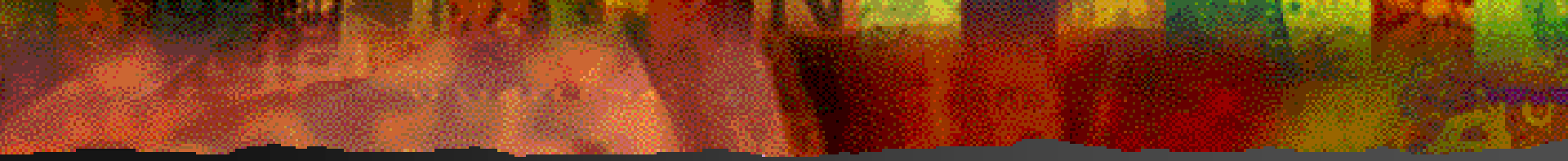


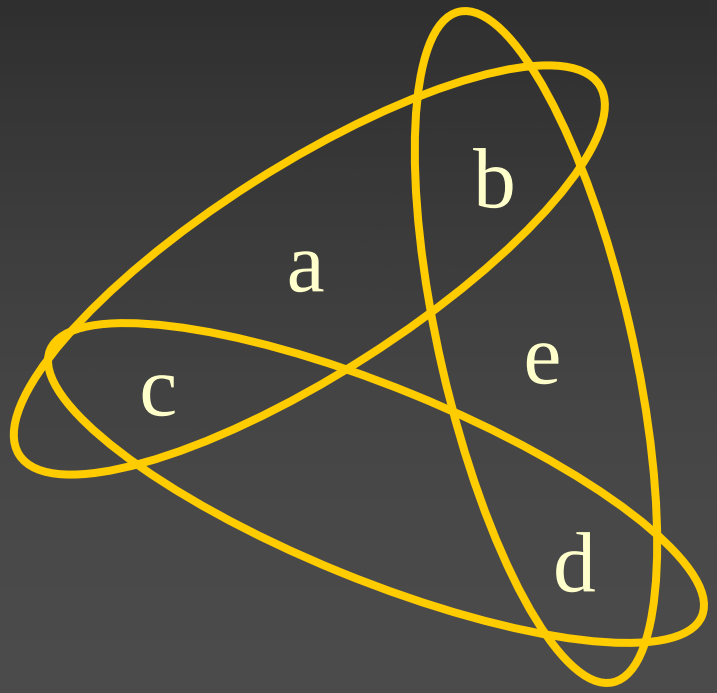
Hypergraph Transversals, Monotone Boolean Formula Dualization, and SAT

Georg Gottlob
University of Oxford & TU Wien

Reports on joint work with Thomas Eiter, Katz Makino, and Enrico Malizia

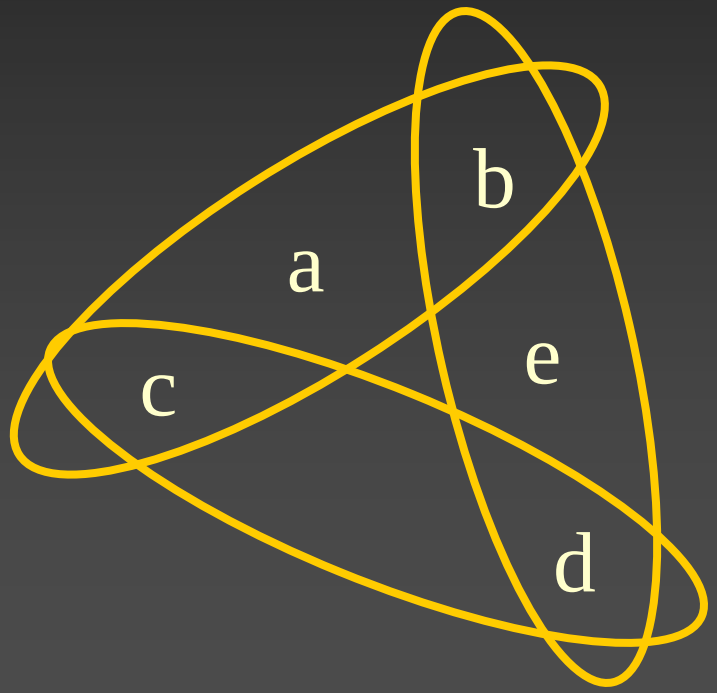
Hypergraphs & Transversals

Hypergraph: H = (Vertices, HyperEdges) = (V(H),E(H)) ≅ E(H)



Hypergraphs & Transversals

Hypergraph: H = (Vertices, HyperEdges) = (V(H), E(H)) ≅ E(H)



TRANSVERSAL = HITTING SET

{d, a, e}

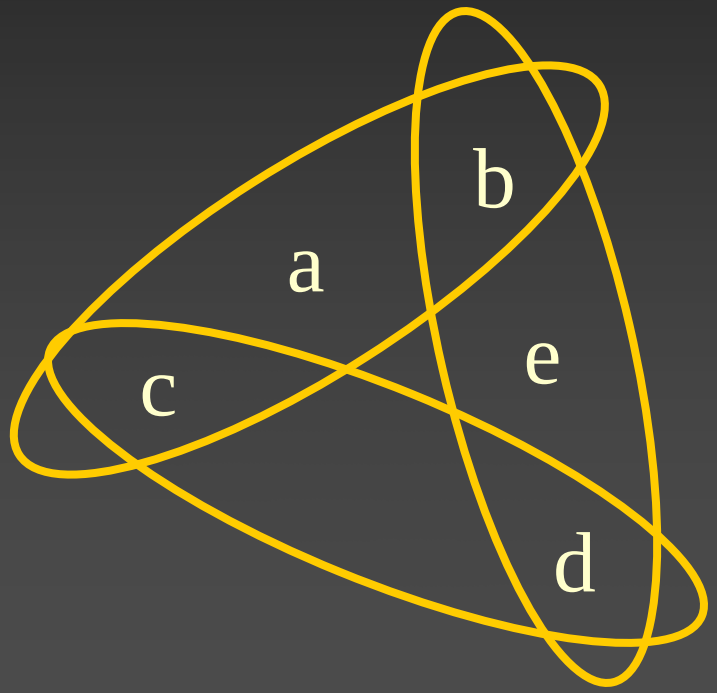
{d, a}

{b,c}

…

Hypergraphs & Transversals

Hypergraph: H = (Vertices, HyperEdges) = (V(H), E(H)) ≅ E(H)



TRANSVERSAL = HITTING SET

{d, a, e}

{d, a}

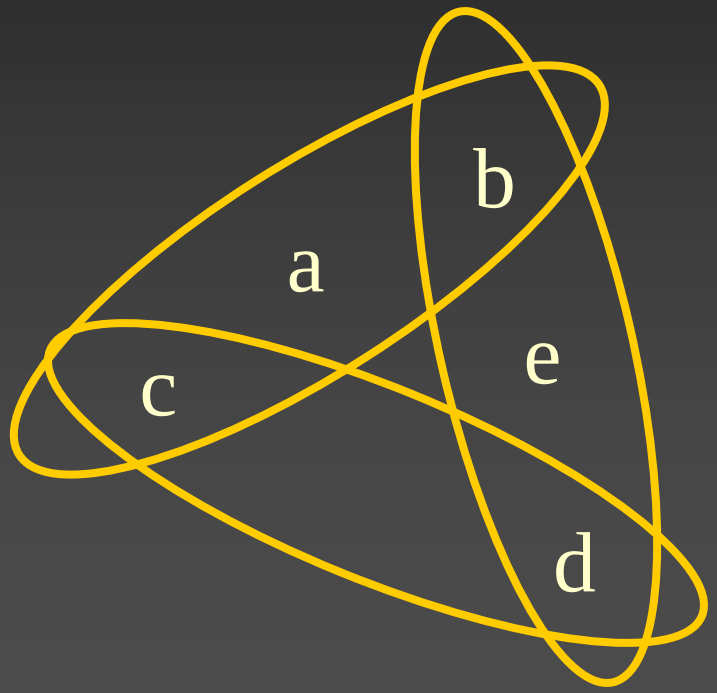
{b,c}

MINIMAL Transversals

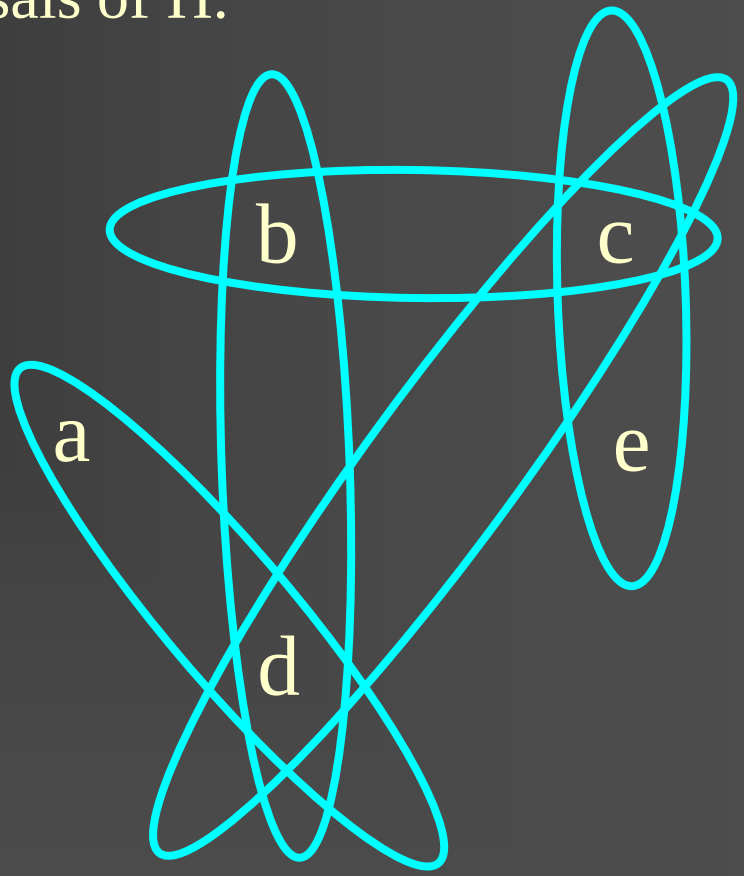
…

Transversal Hypergraph Tr(H)

Tr(H) consists of all minimal transversals of H.



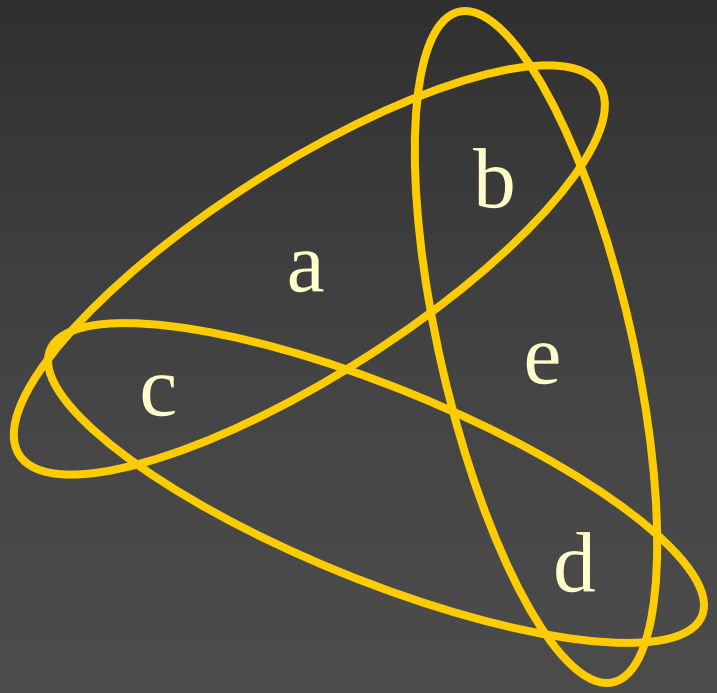
H



G=Tr(H)

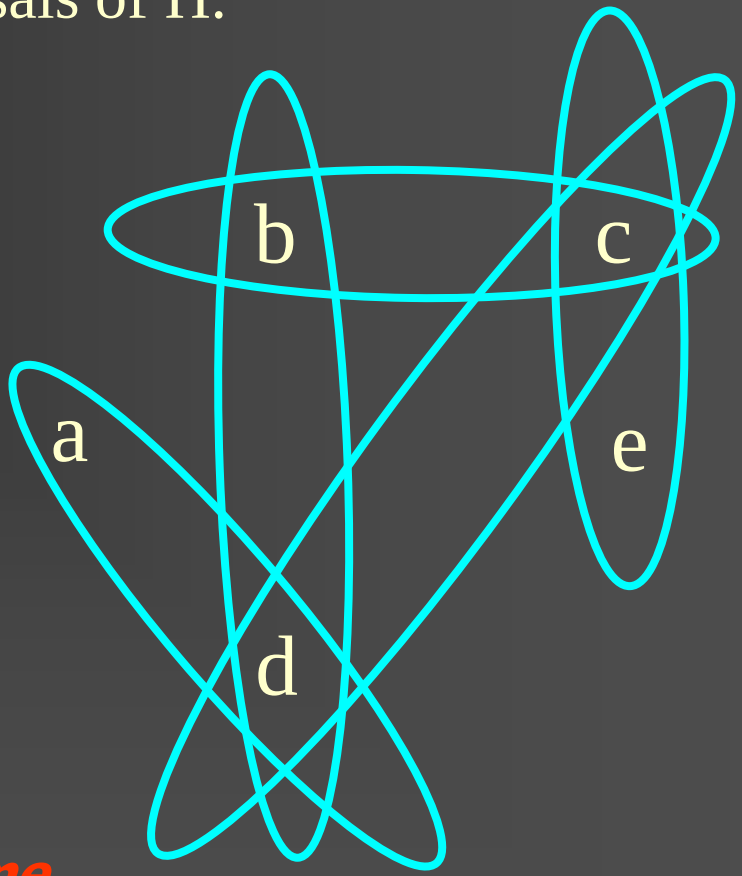
Transversal Hypergraph Tr(H)

Tr(H) consists of all minimal transversals of H.



