DATA516/CSED516
Scalable Data Systems and Algorithms

Lecture 5 Advanced Query Processing

Announcements

- Project proposals were due last week
- HW2 was due yesterday
- HW3 will be posted soon, due on 11/16
- Next paper review due on 11/17

- Next two weeks we have guest lecturers
  - 11/10 Shan Shan Huang on Cloud Databases
  - 11/17 Mingxi Wu on Graph Databases

Today's Lecture

- Brief overview of last week's papers:
  - Snowflake
  - Dremel
- Brief review of query processing
- Begin advanced query processing
  - Today: single server
  - Next week (depending on the guest lecture): distributed processing

Snowflake – Discussion

- ”The Snowflake Elastic Data Warehouse”, Dageville et al., SIGMOD’2016

Snowflake

- It is an SaaS – what is this? Give other examples of types of cloud services…

Snowflake

- It is an SaaS – what is this? Give other examples of types of cloud services…
- SaaS = software as a service
- Other examples:
  - Platform as a service (PaaS):
    e.g. Amazon's EC
  - Infrastructure as a service (virtual machines)
  - Software as a Service
  - Function as a Service: Amazon's Lambda

Snowflake

- Describe Snowflake’s Data Storage

Snowflake

- Describe Snowflake's Data Storage

In class:

- S3:PUT/GET/DELE
- Table → horizontal partition in *files*
- Blobs+PAX
- Temp storage→S3

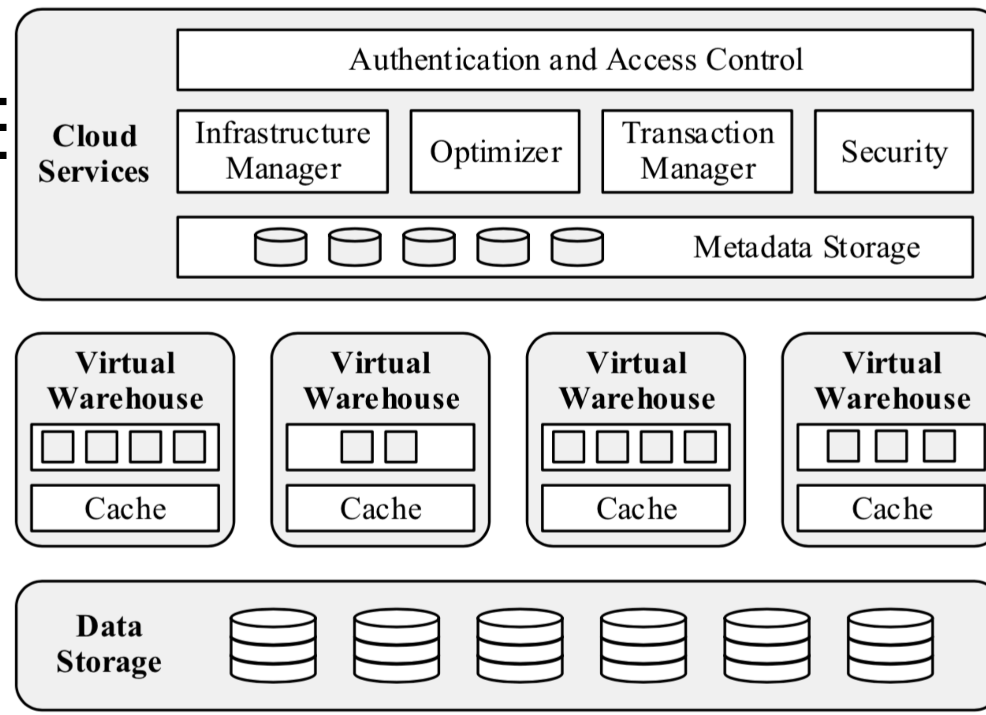


Figure 1: Multi-Cluster, Shared Data Architecture

Snowflake

- Describe Elasticity in Snowflake

- Describe failure handling in Snowflake

Snowflake

- Describe Elasticity in Snowflake
  - Virtual Warehouse (VW) serves one user
  - T-Shirt sizes: X-Small … XX-Large
  - Small query may run on subset of VW
- Describe failure handling in Snowflake

Snowflake

- Describe Elasticity in Snowflake
  - Virtual Warehouse (VW) serves one user
  - T-Shirt sizes: X-Small … XX-Large
  - Small query may run on subset of VW
- Describe failure handling in Snowflake
  - Restart the query
  - No partial retries (like MapReduce or Spark)

Snowflake

- Describe its execution engine

Snowflake

- Describe its execution engine
- Column-oriented (in class)
- Vectorized (“tuple batches” – in class)
- Push-based (in class)

Snowflake

- What does Snowflake use instead of indexes?

Snowflake

- What does Snowflake use instead of indexes?

- “Pruning”: for each file (recall: this is a horizontal partition of a table) and each attribute, it stores the min/max values in that column in that file; may skip files when not needed.

Dremel

- Melnik et al., “Dremel: A Decade of Interactive SQL Analysis at Web Scale”, VLDB’2020
- This is the “ten years best paper award companion paper”; the original Dremel paper was in 2010

Dremel

- Dremel = internal name for Google's BigQuery
- A fully managed, serverless data warehouse for scalable data analytics
- Google's internal history:
  - MapReduce: "databases are bad"
  - BigTable: "tables are OK, but SQL is bad"
  - Dremel: "we love SQL"
  - Spanner: "we love SQL and transactions"

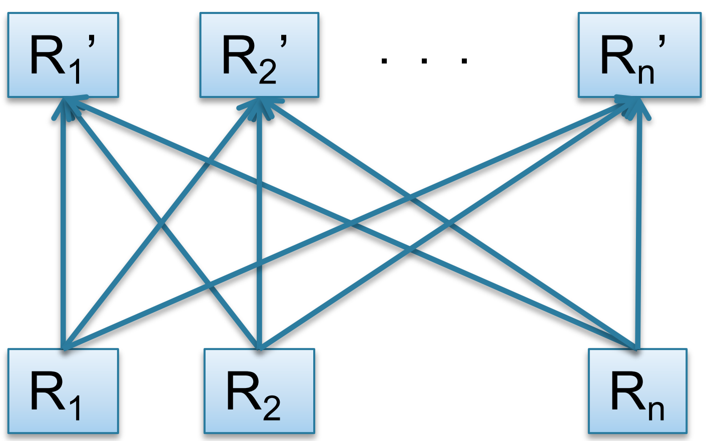
Dremel: Data Storage

- Initially data was stored on local disks
- Migration to Borg meant that the disks were no longer owned by Dremel, but shared among many applications
- Solution: move data to GFS
- But performance degraded: lots work was need to regain performance; but little details given in Sec. 7

Dremel: Disaggregated Shuffle

Standard Shuffle: $O(n^2)$ behavior
E.g. to reshuffle data from 1000 servers, need 10^6 pairwise communications

Disaggregated Shuffle:
add intermediate servers.
E.g. 10 intermediate servers,
hence $2*10*1000 = 2*10^4$ pairwise comm.



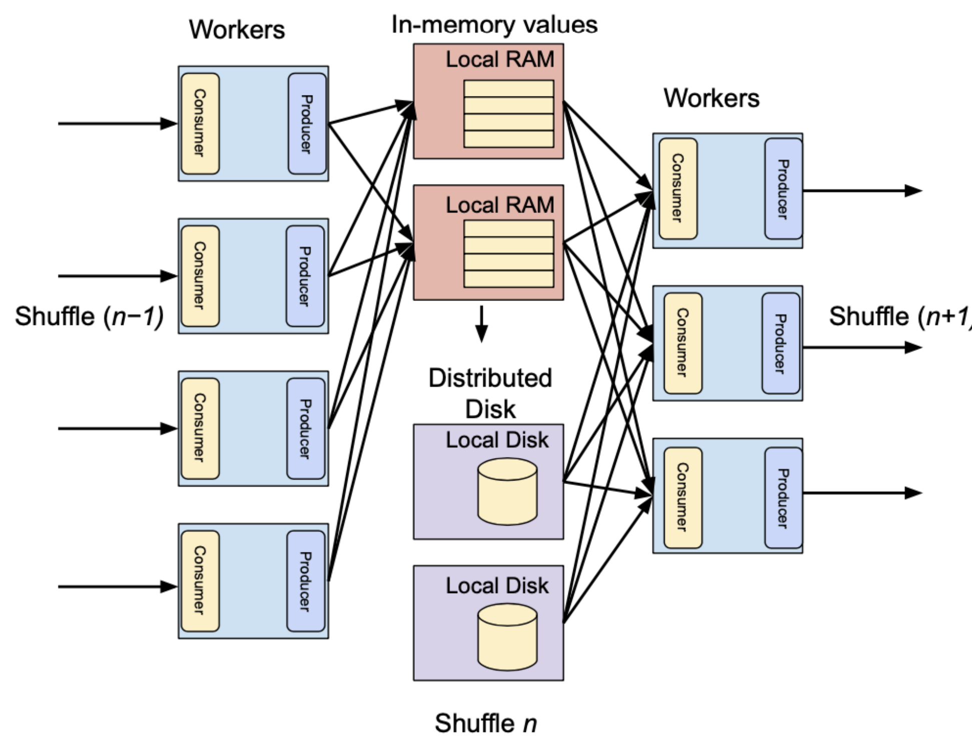


Figure 2: Disaggregated in-memory shuffle

Dremel: Columnar Storage

- We have seen that Snowflake uses PAX-like storage
- Dremel is fully column oriented
- The major novelty is that it stores nested data (Json, or Protobuf) also in column storage
- The original method in 2010 was complicated
- New method is much simpler (next)

Dremel: Columnar Storage

```
DocId: 10                    r1
Name
  Language
    Code: 'en-us'
    Country: 'us'
  Language
    Code: 'en'
  Url: 'http://A'
Name
  Url: 'http://B'
Name
  Language
    Code: 'en-gb'
    Country: 'gb'
```

```
message Document {
  required int64 DocId;
  repeated group Name {
    repeated group Language {
      required string Code;
      optional string Country;
    }
    optional string Url;
  }
}
```

```
DocId: 20                    r2
Name
  Url: 'http://C'
```

Figure 5: Two sample nested records and their schema (based on Figure 2 in [32])

Need to have the schema!

DocId

value	p
10	true
20	true

Name

len
3
1

Name.Url

value	p
http://A	true
http://B	true
	false
http://C	true

Name.Language

len
2
0
1
0

Name.Language.Code

value	p
en-us	true
en	true
en-gb	true

Name.Language.Country

value	p
us	true
	false
gb	true

Figure 7: Columnar representation of the data in Figure 5 showing length (len) and presence (p)

Need to decode all ancestors!

Discussion

```
DocId: 10                r1
Name
  Language
    Code: 'en-us'
    Country: 'us'
  Language
    Code: 'en'
  Url: 'http://A'
Name
  Url: 'http://B'
Name
  Language
    Code: 'en-gb'
    Country: 'gb'
```

```
message Document {
  required int64 DocId;
  repeated group Name {
    repeated group Language {
      required string Code;
      optional string Country;
    }
    optional string Url;
  }
}
```

```
DocId: 20                r2
Name
  Url: 'http://C'
```

Figure 5: Two sample nested records and their schema (based on Figure 2 in [32])

Alternative: normalize the data!

