From Hypertree Width to Submodular Width and Data-dependent Structural Decompositions

Francesco Scarcello, **DIMES, Università della Calabria**

and Georg Gottlob, Vienna, Oxford...

Foundational Challenges in Data and Knowledge Management

Georg Gottlob special event

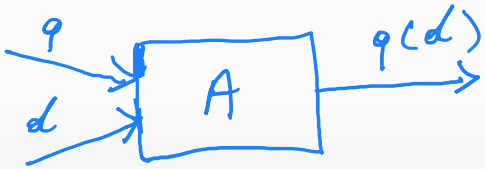
The origins

- I first met Georg Gottlob in Italy, at my first (logic- programming) conference.
  Georg was an invited speaker
- After a few years I visited Georg and Nicola Leone in Vienna
  - Georg was an invited speaker at a conference on Inductive Logic Programming
  - We worked on PAC learning of acyclic clauses
- We started our investigations on acyclic queries and their generalizations
- It was the time of Georgs's Wittgenstein award
- A beatufil and very stimulating period
  - Kurt Gödel Colloquium and Grohe lecture on the Robber and Cops game
- A never ending friendship and fruitful scientific collaboration
  - Databases, Game Theory, Knowledge Representation and Reasoning

The challenge

$\mathcal{C}$: a class of conjunctive queries

$\mathbb{D}$: all possible databases

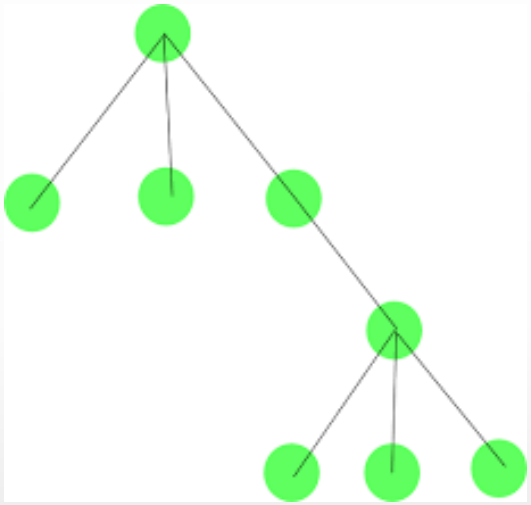


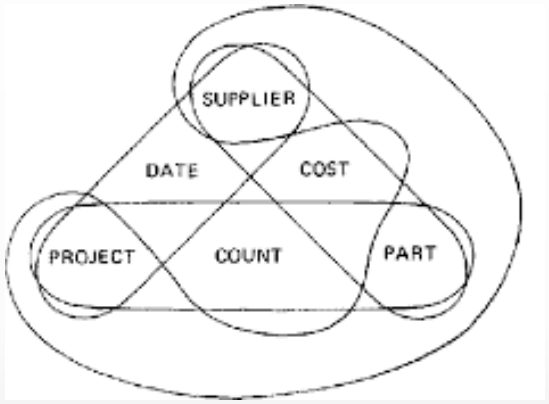
COMBINED COMPLEXITY

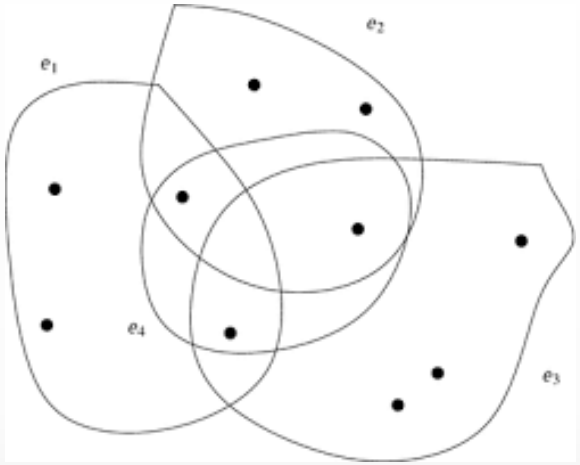
$\mathcal{C}$ IS AN ISLAND OF TRACTABILITY IF THERE EXISTS A POLYNOMIAL TIME ALGORITHM A THAT COMPUTES $q(\mathcal{B})$ $\forall q \in \mathcal{C}$ and $\forall \mathcal{B} \in \mathbb{D}$

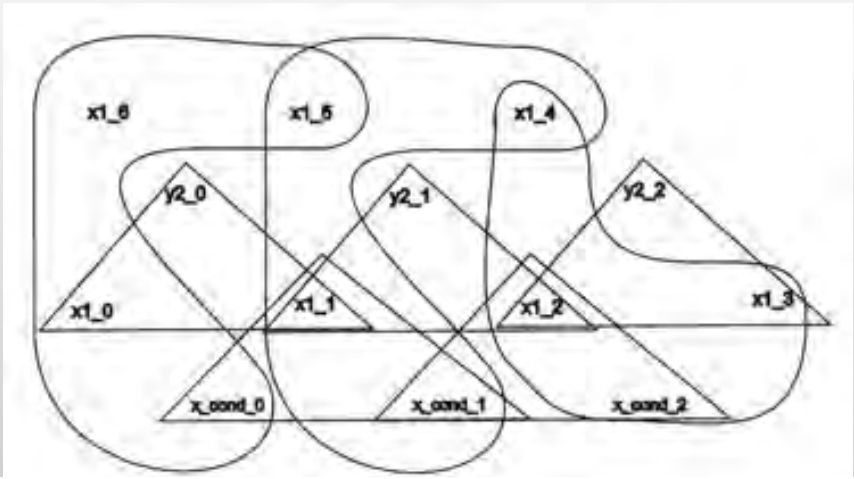
FIND THE LARGEST ISLAND(S) OF TRACTABILITY

Georg: «The evil is in cycles»





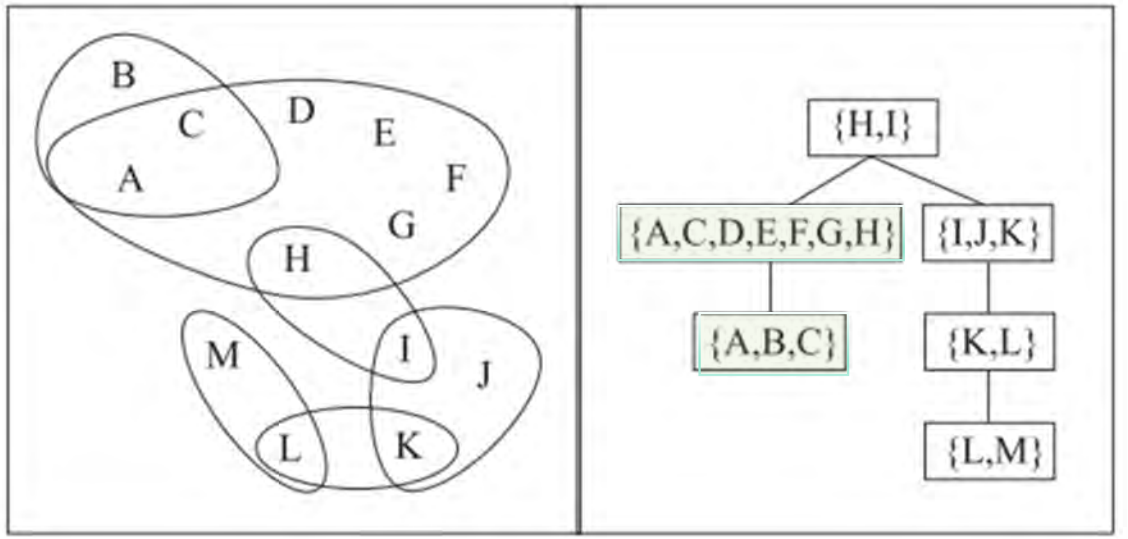


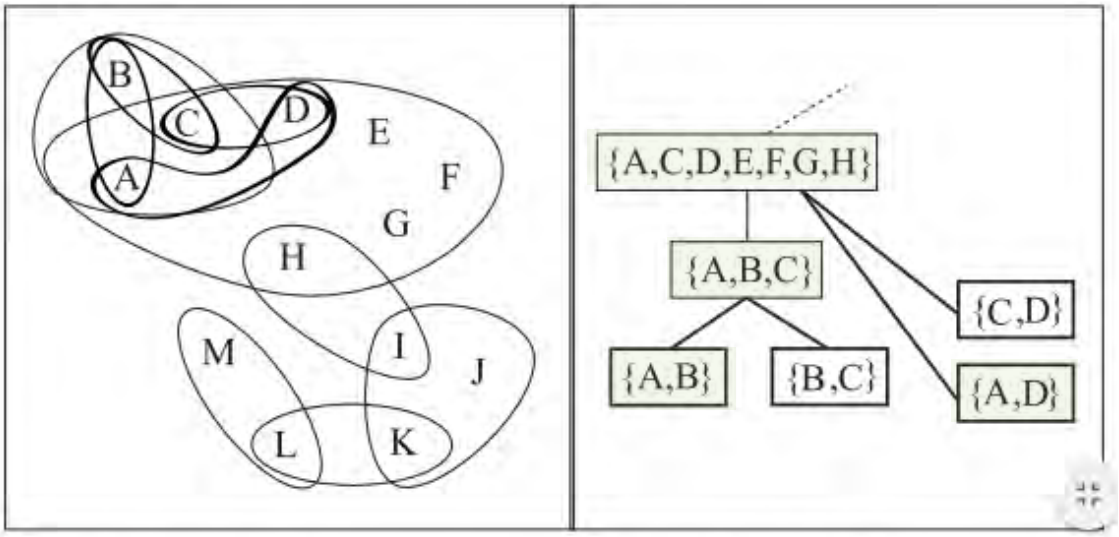


Acyclic queries & join trees

Join tree: a tree whose nodes are labelled by query hyperedges (or query atoms) such that:

- each hyperedge labels some node, and
- For each query variable V, the tree-nodes containing V span a connected subtree (*connectednes condition*)





Acyclic conjunctive query (ACQ):

A Query whose associated hypergraph is acyclic (more precisely,α-acyclic **[Fagin 83]**)

Query acyclicity was independently defined by

- [Beeri et al. STOC81] *acyclic database schemas*, and
- [Goodman & Shmueli 1981, TODS'82] *tree queries*

[Graham; Yu & Özsoyoğlu] *GYO reduction*

A query is acyclic iff it has a join tree

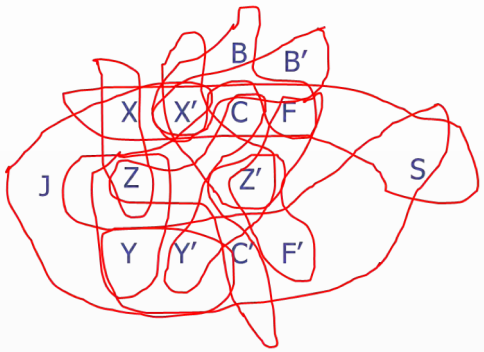
Good properties:

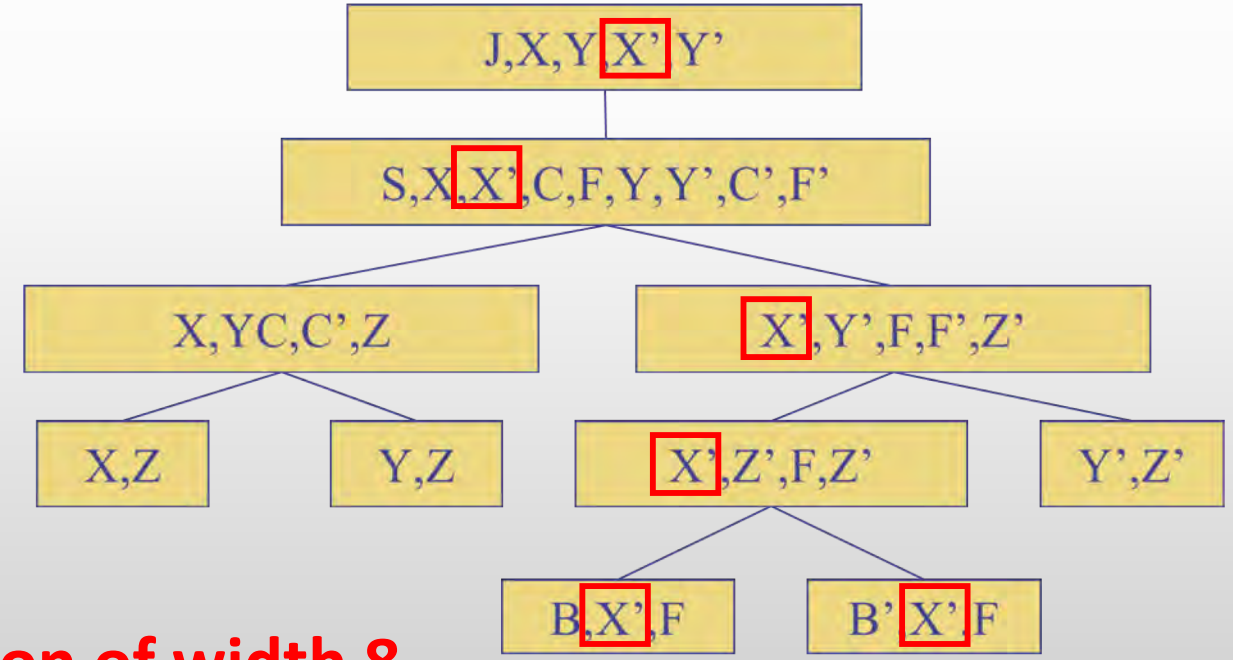
- ACQs can be recognized in polynomial (actually linear) time,
- A join-tree for an ACQ can be built in linear time,
- A Bolean ACQ Q can be answered in time $O(|Q| \times |r_{max}| \times \log |r_{max}|)$ [Yan'81]
- A non-Boolean ACQ can be answered with polynomial delay.

Generalizing acyclicity

Tree Decomposition

$$ans \leftarrow a(S,X,X',C,F) \wedge b(S,Y,Y',C',F') \wedge c(C,C',Z) \wedge d(X,Z) \wedge$$
$$e(Y,Z) \wedge f(F,F',Z') \wedge g(X',Z') \wedge h(Y',Z') \wedge$$
$$j(J,X,Y,X',Y') \wedge p(B,X',F) \wedge q(B',X',F)$$



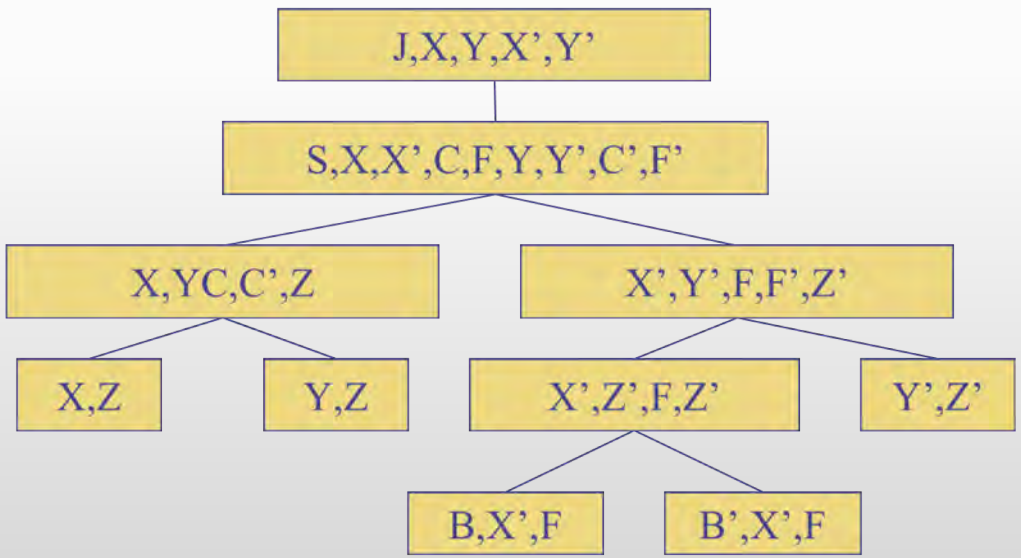


- Variables of each atom covered by some bag
- Connectedness condition

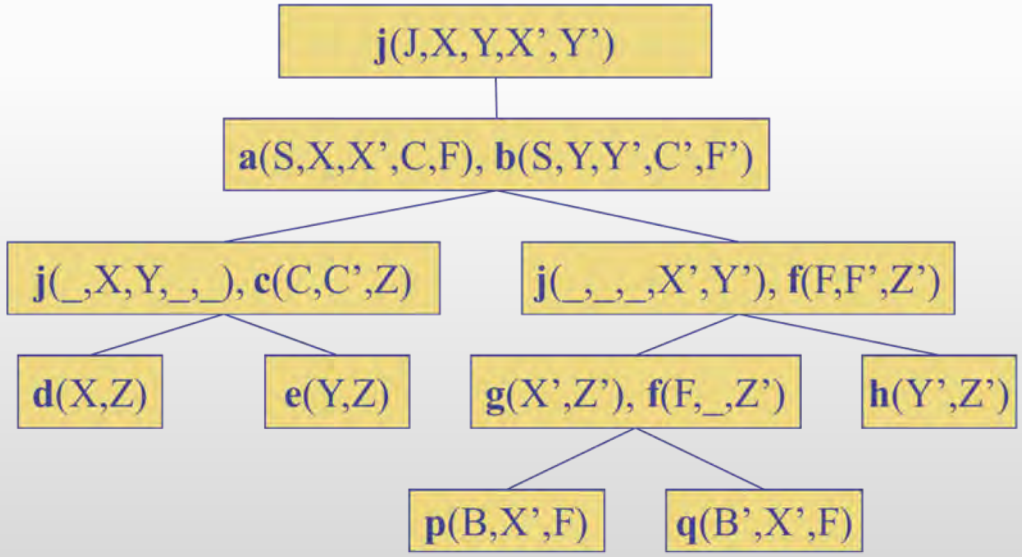
Tree Decomposition of width 8

The power of hyperedges: Generalized Hypertree Decomposition

$$ans \leftarrow a(S,X,X',C,F) \wedge b(S,Y,Y',C',F') \wedge c(C,C',Z) \wedge d(X,Z) \wedge$$
$$e(Y,Z) \wedge f(F,F',Z') \wedge g(X',Z') \wedge h(Y',Z') \wedge$$
$$j(J,X,Y,X',Y') \wedge p(B,X',F) \wedge q(B',X',F)$$



tw=8



ghw=2

Unfortunately:

GHW is NP-hard to compute, even for small width.