H

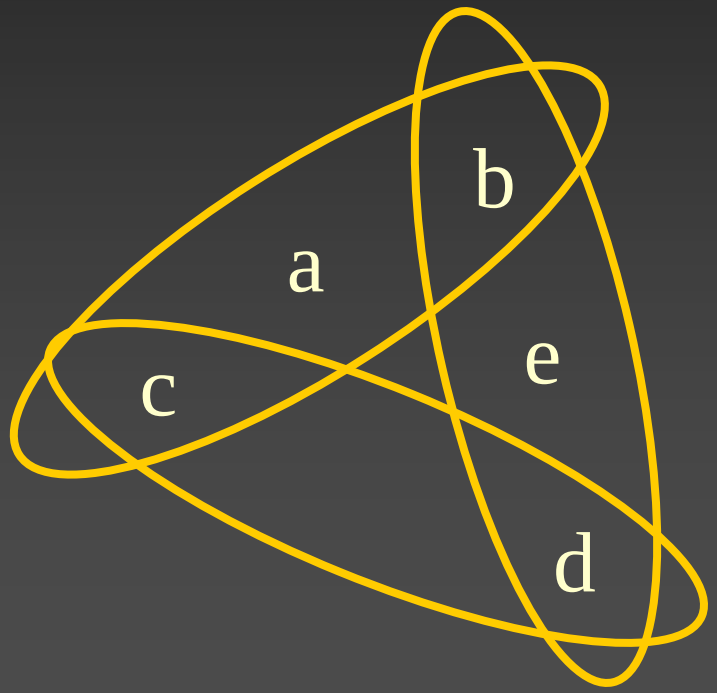
We assume simple hypergraphs!



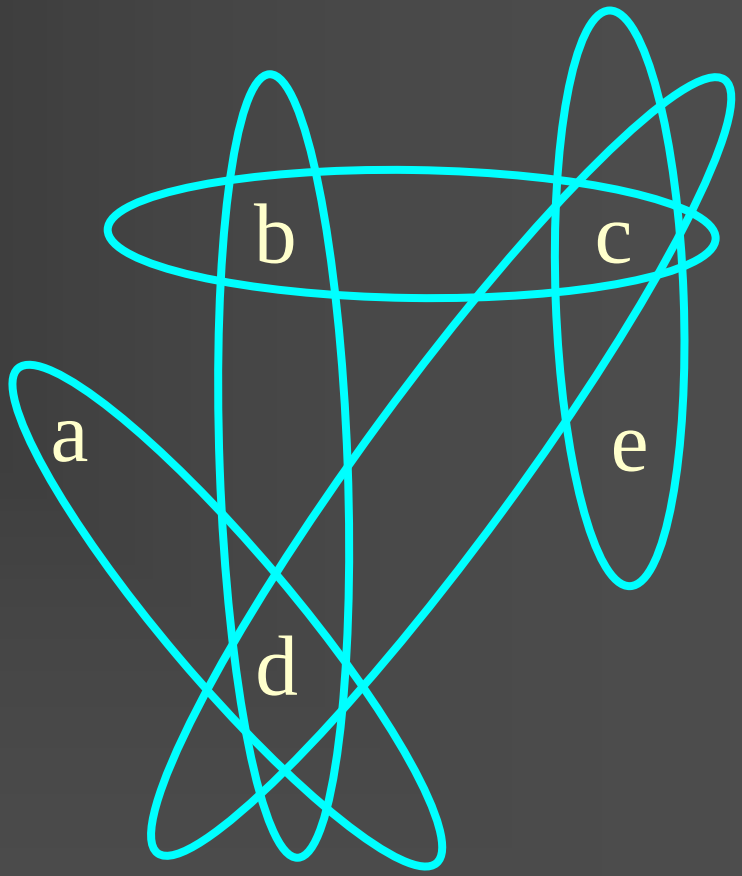
Tr(H)

Transversal Hypergraph: Symmetric Concept

G = Tr(H) ➔ H = Tr(G)

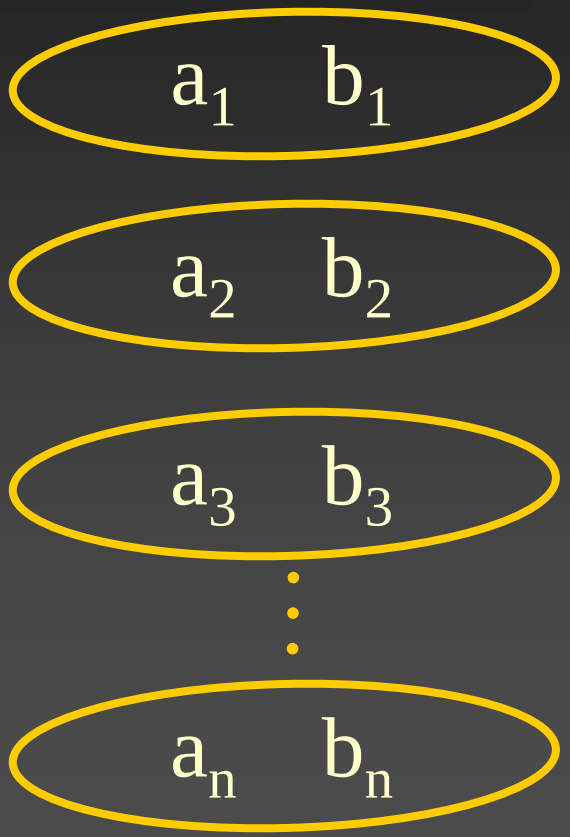


H=Tr(G)



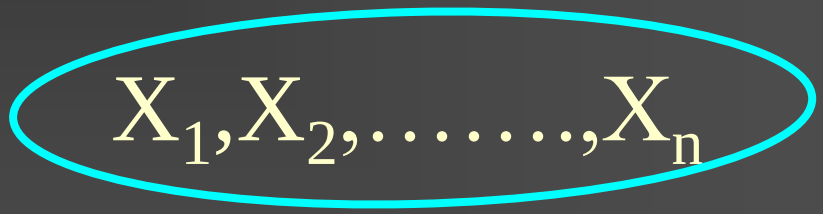
G

Tr(H) may be exponentially large!



H

Tr(H) consists of the 2^n hyperedges



where $X_i = a_i$ or $X_i = b_i$.

Two important problems:

Problem: TRANS-HYP (aka TRANSVERSAL)
Instance: Hypergraphs G, H
Question: Is G = Tr(H) ?

Problem: TRANS-ENUM (Transversal Enumeration)
Instance: Hypergraph H
Output: Tr(H) ?

TRANSVERSAL-complete and TRANS-ENUM-complete problems

Applications

- Satisfiability checking
- Switching theory: Boolean formula dualization
- Data mining
- FD & Armstrong relation generation
- Key detection
- FCA: Pseudo-Intent identification/generation
- Knowledge approximation
- Machine learning
- Quorum determination in distributed DBs
- Model-based diagnosis (Ray Reiter Approach)
- Many others

Applications

- Satisfiability checking
- Switching theory: Boolean formula dualization
- Data mining
- FD & Armstrong relation generation
- Key detection
- FCA: Pseudo-Intent identification/generation
- Knowledge approximation
- Machine learning
- Quorum determination in distributed DBs
- Model-based diagnosis (Ray Reiter Approach)
- Many others

Transversals and Logic

Satisfiability

Monotone Satisfiability MSAT (NP-complete)

$F = (p \vee q \vee r) \wedge (q \vee s \vee t) \wedge (r \vee t) \wedge (\neg q \vee \neg r)$
$\wedge (\neg r \vee \neg s) \wedge (\neg q \vee \neg t) \wedge (\neg p \vee \neg t) \wedge (\neg r \vee \neg t) \wedge (\neg u)$

Transversals and Logic

Satisfiability

Monotone Satisfiability MSAT (NP-complete)

$F = (p \vee q \vee r) \wedge (q \vee s \vee t) \wedge (r \vee t) \wedge (\neg q \vee \neg r)$
$\wedge (\neg r \vee \neg s) \wedge (\neg q \vee \neg t) \wedge (\neg p \vee \neg t) \wedge (\neg r \vee \neg t) \wedge (\neg u)$

LEMMA: MSAT instance F unsatisfiable iff
Tr(positive clauses) $\triangleright$ [negative clauses]

This means that every possible truth value assignment to the positive clauses is contradicted by some negative clause by covering its atoms.

IMSAT: Intersecting MSAT

IMSAT: Monotone Satisfiability MSAT where each positive clause and each negative clause have at least one common atom

$F = (p \vee q \vee r) \wedge (q \vee s \vee t) \wedge (r \vee t) \wedge (\neg q \vee \neg r)$

$\wedge (\neg r \vee \neg s) \wedge (\neg q \vee \neg t) \wedge (\neg p \vee \neg t) \wedge (\neg r \vee \neg t)$

THEOREM: IMSAT instance F unsatisfiable iff

Tr(positive clauses) = [negative clauses]

CORR.: IMSAT is co-TRANS-HYP -complete.

SIMSAT: Symmetric IMSAT

SIMSAT: IMSAT where the negative clauses are obtained from the positive clauses by negating all atoms.

$F = (p \vee q \vee r) \wedge (q \vee s \vee t) \wedge (r \vee t) \wedge$

$(\neg p \vee \neg q \vee \neg r) \wedge (\neg q \vee \neg s \vee \neg t) \wedge (\neg r \vee \neg t)$

THEOREM: SIMSAT is co-TRANS-HYP-complete.

(reduction from SELF-TRANSVERSALITY)

Duality of monotone CNFs

DUAL: Given two monotone prime CNFs F,G
are F and G dual?

$F = (p \vee q \vee r) \wedge (q \vee s \vee t) \wedge (r \vee t)$

$G = (r \vee s) \wedge (q \vee t) \wedge (p \vee t) \wedge (r \vee t)$

F, G dual: dual output on dual truth value assignments.

THEOREM: DUAL is TRANS-HYP-complete,
so is SELF-DUALITY.

DUALIZATION: Compute the dual.

Important in circuit design, etc.

Monotone CNF/DNF equivalence

MCNF/MDNF: Given a monotone prime CNF F and a monotone prime DNF G, is $F \equiv G$?

$F = (p \vee q \vee r) \wedge (q \vee s \vee t) \wedge (r \vee t)$

$G = (r \wedge s) \vee (q \wedge t) \vee (p \wedge t) \vee (r \wedge t)$

THEOREM: MCNF/MDNF TRANS-HYP-complete.

DNF-COMPUTATION: Compute the prime DNF from monotone prime CNF F

Transversals in AI

Theory Revision

Knowledge in Flux

Principle of Minimal Change

Syntactic approach: “Set of Theories Method”

[Fagin&Ullman&Vardi 83; Ginsberg 86]

Theory T, new knowledge p, (inconsistent with T), incorporate p into T.

