}\).

Next: we will prove the AGM bound
Friedgut’s Inequality

Cauchy-Schwartz: \[ \sum_i a_i b_i \leq (\sum_i a_i^2)^{1/2} (\sum_i b_i^2)^{1/2} \]
Friedgut’s Inequality

Cauchy-Schwartz: \[ \sum_i a_i b_i \leq \left( \sum_i a_i^2 \right)^{\frac{1}{2}} \left( \sum_i b_i^2 \right)^{\frac{1}{2}} \]

Triangle: \[ \sum_{i,j,k} a_{ij} b_{jk} c_{ki} \leq \left( \sum_{i,j} a_{ij}^2 \right)^{\frac{1}{2}} \left( \sum_{j,k} b_{jk}^2 \right)^{\frac{1}{2}} \left( \sum_{k,i} c_{ki}^2 \right)^{\frac{1}{2}} \]
Friedgut’s Inequality

Cauchy-Schwartz: \[ \sum_{i} a_{i} b_{i} \leq \left( \sum_{i} a_{i}^{2} \right)^{\frac{1}{2}} \left( \sum_{i} b_{i}^{2} \right)^{\frac{1}{2}} \]

Triangle: \[ \sum_{i,j,k} a_{ij} b_{jk} c_{ki} \leq \left( \sum_{i,j} a_{ij}^{2} \right)^{\frac{1}{2}} \left( \sum_{j,k} b_{jk}^{2} \right)^{\frac{1}{2}} \left( \sum_{k,i} c_{ki}^{2} \right)^{\frac{1}{2}} \]

Hölder \((u + v + w \geq 1)\): \[ \sum_{i} a_{i} b_{i} c_{i} \leq \left( \sum_{i} a_{i}^{u} \right)^{\frac{1}{u}} \left( \sum_{i} b_{i}^{v} \right)^{\frac{1}{v}} \left( \sum_{i} c_{i}^{w} \right)^{\frac{1}{w}} \]
Friedgut’s Inequality

Cauchy-Schwartz: \[ \sum_i a_i b_i \leq \left( \sum_i a_i^2 \right)^{\frac{1}{2}} \left( \sum_i b_i^2 \right)^{\frac{1}{2}} \]

Triangle: \[ \sum_{i,j,k} a_{ij} b_{jk} c_{ki} \leq \left( \sum_{i,j} a_{ij}^2 \right)^{\frac{1}{2}} \left( \sum_{j,k} b_{jk}^2 \right)^{\frac{1}{2}} \left( \sum_{k,i} c_{ki}^2 \right)^{\frac{1}{2}} \]

Hölder \((u + v + w \geq 1)\): \[ \sum_i a_i b_i c_i \leq \left( \sum_i a_i^u \right)^u \left( \sum_i b_i^v \right)^v \left( \sum_i c_i^w \right)^w \]

Theorem (Friedgut’04)

Let \( Q(x) = R_1(x_1), \ldots, R_\ell(x_\ell) \) be a query and \( u_1, \ldots, u_\ell \) be a fractional edge cover. Then:

\[ \sum_x a_{1,x_1} \cdots a_{\ell,x_\ell} \leq \left( \sum_{x_1} a_{1,x_1}^{\frac{1}{u_1}} \right)^{u_1} \cdots \left( \sum_{x_\ell} a_{\ell,x_\ell}^{\frac{1}{u_\ell}} \right)^{u_\ell} \]

What are the queries in the examples above?
Friedgut’s Inequality

Cauchy-Schwartz: \[ \sum_i a_i b_i \leq \left( \sum_i a_i^2 \right)^{\frac{1}{2}} \left( \sum_i b_i^2 \right)^{\frac{1}{2}} \]

Triangle: \[ \sum_{i,j,k} a_{ij} b_{jk} c_{ki} \leq \left( \sum_{i,j} a_{ij}^2 \right)^{\frac{1}{2}} \left( \sum_{j,k} b_{jk}^2 \right)^{\frac{1}{2}} \left( \sum_{k,i} c_{ki}^2 \right)^{\frac{1}{2}} \]

Hölder \((u + v + w \geq 1)\): \[ \sum_i a_i b_i c_i \leq \left( \sum_i a_i^u \right)^u \left( \sum_i b_i^v \right)^v \left( \sum_i c_i^w \right)^w \]

Theorem (Friedgut’04)

Let \( Q(x) = R_1(x_1), \ldots, R_\ell(x_\ell) \) be a query and \( u_1, \ldots, u_\ell \) be a fractional edge cover. Then:
\[
\sum_x a_1,x_1 \cdots a_\ell,x_\ell \leq \left( \sum_{x_1} a_{1,x_1}^{\frac{1}{u_1}} \right)^{u_1} \cdots \left( \sum_{x_\ell} a_{\ell,x_\ell}^{\frac{1}{u_\ell}} \right)^{u_\ell}
\]