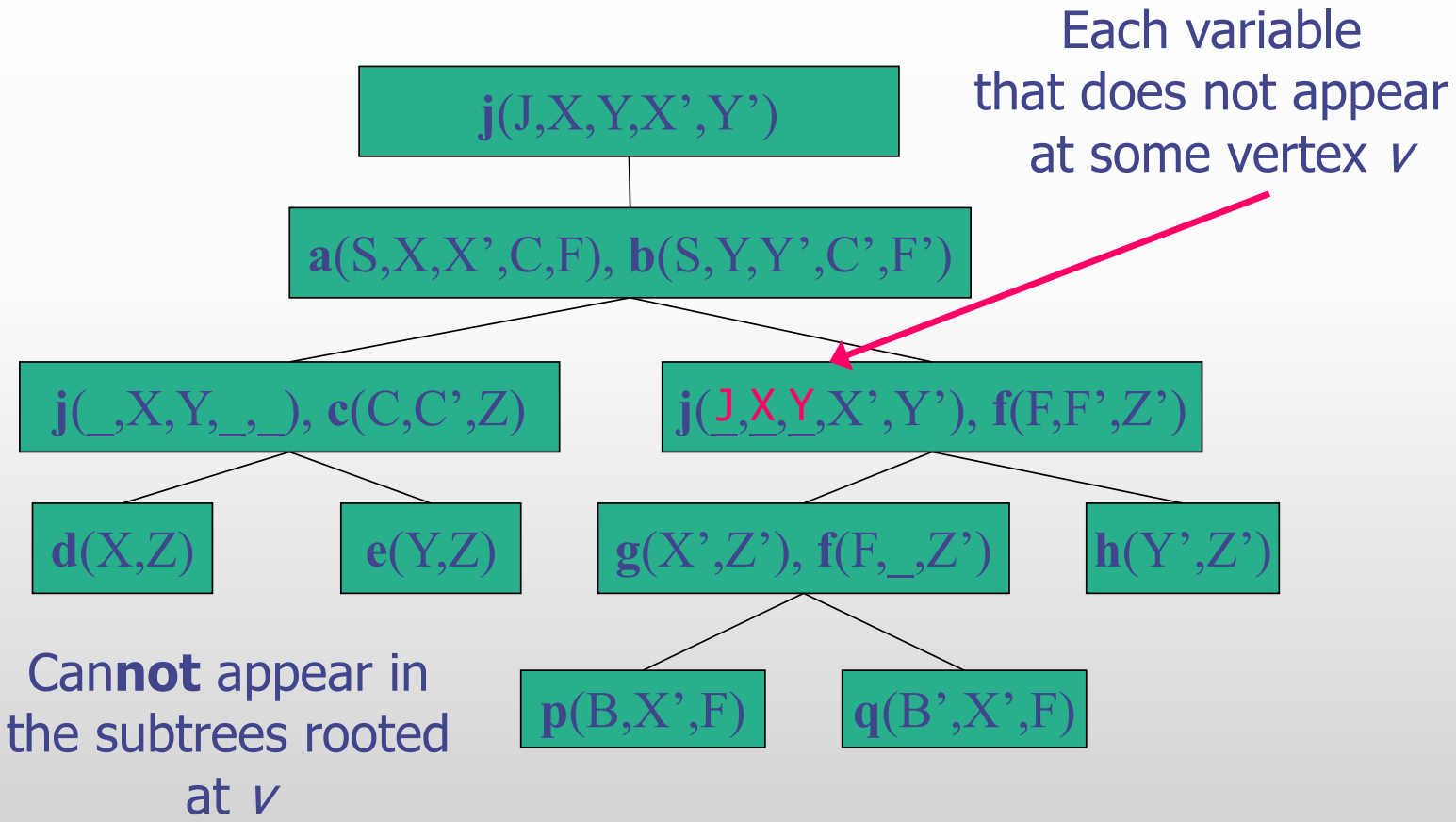
Theorem [G., Miklos, Schwentick 07+09]:

Checking whether ghw(Q)=3 is NP complete

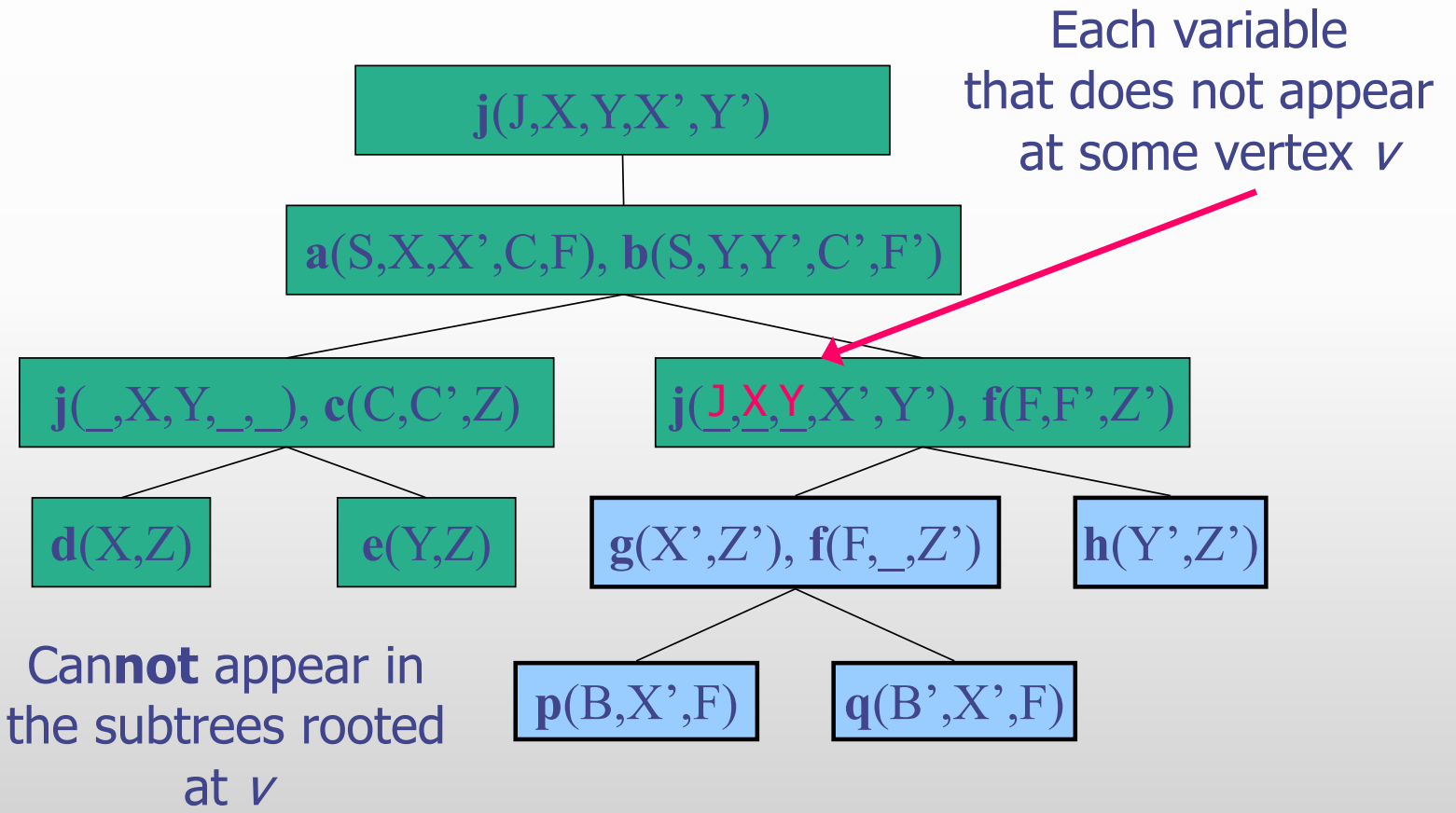
Thus, GHDs do not fulfill criterion 2 (efficient recognizability).

→ Slightly restrict GHDs using a ***special condition***, yielding HDs

Hypertree Decomposition
= GHD +Special Condition



Hypertree Decomposition
= GHD +Special Condition



Good properties of (G)HDs and of queries of bounded (G)HW

✓ GHW and HW generalize acyclicity:
for an acyclic query *Q*, ghw(Q)=hw(Q)=1 .

1. generalization

✓ For fixed *k*, deciding whether
hw(Q) ≤ k is in polynomial time: $O(v \times |Q|^{2k})$.