DocID

DocID
10
20

Name

NameID	DocIDRef	URL
1	10	http://A
2	10	http://B
3	10	NULL
4	20	http://C

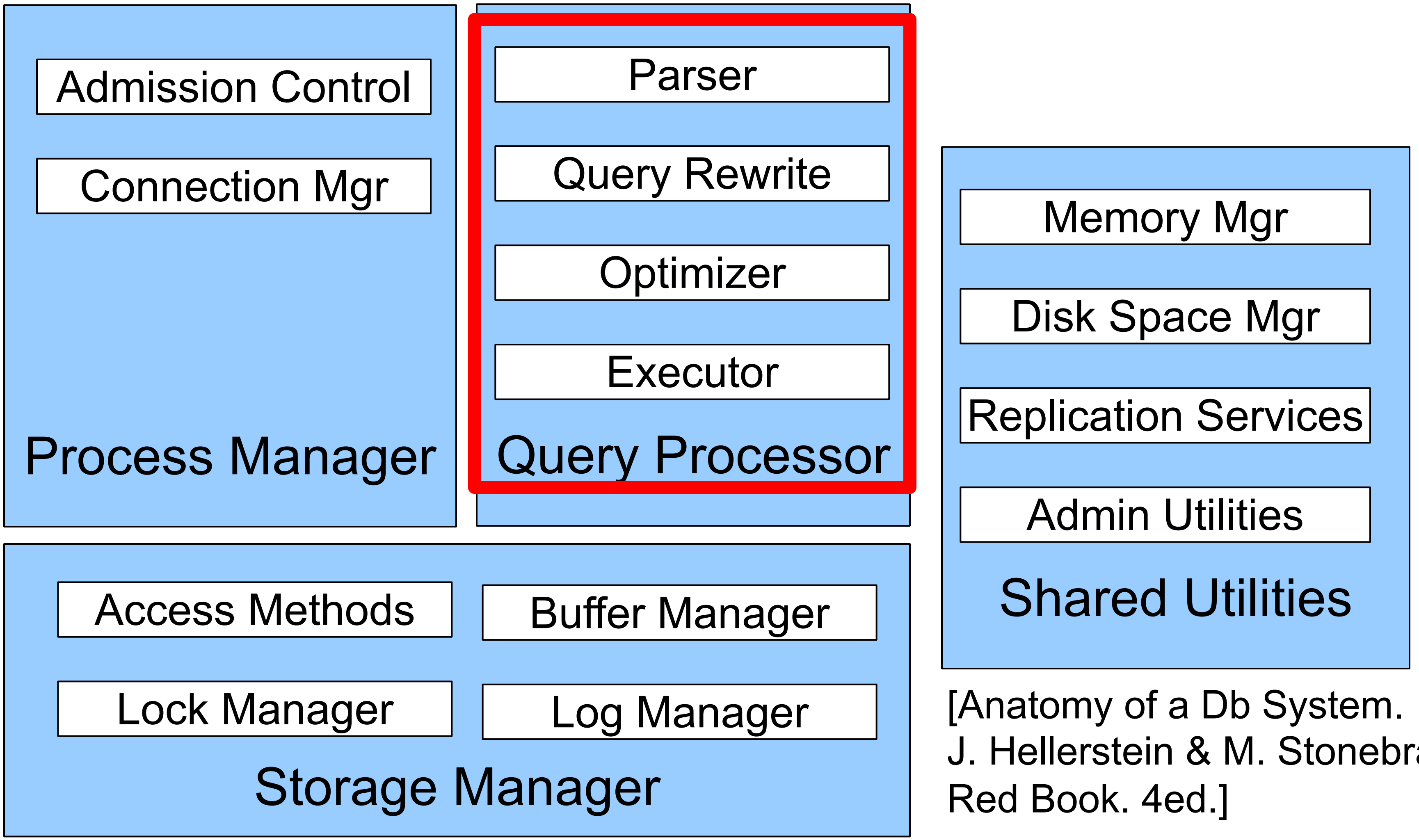
Language

NameIDRef	Code	Country
1	en-us	us
1	en	NULL
3	en-gb	gb

Snowflake maps Json automatically to relations.
How? (I don't know, but you can try to figure it out!)

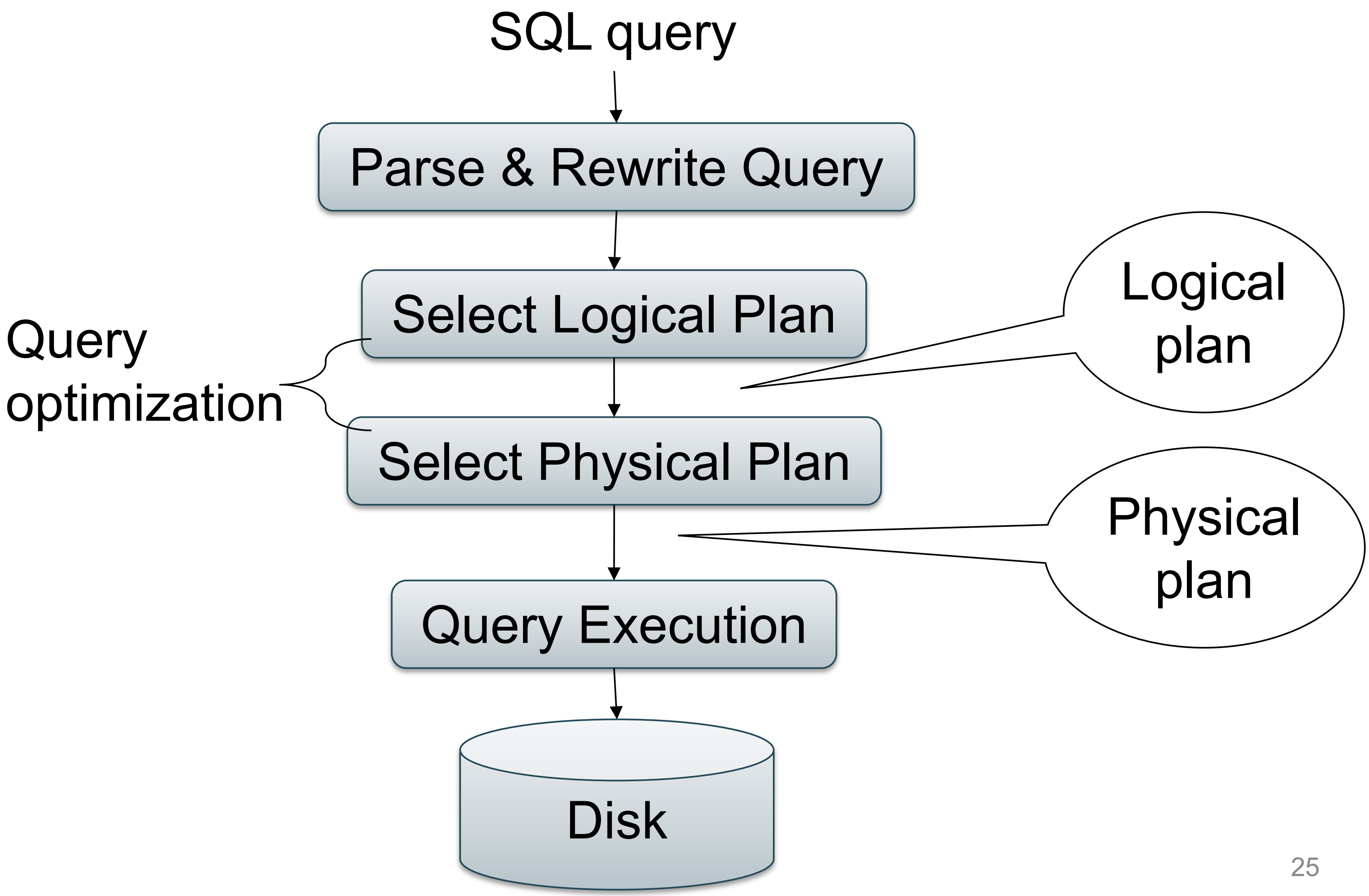
Review of Query Processing

DBMS Architecture



[Anatomy of a Db System.
J. Hellerstein & M. Stonebraker.
Red Book. 4ed.]

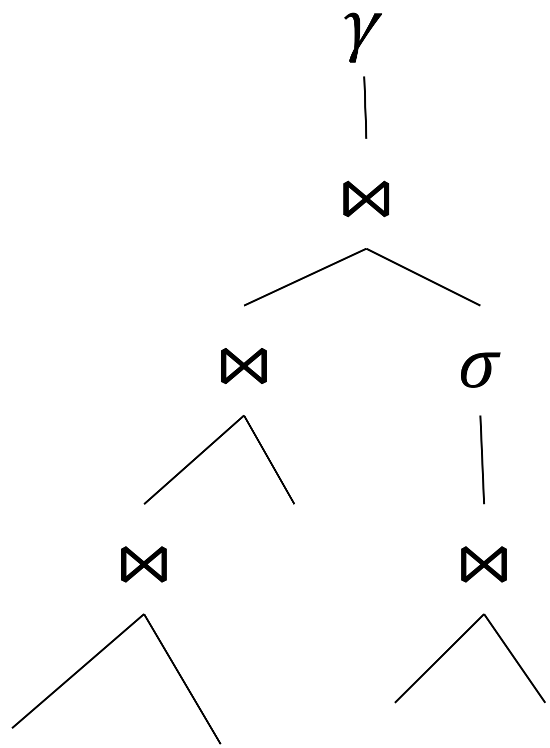
Lifecycle of a Query



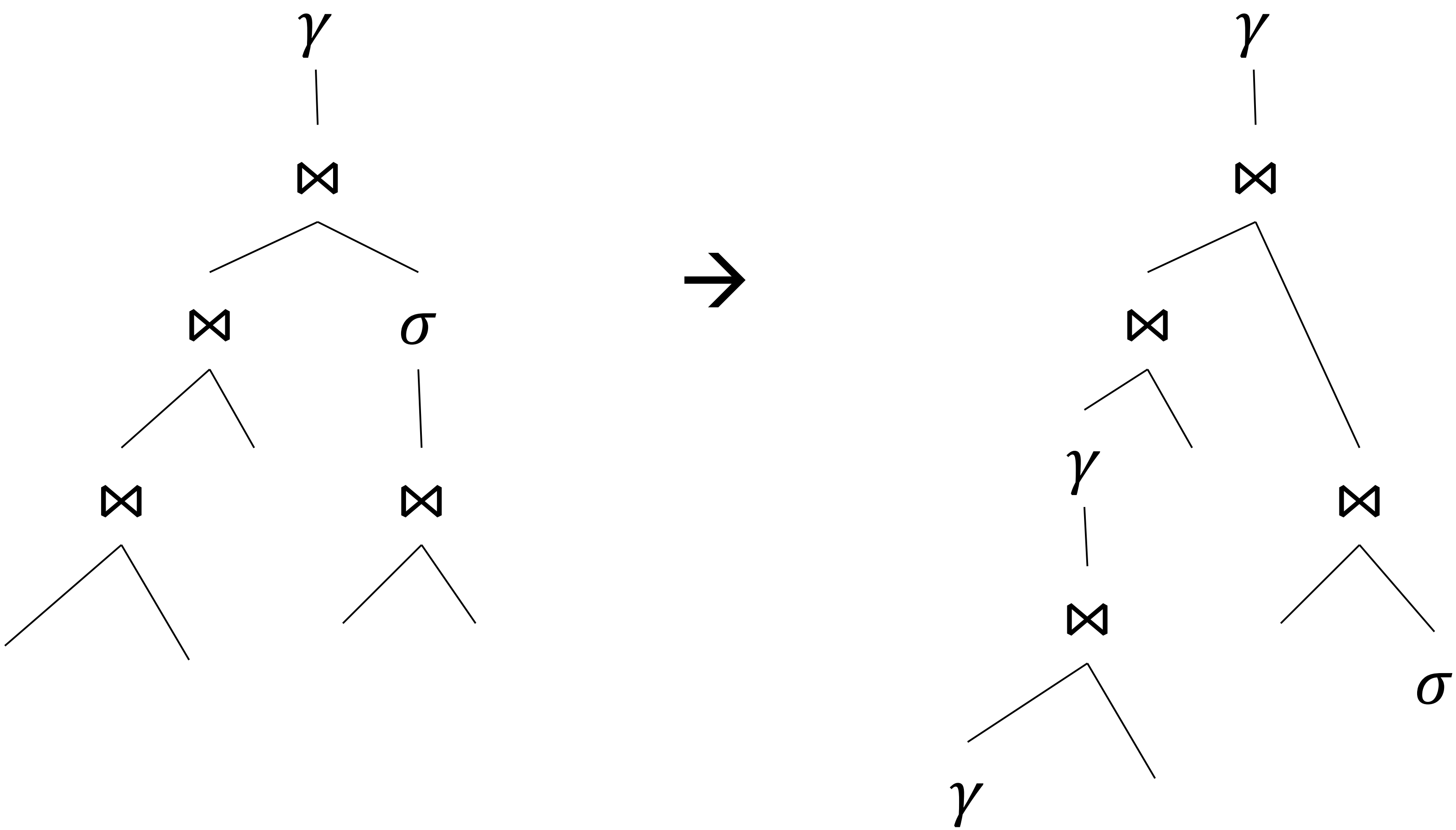
SQL → Query Plan

SELECT …
FROM …
WHERE …
GROUP BY …
HAVING …

→



Optimizer: Plan → Plan



Quick Recap

- Name 3 join processing algorithms
- Speedup, scaleup
- Shared-memory v.s. shared-nothing
- Parallel hash join algorithm
- Discuss skew

Cost Estimater → Large Errors ☹

[Neumann et al., VLDB'2015]