What are the queries in the examples above?
\( Q_{\text{Cauchy-Schwartz}}(x) = R(x), S(x); \)
Friedgut’s Inequality

Cauchy-Schwartz: \[ \sum_{i} a_i b_i \leq \left( \sum_{i} a_i^2 \right)^{\frac{1}{2}} \left( \sum_{i} b_i^2 \right)^{\frac{1}{2}} \]

Triangle:
\[ \sum_{i,j,k} a_{ij} b_{jk} c_{ki} \leq \left( \sum_{i,j} a_{ij}^2 \right)^{\frac{1}{2}} \left( \sum_{j,k} b_{jk}^2 \right)^{\frac{1}{2}} \left( \sum_{k,i} c_{ki}^2 \right)^{\frac{1}{2}} \]

H"older \((u + v + w \geq 1)\):
\[ \sum_{i} a_i b_i c_i \leq \left( \sum_{i} a_i^u \right)^{\frac{1}{u}} \left( \sum_{i} b_i^v \right)^{\frac{1}{v}} \left( \sum_{i} c_i^w \right)^{\frac{1}{w}} \]

Theorem (Friedgut’04)

Let \( Q(x) = R_1(x_1), \ldots, R_\ell(x_\ell) \) be a query and \( u_1, \ldots, u_\ell \) be a fractional edge cover. Then:
\[ \sum_{x_a_1, x_1} \ldots a_\ell, x_\ell \leq \left( \sum_{x_1} a_{1,x_1}^{\frac{1}{u_1}} \right)^{u_1} \ldots \left( \sum_{x_\ell} a_{\ell,x_\ell}^{\frac{1}{u_\ell}} \right)^{u_\ell} \]

What are the queries in the examples above?
\( Q_{\text{Cauchy-Schwartz}}(x) = R(x), S(x); \)
\( Q_{\text{triangle}}(x, y, z) = R(x, y), S(y, z), T(z, x); \)
Friedgut’s Inequality

Cauchy-Schwartz:
\[ \sum_i a_i b_i \leq \left( \sum_i a_i^2 \right)^{\frac{1}{2}} \left( \sum_i b_i^2 \right)^{\frac{1}{2}} \]

Triangle:
\[ \sum_{i,j,k} a_{ij} b_{jk} c_{ki} \leq \left( \sum_i a_{ij}^2 \right)^{\frac{1}{2}} \left( \sum_j b_{jk}^2 \right)^{\frac{1}{2}} \left( \sum_k c_{ki}^2 \right)^{\frac{1}{2}} \]

Hölder \((u + v + w \geq 1)\):
\[ \sum_i a_i b_i c_i \leq \left( \sum_i a_i^u \right)^u \left( \sum_i b_i^v \right)^v \left( \sum_i c_i^w \right)^w \]

Theorem (Friedgut’04)

Let \( Q(x) = R_1(x_1), \ldots, R_\ell(x_\ell) \) be a query and \( u_1, \ldots, u_\ell \) be a fractional edge cover. Then:
\[ \sum_x a_{1,x_1} \cdots a_{\ell,x_\ell} \leq \left( \sum_{x_1} a_{1,x_1}^{\frac{1}{u_1}} \right)^{u_1} \cdots \left( \sum_{x_\ell} a_{\ell,x_\ell}^{\frac{1}{u_\ell}} \right)^{u_\ell} \]

What are the queries in the examples above?

\( Q_{\text{Cauchy-Schwartz}}(x) = R(x), S(x); \)
\( Q_{\text{triangle}}(x, y, z) = R(x, y), S(y, z), T(z, x); \)
\( Q_{\text{Hölder}}(x) = R(x), S(x), T(x) \)
Friedgut’s Inequality – Proof

Query $Q(x) = R_1(x_1), \ldots, R_\ell(x_\ell)$, fractional cover $u_1, \ldots, u_\ell$

\[
\sum_x a_1^{u_1} \cdots a_\ell^{u_\ell} \leq \left( \sum_x a_1^{u_1} \right)^{u_1} \cdots \left( \sum_x a_\ell^{u_\ell} \right)^{u_\ell}
\]

Proof:

by induction on $x$

Base Case.

$\divides x = 1$: $Q(x) = R_1(x), \ldots, R_\ell(x)$, $u_1 + \cdots + u_\ell \geq 1$

Prove:

This is Hölder.

Induction Step.