✓ Computing k-hypertree decompositions -if they exist- is feasible within the same bound.

2. efficient recognizability

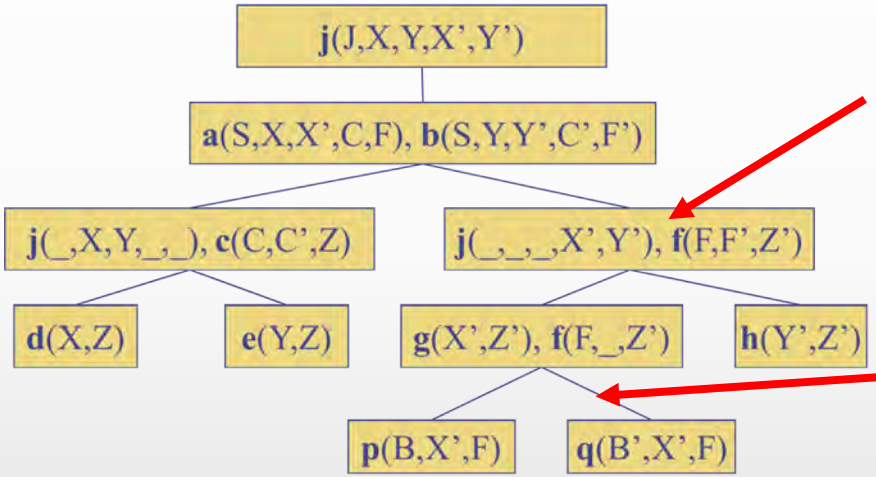
✓ Answering Boolean CQs of ghw or hw k is feasible in time $O(|Q| \times |r_{max}|^k \times \log |r_{max}|)$;
answering non-Boolean CQs with poly delay.

3. efficient query-answering

Hybrid decompositions: matching physical DB parameters

(G)HDs are query plans. Based on available selectivity and cardinality indexes, we can associate a cost to each such decomposition:

D:



Local cost c(n) at each decomposition-node n for joins & projections

Local cost c'(n) at each decomposition-edge e for semijoin

$$cost(D)=\sum_{n\in N} c(n) + \sum_{e\in E} c'(e)$$

Work by Scarcello, Greco, Leone [PODS 04; JCSS 07] :

Theorem: Finding a minimum cost HD (or GHD) is NP-hard.

Note: This is just as bad as classical query optimization.

More surprising: Problem is tractable for a slight restriction of HDs:

Theorem: Finding a minimum cost *normal-form* HD is tractable.

A k-width HD is in normal form iff it is generated by Opt-k-Dekomp

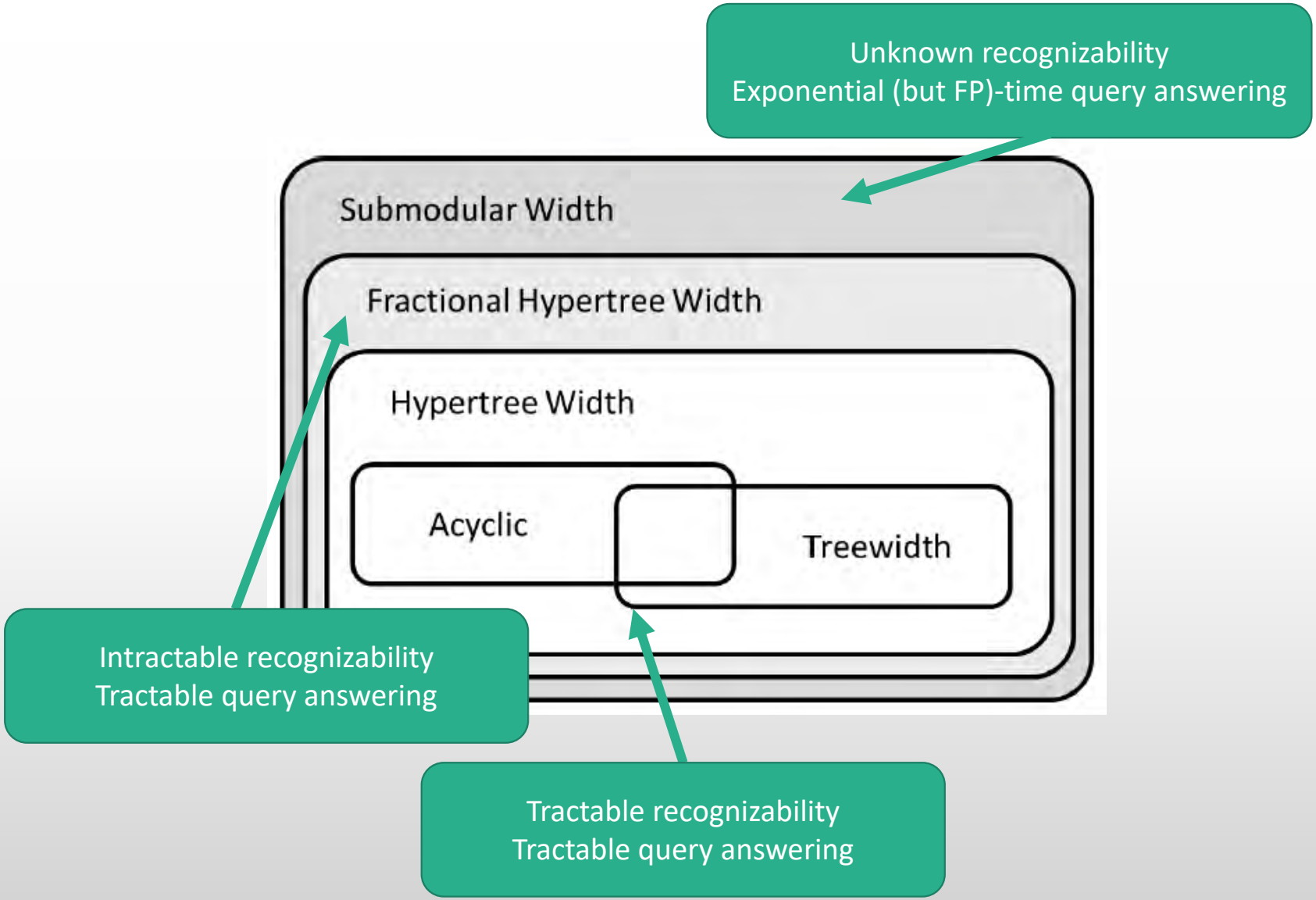
Algorithm Cost-k-Dekomp

More research needed to understand trade-off

Many more works implementing HDs and generalizations

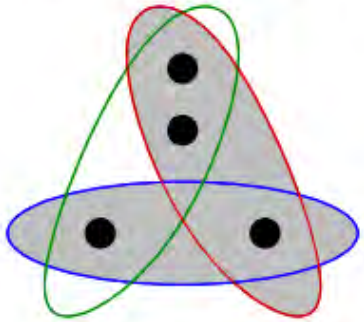
- [Afrati,Joglekar, Ré, Salihoglu,Ullman, ICDT 2017]: GYM: A Multiround Join Algorithm In MapReduce.
- [Koch , Ahmad, Kennedy, Nikolic, Nötzli, Lupei, Shaikhha VLDBJ 2014] DBToaster: higher-order delta processing for dynamic, frequently fresh views.
- [Tu,Ré SIGMOD 2015] DunceCap: Query Plans Using Generalized Hypertree Decompositions
- [Abo Khamis, Ngo, Rudra PODS 2016 Best Paper] FAQ: Questions Asked Frequently
- [Aberger, Tu, Olukotun, Ré, SIGMOD 2016] EmptyHeaded: A Relational Engine for Graph Processing.
- [Joglekar, Puttagunta,Ré, PODS‘16] AJAR: Aggregations and Joins over Annotated Relations
- [Khamis, Ngo, Suciu, PODS‘17] What Do Shannon-type Inequalities, Submodular Width, and Disjunctive Datalog Have to Do with One Another?

Beyond hypertrees (?)

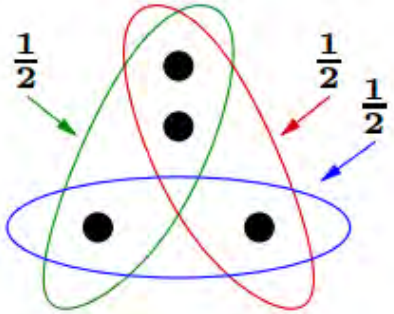

Unknown recognizability
Exponential (but FP)-time query answering
Submodular Width
Fractional Hypertree Width
Hypertree Width
Acyclic
Treewidth
Intractable recognizability
Tractable query answering
Tractable recognizability
Tractable query answering

A **fractional edge cover** is a weight assignment to the edges such that every vertex is covered by total weight at least 1.

$\varrho^*(H)$: smallest total weight of a fractional edge cover.