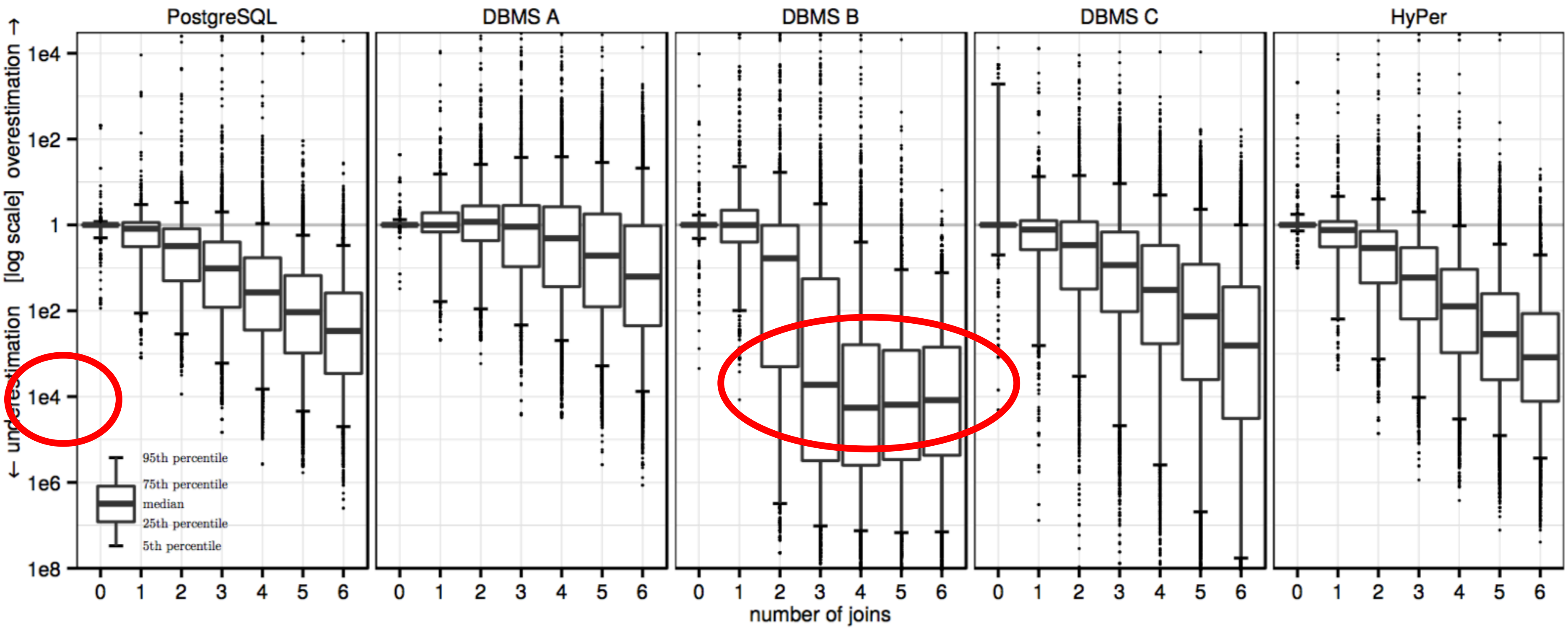


Figure 3: Quality of cardinality estimates for multi-join queries in comparison with the true cardinalities. Each boxplot summarizes the error distribution of all subexpressions with a particular size (over all queries in the workload)

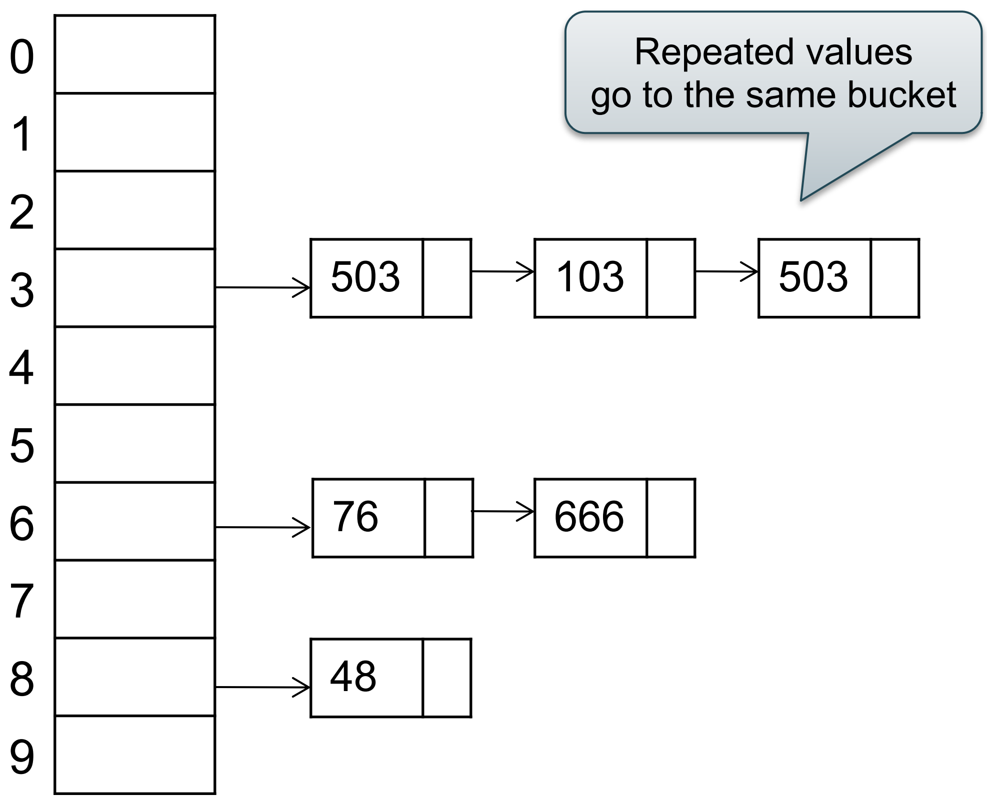
Query Execution

- Nested loop join: $O(N^2)$
- Merge join: $O(N*\log(N))$
- Hash join: $O(N)$ (maybe)

Distributed Query Processing

- Optimizer is the same
- Cost estimator is the same, just harder (why?)
- New operator: shuffle
  - Hash partion → skew is an issue
  - Broadcast → skew no problem, but size is

Skew: Affects any Hash Table



Rule of thumbs:

1. Ideal size of a bucket = N/p
2. When a data value occurs many more times than N/p, we call it "heavy hitter", and there *will be* skew
3. If all data values occur much fewer times than N/p, then there is no skew
4. Last statement is probabilistic; exact calculation uses Cernoff bounds, and is complex

Limitations of Query Processing

- Biggest limitation is the quality of the cardinality estimator
- Another limitation: if the query is *cyclic*, then *any* plan is bad
- Today's topic: advanced techniques
  - Handle cyclic queries well
  - Are more robust to cardinality estimators.

Motivation

- When the joins are "cyclic" then standard query plan are severely suboptimal
- Where we find cyclic queries:
  - Rare events: find actor X who acted in movie Y who was directed by director Z who was married to X
  - Social network analysis: count the number of triangles; the number of 5-cliques, etc
  - "Motif analysis": finding occurrences of certain small subgraphs ("motifs")

A query plans produces large intermediate results; answer is small

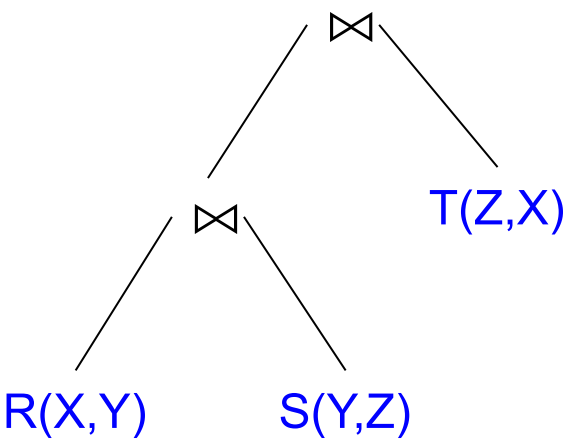
[Ngo'2013]

Example

$Q(X,Y,Z) = R(X,Y) \wedge S(Y,Z) \wedge T(Z,X)$

```
select *                                    -- natural join
from R, S, T
where R.Y = S.Y and S.Z = T.Z and T.X = R.X
```

Query plan

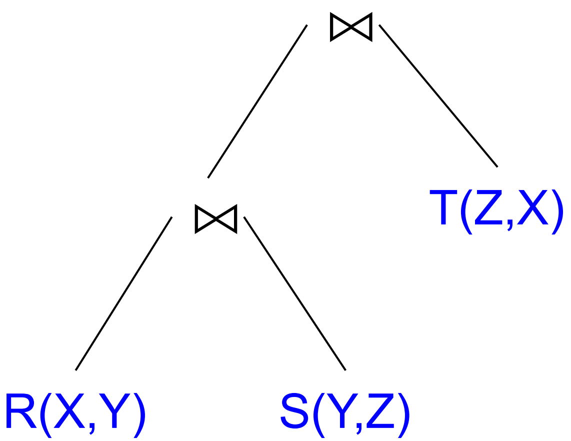


Example

Q(X,Y,Z) = R(X,Y) ∧ S(Y,Z) ∧ T(Z,X)

```
select *                                   -- natural join
from R, S, T
where R.Y = S.Y and S.Z = T.Z and T.X = R.X
```

Query plan



R:

N

X	Y
0	1
0	2
0	3
...	...
0	N/2
1	0
2	0
...	...
N/2	0

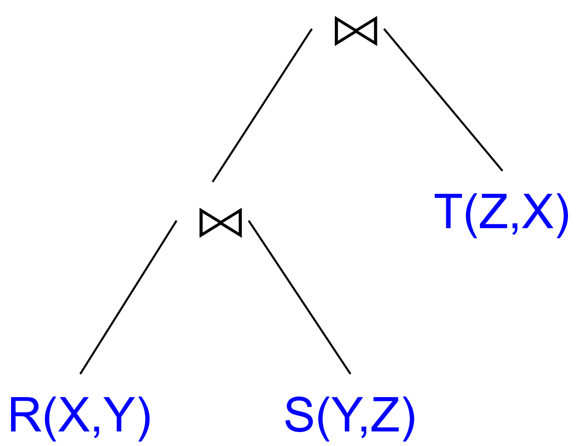
[Ngo'2013]

Example

Q(X,Y,Z) = R(X,Y) ∧ S(Y,Z) ∧ T(Z,X)

```
select *                               -- natural join
from R, S, T
where R.Y = S.Y and S.Z = T.Z and T.X = R.X
```

Query plan



N

R:

X	Y
0	1
0	2
0	3
...	...
0	N/2
1	0
2	0
...	...
N/2	0

S: (same as R)

Y	Z
0	1
0	2
0	3
...	...
0	N/2
1	0
2	0
...	...
N/2	0

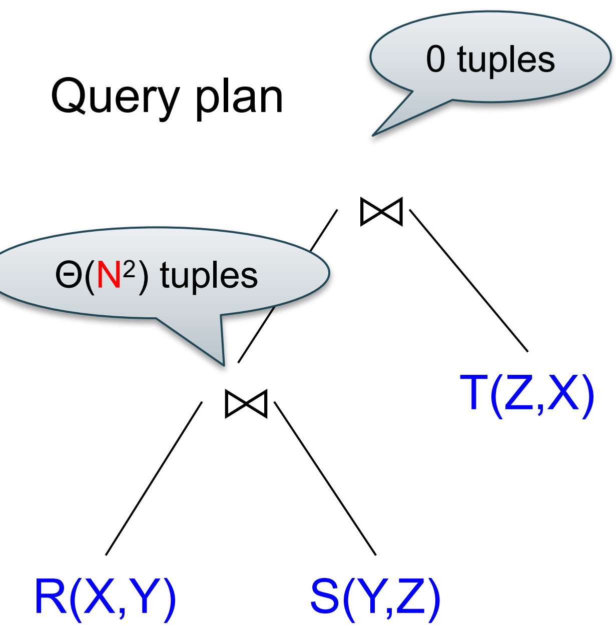
T: (same as R)

Z	X
0	1
0	2
0	3
...	...
0	N/2
1	0
2	0
...	...
N/2	0

Example

Q(X,Y,Z) = R(X,Y) ∧ S(Y,Z) ∧ T(Z,X)

```
select *                                 -- natural join
from R, S, T
where R.Y = S.Y and S.Z = T.Z and T.X = R.X
```



N

R:

X	Y
0	1
0	2
0	3
...	...
0	N/2
1	0
2	0
...	...
N/2	0

S: (same as R)

Y	Z
0	1
0	2
0	3
...	...
0	N/2
1	0
2	0
...	...
N/2	0

T: (same as R)

Z	X
0	1
0	2
0	3
...	...
0	N/2
1	0
2	0
...	...
N/2	0

Optimal Algorithm

To define “optimal” we need to answer two questions:

Q1: How large is the output of a query?