Revised theory: $T \circ p$

T: set of sentences (propositional formulas)

p: propositional update formula

/

$$W(T;p) := \{T' \subseteq T \mid T' \models \neg p \text{ , } T' \text{ maximal } \}$$

$$T \diamond p := \text{(disjunction of all theories in } W(T;p)) \wedge p$$

Simple Example:

$T = \{a, b, a \wedge b \Rightarrow c\}$ $\quad\quad$ $p = \neg c$

$W(T;p) = \{ \{a, b\}, \{a, a \wedge b \Rightarrow c\}, \{b, a \wedge b \Rightarrow c\} \}$

$$T \circ p = [(a \wedge b) \vee (a \wedge (a \wedge b \Rightarrow c)) \vee (b \wedge (a \wedge b \Rightarrow c))] \wedge \neg c$$
$$\Leftrightarrow [(a \wedge b) \vee (a \wedge (b \Rightarrow c)) \vee (b \wedge (a \Rightarrow c))] \wedge \neg c$$
$$\Leftrightarrow (a \vee b) \wedge \neg c$$

THEOREM: For Horn Theories T, computing W(T,p) from T is TRANS-ENUM-Complete

[Gogic & Papadimitriou & Sideri 98]

Data Mining

	CREAM	SUGAR	MILK	HONEY	COFFEE
Client 1	1	0	0	0	1
Client 2	0	0	1	1	0
Client 3	1	1	1	0	1
Client 4	1	1	0	1	1
Client 5	1	1	0	1	0
Client 6	0	0	1	0	1

N=6 rows, Frequency threshold = 0.5, infrequency treshold: 1/6

Maximal frequent sets: {CREAM,COFFEE}, {CREAM, SUGAR}

Minimal infrequent sets: {CREAM, MILK}, {SUGAR,MILK}

Complexity of Data Mining

THEOREM: Additional maximal frequent set (minimal unfrequent set) is co-TRANS-HYP-complete.

THEOREM: Computing maximal frequent sets (minimal unfrequent sets) is TRANS-ENUM-complete.

[Boros,Gurvich,Khachiyan,Makino STACS'02]

Association Rule Discovery, Functional Dependency generation

PART#	NAME	STORE	QUANTITY
13	bolt	A	250
14	hammer	A	30
16	bolt	A	35
13	bolt	B	233
14	hammer	B	20
16	bolt	B	35
13	bolt	C	14
14	hammer	C	250
16	bolt	C	31

Functional Dependencies:

PART# → NAME

PART#,STORE→QUANTITY

Complexity of FD generation

THEOREM: Given a relation R and a set of FDs F, determining if F characterizes R i.e., if F^+=FD(R), is TRANS-HYP hard.

[G.&Libkin, 89]

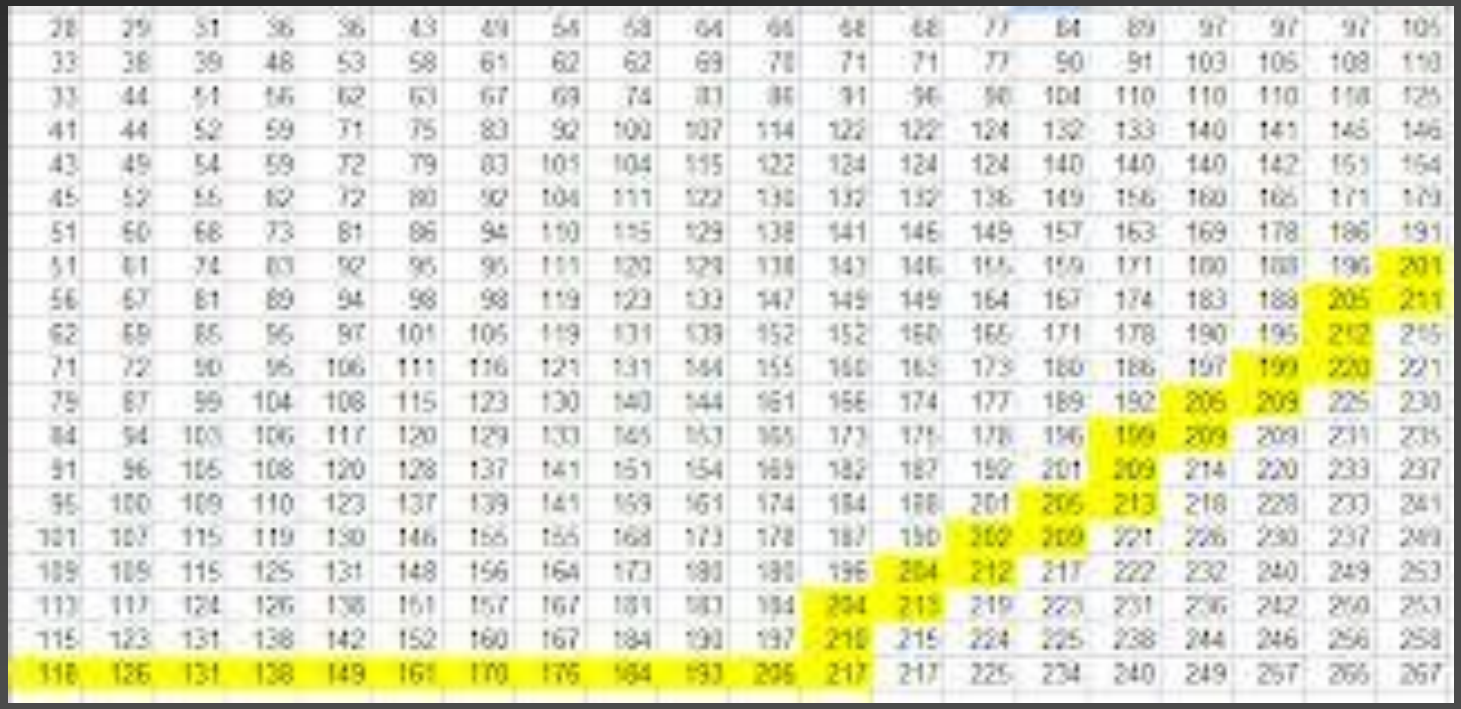
The precise complexity is open!

THEOREM: IF each FD in F has a key as LHS, then the problem is TRANS-HYP-complete.

[Eiter&G., 95]

THEOREM: Generating FD-covers or Armstrong Relations is TRANS-ENUM-hard/cplt..

Analyze large data tables: Generate all keys



Key Identification/ Generation

THEOREM: For relation R and a set of keys, deciding whether there is an additional key is co-TRANS-HYP-complete.

THEOREM: Generating the set of all keys for a relation is TRANS-ENUM-complete.

[Eiter&G., 95]

Machine Learning

THEOREM:

Learning a monotone Boolean function f on n variables by determining $DNF(f)$ and $CNF(f)$ using membership queries is feasible in polynomial time in $n+|DNF(f)|+|CNF(f)|$ iff TRANS-HYP $\in$ PTIME

[Gunopulos, Khardon, Mannila Toivonen PODS'97]

[Domingo, Mishra, Pitt ML'99]

Knowledge Approximation

Given a propositional theory T,

<u>Horn Envelope of T :</u> The Horn theory E(T) such that

$$T \models E(T),$$

and for no Horn theory $T' \not\equiv T$, $T \models T' \models E(T)$.

E(T) unique!

Thus: E(T): strongest Horn Theory approximating T

Efficient sound Reasoning with E(T):

$E(T) \models \varphi \rightarrow T \models \varphi$ Knowledge Compilation

Simple Example:

$T = (a \wedge b \wedge c) \vee (a \wedge \neg b \wedge \neg c) \vee (\neg a \wedge b \wedge c)$

(Note: here T is given in DNF as disjunction of its models)

$E(T) = (b \Rightarrow c) \wedge (c \Rightarrow b)$

Models of E(T):

$E(T) \equiv (a \wedge b \wedge c) \vee$
$(a \wedge \neg b \wedge \neg c) \vee$
$(\neg a \wedge b \wedge c) \vee$
$(\neg a \wedge \neg b \wedge \neg c)$

THEOREM: The problem of computing a Horn envelope from the models of T is TRANS-ENUM-complete.

[Kavvadias, Papadimitriou & Sideri 93]

Structural Complexity -- Progress

THEOREM [Fredman, Khachiyan 96]

TRANS-HYP is in DTIME($n^{o(\log n)}$)

Observation: $G=Tr(H) \rightarrow \exists\, S \in G \cup H : |S| \leq \log n$

It follows that some vertex a of the small edge S occurs rather frequently in the other hypergraph!

Use this vertex for a smart *divide-and-conquer* algorithm.

Input volume: $v = |G| * |H|$

Complexity? Open for many years!

Problem: TRANSVERSAL
Instance: Hypergraphs G, H
Question: Is G = Tr(H) ?

Obviously in co-NP

Complexity? Open for many years!

Problem: TRANS-HYP
Instance: Hypergraphs G, H
Question: Is G = Tr(H) ?

Obviously in co-NP

Problem: TRANS-ENUM (Transversal Enumeration)
Instance: Hypergraph H
Output: Tr(H) ?

TRANSVERSAL $\in$ PTIME $\Leftrightarrow$ TRANS-ENUM $\in$ Total-PTIME

[Bioch & Ibaraki 95]

Complexity of TRANS-HYP

Problem: TRANS-HYP
Instance: Hypergraphs G, H
Question: Is G = Tr(H) ?

TRANS-HYP obviously in co-NP.

TRANS-HYP in **DTIME**($n^{\log n}$) [Fredman&Khachiyan 96]

Interesting polynomial subclasses are known
(Some will be discussed here)

Complexity Results

Concrete: Tractable Cases

[Eiter & G., SIAM J. Comput., 24(6), 1995]
[Eiter, G., Makino STOC'02;
full paper: SIAM J. Comput. 32(2): 2003]

Structural: Limited nondeterminism

[G., Malizia LICS'14];
Full paper: SIAM J. Comput. 47(2) (2018)]

New tractable cases / Algorithm

Let $H = \{E_1, E_2, \ldots, E_m\}$

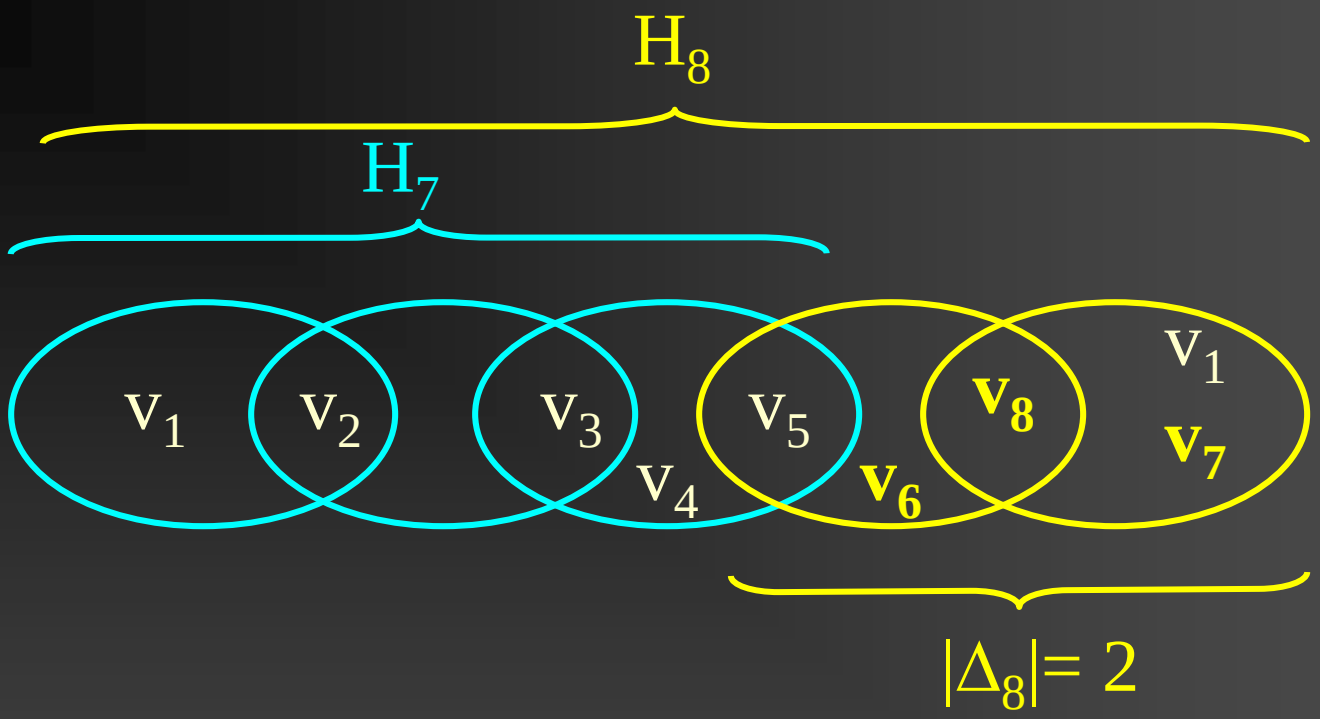
Fix an ordering $<$ on the vertices of a hypergraph H:

$$v_1 < v_2 < v_3 < \ldots < v_n$$

Define $H_i = \{E \in H \mid E \subseteq \{v_1, v_2, v_3, \ldots, v_i\}\}$

Note that $H_0 = \{\}$ and $H_n = H$

Define $\Delta_i = H_i - H_{i-1}$



Δ_i: hyperedges contained in H_i but not in H_{i-1}

Each transversal of H_i can be extended to one of H_{i+1}

Moreover, $tr(H_{i+1}) \subseteq [\ tr(H_i) + tr(\Delta_{i+1})\ ]$

→For small Δ, polytime incremental transversal computation

k-degenerate hypergraphs

Hypergraph $H = (V,E)$, is k-degenerate

if there exists an ordering "<" on V such that

$\forall\ i \leq |V|,\ |\Delta_i| \leq k.$

k-degeneracy can be tested in polynomial time.