Pick a variable $x$, and remove it. For example, $Q(x, y, z) = R(x, y), S(y, z), T(z, x)$ becomes $Q'(y, z) = R'(y), S(y, z), T'(z)$.
Friedgut’s Inequality – Proof

Query \( Q(x) = R_1(x_1), \ldots, R_\ell(x_\ell) \), fractional cover \( u_1, \ldots, u_\ell \)

\[
\sum_x a_{1,x_1}^{u_1} \cdots a_{\ell,x_\ell}^{u_\ell} \leq \left( \sum_{x_1} a_{1,x_1} \right)^{u_1} \cdots \left( \sum_{x_\ell} a_{\ell,x_\ell} \right)^{u_\ell}
\]

**Proof:** by induction on \(|x|\)
Friedgut’s Inequality – Proof

Query \( Q(x) = R_1(x_1), \ldots, R_\ell(x_\ell) \), fractional cover \( u_1, \ldots, u_\ell \)

\[
\sum_x a_{1,x_1}^{u_1} \ldots a_{\ell,x_\ell}^{u_\ell} \leq \left( \sum_x a_{1,x} \right)^{u_1} \ldots \left( \sum_x a_{\ell,x} \right)^{u_\ell}
\]

Proof: by induction on \( |x| \)

Base Case. \( |x| = 1: Q(x) = R_1(x), \ldots, R_\ell(x), u_1 + \ldots + u_\ell \geq 1 \)

Prove: \( \sum_x a_{1,x}^{u_1} \ldots a_{\ell,x}^{u_\ell} \leq \left( \sum_x a_{1,x} \right)^{u_1} \ldots \left( \sum_x a_{\ell,x} \right)^{u_\ell} \) This is Hölder.
Friedgut’s Inequality – Proof

Query \( Q(x) = R_1(x_1), \ldots, R_\ell(x_\ell) \), fractional cover \( u_1, \ldots, u_\ell \)

\[
\sum_x a_{1,x_1}^{u_1} \cdots a_{\ell,x_\ell}^{u_\ell} \leq \left( \sum_{x_1} a_{1,x_1} \right)^{u_1} \cdots \left( \sum_{x_\ell} a_{\ell,x_\ell} \right)^{u_\ell}
\]

**Proof:** by induction on \( |x| \)

**Base Case.** \( |x| = 1 \): \( Q(x) = R_1(x), \ldots, R_\ell(x), u_1 + \ldots + u_\ell \geq 1 \)

Prove: \( \sum_x a_{1,x}^{u_1} \cdots a_{\ell,x}^{u_\ell} \leq \left( \sum_x a_{1,x} \right)^{u_1} \cdots \left( \sum_x a_{\ell,x} \right)^{u_\ell} \) This is Hölder.

**Induction Step.** Pick a variable \( x \), and remove it. For example, \( Q(x,y,z) = R(x,y), S(y,z), T(z,x) \) becomes \( Q'(y,z) = R'(y), S(y,z), T'(z) \)
Friedgut’s Inequality – Proof

Query $Q(x) = R_1(x_1), \ldots, R_\ell(x_\ell)$, fractional cover $u_1, \ldots, u_\ell$

$$\sum_x a_{1,x_1}^{u_1} \cdots a_{\ell,x_\ell}^{u_\ell} \leq (\sum_x a_{1,x_1})^{u_1} \cdots (\sum_x a_{\ell,x_\ell})^{u_\ell}$$

**Proof:** by induction on $|x|$

**Base Case.** $|x| = 1$: $Q(x) = R_1(x), \ldots, R_\ell(x)$, $u_1 + \ldots + u_\ell \geq 1$

Prove: $\sum_x a_{1,x}^{u_1} \cdots a_{\ell,x}^{u_\ell} \leq (\sum_x a_{1,x})^{u_1} \cdots (\sum_x a_{\ell,x})^{u_\ell}$ This is Hölder.

**Induction Step.** Pick a variable $x$, and remove it. For example, $Q(x, y, z) = R(x, y), S(y, z), T(z, x)$ becomes $Q'(y, z) = R'(y), S(y, z), T'(z)$

$$\sum_{x y z} a_{x y}^{u_1} b_{y z}^{u_2} c_{z x}^{u_3} = \sum_{y z} b_{y z}^{u_2} \sum_x a_{x y}^{u_1} c_{z x}^{u_3} \quad \text{group by } \sum_x$$
Friedgut’s Inequality – Proof

Query \( Q(x) = R_1(x_1), \ldots, R_\ell(x_\ell) \), fractional cover \( u_1, \ldots, u_\ell \)

\[
\sum_x a_{1,x_1}^{u_1} \cdots a_{\ell,x_\ell}^{u_\ell} \leq (\sum_{x_1} a_{1,x_1})^{u_1} \cdots (\sum_{x_\ell} a_{\ell,x_\ell})^{u_\ell}
\]

Proof: by induction on \( |x| \)

**Base Case.** \( |x| = 1 \): \( Q(x) = R_1(x), \ldots, R_\ell(x) \), \( u_1 + \ldots + u_\ell \geq 1 \)

Prove: \( \sum_x a_{1,x}^{u_1} \cdots a_{\ell,x}^{u_\ell} \leq (\sum_x a_{1,x})^{u_1} \cdots (\sum_x a_{\ell,x})^{u_\ell} \) This is Hölder.