Q2: How can we compute it in time no larger than the largest output?

Full Conjunctive Query

- Full CQ = all variables occur in the head

$$Q(X_1, \ldots, X_k) \ = \ R_1(Vars_1) \wedge \cdots \wedge R_m(Vars_m)$$

- Equivalent SQL:

```
select   *
from     R1, R2, ...
where    [join and selection conditions]
```

Worst-Case Optimality

Fix input statistics for D

- |R|, |S| ≤ N
- How large are the answers to these queries?

$Q(X,Y,Z) = R(X,Y) \wedge S(Y,Z)$

No other info: $|Q(D)| \leq N^2$

Worst-Case Optimality

Fix input statistics for D

- |R|, |S| ≤ N
- How large are the answers to these queries?

$Q(X, Y, Z) = R(X, Y) \wedge S(Y, Z)$

No other info: $|Q(D)| \leq N^2$

- S.Y is a key: $|Q(D)| \leq N$

Worst-Case Optimality

Fix input statistics for D

- |R|, |S| ≤ N
- How large are the answers to these queries?

$Q(X, Y, Z) = R(X, Y) \wedge S(Y, Z)$

No other info: $|Q(D)| \leq N^2$

- S.Y is a key: $|Q(D)| \leq N$
- S.Y has degree ≤ d: $|Q(D)| \leq d \times N$

Worst-Case Optimality

Fix input statistics for D

- |R|, |S| ≤ N
- How large are the answers to these queries?

$Q(X,Y,Z) = R(X,Y) \wedge S(Y,Z)$

No other info: $|Q(D)| \leq N^2$

- S.Y is a key: $|Q(D)| \leq N$
- S.Y has degree ≤ d: $|Q(D)| \leq d \times N$

$Q(X,Y,Z) = R(X,Y) \wedge S(Y,Z) \wedge T(Z,X)$

No other info:

Worst-Case Optimality

Fix input statistics for D

- |R|, |S| ≤ N
- How large are the answers to these queries?

$Q(X,Y,Z) = R(X,Y) \wedge S(Y,Z)$

No other info: $|Q(D)| \leq N^2$

- S.Y is a key: $|Q(D)| \leq N$
- S.Y has degree ≤ d: $|Q(D)| \leq d \times N$

$Q(X,Y,Z) = R(X,Y) \wedge S(Y,Z) \wedge T(Z,X)$

No other info: $|Q(D)| \leq N^{3/2}$

Worst-Case Optimality

Fix input statistics for D

- |R|, |S| ≤ N
- How large are the answers to these queries?

$Q(X,Y,Z) = R(X,Y) \wedge S(Y,Z)$

No other info: $|Q(D)| \leq N^2$

- S.Y is a key: $|Q(D)| \leq N$
- S.Y has degree ≤ d: $|Q(D)| \leq d \times N$

$Q(X,Y,Z) = R(X,Y) \wedge S(Y,Z) \wedge T(Z,X)$

No other info: $|Q(D)| \leq N^{3/2}$

WOW!

The Two Questions

Q1: Given statistics, what is $\max(|Q(D)|)$?

Q2: How can we compute Q in time $O(\max(|Q(D)|))$?

Simple Fact #1

- Consider any query:

$$Q(X_1, \dots, X_k) = R_1(Vars_1) \land \cdots \land R_m(Vars_m)$$

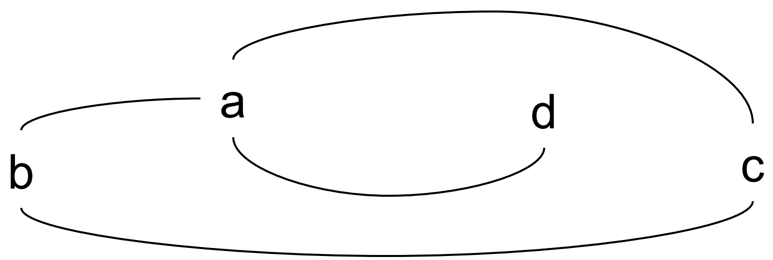
- Its output size is no larger than the product of all cardinalities:

$$|Q| \leq |R_1| \times \cdots \times |R_m|$$

Why?

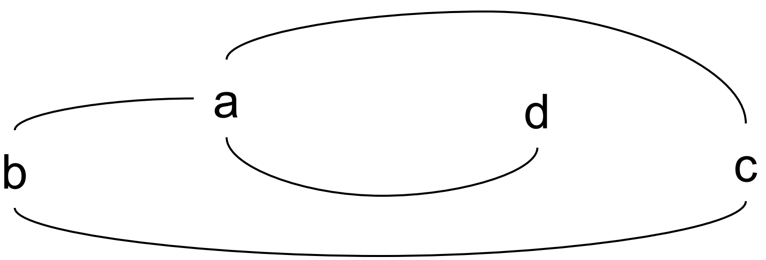
Graphs and Hypergraphs

- An undirected graph G = (V, E) where each edge e ∈ E is a set of two nodes

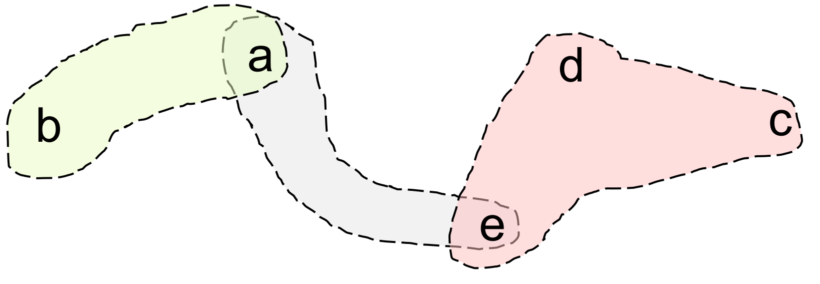


Graphs and Hypergraphs

- An undirected graph G = (V, E) where each edge e ∈ E is a set of two nodes

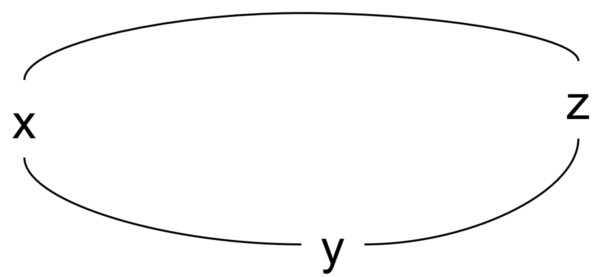


- A hypergraph is G = (V, E) where each edge is some set (of 1 or 2 or >2 nodes)

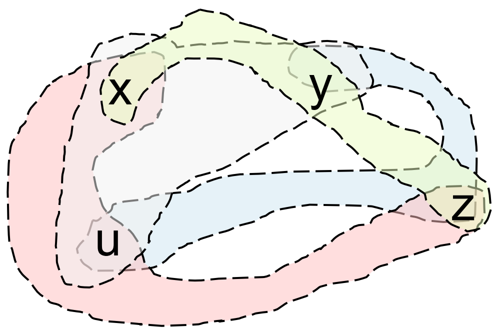


Conjunctive Queries are Hypergraphs

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

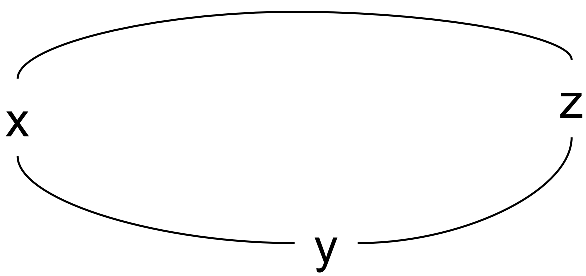


$Q(x, y, z) = A(x, y, z) \wedge B(x, y, u) \wedge C(x, z, u) \wedge D(y, z, u)$



Edge Cover

- An *edge cover* of a (hyper)graph is a subset of edges that contain all the vertices

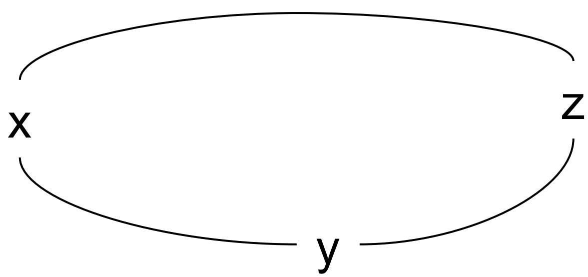


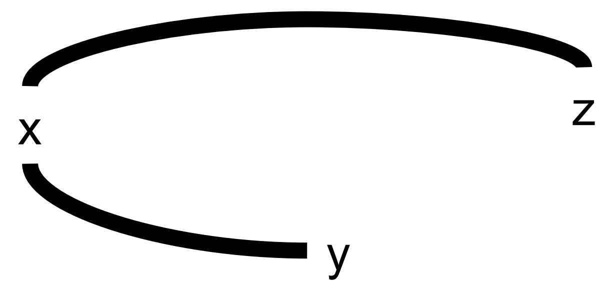
Edge Cover

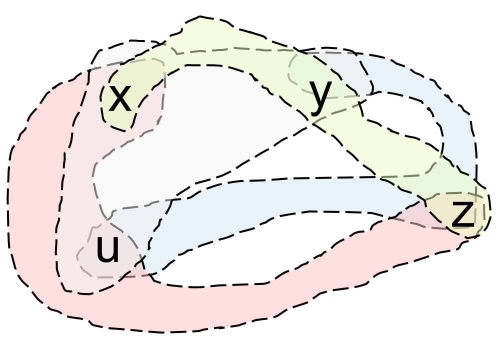
- An *edge cover* of a (hyper)graph is a subset of edges that contain all the vertices

Edge Cover

- An *edge cover* of a (hyper)graph is a subset of edges that contain all the vertices







Edge Cover

- An *edge cover* of a (hyper)graph is a subset of edges that contain all the vertices

Simple Fact #2

- Consider any query:

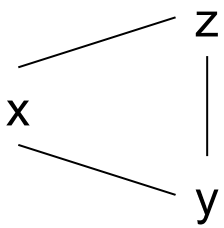
$$Q(X_1, \ldots, X_k) \;=\; R_1(Vars_1) \wedge \cdots \wedge R_m(Vars_m)$$

- Let $R_{i_1}, R_{i_2}, \ldots, R_{i_n}$ be an edge cover. Then the output size is no larger than their product:

$$|Q| \;\leq\; |R_{i_1}| \times \cdots \times |R_{i_n}|$$

Why?

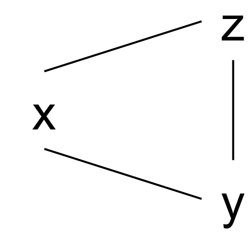
Examples



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

- Edge covers:
  $R(x, y) \wedge S(y, z)$

Examples

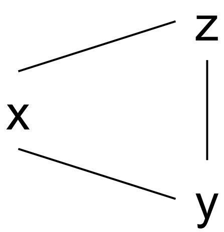


$Q(x,y,z) = R(x,y) \wedge S(y,z) \wedge T(z,x)$

- Edge covers:
  $R(x,y) \wedge S(y,z)$

$|Q| \leq |R| \times |S|$

Examples

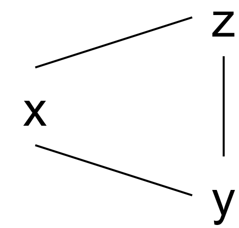


$Q(x,y,z) \ = \ R(x,y) \wedge S(y,z) \wedge T(z,x)$

- Edge covers:
  $R(x,y) \wedge S(y,z)$ or $R(x,y) \wedge T(z,x)$

$|Q| \leq |R| \times |S|$

Examples



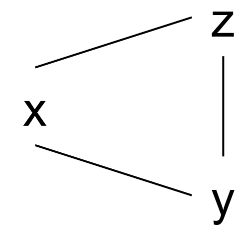
$$Q(x, y, z) = R(x, y) \wedge S(y, z) \wedge T(z, x)$$

- Edge covers:
  $R(x, y) \wedge S(y, z)$ or $R(x, y) \wedge T(z, x)$

$|Q| \leq |R| \times |S|$

$|Q| \leq |R| \times |T|$

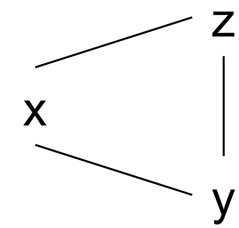
Examples



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

- Edge covers:
  $R(x, y) \wedge S(y, z)$ or $R(x, y) \wedge T(z, x)$ or $S(y, z) \wedge T(z, x)$