$\varrho(H) = 2$



$\varrho^*(H) = 1.5$

© [Marx 2005]

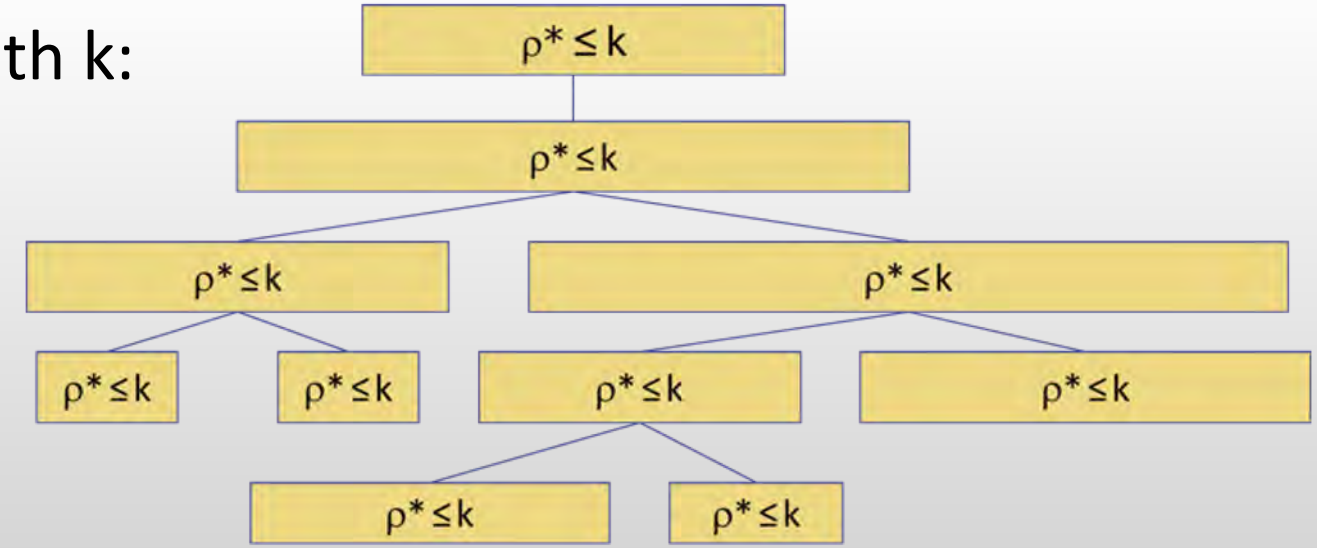
Theorem [Grohe & Marx 06] : The answer to a query of fractional cover weight $\rho^*(Q)$ can be computed in time $|Q| \times rmax^{\rho^*(Q)+O(1)}$

Observe: $\{Q|\ \rho^*(Q) \leq k\}$ and $\{Q|\ hw(Q) \leq k\}$ are incomparable.

To combine the two notions profitably, Grohe and Marx defined

Fractional Hypertree Decompositions (FHDs) and correspondingly FHW

FHD of width k:



The AGM bound [Atezerias, Grohe, Marx '08]

Let q be a full conjunctive query. For every fractional edge cover $\mathbf{u}$ of q, we have:

$$|q| \leq \prod_{j=1}^{\ell} N_j^{u_j}$$

$$\begin{aligned} \min \quad & \sum_j \log_2(N_j) \cdot u_j \\ \text{s.t.} \forall x \in vars(q) : & \sum_{j: x \in vars(R_j)} u_j \geq 1 \\ & \forall R_j : u_j \geq 0 \end{aligned}$$

The AGM bound is tight

Coloring bound [G. Gottlob, S.T. Lee, G. Valiant, P. Valiant '12]

$$\textit{Given query } Q = R_0(u_0) \leftarrow R_{i_1}(u_1) \wedge \ldots \wedge R_{i_m}(u_m)$$

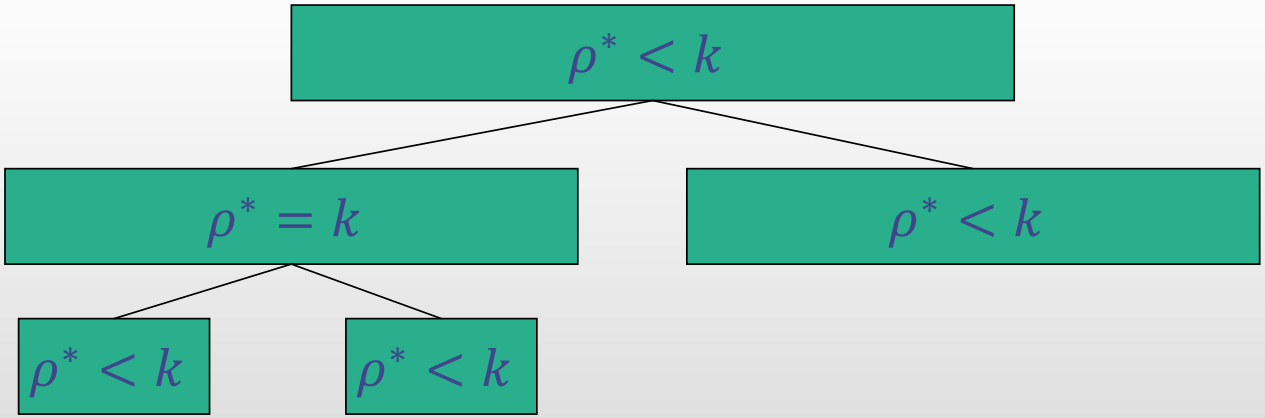
$$\begin{aligned} \textit{maximize} \quad & \textstyle\sum_{i:X_i \in u_0} x_i \\ \textit{subject to} \quad & \textstyle\sum_{i:X_i \in u_j} x_i \leq 1 \qquad \forall j \geq 1 \\ & x_i \geq 0. \end{aligned}$$

This bound is designed to work with output variables, and in the presence of keys and functional dependencies

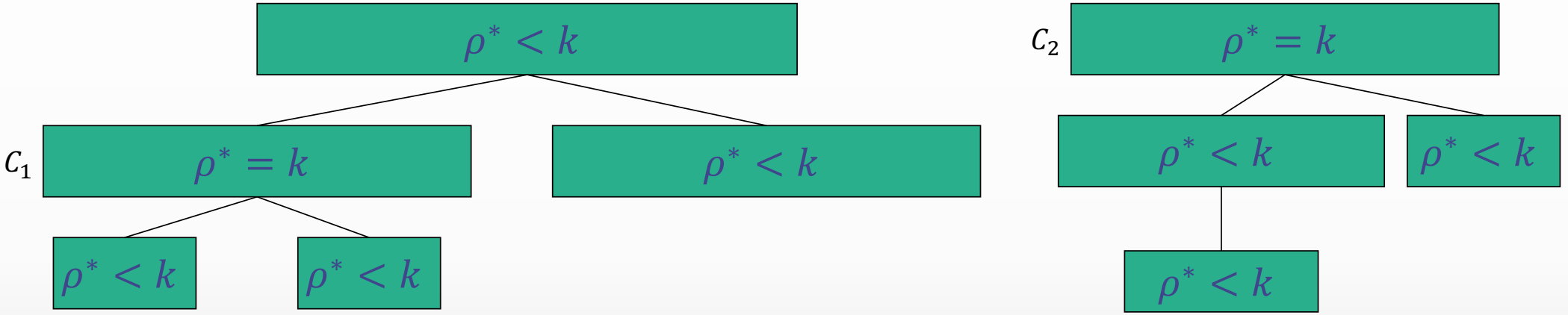
Fractional hypertree decompositions

- Assume the fractional hypertree width is k
- Then, at least one vertex has a cover equal to k
- There exists a database that meets the worst-case bound

FHD of width k:

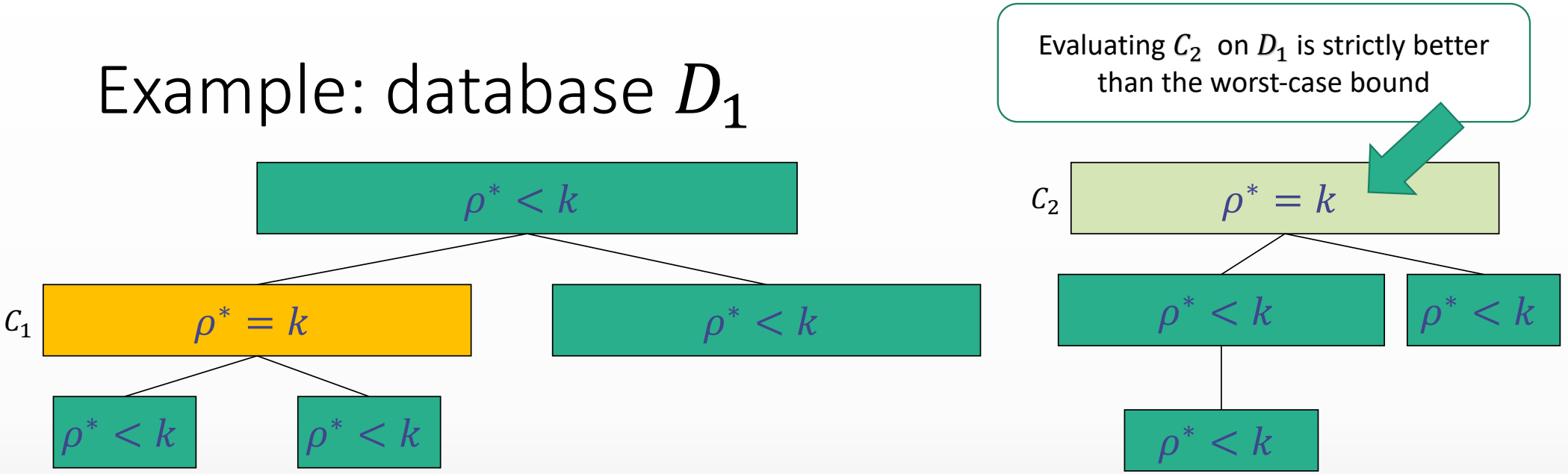


Why we can go beyond FHDs



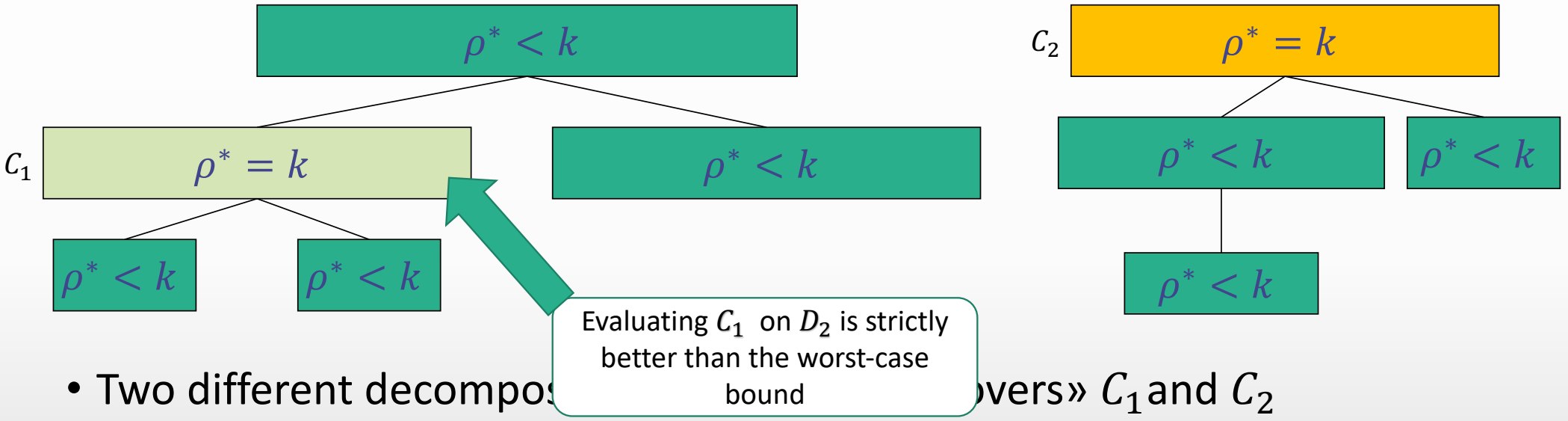
- Two different decompositions with «critical covers» C_1 and C_2
- From the AGM bound: there exists a database D_1 where C_1 meets the worst case bound and a database D_2 where C_2 meets the bound

Example: database D_1



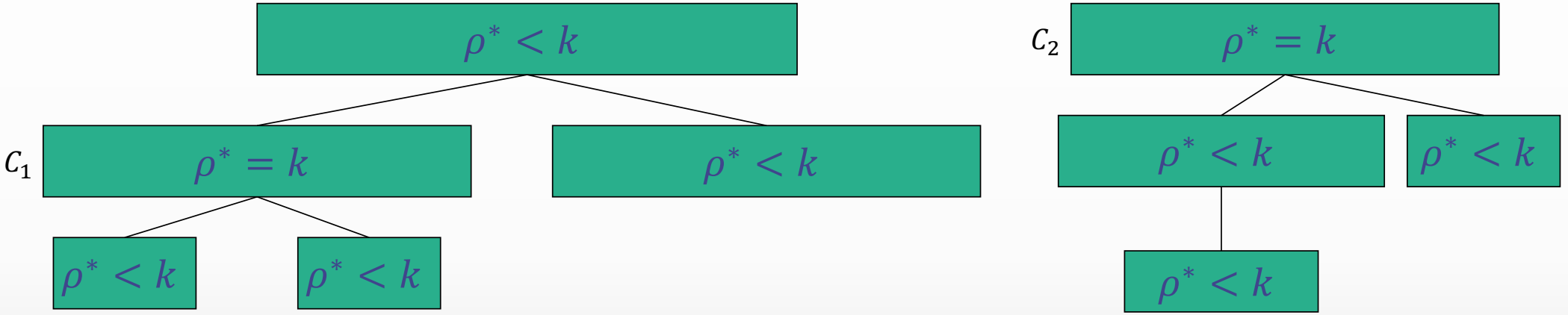
- Two different decompositions with «critical covers» C_1 and C_2
- From the AGM bound: there exists a database D_1 where C_1 meets the worst case bound and a database D_2 where C_2 meets the bound

Example: database D_2



- Two different decompos…overs» C_1 and C_2
- From the AGM bound: there exists a database D_1 where C_1 meets the worst case bound and a database D_2 where C_2 meets the bound

Why we can go beyond FHDs



- In general, we have different decompositions with different «critical covers»
- It is possible that there exists no database that simultaneusly meets the worst-case AGM bound on **all** hypertree decompositions!

What can we do?

- Given a query and a database, choose the best possible decomposition
  → use weighted hypertree decompositions

Hybrid approach

A more powerful (structural) width?

Think of a measure (width) such that, for every database,

there is a hypertree decomposition whose width does not exceed a given threshold

For some classes of queries, such a width can be strictly smaller than the fractional hypertree width

- See [Khamis, Ngo, Suciu 2017]

A case study: cycles

$q(x_1, x_2, x_3) \leftarrow r_1(x_1, x_2) \wedge r_2(x_2, x_3) \wedge r_3(x_1, x_3)$

Assume $|r_i| = N$

DIFFERENT POSSIBLE RELATIONS

1	1
2	1
3	2
4	3

1	1
1	2
2	1
2	2

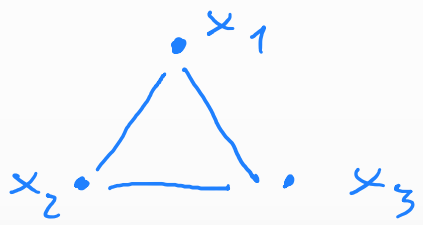
1	1
1	2
1	3
1	4

1	1
2	2
3	3
4	4

A case study: cycles

$q(x_1, x_2, x_3) \leftarrow r_1(x_1, x_2) \wedge r_2(x_2, x_3) \wedge r_3(x_1, x_3)$

Assume $|r_i| = N$



1	1
1	2
2	1
2	2

In this case, every variable can take at most $\sqrt{N}$ values, and $|q(b)| \leq N^{\frac{3}{2}}$

↑

AGM Bound

A case study: cycles

$q(x_1, x_2, x_3) \leftarrow r_1(x_1, x_2) \wedge r_2(x_2, x_3) \wedge r_3(x_1, x_3)$

Assume $|r_i| = N$

$r_1 = r_3$

1	1
1	2
1	3
1	4

In this case, x_1 take just one value and

$|q(\mathcal{D})| \leq |\pi_{x_1} r_1 \times r_2| = |r_2|$

A case study: cycles

$q(x_1, x_2, x_3) \leftarrow r_1(x_1, x_2) \wedge r_2(x_2, x_3) \wedge r_3(x_1, x_3)$

Assume $|r_i| = N$

In this case, we have keys, and

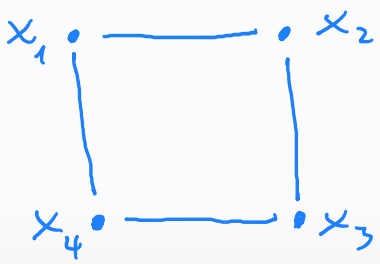
$|q(D)| \leq |r_i|$

r_i :

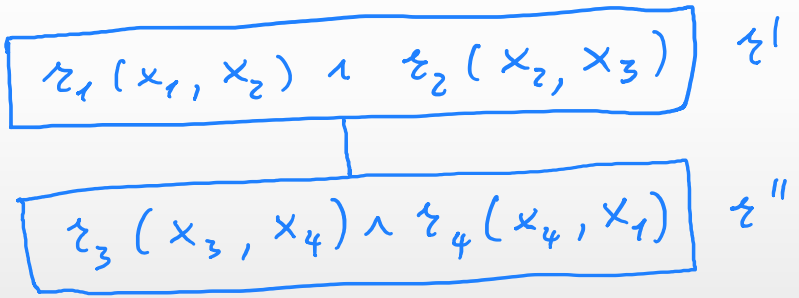
1	1
2	2
3	3
4	4

Cycles and decompositions

$q \leftarrow r_1(x_1, x_2) \wedge r_2(x_2, x_3) \wedge \cdots \wedge r_p(x_p, x_1)$