**Induction Step.** Pick a variable \( x \), and remove it. For example, \( Q(x, y, z) = R(x, y), S(y, z), T(z, x) \) becomes \( Q'(y, z) = R'(y), S(y, z), T'(z) \)

\[
\sum_{xyz} a_{xy}^{u_1} b_{yz}^{u_2} c_{zx}^{u_3} = \sum_{yz} b_{yz}^{u_2} \sum_{x} a_{xy}^{u_1} c_{zx}^{u_3}
\]

\[
\leq \sum_{yz} b_{yz}^{u_2} (\sum_{x} a_{xy})^{u_1} (\sum_{x} c_{zx})^{u_3}
\]

Hölder \( u_1 + u_3 \geq 1 \)
Friedgut’s Inequality – Proof

Query \( Q(x) = R_1(x_1), \ldots, R_\ell(x_\ell) \), fractional cover \( u_1, \ldots, u_\ell \)

\[
\sum_x a_{1,x_1}^{u_1} \cdots a_{\ell,x_\ell}^{u_\ell} \leq \left( \sum_x a_{1,x_1} \right)^{u_1} \cdots \left( \sum_x a_{\ell,x_\ell} \right)^{u_\ell}
\]

Proof: by induction on \(|x|\)

Base Case. \(|x| = 1\): \( Q(x) = R_1(x), \ldots, R_\ell(x) \), \( u_1 + \ldots + u_\ell \geq 1 \)

Prove: \( \sum_x a_{1,x}^{u_1} \cdots a_{\ell,x}^{u_\ell} \leq \left( \sum_x a_{1,x} \right)^{u_1} \cdots \left( \sum_x a_{\ell,x} \right)^{u_\ell} \) This is Hölder.

**Induction Step.** Pick a variable \( x \), and remove it. For example, \( Q(x, y, z) = R(x, y), S(y, z), T(z, x) \) becomes \( Q'(y, z) = R'(y), S(y, z), T'(z) \)

\[
\sum_{x,y,z} a_{xy}^{u_1} b_{yz}^{u_2} c_{zx}^{u_3} = \sum_{y,z} b_{yz}^{u_2} \sum_x a_{xy}^{u_1} c_{zx}^{u_3}
\]

\[
\leq \sum_{y,z} b_{yz}^{u_2} \left( \sum_x a_{xy} \right)^{u_1} \left( \sum_x c_{zx} \right)^{u_3}
\]

\[
= \sum_{y,z} b_{yz}^{u_2} A_y^{u_1} C_z^{u_3} \leq \left( \sum_{y,z} b_{yz} \right)^{u_2} \left( \sum_y A_y \right)^{u_1} \left( \sum_z C_z \right)^{u_3}
\]

Hölder \( u_1 + u_3 \geq 1 \)

Induction for \( Q' \)
Friedgut’s Inequality – Proof

Query $Q(x) = R_1(x_1), \ldots, R_\ell(x_\ell)$, fractional cover $u_1, \ldots, u_\ell$

$$\sum_x a_{1,x_1}^{u_1} \cdots a_{\ell,x_\ell}^{u_\ell} \leq \left( \sum_x a_{1,x_1} \right)^{u_1} \cdots \left( \sum_x a_{\ell,x_\ell} \right)^{u_\ell}$$

Proof: by induction on $|x|$

Base Case. $|x| = 1$: $Q(x) = R_1(x), \ldots, R_\ell(x), u_1 + \ldots + u_\ell \geq 1$

Prove: $\sum_x a_{1,x_1}^{u_1} \cdots a_{\ell,x_\ell}^{u_\ell} \leq \left( \sum_x a_{1,x} \right)^{u_1} \cdots \left( \sum_x a_{\ell,x} \right)^{u_\ell}$ This is Hölder.

Induction Step. Pick a variable $x$, and remove it. For example, $Q(x, y, z) = R(x, y), S(y, z), T(z, x)$ becomes $Q'(y, z) = R'(y), S(y, z), T'(z)$

$$\sum_{xyz} a_{xy}^{u_1} b_{yz}^{u_2} c_{zx}^{u_3} = \sum_{yz} b_{yz}^{u_2} \sum_{x} a_{xy}^{u_1} c_{zx}^{u_3}$$

$$\leq \sum_{yz} b_{yz}^{u_2} \left( \sum_x a_{xy} \right)^{u_1} \left( \sum_x c_{zx} \right)^{u_3}$$

Hölder $u_1 + u_3 \geq 1$

$$\sum_{yz} b_{yz}^{u_2} A_y^{u_1} C_z^{u_3} \leq \left( \sum_{yz} b_{yz} \right)^{u_2} \left( \sum_y A_y \right)^{u_1} \left( \sum_z C_z \right)^{u_3}$$