An appropriate ordering < can be computed in polynomial time.

THEOREM: The transversals of a k-degenerate hypergraph can be computed in incremental polynomial time.

COROLL.: TRANS-HYP is in PTIME for all classes of instances (H,G) where H or G is k-degenerate.

New tractable classes

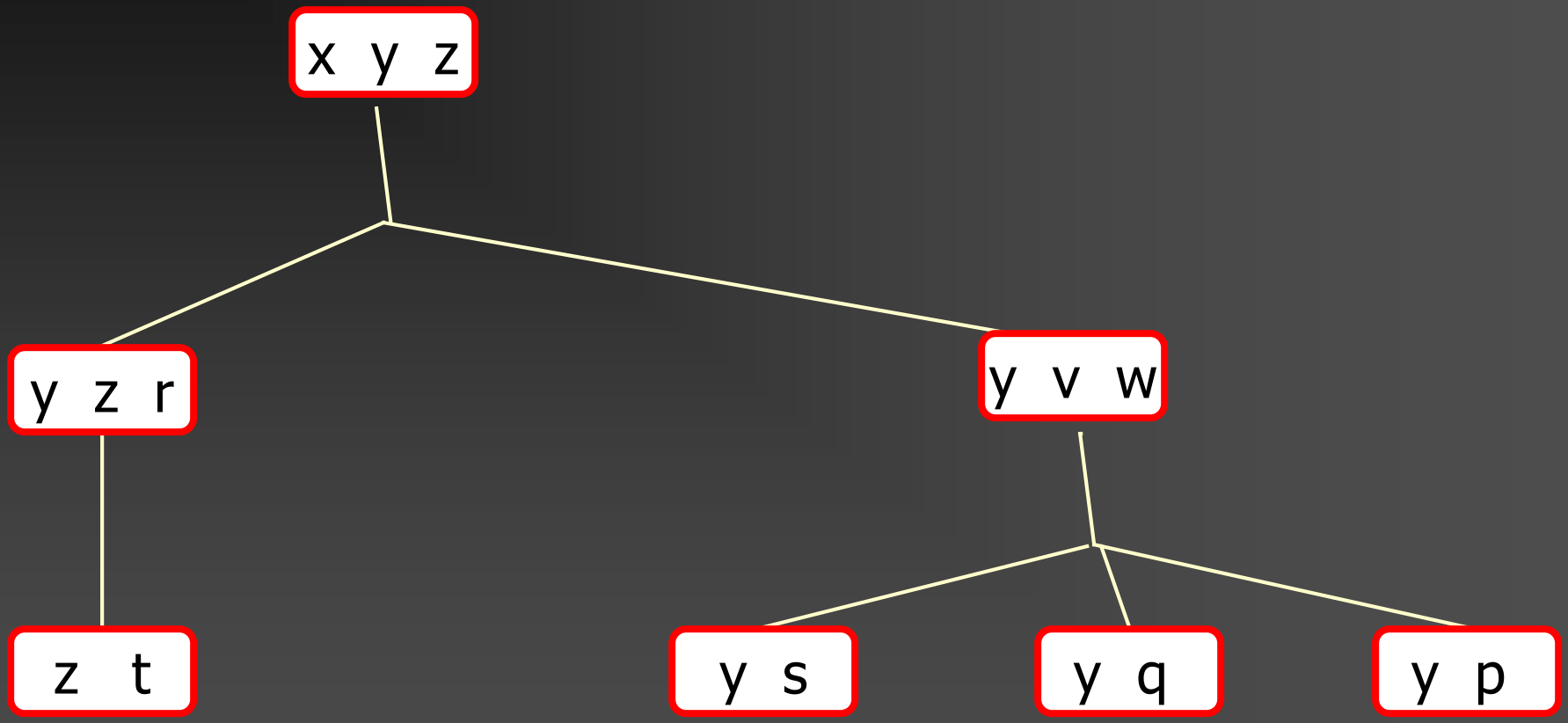
THEOREM

HYP-TRANS and TRANS-ENUM are solvable in PTIME for the following hypergraph classes:

- Read-k hypergraphs (k-degenerate)
- α-acyclic hypergraphs (1-degenerate)
- Hypergraphs of treewidth k (2^k-degenerate)

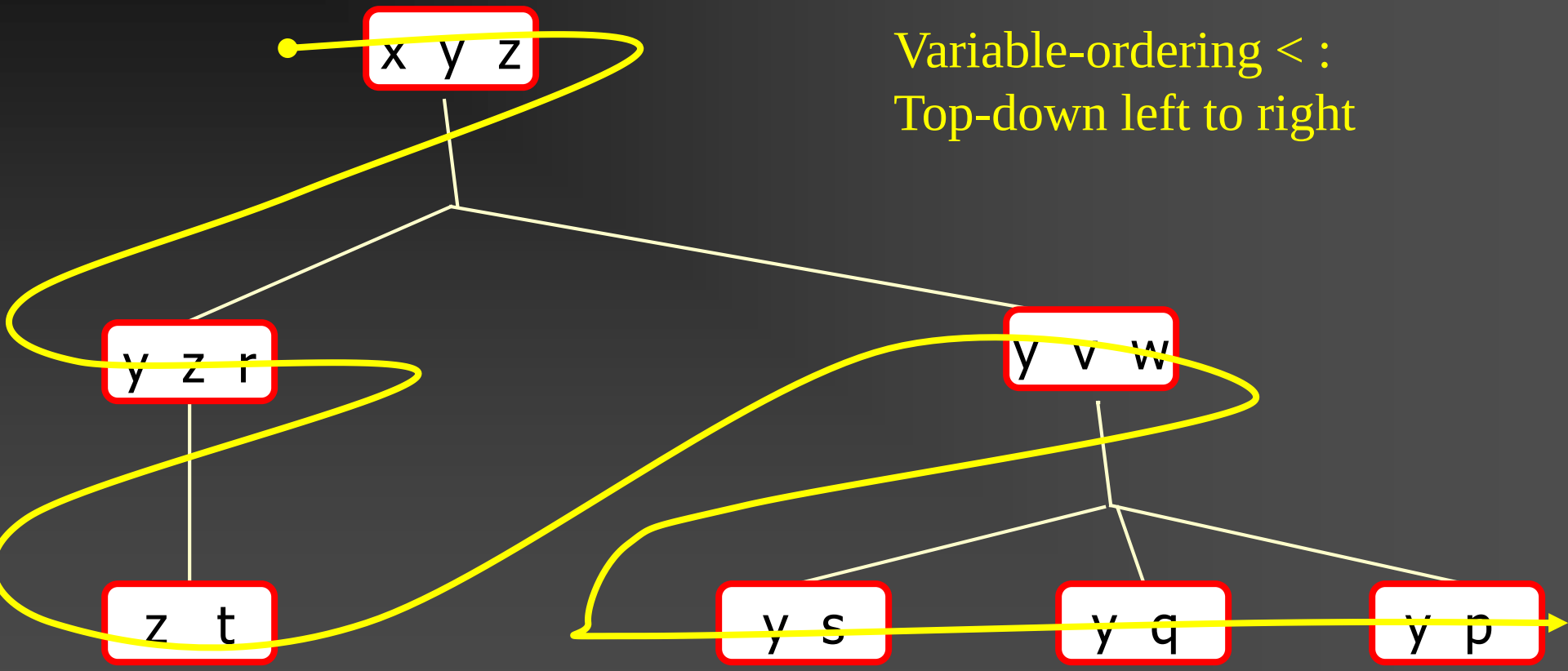
Treewidth of H : treewidth of incidence graph of H.

Acyclic hypergraph: Join Tree



We assume w.l.o.g. a simple hypergraph.