Examples

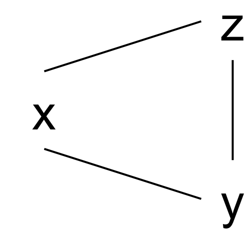


$Q(x, y, z) \; = \; R(x, y) \wedge S(y, z) \wedge T(z, x)$

- Edge covers:
  $R(x, y) \wedge S(y, z)$ or $R(x, y) \wedge T(z, x)$ or $S(y, z) \wedge T(z, x)$

$$|Q| \; \leq \; \min(|R| \times |S|, |R| \times |T|, |S| \times |T|)$$

Examples



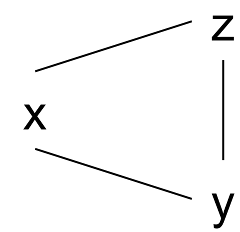
$Q(x, y, z) \ = \ R(x, y) \wedge S(y, z) \wedge T(z, x)$

- Edge covers:
  $R(x, y) \wedge S(y, z)$ or $R(x, y) \wedge T(z, x)$ or $S(y, z) \wedge T(z, x)$

$$|Q| \ \leq \ \min(|R| \times |S|, |R| \times |T|, |S| \times |T|)$$

$Q(x, y, z, u) \ = \ A(x, y, z) \wedge B(x, y, u) \wedge C(x, z, u) \wedge D(y, z, u)$

Examples



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

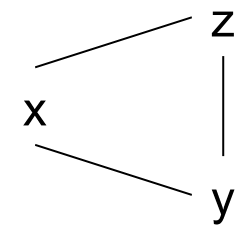
- Edge covers:
  $R(x, y) \wedge S(y, z)$ or $R(x, y) \wedge T(z, x)$ or $S(y, z) \wedge T(z, x)$

$$|Q| \ \leq \ \min(|R| \times |S|, |R| \times |T|, |S| \times |T|)$$

$Q(x, y, z, u) \ = \ A(x, y, z) \wedge B(x, y, u) \wedge C(x, z, u) \wedge D(y, z, u)$

- Edge covers:
  $A(x, y, z) \wedge B(x, y, u)$ or $A(x, y, z) \wedge C(x, z, u)$ or …

Examples



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

- Edge covers:
  $R(x,y) \wedge S(y,z)$ or $R(x,y) \wedge T(z,x)$ or $S(y,z) \wedge T(z,x)$

$$|Q| \;\le\; \min(|R|\times|S|, |R|\times|T|, |S|\times|T|)$$

$Q(x,y,z,u) \; = \; A(x,y,z) \wedge B(x,y,u) \wedge C(x,z,u) \wedge D(y,z,u)$

- Edge covers:
  $A(x,y,z) \wedge B(x,y,u)$ or $A(x,y,z) \wedge C(x,z,u)$ or …

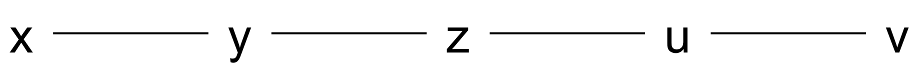
$$|Q| \;\le\; \min(|A|\times|B|, |A|\times|C|, \dots)$$

More examples

Assume all relations have size N.

What are the maximum output sizes?

- $Q(x,y,z,u,v) = R(x,y) \wedge S(y,z) \wedge T(z,u) \wedge K(u,v)$



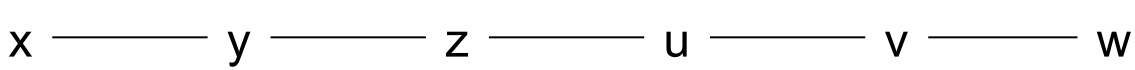
More examples

Assume all relations have size N.

What are the maximum output sizes?

- $Q(x, y, z, u, v) = R(x, y) \wedge S(y, z) \wedge T(z, u) \wedge K(u, v)$

- $Q(x, y, z, u, v, w) =$
  $R(x, y) \wedge S(y, z) \wedge T(z, u) \wedge K(u, v) \wedge L(v, w)$



More examples

Assume all relations have size N.

What are the maximum output sizes?

- $Q(x,y,z,u,v) = R(x,y) \wedge S(y,z) \wedge T(z,u) \wedge K(u,v)$

x —— y —— z —— u —— v

Answer: N^3
in both cases

- $Q(x,y,z,u,v,w) =$
  $R(x,y) \wedge S(y,z) \wedge T(z,u) \wedge K(u,v) \wedge L(v,w)$

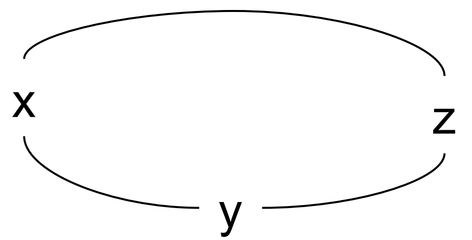
x —— y —— z —— u —— v —— w

Fractional Edge Cover

- A <u>*fractional edge cover*</u> of a (hyper)graph is a set of non-negative numbers w_e, one for each edge e, such that, for every vertex v: $\sum_{e:v \in e} w_e \geq 1$

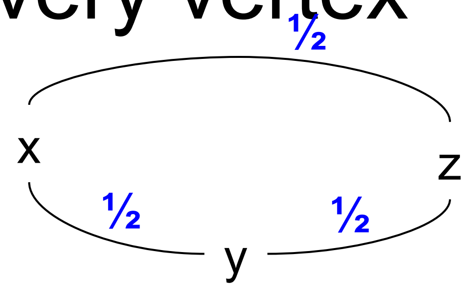
Fractional Edge Cover

- A *fractional edge cover* of a (hyper)graph is a set of non-negative numbers w_e, one for each edge e, such that, for every vertex v: $\sum_{e:v\in e} w_e \geq 1$



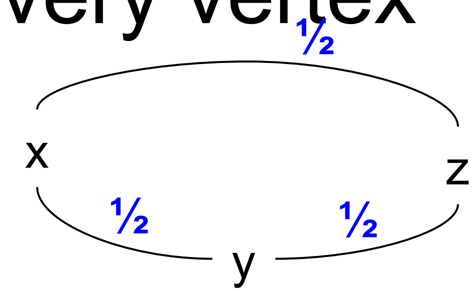
Fractional Edge Cover

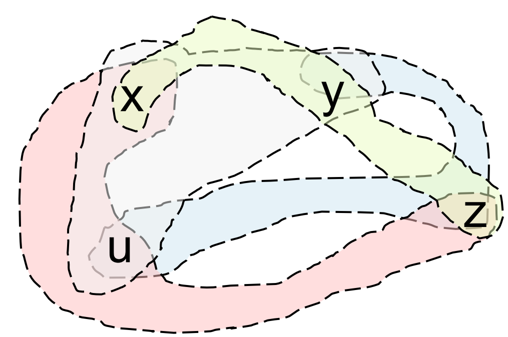
- A *fractional edge cover* of a (hyper)graph is a set of non-negative numbers w_e, one for each edge e, such that, for every vertex v: $\sum_{e:v\in e} w_e \geq 1$



Fractional Edge Cover

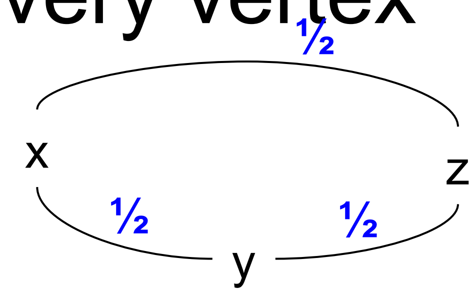
- A *fractional edge cover* of a (hyper)graph is a set of non-negative numbers w_e, one for each edge e, such that, for every vertex v: $\sum_{e:v\in e} w_e \geq 1$

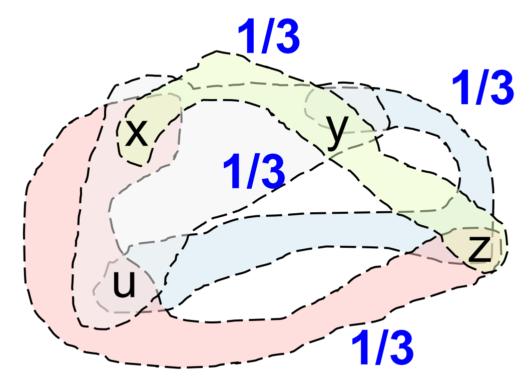




Fractional Edge Cover

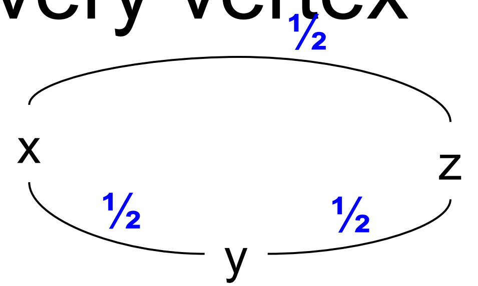
- A *fractional edge cover* of a (hyper)graph is a set of non-negative numbers w_e, one for each edge e, such that, for every vertex v: $\sum_{e:v \in e} w_e \geq 1$



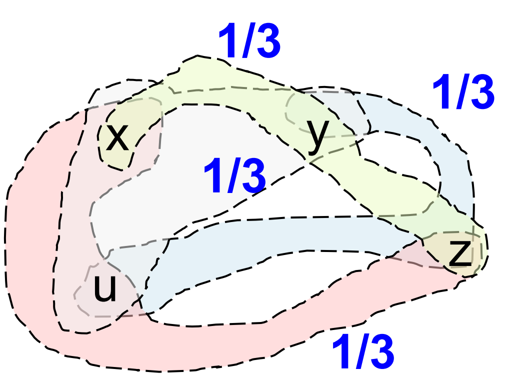


Fractional Edge Cover

- A *fractional edge cover* of a (hyper)graph is a set of non-negative numbers w_e, one for each edge e, such that, for every vertex v: $\sum_{e:v\in e} w_e \geq 1$



- **Fact**: every edge cover is also a fractional edge cover. Why?



Not so Simple Fact #3

- Consider any query:

$$Q(X_1, \ldots, X_k) = R_1(Vars_1) \wedge \cdots \wedge R_m(Vars_m)$$

- Let $w_1, w_2, \ldots, w_m$ be a fractional edge cover. Then the output size is no larger than:

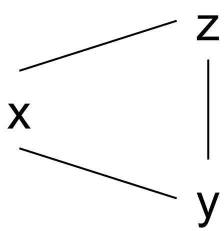
$$|Q| \leq |R_1|^{w_1} \times \cdots \times |R_m|^{w_m}$$

Examples

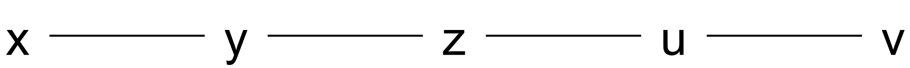
Assume all relations have size N

What are the maximum sizes?

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



- $Q(x, y, z, u, v) = R(x, y) \land S(y, z) \land T(z, u) \land K(u, v)$

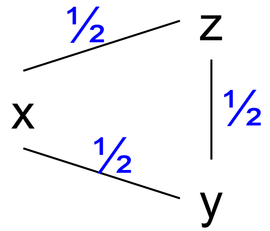


Examples

Assume all relations have size N

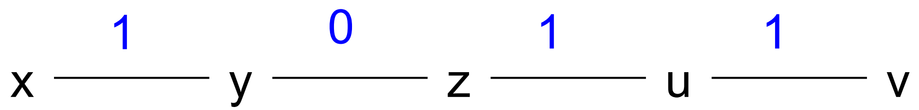
What are the maximum sizes?