Induction for $Q'$

$$= \left( \sum_{yz} b_{yz} \right)^{u_2} \left( \sum_{xy} a_{xy} \right)^{u_1} \left( \sum_{zx} c_{zx} \right)^{u_3}$$
Friedgut’s Inequality – Proof

Query \( Q(x) = R_1(x_1), \ldots, R_\ell(x_\ell) \), fractional cover \( u_1, \ldots, u_\ell \)

\[
\sum_x a_{1,x_1}^{u_1} \cdots a_{\ell,x_\ell}^{u_\ell} \leq \left( \sum_x a_{1,x_1} \right)^{u_1} \cdots \left( \sum_x a_{\ell,x_\ell} \right)^{u_\ell}
\]

**Proof:** by induction on \( |x| \)

**Base Case.** \( |x| = 1: \) \( Q(x) = R_1(x), \ldots, R_\ell(x) \), \( u_1 + \ldots + u_\ell \geq 1 \)

Prove: \( \sum_x a_{1,x_1}^{u_1} \cdots a_{\ell,x_\ell}^{u_\ell} \leq \left( \sum_x a_{1,x_1} \right)^{u_1} \cdots \left( \sum_x a_{\ell,x_\ell} \right)^{u_\ell} \) This is Hölder.

**Induction Step.** Pick a variable \( x \), and remove it. For example, \( Q(x,y,z) = R(x,y), S(y,z), T(z,x) \) becomes \( Q'(y,z) = R'(y), S(y,z), T'(z) \)

\[
\sum_{xyz} a_{xy}^{u_1} b_{yz}^{u_2} c_{zx}^{u_3} = \sum_{yz} b_{yz}^{u_2} \sum_{x} a_{xy}^{u_1} c_{zx}^{u_3}
\]

\[
\leq \sum_{yz} b_{yz}^{u_2} \left( \sum_{x} a_{xy} \right)^{u_1} \left( \sum_{x} c_{zx} \right)^{u_3}
\]

\[
= \sum_{yz} b_{yz}^{u_2} A_y^{u_1} C_z^{u_3} \leq \left( \sum_{yz} b_{yz} \right)^{u_2} \left( \sum_{y} A_y \right)^{u_1} \left( \sum_{z} C_z \right)^{u_3}
\]

\[
= \left( \sum_{yz} b_{yz} \right)^{u_2} \left( \sum_{xy} a_{xy} \right)^{u_1} \left( \sum_{zx} c_{zx} \right)^{u_3}
\]

Hölder \( u_1 + u_3 \geq 1 \)

Induction for \( Q' \)

**QED**
The AGM Inequality – Proof

Query $Q(x) = R_1(x_1), \ldots, R_\ell(x_\ell)$, fractional cover $u_1, \ldots, u_\ell$

Sizes $|R_1| = m_1, \ldots, |R_\ell| = m_\ell$

Prove $|Q| \leq m_1^{u_1} \cdots m_\ell^{u_\ell}$

Let $\text{Dom} = \text{the domain of all constants in the relations } R_1, \ldots, R_\ell$.

For every $j = 1, \ldots, \ell$, and every tuple $x_j \in \text{Dom}^{|x_j|}$, define:

$$a_{j,x_j} = \begin{cases} 
1 & \text{if the tuple } x_j \text{ belongs to } R_j \\
0 & \text{otherwise}
\end{cases}$$

Then: $m_j = |R_j| = \sum_{x_j \in \text{Dom}^{|x_j|}} a_{j,x_j}$, $|Q| = \sum_{x \in \text{Dom}^{|x|}} a_{1,x_1} \cdots a_{\ell,x_\ell}$

Now use Friedgut’s inequality:

$$|Q| = \sum_x a_{1,x_1}^{u_1} \cdots a_{\ell,x_\ell}^{u_\ell} \leq (\sum_{x_1} a_{1,x_1}^{u_1})^{u_1} \cdots (\sum_{x_\ell} a_{\ell,x_\ell}^{u_\ell})^{u_\ell} = m_1^{u_1} \cdots m_\ell^{u_\ell}$$

QED
The AGM Inequality – Proof

Query \( Q(x) = R_1(x_1), \ldots, R_\ell(x_\ell) \), fractional cover \( u_1, \ldots, u_\ell \)

Sizes \( |R_1| = m_1, \ldots, |R_\ell| = m_\ell \)

Prove \( |Q| \leq m_1^{u_1} \ldots m_\ell^{u_\ell} \)

Let \( \text{Dom} = \) the domain of all constants in the relations \( R_1, \ldots, R_\ell \).