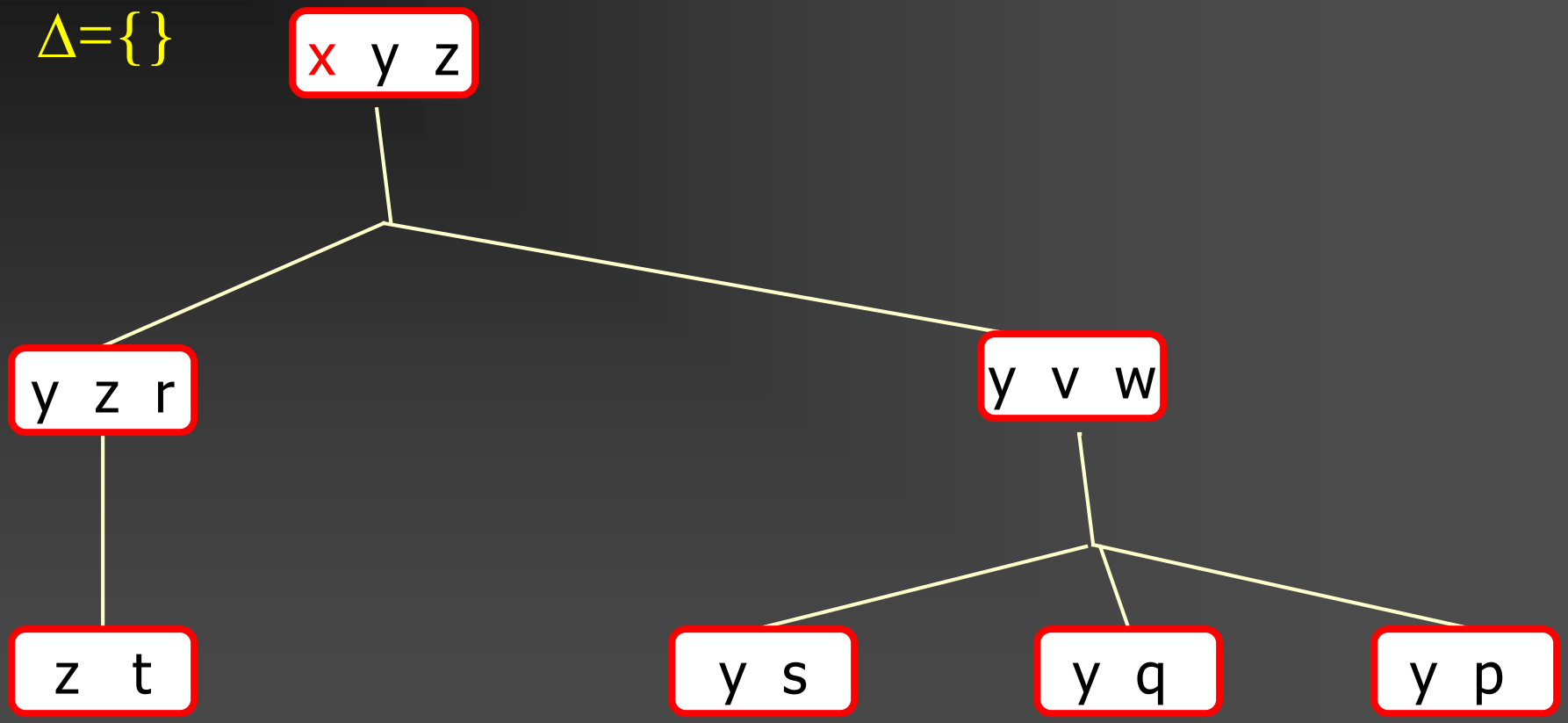
Acyclic hypergraph: Join Tree



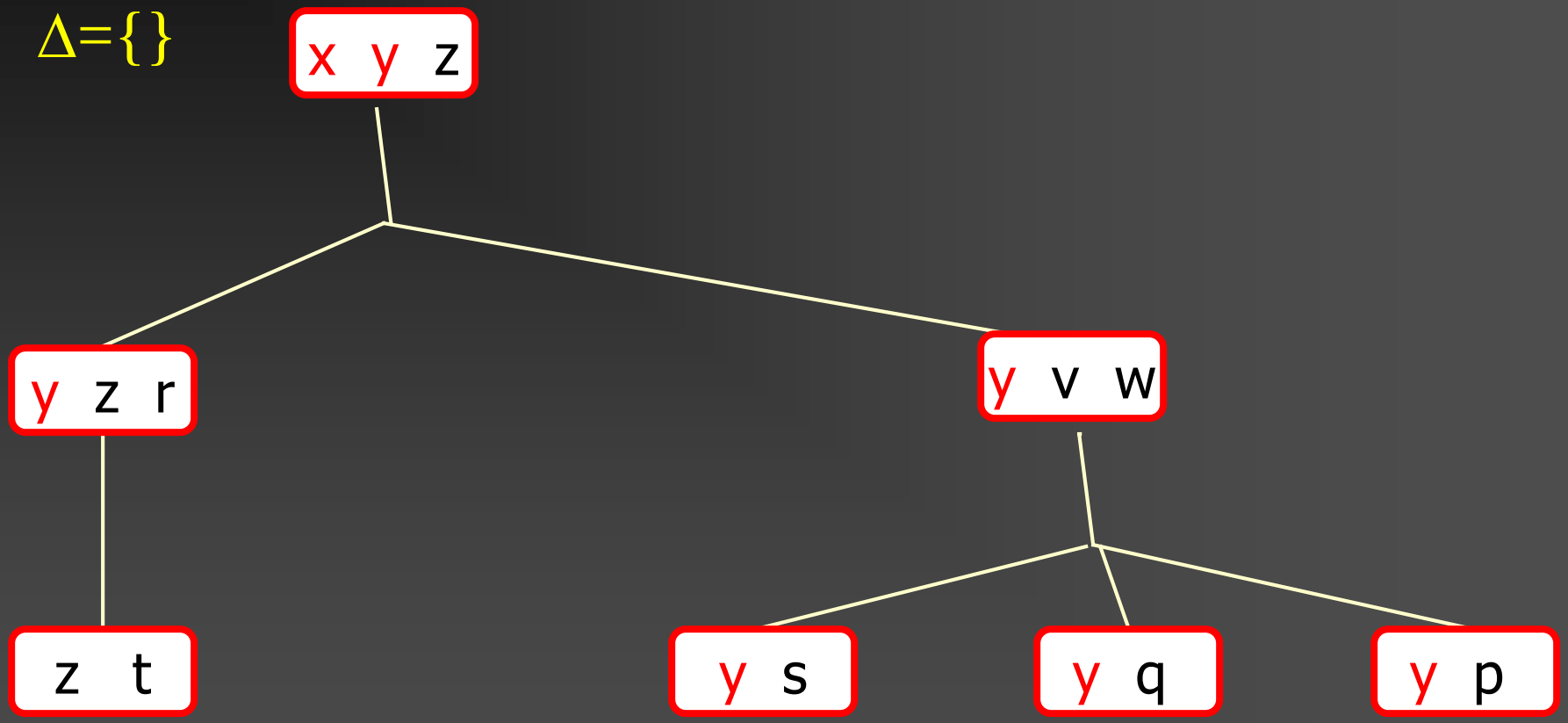
Variable-ordering < :
Top-down left to right

We assume w.l.o.g. a simple hypergraph.