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



Answer: $|Q| \leq N^{3/2}$

- $Q(x, y, z, u, v) = R(x, y) \land S(y, z) \land T(z, u) \land K(u, v)$



Answer: $|Q| \leq N^{3}$

Discussion

- When all relations have the same cardinality N:
  - Choose the fractional edge cover $w_1, w_2, \dots, w_m$ with the smallest sum.
  - This sum is denoted ρ^*, hence $|Q| \leq N^{\rho^*}$

- When relations have different cardinalities, try all edge covers, use minimum bound

Examples

Query	$w_1, w_2, \ldots, w_m$	$\lvert R_1\rvert^{w1} \times \cdots \times \lvert R_m\rvert^{wm}$	$\lvert Q\rvert \leq \ldots$
$R(x,y) \wedge S(y,z)$	1,1	$\lvert R\rvert \times \lvert S\rvert$	$\leq \lvert R\rvert \times \lvert S\rvert$

Examples

Query	$w_1, w_2, \ldots, w_m$	$\lvert R_1\rvert^{w1 \times \cdots} \times \lvert R_m\rvert^{wm}$	$\lvert Q\rvert \leq \ldots$
$R(x,y) \wedge S(y,z)$	1,1	$\lvert R\rvert \times \lvert S\rvert$	$\leq \lvert R\rvert \times \lvert S\rvert$
$R(x,y) \wedge S(y,z) \wedge T(z,x)$	½, ½, ½		
	1,1,0		
	1,0,1		
	0,1,1		

Examples

Query	$w_1, w_2, \dots, w_m$	$\lvert R_1\rvert^{w1} \times \cdots \times \lvert R_m\rvert^{wm}$	$\lvert Q\rvert \leq \dots$
$R(x,y) \wedge S(y,z)$	1,1	$\lvert R\rvert \times \lvert S\rvert$	$\leq \lvert R\rvert \times \lvert S\rvert$
$R(x,y) \wedge S(y,z) \wedge T(z,x)$	½, ½, ½	$(\lvert R\rvert \times \lvert S\rvert \times \lvert T\rvert)^{½}$	
	1,1,0	|R|×|S|	
	1,0,1	|R|×|T|	
	0,1,1	|S|×|T|	

Examples

Query	$w_1, w_2, \ldots, w_m$	$\lvert R_1\rvert^{w1} \times \cdots \times \lvert R_m\rvert^{wm}$	$\lvert Q\rvert \le \ldots$
$R(x,y) \wedge S(y,z)$	1,1	$\lvert R\rvert \times \lvert S\rvert$	$\le \lvert R\rvert \times \lvert S\rvert$
$R(x,y) \wedge S(y,z) \wedge T(z,x)$	½, ½, ½	$(\lvert R\rvert \times \lvert S\rvert \times \lvert T\rvert)^{½}$	$\le \min($ $(\lvert R\rvert \times \lvert S\rvert \times \lvert T\rvert)^{½}$, $\lvert R\rvert \times \lvert S\rvert$, $\lvert R\rvert \times \lvert T\rvert$, $\lvert S\rvert \times \lvert T\rvert)$
	1,1,0	$\lvert R\rvert \times \lvert S\rvert$	
	1,0,1	$\lvert R\rvert \times \lvert T\rvert$	
	0,1,1	$\lvert S\rvert \times \lvert T\rvert$	

Examples

Query	$w_1, w_2, \ldots, w_m$	$\lvert R_1\rvert^{w1} \times \cdots \times \lvert R_m\rvert^{wm}$	$\lvert Q\rvert \le \ldots$
$R(x,y) \wedge S(y,z)$	1,1	$\lvert R\rvert \times \lvert S\rvert$	$\le \lvert R\rvert \times \lvert S\rvert$
$R(x,y) \wedge S(y,z) \wedge T(z,x)$	½, ½, ½	$(\lvert R\rvert \times \lvert S\rvert \times \lvert T\rvert)^{½}$	$\le \min($ $(\lvert R\rvert \times \lvert S\rvert \times \lvert T\rvert)^{½}$, $\lvert R\rvert \times \lvert S\rvert$, $\lvert R\rvert \times \lvert T\rvert$, $\lvert S\rvert \times \lvert T\rvert)$
	1,1,0	$\lvert R\rvert \times \lvert S\rvert$	
	1,0,1	$\lvert R\rvert \times \lvert T\rvert$	
	0,1,1	$\lvert S\rvert \times \lvert T\rvert$	
$A(x,y,z) \wedge B(x,y,u)$ $\wedge C(x,z,u) \wedge D(y,z,u)$			

Examples

Query	$w_1, w_2, \ldots, w_m$	$\lvert R_1\rvert^{w1 \times \cdots} \times \lvert R_m\rvert^{wm}$	$\lvert Q\rvert \leq \ldots$
$R(x,y) \wedge S(y,z)$	1,1	$\lvert R\rvert \times \lvert S\rvert$	$\leq \lvert R\rvert \times \lvert S\rvert$
$R(x,y) \wedge S(y,z) \wedge T(z,x)$	½, ½, ½	$(\lvert R\rvert \times \lvert S\rvert \times \lvert T\rvert)^{½}$	$\leq \min($ $(\lvert R\rvert \times \lvert S\rvert \times \lvert T\rvert)^{½}$, $\lvert R\rvert \times \lvert S\rvert$, $\lvert R\rvert \times \lvert T\rvert$, $\lvert S\rvert \times \lvert T\rvert)$
	1,1,0	|R|×|S|	
	1,0,1	|R|×|T|	
	0,1,1	|S|×|T|	
$A(x,y,z) \wedge B(x,y,u) \wedge C(x,z,u) \wedge D(y,z,u)$	1/3, 1/3, 1/3, 1/3		
	1,1,0,0		
	1,0,1,0		
	…		

Examples

Query	$w_1, w_2, \ldots, w_m$	$\lvert R_1\rvert^{w1} \times \cdots \times \lvert R_m\rvert^{wm}$	$\lvert Q\rvert \leq$ ...
$R(x,y) \wedge S(y,z)$	1,1	$\lvert R\rvert \times \lvert S\rvert$	$\leq \lvert R\rvert \times \lvert S\rvert$
$R(x,y) \wedge S(y,z) \wedge T(z,x)$	½, ½, ½	$(\lvert R\rvert \times \lvert S\rvert \times \lvert T\rvert)^{½}$	$\leq \min($ $(\lvert R\rvert \times \lvert S\rvert \times \lvert T\rvert)^{½}$, $\lvert R\rvert \times \lvert S\rvert$, $\lvert R\rvert \times \lvert T\rvert$, $\lvert S\rvert \times \lvert T\rvert)$
	1,1,0	|R|×|S|	
	1,0,1	|R|×|T|	
	0,1,1	|S|×|T|	
$A(x,y,z) \wedge B(x,y,u)$ $\wedge\, C(x,z,u) \wedge D(y,z,u)$	1/3, 1/3, 1/3, 1/3	$(\lvert A\rvert \times \lvert B\rvert \times \lvert C\rvert \times \lvert D\rvert)^{1/3}$	min(…)
	1,1,0,0	$\lvert A\rvert \times \lvert B\rvert$	
	1,0,1,0	$\lvert A\rvert \times \lvert C\rvert$	
	…	…	

Examples

Query	$w_1, w_2, \dots, w_m$	$\lvert R_1\rvert^{w1} \times \cdots \times \lvert R_m\rvert^{wm}$	$\lvert Q\rvert \le$ ...
$R(x,y) \wedge S(y,z)$	1,1	$\lvert R\rvert \times \lvert S\rvert$	$\le \lvert R\rvert \times \lvert S\rvert$
$R(x,y) \wedge S(y,z) \wedge T(z,x)$	½, ½, ½	$(\lvert R\rvert \times \lvert S\rvert \times \lvert T\rvert)^{½}$	$\le \min($ $(\lvert R\rvert \times \lvert S\rvert \times \lvert T\rvert)^{½}$, $\lvert R\rvert \times \lvert S\rvert$, $\lvert R\rvert \times \lvert T\rvert$, $\lvert S\rvert \times \lvert T\rvert)$
	1,1,0	|R|×|S|	
	1,0,1	|R|×|T|	
	0,1,1	|S|×|T|	
$A(x,y,z) \wedge B(x,y,u)$ $\wedge C(x,z,u) \wedge D(y,z,u)$	1/3, 1/3, 1/3, 1/3	$(\lvert A\rvert \times \lvert B\rvert \times \lvert C\rvert \times \lvert D\rvert)^{1/3}$	min(…)
	1,1,0,0	$\lvert A\rvert \times \lvert B\rvert$	
	1,0,1,0	$\lvert A\rvert \times \lvert C\rvert$	
	…	…	
$R(x,y) \wedge S(y,z) \wedge T(z,u)$ $\wedge K(u,v)$			

Examples

Query	$w_1, w_2, \dots, w_m$	$\|R_1\|^{w1} \times \cdots \times \|R_m\|^{wm}$	$\|Q\| \leq$ ...
$R(x,y) \wedge S(y,z)$	1,1	$\|R\| \times \|S\|$	$\leq \|R\| \times \|S\|$
$R(x,y) \wedge S(y,z) \wedge T(z,x)$	½, ½, ½	$(\|R\| \times \|S\| \times \|T\|)^{½}$	$\leq \min($ $(\|R\| \times \|S\| \times \|T\|)^{½}$, $\|R\| \times \|S\|$, $\|R\| \times \|T\|$, $\|S\| \times \|T\|)$
	1,1,0	|R|×|S|	
	1,0,1	|R|×|T|	
	0,1,1	|S|×|T|	
$A(x,y,z) \wedge B(x,y,u) \wedge C(x,z,u) \wedge D(y,z,u)$	1/3, 1/3, 1/3, 1/3	$(\|A\| \times \|B\| \times \|C\| \times \|D\|)^{1/3}$	min(…)
	1,1,0,0	$\|A\| \times \|B\|$	
	1,0,1,0	$\|A\| \times \|C\|$	
	…	…	
$R(x,y) \wedge S(y,z) \wedge T(z,u) \wedge K(u,v)$	1,0,1,1	$\|R\| \times \|T\| \times \|K\|$	
	1,1,0,1	$\|R\| \times \|S\| \times \|K\|$	
	1, ½, ½, 1		

Examples

Query	$w_1, w_2, \ldots, w_m$	$\lvert R_1\rvert^{w1} \times \cdots \times \lvert R_m\rvert^{wm}$	$\lvert Q\rvert \le \ldots$
$R(x,y) \wedge S(y,z)$	1,1	$\lvert R\rvert \times \lvert S\rvert$	$\le \lvert R\rvert \times \lvert S\rvert$
$R(x,y) \wedge S(y,z) \wedge T(z,x)$	½, ½, ½	$(\lvert R\rvert \times \lvert S\rvert \times \lvert T\rvert)^{½}$	$\le \min($ $(\lvert R\rvert \times \lvert S\rvert \times \lvert T\rvert)^{½}$, $\lvert R\rvert \times \lvert S\rvert$, $\lvert R\rvert \times \lvert T\rvert$, $\lvert S\rvert \times \lvert T\rvert)$
	1,1,0	|R|×|S|	
	1,0,1	|R|×|T|	
	0,1,1	|S|×|T|	
$A(x,y,z) \wedge B(x,y,u) \wedge C(x,z,u) \wedge D(y,z,u)$	1/3, 1/3, 1/3, 1/3	$(\lvert A\rvert \times \lvert B\rvert \times \lvert C\rvert \times \lvert D\rvert)^{1/3}$	min(…)
	1,1,0,0	$\lvert A\rvert \times \lvert B\rvert$	
	1,0,1,0	$\lvert A\rvert \times \lvert C\rvert$	
	…	…	
$R(x,y) \wedge S(y,z) \wedge T(z,u) \wedge K(u,v)$	1,0,1,1	$\lvert R\rvert \times \lvert T\rvert \times \lvert K\rvert$	
	1,1,0,1	$\lvert R\rvert \times \lvert S\rvert \times \lvert K\rvert$	
	1, ½, ½, 1	(no need; why?)	

Examples

Query	$w_1, w_2, \ldots, w_m$	$\lvert R_1\rvert^{w1} \times \cdots \times \lvert R_m\rvert^{wm}$	$\lvert Q\rvert \leq \ldots$
$R(x,y) \wedge S(y,z)$	1,1	$\lvert R\rvert \times \lvert S\rvert$	$\leq \lvert R\rvert \times \lvert S\rvert$
$R(x,y) \wedge S(y,z) \wedge T(z,x)$	½, ½, ½	$(\lvert R\rvert \times \lvert S\rvert \times \lvert T\rvert)^{½}$	$\leq \min($ $(\lvert R\rvert \times \lvert S\rvert \times \lvert T\rvert)^{½}$, $\lvert R\rvert \times \lvert S\rvert$, $\lvert R\rvert \times \lvert T\rvert$, $\lvert S\rvert \times \lvert T\rvert)$
	1,1,0	|R|×|S|	
	1,0,1	|R|×|T|	
	0,1,1	|S|×|T|	
$A(x,y,z) \wedge B(x,y,u) \wedge C(x,z,u) \wedge D(y,z,u)$	1/3, 1/3, 1/3, 1/3	$(\lvert A\rvert \times \lvert B\rvert \times \lvert C\rvert \times \lvert D\rvert)^{1/3}$	min(…)
	1,1,0,0	$\lvert A\rvert \times \lvert B\rvert$	
	1,0,1,0	$\lvert A\rvert \times \lvert C\rvert$	
	…	…	
$R(x,y) \wedge S(y,z) \wedge T(z,u) \wedge K(u,v)$	1,0,1,1	$\lvert R\rvert \times \lvert T\rvert \times \lvert K\rvert$	$\min(\lvert R\rvert \times \lvert T\rvert \times \lvert K\rvert$, $\lvert R\rvert \times \lvert S\rvert \times \lvert K\rvert)$
	1,1,0,1	$\lvert R\rvert \times \lvert S\rvert \times \lvert K\rvert$	
	1, ½, ½, 1	(no need; why?)	