For every \( j = 1, \ldots, \ell \), and every tuple \( x_j \in \text{Dom} \setminus x_j \), define:

\[
a_{j,x_j} = \begin{cases} 
1 & \text{if the tuple } x_j \text{ belongs to } R_j \\
0 & \text{otherwise} 
\end{cases}
\]

Then: \( m_j = |R_j| = \sum_{x_j \in \text{Dom}} a_{j,x_j}, \ |Q| = \sum_{x \in \text{Dom}} a_{1,x_1} \ldots a_{\ell,x_\ell} \)

Now use Friedgut’s inequality:

\[
|Q| = \sum_{x} a_{1,x_1}^{u_1} \ldots a_{\ell,x_\ell}^{u_\ell} \leq (\sum_{x_1} a_{1,x_1})^{u_1} \ldots (\sum_{x_\ell} a_{\ell,x_\ell})^{u_\ell} = m_1^{u_1} \ldots m_\ell^{u_\ell}
\]

QED
The AGM Inequality – Proof

Query \( Q(x) = R_1(x_1), \ldots, R_\ell(x_\ell) \), fractional cover \( u_1, \ldots, u_\ell \)

Sizes \( |R_1| = m_1, \ldots, |R_\ell| = m_\ell \)

Prove \( |Q| \leq m_1^{u_1} \cdots m_\ell^{u_\ell} \)

Let \( \text{Dom} = \) the domain of all constants in the relations \( R_1, \ldots, R_\ell \).

For every \( j = 1, \ldots, \ell \), and every tuple \( x_j \in \text{Dom}^{x_j} \), define:

\[
a_{j,x_j} = \begin{cases} 
1 & \text{if the tuple } x_j \text{ belongs to } R_j \\
0 & \text{otherwise}
\end{cases}
\]

Then: \( m_j = |R_j| = \sum_{x_j \in \text{Dom}^{x_j}} a_{j,x_j} \), \( |Q| = \sum_{x \in \text{Dom}^x} a_{1,x_1} \cdots a_{\ell,x_\ell} \)

Now use Friedgut’s inequality:

\[
|Q| = \sum_{x} a_{1,x_1}^{u_1} \cdots a_{\ell,x_\ell}^{u_\ell} \leq (\sum_{x_1} a_{1,x_1})^{u_1} \cdots (\sum_{x_\ell} a_{\ell,x_\ell})^{u_\ell} = m_1^{u_1} \cdots m_\ell^{u_\ell}
\]

QED
The AGM Inequality – Proof

Query \( Q(\mathbf{x}) = R_1(\mathbf{x}_1), \ldots, R_\ell(\mathbf{x}_\ell) \), fractional cover \( u_1, \ldots, u_\ell \)

Sizes \( |R_1| = m_1, \ldots, |R_\ell| = m_\ell \)

Prove \( |Q| \leq m_1^{u_1} \cdots m_\ell^{u_\ell} \)

Let \( \text{Dom} \) = the domain of all constants in the relations \( R_1, \ldots, R_\ell \).

For every \( j = 1, \ldots, \ell \), and every tuple \( \mathbf{x}_j \in \text{Dom} |\mathbf{x}_j| \), define:

\[
a_{j,\mathbf{x}_j} = \begin{cases} 
1 & \text{if the tuple } \mathbf{x}_j \text{ belongs to } R_j \\
0 & \text{otherwise}
\end{cases}
\]

Then: \( m_j = |R_j| = \sum_{\mathbf{x}_j \in \text{Dom} |\mathbf{x}_j|} a_{j,\mathbf{x}_j}, \quad |Q| = \sum_{\mathbf{x} \in \text{Dom} |\mathbf{x}|} a_{1,\mathbf{x}_1} \cdots a_{\ell,\mathbf{x}_\ell} \)

Now use Friedgut’s inequality:

\[
|Q| = \sum_{\mathbf{x}} a_{1,\mathbf{x}_1}^{u_1} \cdots a_{\ell,\mathbf{x}_\ell}^{u_\ell} \leq \left( \sum_{\mathbf{x}_1} a_{1,\mathbf{x}_1} \right)^{u_1} \cdots \left( \sum_{\mathbf{x}_\ell} a_{\ell,\mathbf{x}_\ell} \right)^{u_\ell} = m_1^{u_1} \cdots m_\ell^{u_\ell}
\]

QED
The AGM Inequality – Proof

Query \( Q(x) = R_1(x_1), \ldots, R_\ell(x_\ell) \), fractional cover \( u_1, \ldots, u_\ell \)

Sizes \( |R_1| = m_1, \ldots, |R_\ell| = m_\ell \)

Prove \( |Q| \leq m_1^{u_1} \cdots m_\ell^{u_\ell} \)

Let \( \text{Dom} = \) the domain of all constants in the relations \( R_1, \ldots, R_\ell \).