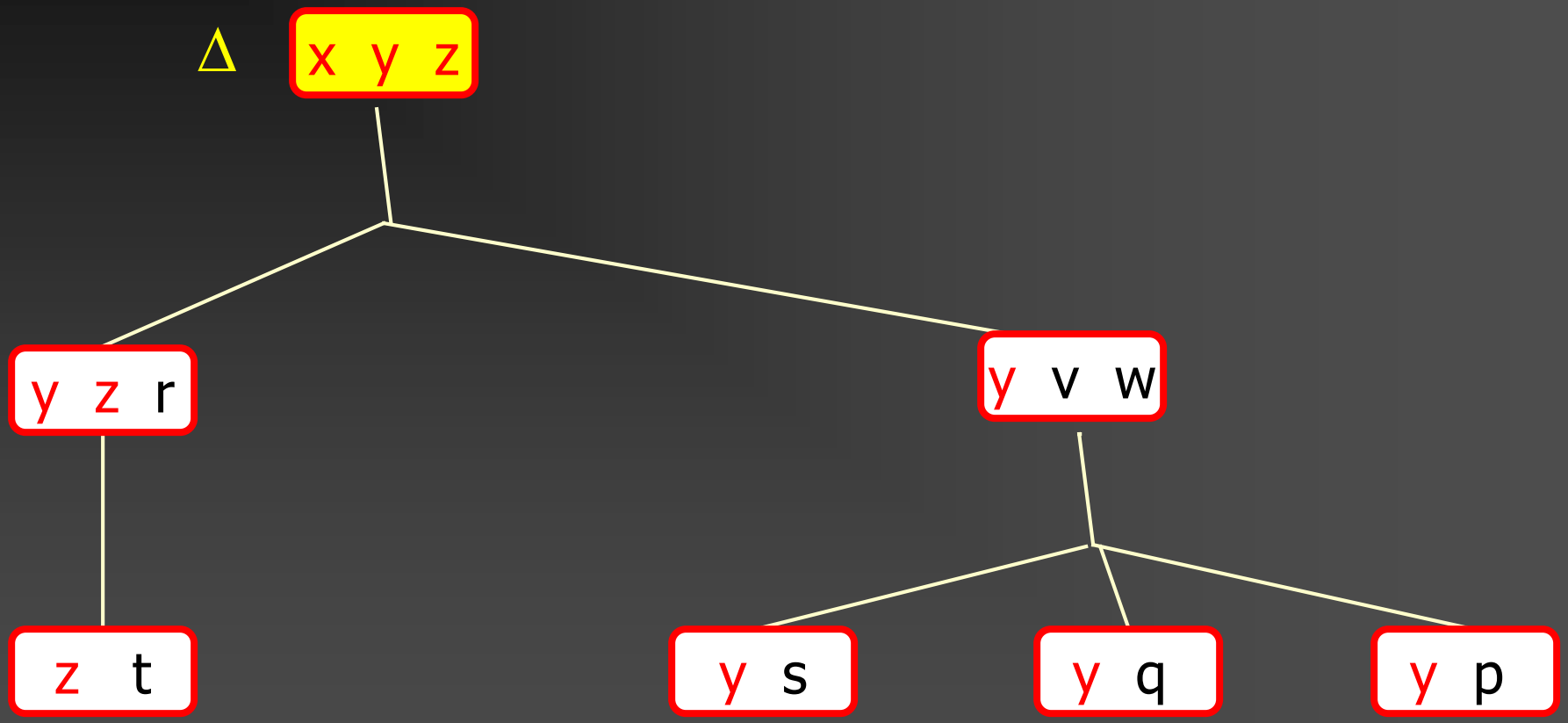
Acyclic hypergraph: Join Tree



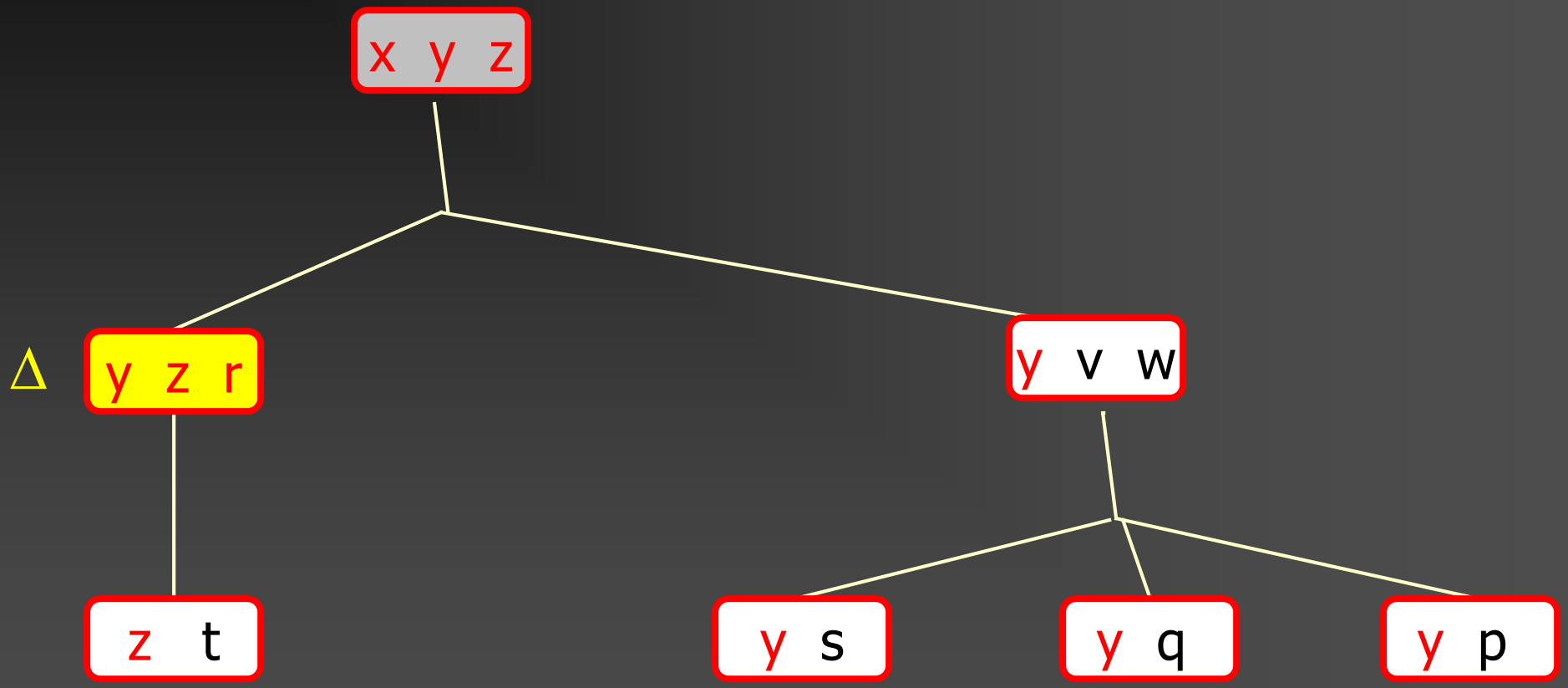
Acyclic hypergraph: Join Tree



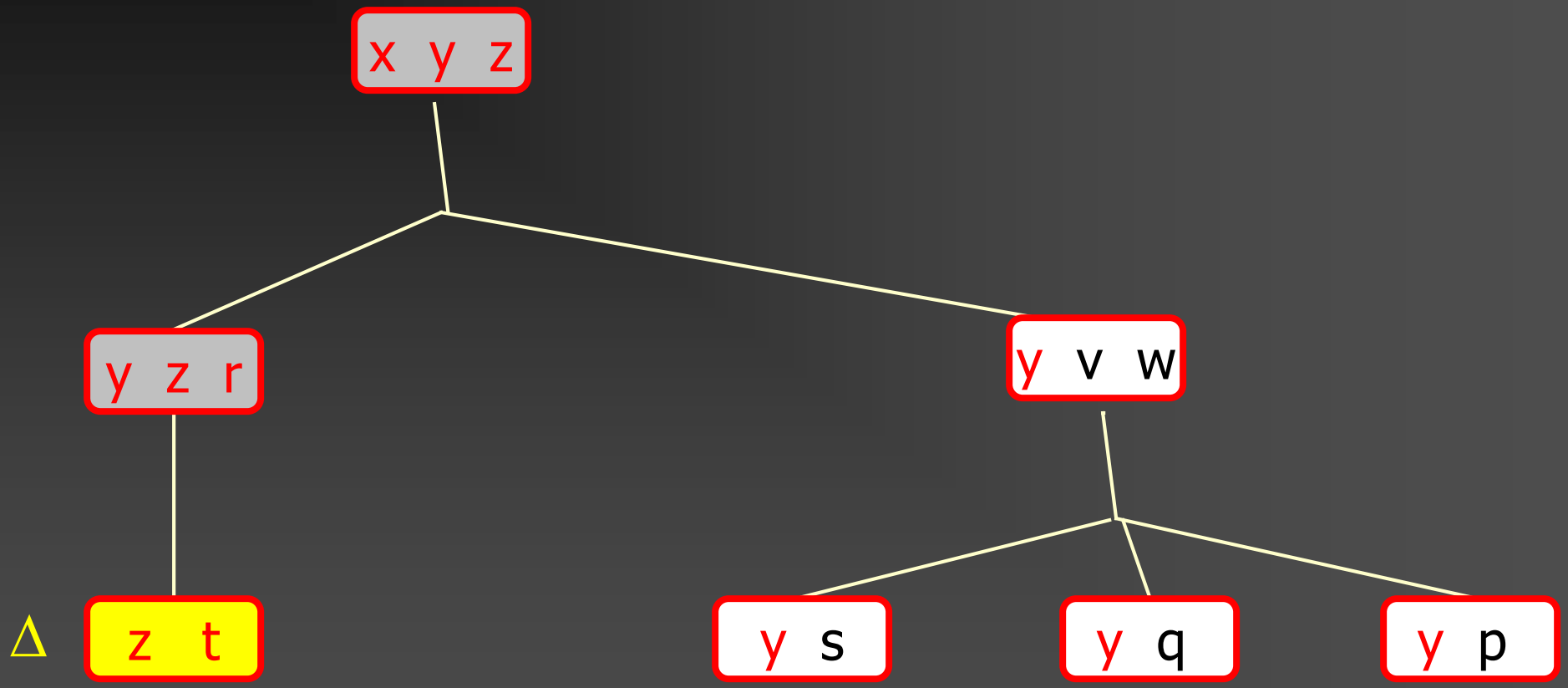
Acyclic hypergraph: Join Tree



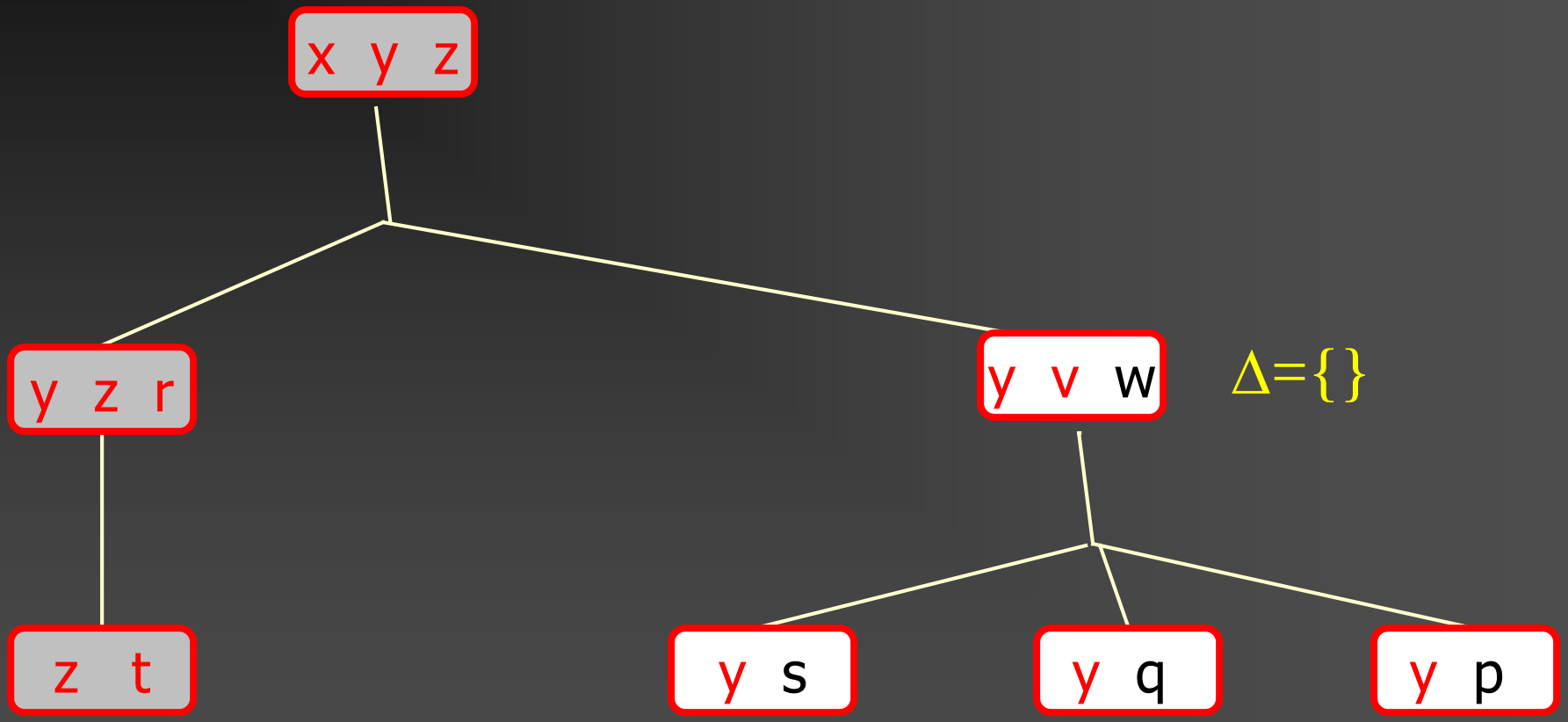
Acyclic hypergraph: Join Tree



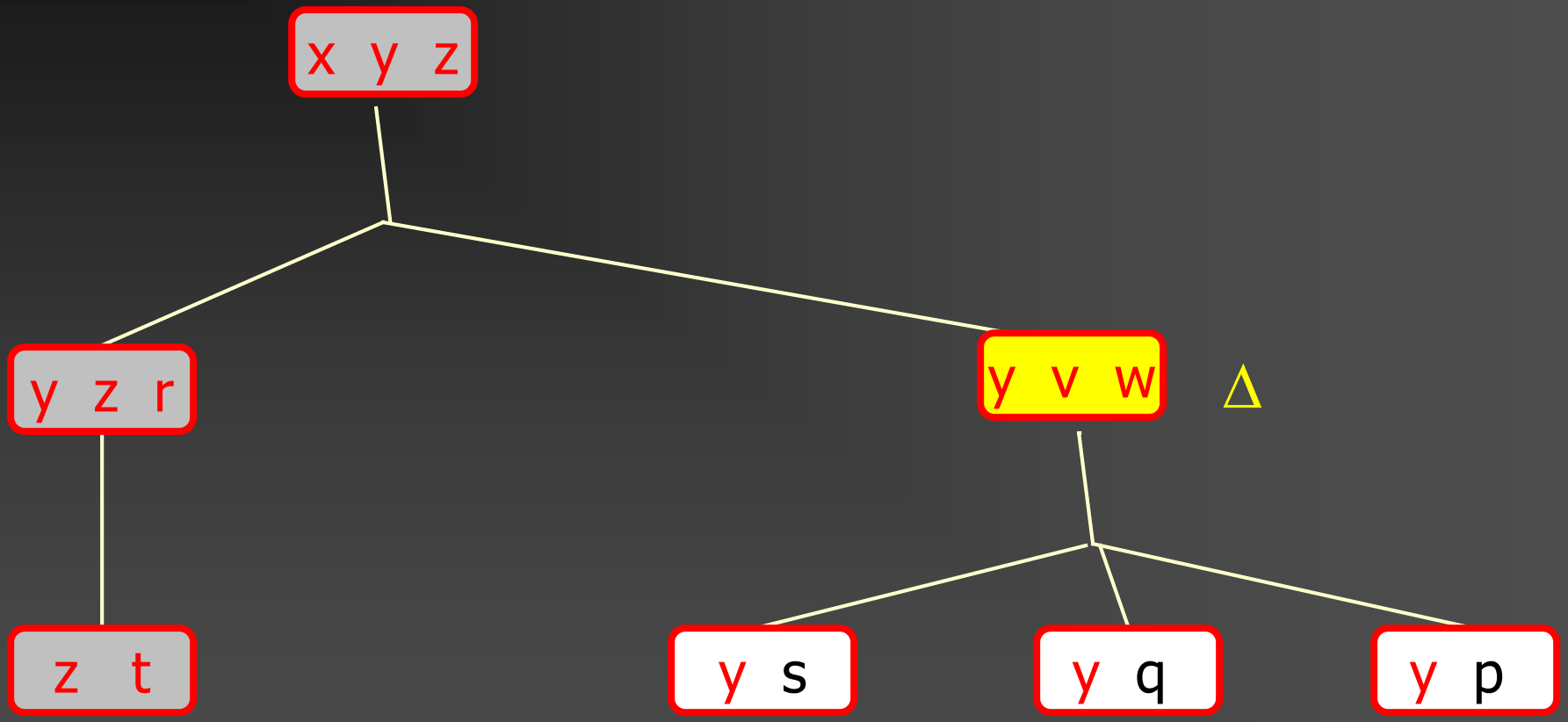
Acyclic hypergraph: Join Tree



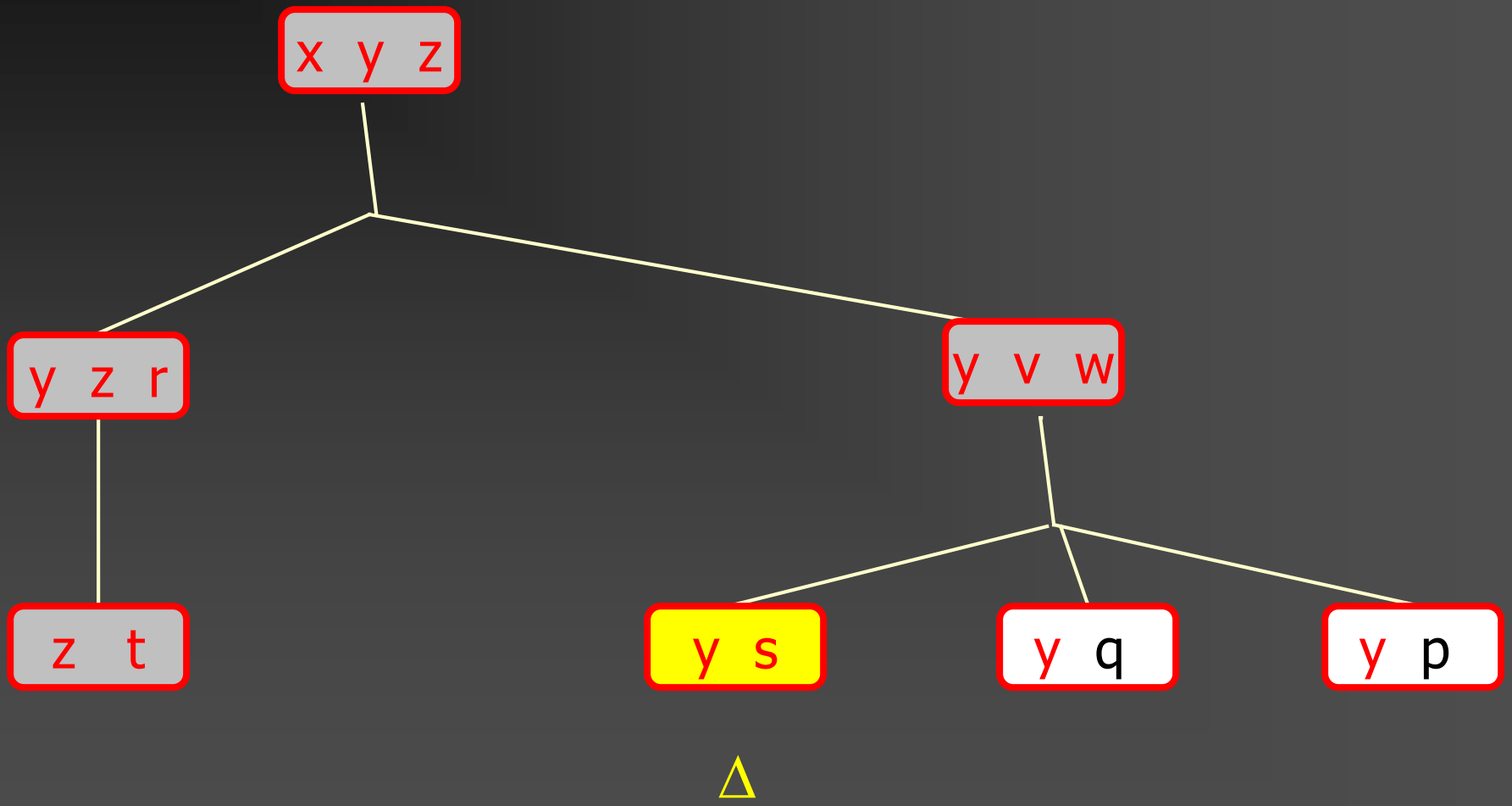
Acyclic hypergraph: Join Tree



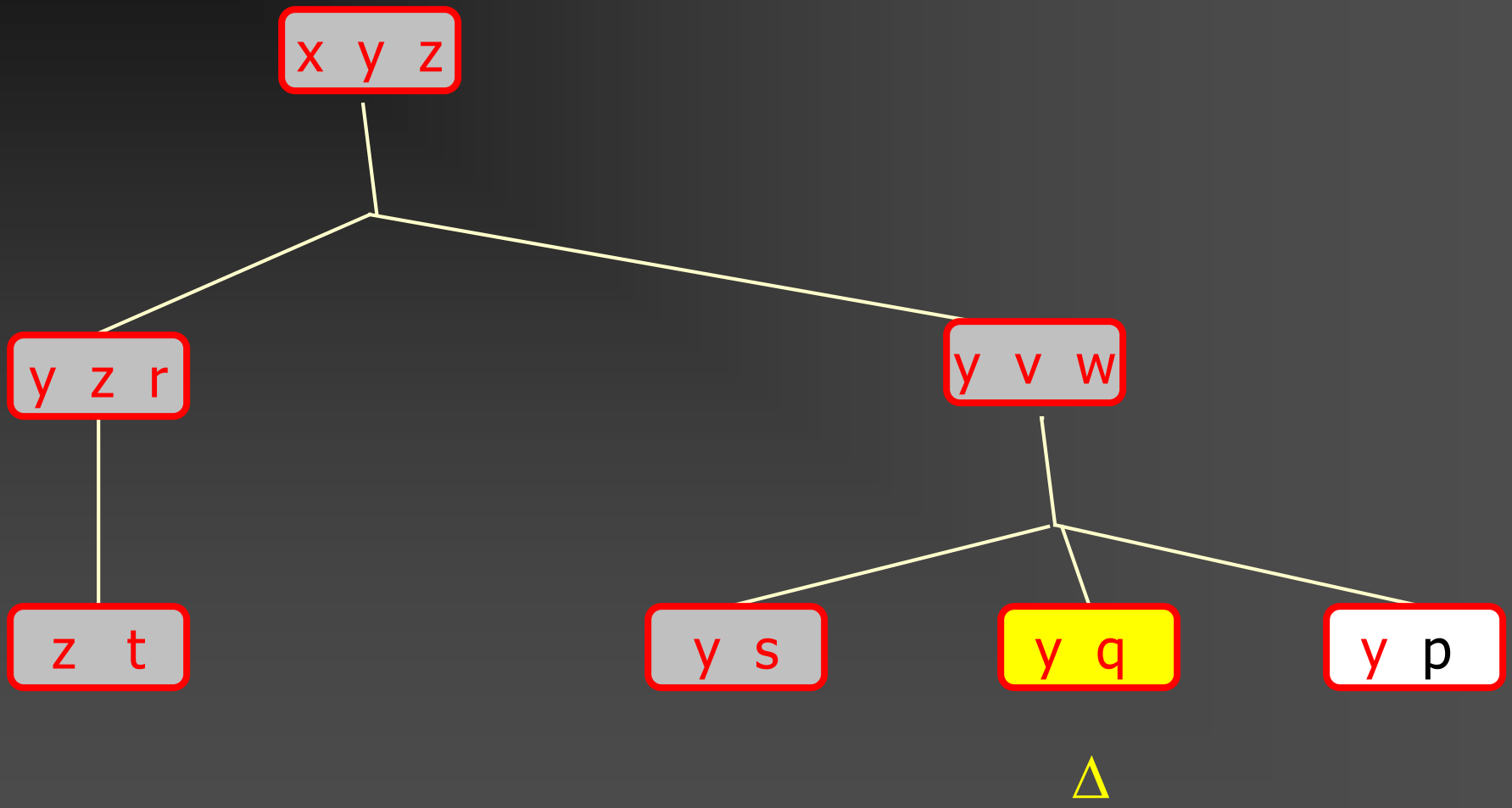
Acyclic hypergraph: Join Tree