Upper Bound of a Query

Theorem $|Q| \leq \min_{w_1,\cdots,w_m} |R_1|^{w_1} \times \cdots \times |R_m|^{w_m}$

This is called the AGM bound* of Q. It is tight.

Note: it suffices to consider only those fractional edge covers $w_1, \ldots, w_m$ that are not convex combinations of others

We will prove tightness on a special case.

But first, let's discuss an algorithm for computing Q with this runtime

*Atserias, Grohe, Marx introduced this bound

AGM(Q) = $\min_{w_1,\cdots,w_m} |R_1|^{w_1} \times \cdots \times |R_m|^{w_m}$

Generic Join – Overview

- Choose a variable order
- Sort every relation R_i according to this order: time is $O(|R_i| \log |R_i|) = \tilde{O}(|R_i|)$
- *Generic join* assumes relations are sorted; it computes Q in time $\tilde{O}(\text{AGM}(Q))$
- "Worst case optimal"

Generic Join – The Intersection

Intersection is the main building block of G.J.

$Q(x) = R(x) \wedge S(x)$

- Discuss merge-join in class – what is runtime?

Generic Join – The Intersection

Intersection is the main building block of G.J.

$Q(x) = R(x) \wedge S(x)$

- Discuss merge-join in class – what is runtime?
- Edge covers of Q: 1,0 and 0,1; $|Q| \leq \min(|R|, |S|)$

Generic Join – The Intersection

Intersection is the main building block of G.J.

$Q(x) = R(x) \wedge S(x)$

- Discuss merge-join in class – what is runtime?

- Edge covers of Q: 1,0 and 0,1; $|Q| \leq \min(|R|, |S|)$
- Discuss improved merge-join in class
  Runtime: $\tilde{O}(\min(|R|, |S|))$

Generic Join Algorithm

Let x be the first variable
Let R_{i_1}, R_{i_2}, ... be all relations containing x
Compute $D = \Pi_x(R_{i_1}) \cap \Pi_x(R_{i_2}) \cap \cdots$
for every value $v \in D$ do:
    Compute Q,
    where R_{i_1}, R_{i_2}, ... are restricted to $x = v$

needs to be done in time $\tilde{O}(min_j \, \Pi_x(R_j))$

Generic Join Example

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

Generic Join Example

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

$A = \Pi_x(R(x, y)) \cap \Pi_x(T(z, x))$

Generic Join Example

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

$A = \Pi_x(R(x, y)) \cap \Pi_x(T(z, x))$

for a in A **do**

/* compute $Q(a, y, z) = R(a, y) \wedge S(y, z) \wedge T(z, a)$ */

$B = \Pi_y(R(a, y)) \cap \Pi_y(S(y, z))$

Generic Join Example

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

$A = \Pi_x(R(x,y)) \cap \Pi_x(T(z,x))$

for a in A **do**

/* compute $Q(a,y,z) = R(a,y) \wedge S(y,z) \wedge T(z,a)$ */

$B = \Pi_y(R(a,y)) \cap \Pi_y(S(y,z))$

for b in B **do**

/* compute $Q(a,b,z) = R(a,b) \wedge S(b,z) \wedge T(z,a)$ */

Generic Join Example

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

$A = \Pi_x(R(x,y)) \cap \Pi_x(T(z,x))$

for a in A **do**

/* compute $Q(a,y,z) = R(a,y) \wedge S(y,z) \wedge T(z,a)$ */

$B = \Pi_y(R(a,y)) \cap \Pi_y(S(y,z))$

for b in B **do**

/* compute $Q(a,b,z) = R(a,b) \wedge S(b,z) \wedge T(z,a)$ */

$C = \Pi_z(S(b,z)) \cap \Pi_z(T(z,a))$

for c in C **do**

output (a,b,c)

Generic Join Example

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

$A = \Pi_x(R(x, y)) \cap \Pi_x(T(z, x))$
for a in A **do**
    /* compute $Q(a, y, z) = R(a, y) \wedge S(y, z) \wedge T(z, a)$ */
    $B = \Pi_y(R(a, y)) \cap \Pi_y(S(y, z))$
    for b in B **do**
    /* compute $Q(a, b, z) = R(a, b) \wedge S(b, z) \wedge T(z, a)$ */
        $C = \Pi_z(S(b, z)) \cap \Pi_z(T(z, a))$
        for c in C **do**
            output (a,b,c)

Runs in time $\tilde{O}(AGM(Q))$

Discussion

- All relations need to be presorted, or indexed
- Runtime is guaranteed to be worst-case optimal, *no matter* what variable order we choose
- In practice, the variable order *does matter*; in class: discuss $R(x,y) \wedge S(y,z)$

Comparison to Naïve Nested Loop

Naïve nested loop:

```
// tuple at a time:
For t1 in R1 do
  for t2 in R2 do
    …
```

Comparison to Naïve Nested Loop

Naïve nested loop:

```
// tuple at a time:
For t1 in R1 do
  for t2 in R2 do
    …

// value at a time:
For x in Domain do
  For y in Domain do
    For z in Domain do
      ...
```

Comparison to Naïve Nested Loop

Naïve nested loop:

```
// tuple at a time:
For t1 in R1 do
  for t2 in R2 do
    …

// value at a time:
For x in Domain do
  For y in Domain do
    For z in Domain do
      ...
```

Generic-join

```
A = ∩ domains for x
For x in A do
   B = ∩ domains for y
   For y in B do
      C = ∩ domains for z
      For z in C do
      ...
```

Tightness

- There exists instances $R_1, R_2,$ ... such that the size of the query's output is AGM(Q)

- Proof is simple and instructive; we will show for special case $|R_1| = \cdots = |R_m| = N$

- In this case $AGM(Q) = N^{\rho^*}$

Fractional Edge Covering Number

- The fractional edge covering number of a hypergraph is $\rho^* = \min \sum_e w_e$, where the minimum is over all fractional edge covers of the hypergraph.

Fact Assume $|R_1| = \ldots = |R_m| = N$. Then $AGM(Q) = N^{\rho^*}$

Fractional Vertex Packing

- A <u>*fractional vertex packing*</u> of a (hyper)graph is a set of non-negative numbers v_x, one for each node x, such that, for every edge e: $\sum_{x:\, x \in e} v_x \leq 1$

Fractional Vertex Packing

- A *fractional vertex packing* of a (hyper)graph is a set of non-negative numbers v_x, one for each node x, such that, for every edge e: $\sum_{x:\, x \in e} v_x \leq 1$

Fact For any v, w: $\sum_x v_x \leq \sum_e w_e$

Fractional Vertex Packing

- A *fractional vertex packing* of a (hyper)graph is a set of non-negative numbers v_x, one for each node x, such that, for every edge e: $\sum_{x:\, x \in e} v_x \leq 1$

Fact For any v, w: $\sum_x v_x \leq \sum_e w_e$

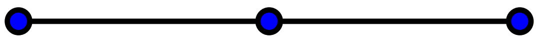
Theorem $\max_v \sum_x v_x = \rho^* = \min_w \sum_e w_e$

Fractional Vertex Packing

- A *fractional vertex packing* of a (hyper)graph is a set of non-negative numbers v_x, one for each node x, such that, for every edge e: $\sum_{x:\, x \in e} v_x \leq 1$

Fact For any v, w: $\sum_x v_x \leq \sum_e w_e$

Theorem $\max_v \sum_x v_x = \rho^* = \min_w \sum_e w_e$



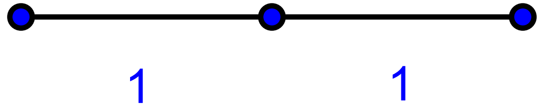
Fractional Vertex Packing

- A *fractional vertex packing* of a (hyper)graph is a set of non-negative numbers v_x, one for each node x, such that, for every edge e: $\sum_{x:\, x \in e} v_x \leq 1$

Fact For any v, w: $\sum_x v_x \leq \sum_e w_e$

Theorem $\max_v \sum_x v_x = \rho^* = \min_w \sum_e w_e$

$\rho^* = 1$

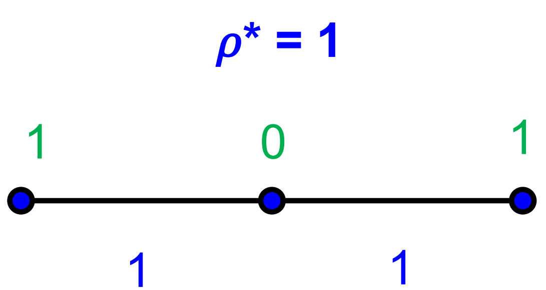


Fractional Vertex Packing

- A *fractional vertex packing* of a (hyper)graph is a set of non-negative numbers v_x, one for each node x, such that, for every edge e: $\sum_{x:\, x \in e} v_x \leq 1$

Fact For any v, w: $\sum_x v_x \leq \sum_e w_e$

Theorem $\max_v \sum_x v_x = \rho^* = \min_w \sum_e w_e$

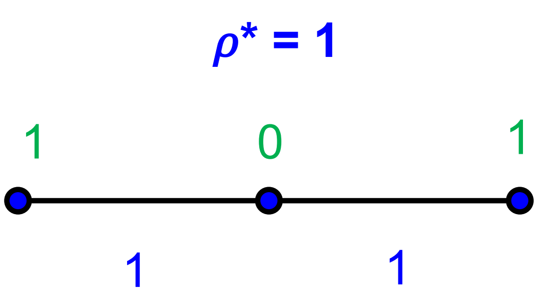


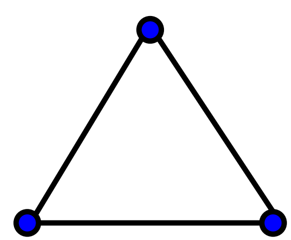
Fractional Vertex Packing

- A <u>*fractional vertex packing*</u> of a (hyper)graph is a set of non-negative numbers v_x, one for each node x, such that, for every edge e: $\sum_{x:\, x \in e} v_x \leq 1$

Fact For any v, w: $\sum_x v_x \leq \sum_e w_e$

Theorem $\max_v \sum_x v_x = \rho^* = \min_w \sum_e w_e$





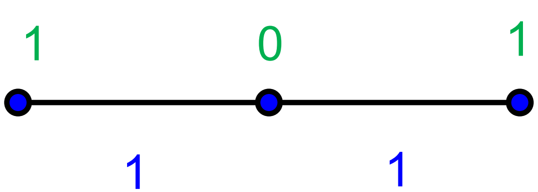
Fractional Vertex Packing

- A *fractional vertex packing* of a (hyper)graph is a set of non-negative numbers v_x, one for each node x, such that, for every edge e: $\sum_{x:\, x \in e} v_x \leq 1$

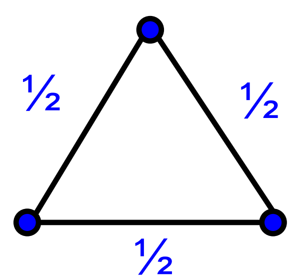
Fact For any v, w: $\sum_x v_x \leq \sum_e w_e$

Theorem $\max_v \sum_x v_x = \rho^* = \min_w \sum_e w_e$

$\rho^* = 1$



$\rho^* = 3/2$

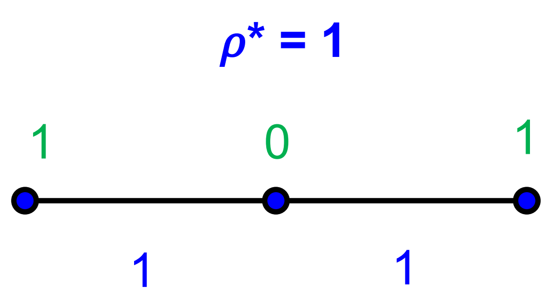


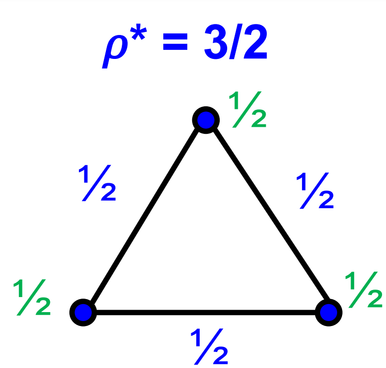
Fractional Vertex Packing

- A *fractional vertex packing* of a (hyper)graph is a set of non-negative numbers v_x, one for each node x, such that, for every edge e: $\sum_{x:\, x \in e} v_x \leq 1$