For every \( j = 1, \ldots, \ell \), and every tuple \( x_j \in \text{Dom}|_{x_j} \), define:

\[
a_{j,x_j} = \begin{cases} 
1 & \text{if the tuple } x_j \text{ belongs to } R_j \\
0 & \text{otherwise}
\end{cases}
\]

Then: \( m_j = |R_j| = \sum_{x_j \in \text{Dom}|_{x_j}} a_{j,x_j}, \quad |Q| = \sum_{x \in \text{Dom}|_x} a_{1,x_1} \cdots a_{\ell,x_\ell} \)

Now use Friedgut’s inequality:
The AGM Inequality – Proof

Query $Q(x) = R_1(x_1), \ldots, R_\ell(x_\ell)$, fractional cover $u_1, \ldots, u_\ell$

Sizes $|R_1| = m_1, \ldots, |R_\ell| = m_\ell$

Prove $|Q| \leq m_1^{u_1} \cdots m_\ell^{u_\ell}$

Let $\text{Dom} = \text{the domain of all constants in the relations } R_1, \ldots, R_\ell$.

For every $j = 1, \ldots, \ell$, and every tuple $x_j \in \text{Dom}^{\{|x_j|\}}$, define:

$$a_{j,x_j} = \begin{cases} 
1 & \text{if the tuple } x_j \text{ belongs to } R_j \\
0 & \text{otherwise}
\end{cases}$$

Then: $m_j = |R_j| = \sum_{x_j \in \text{Dom}^{\{|x_j|\}}} a_{j,x_j}$, $|Q| = \sum_{x \in \text{Dom}^{\{|x|\}}} a_{1,x_1} \cdots a_{\ell,x_\ell}$

Now use Friedgut’s inequality:

$$|Q| = \sum_x a_{1,x_1}^{u_1} \cdots a_{\ell,x_\ell}^{u_\ell} \leq (\sum_{x_1} a_{1,x_1}^{u_1})^{u_1} \cdots (\sum_{x_\ell} a_{\ell,x_\ell}^{u_\ell})^{u_\ell} = m_1^{u_1} \cdots m_\ell^{u_\ell}$$
The AGM Inequality – Proof

Query $Q(x) = R_1(x_1), \ldots, R_\ell(x_\ell)$, fractional cover $u_1, \ldots, u_\ell$

Sizes $|R_1| = m_1, \ldots, |R_\ell| = m_\ell$

Prove $|Q| \leq m_1^{u_1} \cdots m_\ell^{u_\ell}$

Let $\text{Dom} = \text{the domain of all constants in the relations } R_1, \ldots, R_\ell$.

For every $j = 1, \ldots, \ell$, and every tuple $x_j \in \text{Dom} |x_j|$, define:

$$a_{j, x_j} = \begin{cases} 
1 & \text{if the tuple } x_j \text{ belongs to } R_j \\
0 & \text{otherwise}
\end{cases}$$

Then: $m_j = |R_j| = \sum_{x_j \in \text{Dom} |x_j|} a_{j, x_j}$, $|Q| = \sum_{x \in \text{Dom} |x|} a_{1, x_1} \cdots a_{\ell, x_\ell}$

Now use Friedgut’s inequality:

$$|Q| = \sum_x a_{1, x_1}^{u_1} \cdots a_{\ell, x_\ell}^{u_\ell} \leq (\sum_{x_1} a_{1, x_1})^{u_1} \cdots (\sum_{x_\ell} a_{\ell, x_\ell})^{u_\ell} = m_1^{u_1} \cdots m_\ell^{u_\ell}$$

QED
Computing Full Conjunctive Queries

- Recall: all database systems compute one join at a time
- This may be much larger than the maximum output size, $AGM(Q)$.
- Goal: design an algorithm that runs in time $AGM(Q)$.

*Worst-Case-Optimal* algorithm: runs in time $AGM(Q)$. 
Worst-Case Optimal Algorithm

History:

- An algorithm that runs in time $O(n \cdot AGM(Q))$ was given in [AGM’2013].

- First worst-case optimal algorithm that was published: the NPRR algorithm by Ngo, Porat, Ré, Rudra, in PODS’2012. It is complex.

- Earlier algorithm *Leapfrog Trie-join* (LFTJ), by LogicBlox. Veldhuizen proved in ICDT’2014 that LFTJ is also worst case optimal.

- Ngo, Ré, Rudra gave a very simple worst-case algorithm, with a very simple optimality proof, in SIGMOD Records’2013. The algorithm is called *Generic Join*.

Next: we discuss Generic-Join
**Generic Join**

Compute \( Q(x) = R_1(x_1), \ldots, R_\ell(x_\ell) \)

If \(|x| = 1\) then return \( R_1 \cap \cdots \cap R_\ell \).

Otherwise, choose a variable \( x \), occurring in atoms \( R_{i_1}, \ldots, R_{i_k} \)

- Compute \( A = \Pi_x(R_{i_1}) \cap \cdots \cap \Pi_x(R_{i_k}) \)
- For each \( a \in A \), compute \( \text{Result}_a = Q[a/x] \) using \textit{Generic-Join}
- Return \( \bigcup_a \text{Result}_a \).