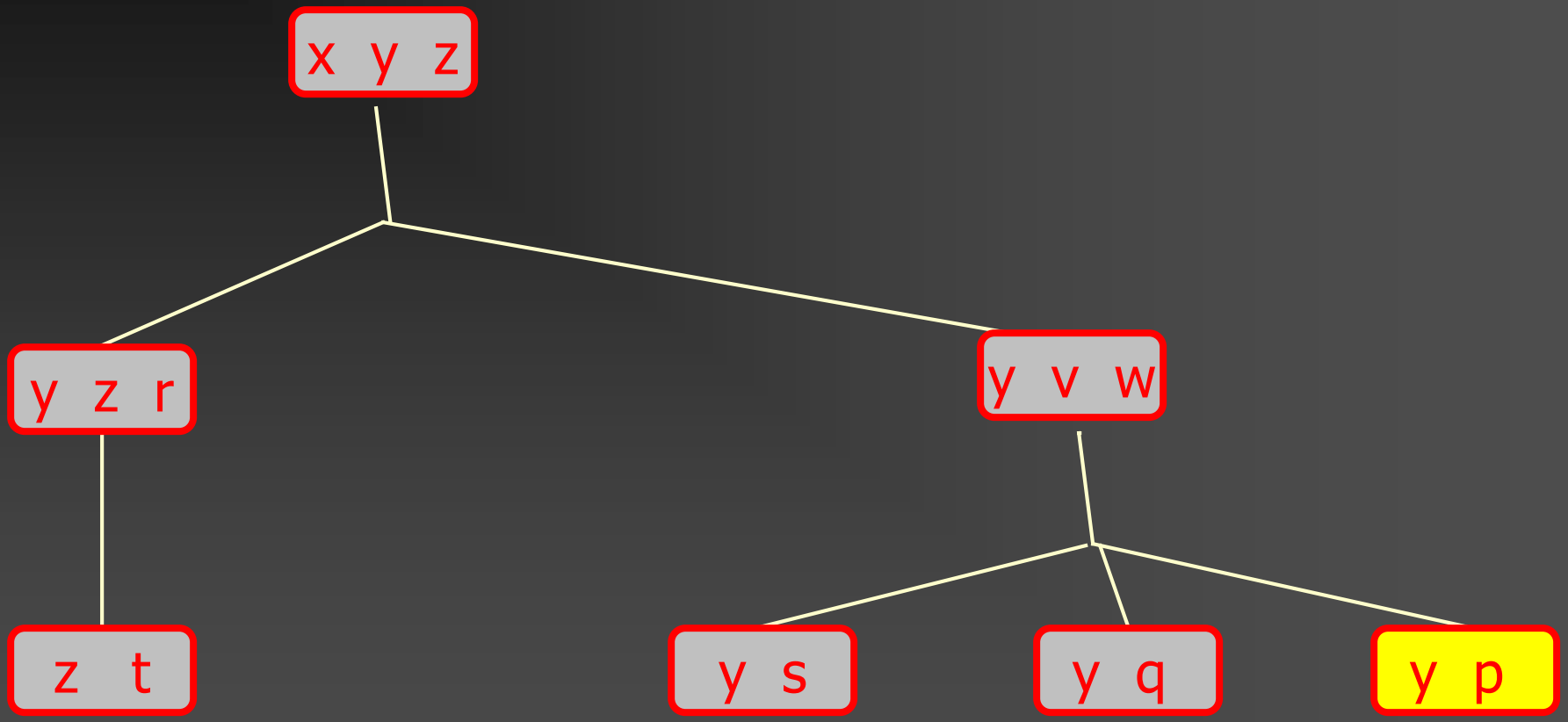
Acyclic hypergraph: Join Tree



Acyclic hypergraph: Join Tree

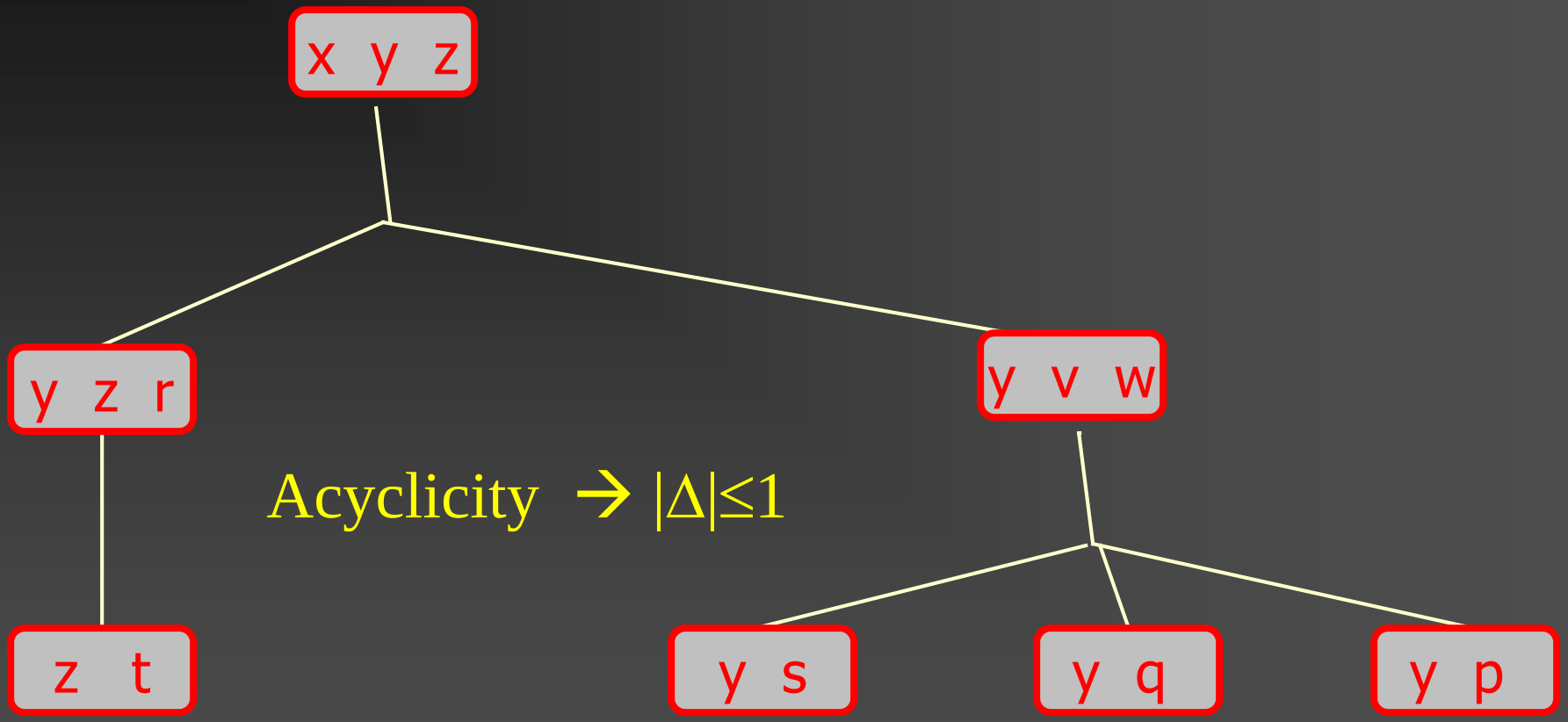


Acyclic hypergraph: Join Tree



Δ

Acyclic hypergraph: Join Tree



Acyclicity → $|\Delta| \leq 1$

Every acyclic (simple) hypergraph is 1-degenerate.

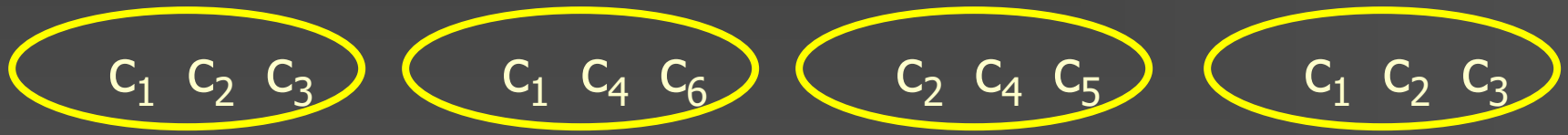
Negative result: Hypertree width

[G., Leone, Scarcello 99]

Hypetreewidth 2 =/=> bounded degeneracy

THEOREM: TRANS-ENUM remains TRANS-ENUM-hard even if H has hypertreewidth 2.