Fact For any v, w: $\sum_x v_x \leq \sum_e w_e$

Theorem $\max_v \sum_x v_x = \rho^* = \min_w \sum_e w_e$



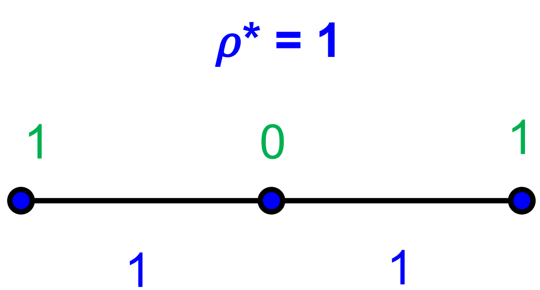


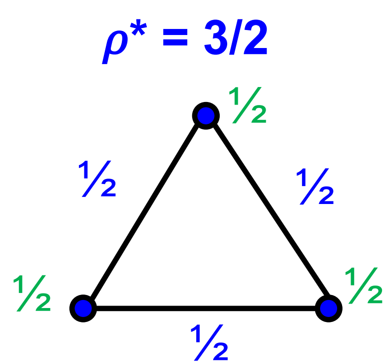
Fractional Vertex Packing

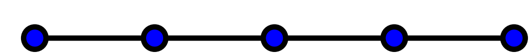
- A ***fractional vertex packing*** of a (hyper)graph is a set of non-negative numbers v_x, one for each node x, such that, for every edge e: $\sum_{x:\, x \in e} v_x \leq 1$

Fact For any v, w: $\sum_x v_x \leq \sum_e w_e$

Theorem $\max_v \sum_x v_x = \rho^* = \min_w \sum_e w_e$







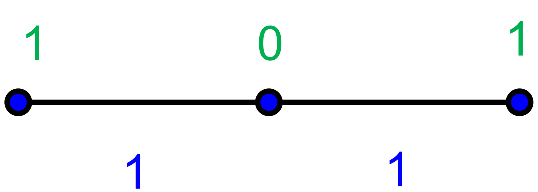
Fractional Vertex Packing

- A *fractional vertex packing* of a (hyper)graph is a set of non-negative numbers v_x, one for each node x, such that, for every edge e: $\sum_{x:\, x \in e} v_x \leq 1$

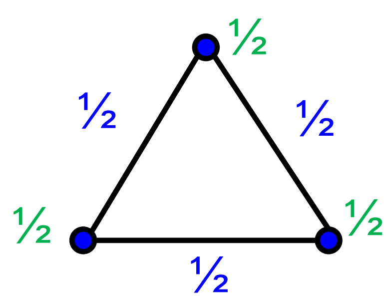
Fact For any v, w: $\sum_x v_x \leq \sum_e w_e$

Theorem $\max_v \sum_x v_x = \rho^* = \min_w \sum_e w_e$

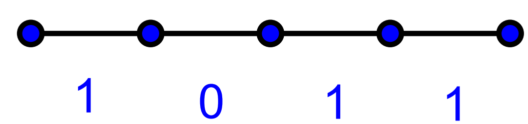
$\rho^* = 1$



$\rho^* = 3/2$



$\rho^* = 3$

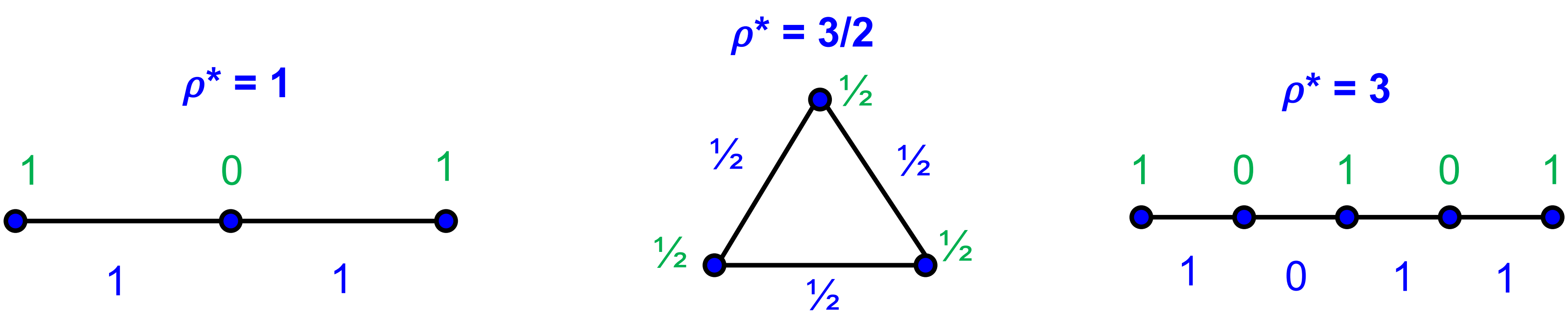


Fractional Vertex Packing

- A *fractional vertex packing* of a (hyper)graph is a set of non-negative numbers v_x, one for each node x, such that, for every edge e: $\sum_{x:\, x \in e} v_x \leq 1$

Fact For any v, w: $\sum_x v_x \leq \sum_e w_e$

Theorem $\max_v \sum_x v_x = \rho^* = \min_w \sum_e w_e$



The Bound is Tight

Fact Fix a fractional vertex packing v = $(v_x)_{x \in Nodes}$.
Then there exists a database such that
$|R_1| \leq N, \ldots, |R_m| \leq N$ and $|Q| = N^{\sum_x v_x}$

The Bound is Tight

Fact Fix a fractional vertex packing v = $(v_x)_{x \in Nodes}$.
Then there exists a database such that
$R_1| \leq N$, …, $|R_m| \leq N$ and $|Q| = N^{\Sigma_x v_x}$

Proof. For every relation R_j with variables $x_{i_1}, x_{i_2}, \ldots$
define the instance $|R_j| = [N^{v_{i_1}}] \times [N^{v_{i_2}}] \times \cdots$
where [k] = {1,2,…,k}.

The Bound is Tight

Fact Fix a fractional vertex packing v = $(v_x)_{x \in Nodes}$.
Then there exists a database such that
$|R_1| \leq N$, …, $|R_m| \leq N$ and $|Q| = N^{\Sigma_x v_x}$

Proof. For every relation R_j with variables $x_{i_1}, x_{i_2}, \ldots$
define the instance $|R_j| = [N^{v_{i_1}}] \times [N^{v_{i_2}}] \times \cdots$
where [k] = {1,2,…,k}. Then:
(a) $|R_j| = N^{v_{i_1}+v_{i_2}+\cdots} \leq N$ (why?)

The Bound is Tight

Fact Fix a fractional vertex packing v = $(v_x)_{x \in Nodes}$.
Then there exists a database such that
$|R_1| \leq N$, …, $|R_m| \leq N$ and $|Q| = N^{\sum_x v_x}$

Proof. For every relation R_j with variables $x_{i_1}, x_{i_2}, \ldots$
define the instance $|R_j| = [N^{v_{i_1}}] \times [N^{v_{i_2}}] \times \cdots$
where [k] = {1,2,…,k}. Then:
$(a) |R_j| = N^{v_{i_1} + v_{i_2} + \cdots} \leq N$ (why?)
$(b) |Q| = N^{\sum_x v_x}$ (why?)

Example

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

- Assume $|R|=|S|=|T|=N$.

Example

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

- Assume $|R|=|S|=|T|=N$.
- Optimal vertex packing: $v_x = v_y = v_z = 1/2$

Example

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

- Assume $|R|=|S|=|T|=N$.
- Optimal vertex packing: $v_x = v_y = v_z = 1/2$
- Define: $D_x = [N^{½}] = \{1, 2, \ldots, N^{½}\}$
  $D_y = [N^{½}]$
  $D_z = [N^{½}]$

Example

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

- Assume $|R|=|S|=|T|=N$.
- Optimal vertex packing: $v_x = v_y = v_z = 1/2$
- Define: $D_x = [N^{½}] = \{1, 2, \ldots, N^{½}\}$
  $D_y = [N^{½}]$
  $D_z = [N^{½}]$
- Define $R = D_x \times D_y$, $S = D_y \times D_z$, $T = D_z \times D_x$.

Example

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

- Assume $|R|=|S|=|T|=N$.
- Optimal vertex packing: $v_x = v_y = v_z = 1/2$
- Define: $D_x = [N^{½}] = \{1, 2, \ldots, N^{½}\}$
  $D_y = [N^{½}]$
  $D_z = [N^{½}]$
- Define $R = D_x \times D_y$, $S = D_y \times D_z$, $T = D_z \times D_x$.
- Then $|R| = |S| = |T| = N$,
  $Q = D_x \times D_y \times D_z$ and $|Q| = N^{3/2}$

Keys

$R(X,Y) \wedge S(Y,Z)$, $|R|, |S| \leq N$

- No other info: $|Q(D)| \leq N^2$
- S.Y is a key: $|Q(D)| \leq N$

The *Query Expansion* method:

- If Y is a key in some relation S, then add all attributes of S relations containing Y
- Compute AGM($Q^{expanded}$)

Examples

Y is a key

$$Q(X, Y, Z) = R(X, Y) \wedge S(\underline{Y}, Z)$$

Examples

Y is a key

$$Q(X,Y,Z) \;=\; R(X,Y) \wedge S(\underline{Y},Z)$$

- $Q^{exp}(X,Y,Z) \;=\; R(X,Y,Z) \wedge S(Y,Z)$,

Examples

Y is a key

$Q(X, Y, Z) = R(X, Y) \wedge S(\underline{Y}, Z)$

- $Q^{exp}(X, Y, Z) = R(X, Y, Z) \wedge S(Y, Z)$,
- Edge cover: 1,0

Examples

Y is a key

$Q(X, Y, Z) = R(X, Y) \wedge S(\underline{Y}, Z)$

- $Q^{exp}(X, Y, Z) = R(X, Y, Z) \wedge S(Y, Z)$,
- Edge cover: 1,0
- $AGM(Q^{exp}) = |R|$

Examples

Y is a key

$Q(X, Y, Z) = R(X, Y) \wedge S(\underline{Y}, Z)$

- $Q^{exp}(X, Y, Z) = R(X, Y, Z) \wedge S(Y, Z)$,
- Edge cover: 1,0
- $AGM(Q^{exp}) = |R|$

$Q(X, Y, Z) = R(X, Y) \wedge S(\underline{Y}, Z) \wedge T(Z, X)$

Examples

Y is a key

$Q(X, Y, Z) = R(X, Y) \wedge S(\underline{Y}, Z)$

- $Q^{exp}(X, Y, Z) = R(X, Y, Z) \wedge S(Y, Z)$,
- Edge cover: 1,0
- $AGM(Q^{exp}) = |R|$

$Q(X, Y, Z) = R(X, Y) \wedge S(\underline{Y}, Z) \wedge T(Z, X)$

- $Q^{exp}(X, Y, Z) = R(X, Y, Z) \wedge S(Y, Z) \wedge T(Z, X)$
- Edge covers: 1,0,0 or 0,1,1

Examples

Y is a key

$Q(X,Y,Z) = R(X,Y) \wedge S(\underline{Y},Z)$

- $Q^{exp}(X,Y,Z) = R(X,Y,Z) \wedge S(Y,Z)$,
- Edge cover: 1,0
- $AGM(Q^{exp}) = |R|$

$Q(X,Y,Z) = R(X,Y) \wedge S(\underline{Y},Z) \wedge T(Z,X)$

- $Q^{exp}(X,Y,Z) = R(X,Y,Z) \wedge S(Y,Z) \wedge T(Z,X)$
- Edge covers: 1,0,0 or 0,1,1

Examples

Y is a key

$Q(X,Y,Z) = R(X,Y) \wedge S(\underline{Y},Z)$

- $Q^{exp}(X,Y,Z) = R(X,Y,Z) \wedge S(Y,Z),$
- Edge cover: 1,0
- $AGM(Q^{exp}) = |R|$

$Q(X,Y,Z) = R(X,Y) \wedge S(\underline{Y},Z) \wedge T(Z,X)$

- $Q^{exp}(X,Y,Z) = R(X,Y,Z) \wedge S(Y,Z) \wedge T(Z,X)$
- Edge covers: 1,0,0 or 0,1,1
- $AGM(Q^{exp}) = \min(|R|, |S| \times |T|)$

Summary

Given cardinalities of all input tables:

- AGM bound gives upper bound on query size
- GJ computes the query in this time

Generic Join:

- A nested loop algorithm
- No longer one-join-at-a-time
- Theoretical optimality means it will be efficient for very expensive queries; less so for cheaper queries