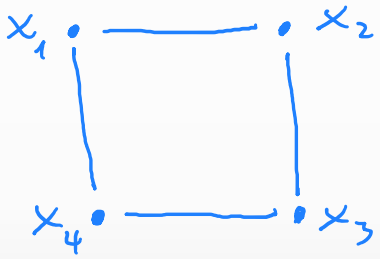
Assume $|r_i| = N$



The cost of cycles

$$q \leftarrow r_1(x_1, x_2) \wedge r_2(x_2, x_3) \wedge \cdots \wedge r_p(x_p, x_1)$$

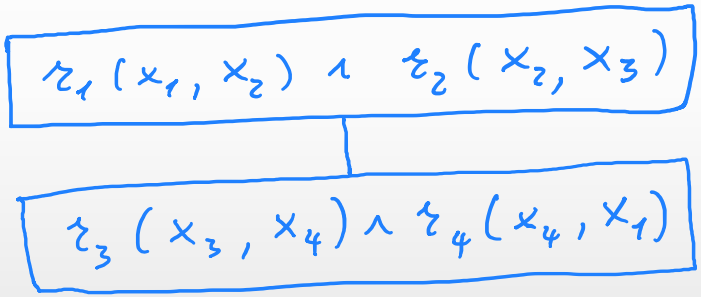
Assume $|r_i| = N$



EVERY CYCLE CAN BE SOLVED IN

$O(N^2)$

(FRACTIONAL) HYPERTREE WIDTH : 2

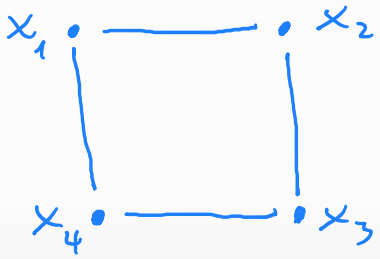


NOTE THAT THE WORST-CASE DATABASE MUST HAVE MANY VALUES FOR x_1 AND x_3, AND x_2 IS A HEAVY HITTER AND SHOULD HAVE FEW VALUES

The cost of cycles

$$q \leftarrow r_1(x_1, x_2) \wedge r_2(x_2, x_3) \wedge \cdots \wedge r_p(x_p, x_1)$$

Assume $|r_i| = N$



$r_1(x_1, x_2) \wedge r_2(x_2, x_3)$ — r'

$r_3(x_3, x_4) \wedge r_4(x_4, x_1)$ — r''

EVERY CYCLE CAN BE SOLVED IN

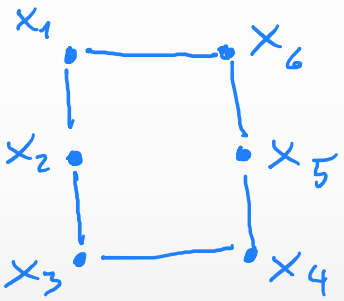
$O(N^2)$

Actually, for $r = 4$, it can be done in $O(N^{\frac{3}{2}})$

→ FOR EVERY GIVEN DATABASE

The cost of cycles

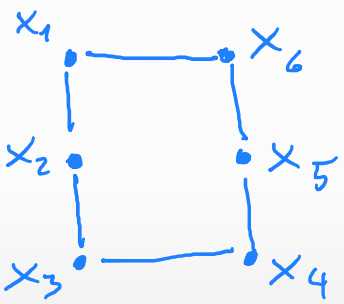
$$q \leftarrow r_1(x_1, x_2) \wedge r_2(x_2, x_3) \wedge \cdots \wedge r_p(x_p, x_1)$$

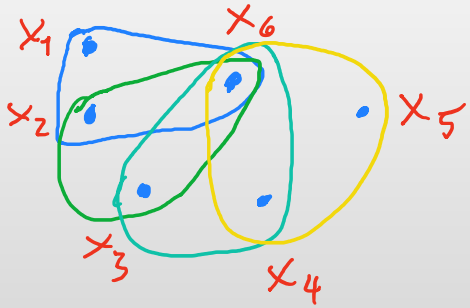


- Consider the case *p=6*
- Consider one variable, say x6, with a few values in its active domain

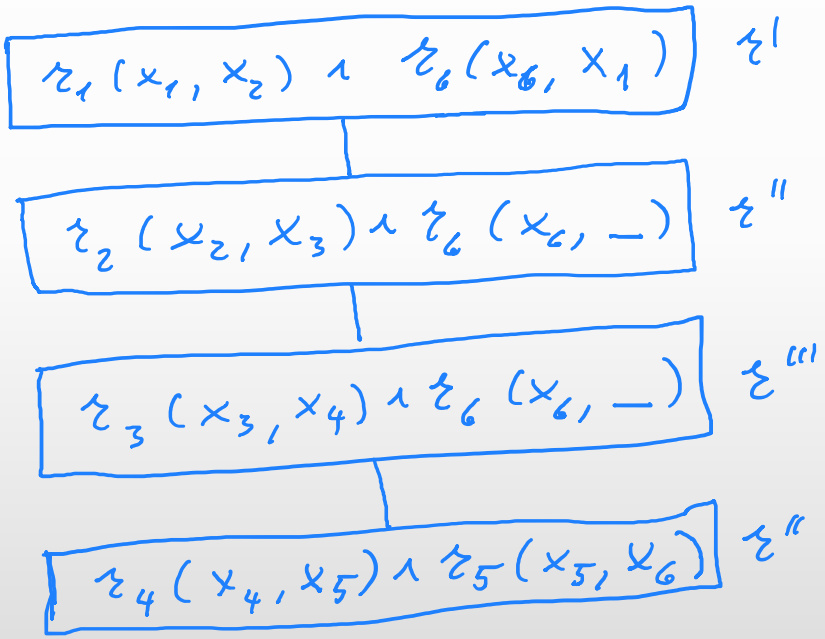
The cost of cycles

$q \leftarrow r_1(x_1, x_2) \wedge r_2(x_2, x_3) \wedge \cdots \wedge r_p(x_p, x_1)$



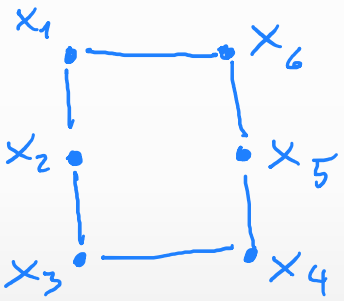


WITH OUR ASSUMPTION ON x_6 THE COST IS ALMOST LINEAR



The cost of cycles

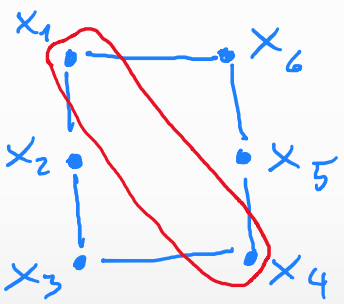
$$q \leftarrow r_1(x_1, x_2) \wedge r_2(x_2, x_3) \wedge \cdots \wedge r_p(x_p, x_1)$$



- Consider the case *p=6*
- Assume now that all variables have many values. In this case, they should be “almost keys”, and a different kind of decomposition should be considered

The cost of cycles

$$q \leftarrow r_1(x_1, x_2) \wedge r_2(x_2, x_3) \wedge \cdots \wedge r_p(x_p, x_1)$$



$$r_1(x_1, x_2) \wedge r_2(x_2, x_3) \wedge r_3(x_3, x_4)$$

$$r_4(x_4, x_5) \wedge r_5(x_5, x_6) \wedge r_6(x_6, x_1)$$

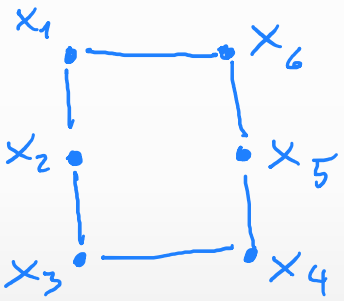
WE HALVE THE CYCLE IN TWO SUBPROBLEMS HAVING AT MOST $\left\lceil \frac{p}{2} \right\rceil$ ATOMS

If we have quasi-keys of maximum degree *d*, the cost of each node is Nd^3

(*d=1* for actual keys)

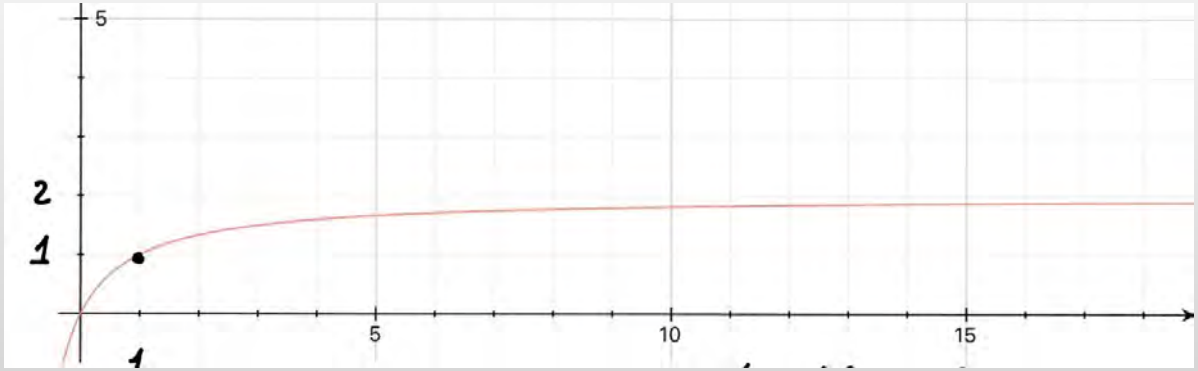
The cost of cycles

$$q \leftarrow r_1(x_1, x_2) \wedge r_2(x_2, x_3) \wedge \cdots \wedge r_p(x_p, x_1)$$