Runtime: \( O(\text{AGM}(Q)) \) (Recall: we ignore log-factors)

\[ Q(x, y, z) = R(x, y), S(y, z), T(z, x) \]

- Compute \( A = \Pi_x(R) \cap \Pi_x(T) = \{a_1, \ldots, a_n\} \)
- For each \( a_i \in A \), denote \( R'(y) = R(a_i, y), T'(z) = T(z, a_i) \)
  - Compute \( \text{Result}_i(a_i, y, z) = R'(y), S(y, z), T'(z) \)
- Return \( \bigcup_i \text{Result}_i \)

Runtime: \( O(m^{3/2}) \) assuming \( |R| = |S| = |T| = m \).
Generic Join

Compute $Q(x) = R_1(x_1), \ldots, R_\ell(x_\ell)$

If $|x| = 1$ then return $R_1 \cap \cdots \cap R_\ell$.
Otherwise, choose a variable $x$, occurring in atoms $R_{i_1}, \ldots, R_{i_k}$

- Compute $A = \Pi_x(R_{i_1}) \cap \cdots \cap \Pi_x(R_{i_k})$
- For each $a \in A$, compute $\text{Result}_a = Q[a/x]$ using $\text{Generic-Join}$
- Return $\bigcup_a \text{Result}_a$.

Runtime: $O(AGM(Q))$ (Recall: we ignore log-factors)

$Q(x, y, z) = R(x, y), S(y, z), T(z, x)$

- Compute $A = \Pi_x(R) \cap \Pi_x(T) = \{a_1, \ldots, a_n\}$
- For each $a_i \in A$, denote $R'(y) = R(a_i, y), T'(z) = T(z, a_i)$
  Compute $\text{Result}_i(a_i, y, z) = R'(y), S(y, z), T'(z)$
- Return $\bigcup_i \text{Result}_i$

Runtime: $O(m^{3/2})$ assuming $|R| = |S| = |T| = m$. 
**Generic Join**

Compute \( Q(x) = R_1(x_1), \ldots, R_\ell(x_\ell) \)

If \(|x| = 1\) then return \( R_1 \cap \cdots \cap R_\ell \).

Otherwise, choose a variable \( x \), occurring in atoms \( R_{i_1}, \ldots, R_{i_k} \)

- Compute \( A = \Pi_x(R_{i_1}) \cap \cdots \cap \Pi_x(R_{i_k}) \)
- For each \( a \in A \), compute \( \text{Result}_a = Q[a/x] \) using \( \text{Generic-Join} \)
- Return \( \bigcup_a \text{Result}_a \).

Runtime: \( O(AGM(Q)) \) (Recall: we ignore log-factors)

\[ Q(x, y, z) = R(x, y), S(y, z), T(z, x) \]

- Compute \( A = \Pi_x(R) \cap \Pi_x(T) = \{a_1, \ldots, a_n\} \)
- For each \( a_i \in A \), denote \( R'(y) = R(a_i, y), T'(z) = T(z, a_i) \)
  - Compute \( \text{Result}_i(a_i, y, z) = R'(y), S(y, z), T'(z) \)
- Return \( \bigcup_i \text{Result}_i \)

Runtime: \( O(m^{3/2}) \) assuming \( |R| = |S| = |T| = m \).
Discussion: Generic Join v.s. Yannakakis’ Algorithm

[Yannakakis’82] described an algorithm for computing any acyclic query in time $O(|\text{Input}| + |\text{Output}|)$. Basic idea: first perform a semijoin reduction to ensure that all intermediate results are $\leq |\text{Output}|$, then compute the query in standard fashion, one join at a time.

$$Q(x_0, x_1, x_2, x_3, x_4, x_5) =$$
$$R_1(x_0, x_1), R_2(x_1, x_2), R_3(x_2, x_3), R_4(x_3, x_4), R_5(x_4, x_5)$$

$$|R_1| = \ldots = |R_5| = m, \text{ AGM}(Q) = m^3 \text{ (optimal cover: } (1, 0, 1, 0, 1))$$.

There are instances where $Q = \emptyset$, hence Yannakakis’ algorithm takes time $O(m)$, yet Generic-join takes time $\Omega(m^3)$ (Discuss in class).

Newer work on instance-optimal join algorithms [Ngo’2014]
Summary of Lecture 1

- Joins, and conjunctive queries are very important: in SQL, in data analytics, everywhere.

- All traditional query processing algorithms compute one join at a time (except LogicBlox!): suboptimal.

- The AGM bound gives a tight upper bound on the query size, expressed in terms of fractional edge cover.

- The Generic-Join algorithm computes the query in time bounded by the AGM bound: hence worst-case optimal.