We can show that the following «width» bound holds:

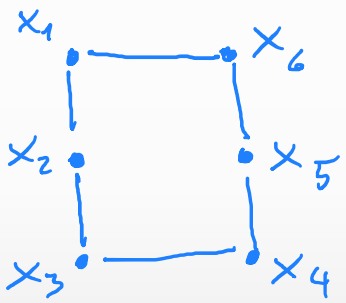
$$2 - \frac{1}{\lceil \frac{p}{2} \rceil}$$



The cost of cycles

$q \leftarrow r_1(x_1, x_2) \wedge r_2(x_2, x_3) \wedge \cdots \wedge r_p(x_p, x_1)$

Assume $|r_i| = N$



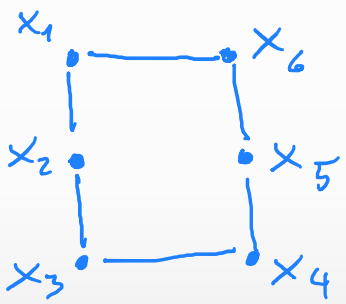
SUBQUADRATIC FOR EVERY CLASS C_p of cycles (having length at most p)

To evaluate such a query, we get a cost of

$$O(p^{1-1/\lceil p/2 \rceil} \cdot N^{2-1/\lceil p/2 \rceil})$$

The cost of cycles

$$q \leftarrow r_1(x_1, x_2) \wedge r_2(x_2, x_3) \wedge \cdots \wedge r_p(x_p, x_1)$$



Assume $|r_i| = N$

THE SUBQUADRATIC BOUND COMES FROM MARX'S SUBMODULAR WIDTH:

- Horizontal fragments
- different decompositions

To evaluate such a query, we get a cost of

$$O(p^{1-1/\lceil p/2 \rceil} \cdot N^{2-1/\lceil p/2 \rceil})$$

Surprising results

- The submodular width [Marx 2013] is based on data-dependent decompositions, but
  - **It provides a structural worst-case bound**

- Our first feeling was that (hypertree) decompositions techniques were practically useful for long and complex queries, but
  - **It turns out that we can get important results even for simple queries, such as the short cycles**

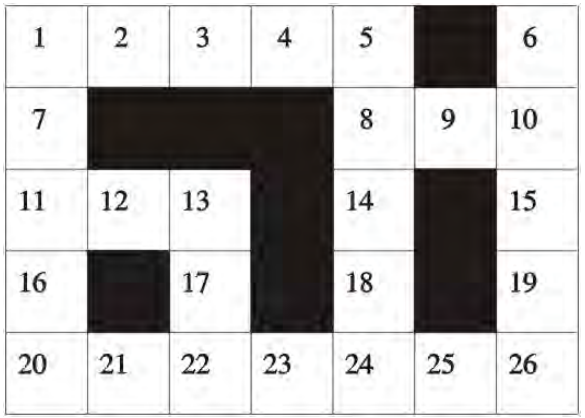
Some hot issues for future research

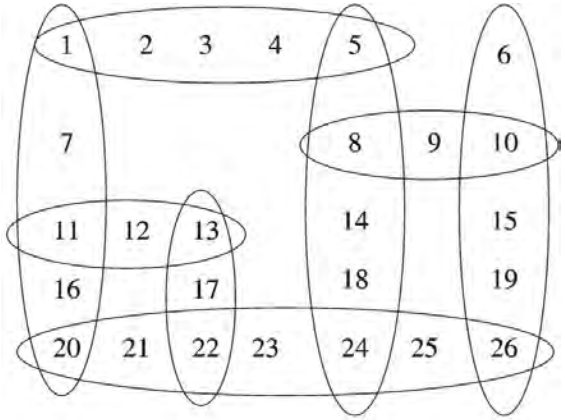
- The challenge: when conjunctive queries are tractable?
- Can minimum cost decompositions be well approximated?
- Better integrate classical query optimization methods with HDs
- Provide efficient FPT algorithms that match the submodular width
  - N.B.: the PANDA algorithm [Khamis, Ngo, Suciu 2017] is not far from that, but the $\tilde{O}$ notation hides a $\log(n)^{|f(q)|}$ factor (it is not FPT)

Appendix

Can HDs be applied outside CQs?

Example of CSP: Crossword Puzzle





1h: PARIS
PANDA
LAURA
ANITA

1v: LIMBO
LINGO
PETRA
PAMPA
PETER

and so on

Constraint satisfaction problems: Renault example(1/2)

- Renault Megane configuration *[Amilhastre, Fargier, Marquis AIJ, 2002]* Used in CSP competitions and as a benchmark problem

- Variables encode type of engine, country, options like air cooling, etc.

- The considered instances consist of about **150 atoms/constraints** and **110 variables**, with database instances where **attributes** have at most **42** distinct values, and the largest constraint relation contains **48721 tuples**.

Constraint satisfaction problems: Renault example (2/2)

- We discovered that the generalized hypertree width is 3 for most instances (with a maximum of 4).
- The total number of solutions is about $2 \cdot 10^{12}$, however this information is not very meaningful, because of the many auxiliary variables occurring in the problem.
- Rather, by using the algorithms based on generalized hypertree decompositions, it is possible to compute the solutions of these instances (or just their number) over the actual variables of interest.
- None of the other available engines that we know, either in the database or in the CSP community, were able to compute such a result for those large instances.

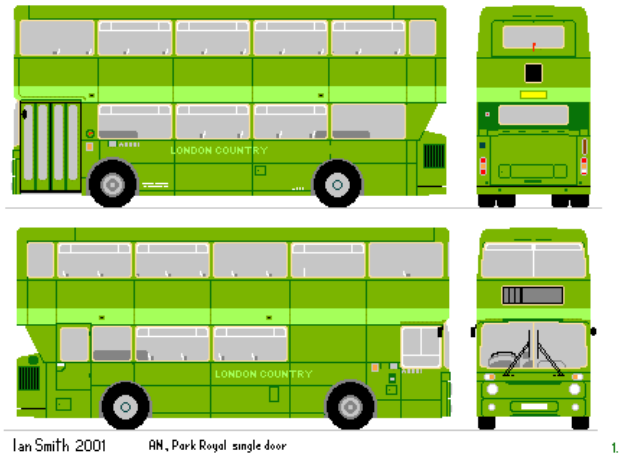
For more information, we refer to the Hypertree Decomposition web-page:
http://www.dimes.unical.it/scarcello/Hypertrees/

Combinatorial Auctions

Bidders can place bids on
Packages of items.
Winner determination: Choose a set of
compatible bids of maximum revenue
or minimum cost.
For classical auctions, winner determination
is obviously tractable. Not so for CAs.

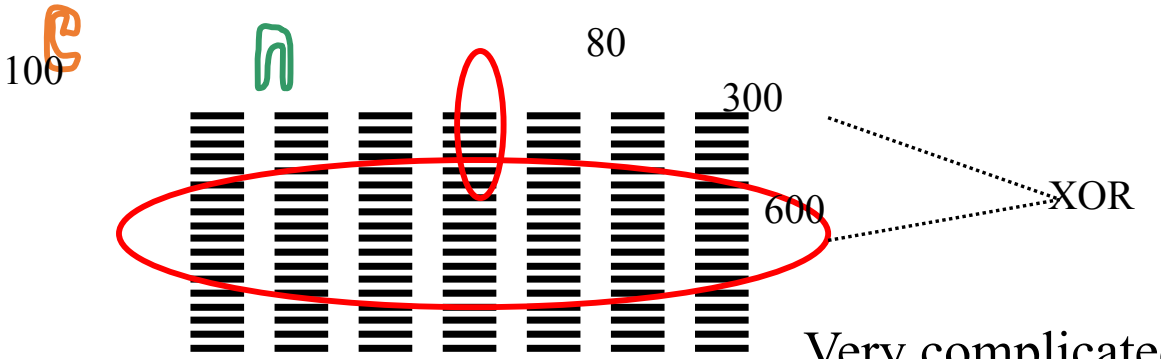
Interesting tractable results based on the hypertree width
of the dual hypergraphs [Gottlob & Greco, JACM 2013]

Applications in different domains



London Regional Transports:
Combinatorial auctions of bus routes.
Private bus companies bid on bundles of routes.

Airport slot auction



Very complicated hypergraphs!