Take an arbitrary hypergraph H:



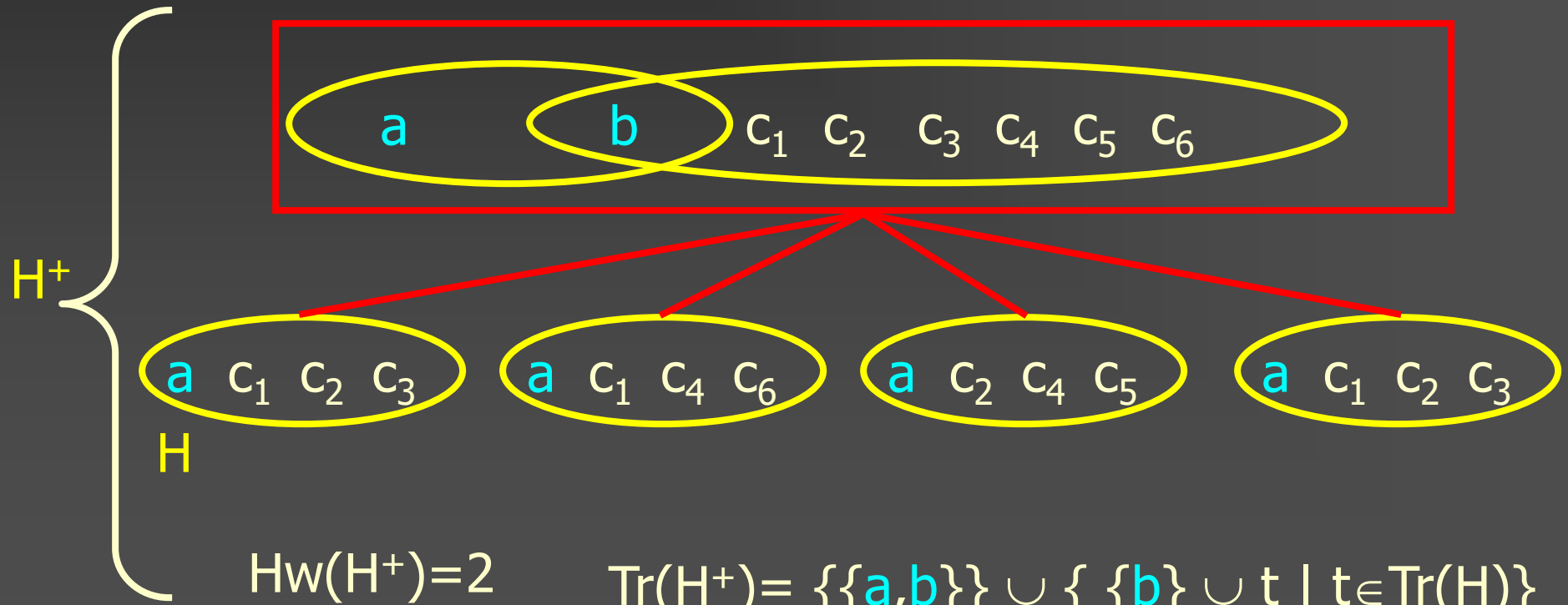
H

Negative result: Hypertree width

[G., Leone, Scarcello 99]

Hypetreewidth 2 =/=> bounded degeneracy

THEOREM: TRANS-ENUM remains TRANS-ENUM-hard even if H has hypertreewidth 2.



$Hw(H^+)=2$

$Tr(H^+)= \{\{a,b\}\} \cup \{ \{b\} \cup t \mid t \in Tr(H)\}$

Complexity of TRANS-HYP

Problem: TRANS-HYP
Instance: Hypergraphs G, H
Question: Is G = Tr(H) ?

TRANS-HYP obviously in co-NP.

TRANS-HYP in **DTIME**($n^{\log n}$) [Fredman&Khachiyan 96]

More Recent Complexity Results

Structural: Upper bounds

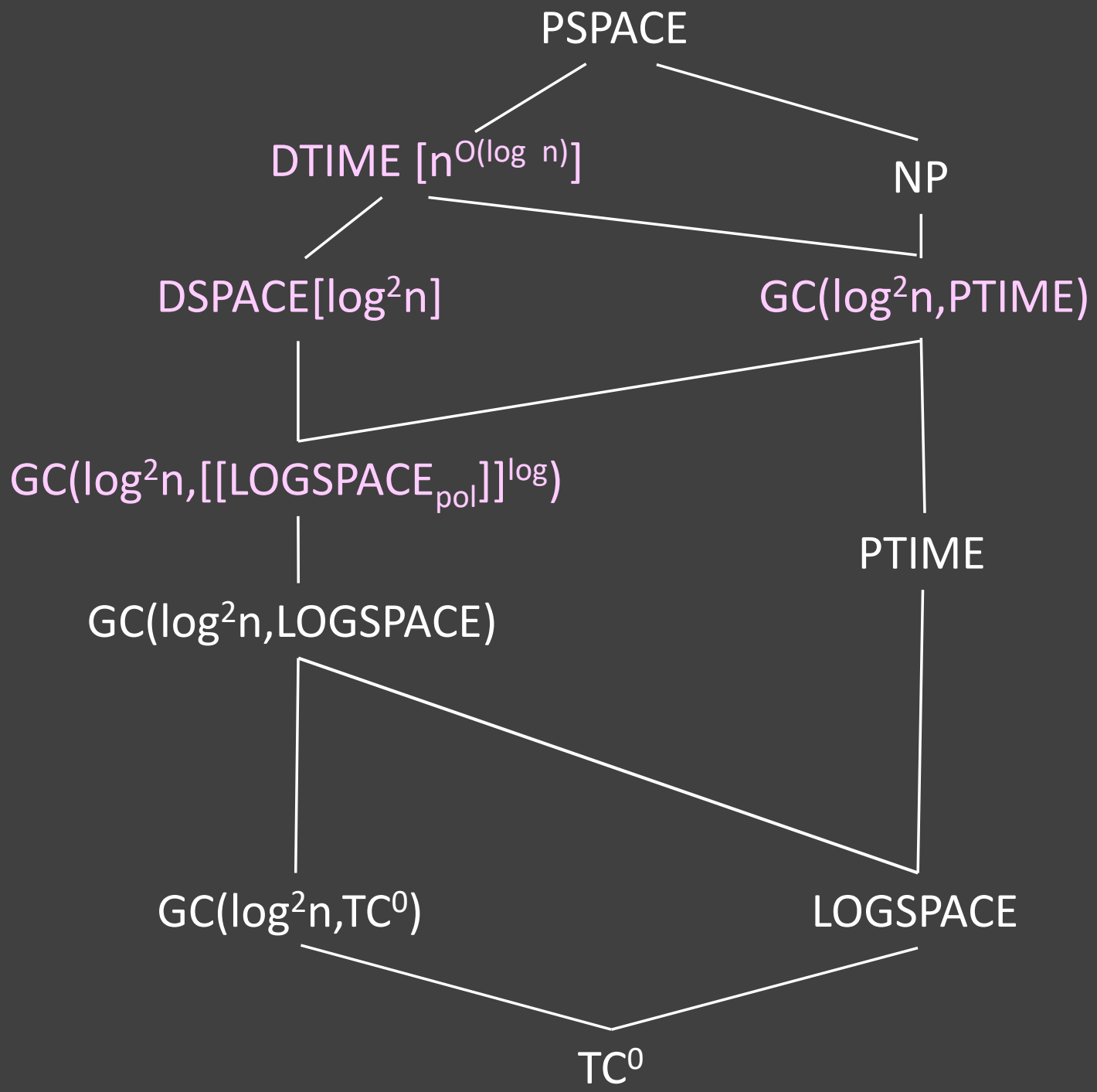
in GC($\log^2$ n, PTIME) — [Eiter, G., Makino STOC'02]; [Kavvadias, Stavropoulos. IPL 2003]

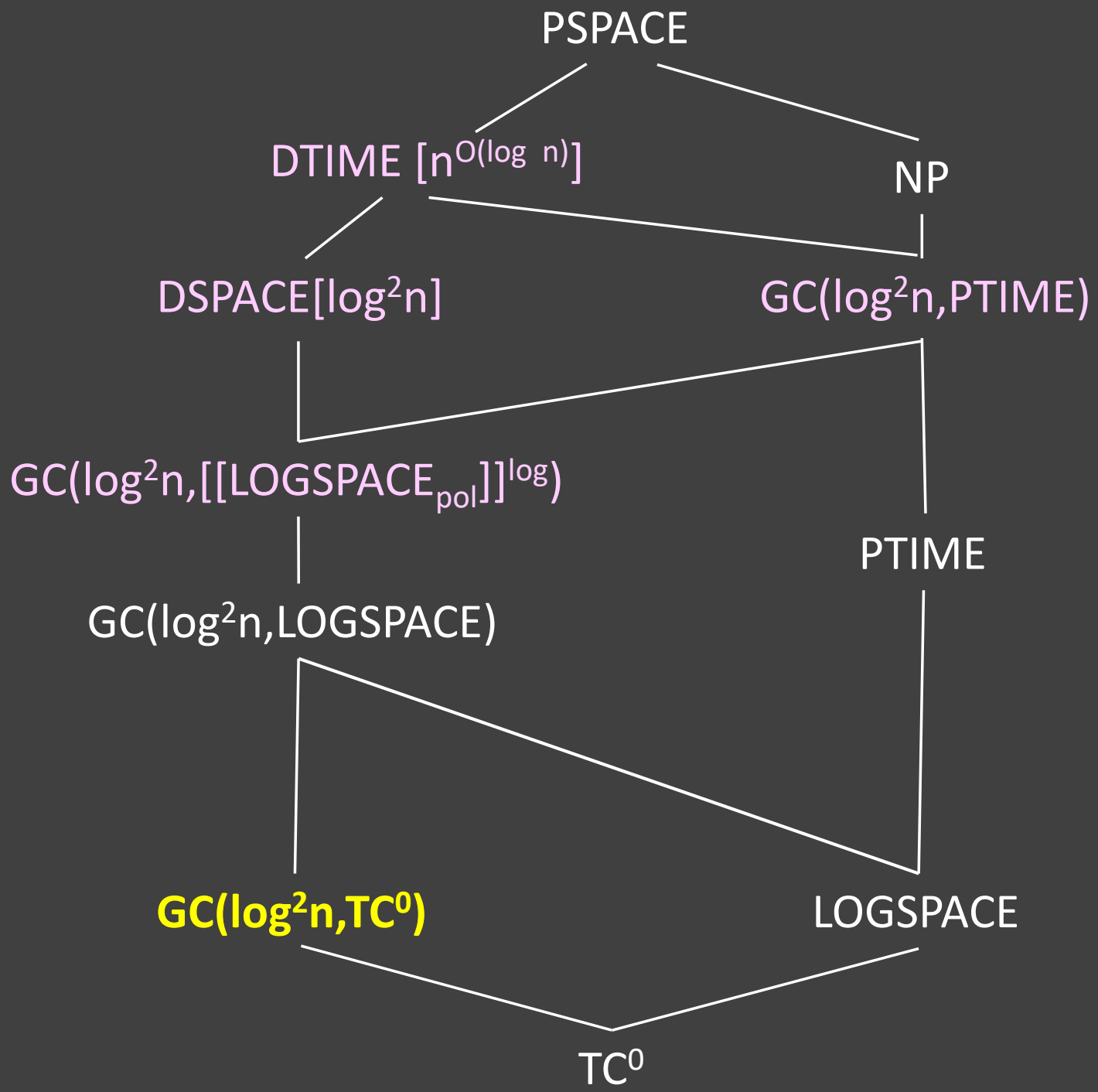
in LOG^{2}SPACE — [G. PODS'13]

actually in LOG^{2}SPACE $\cap$ GC($\log^2$(n),PTIME) — [G.,Malizia LICS'14]

Main result:

THEOREM: co-TRANS-HYP is in in GC($\log^2$ n, TC^0)


PSPACE
DTIME [n^O(log n)]
NP
DSPACE[log^2n]
GC(log^2n,PTIME)
GC(log^2n,[[LOGSPACE_pol]]^log)
PTIME
GC(log^2n,LOGSPACE)
GC(log^2n,TC^0)
LOGSPACE
TC^0


PSPACE
DTIME [n^O(log n)]
NP
DSPACE[log^2n]
GC(log^2n,PTIME)
GC(log^2n,[[LOGSPACE_pol]]^log)
PTIME
GC(log^2n,LOGSPACE)
GC(log^2n,TC^0)
LOGSPACE
TC^0

The basic self-reduction

The basic recursion scheme.

Let G and H be two hypergraphs over vertex set V, then

$$Tr(G \cup H) = [Tr(G) \oplus Tr(H)]_{min}$$

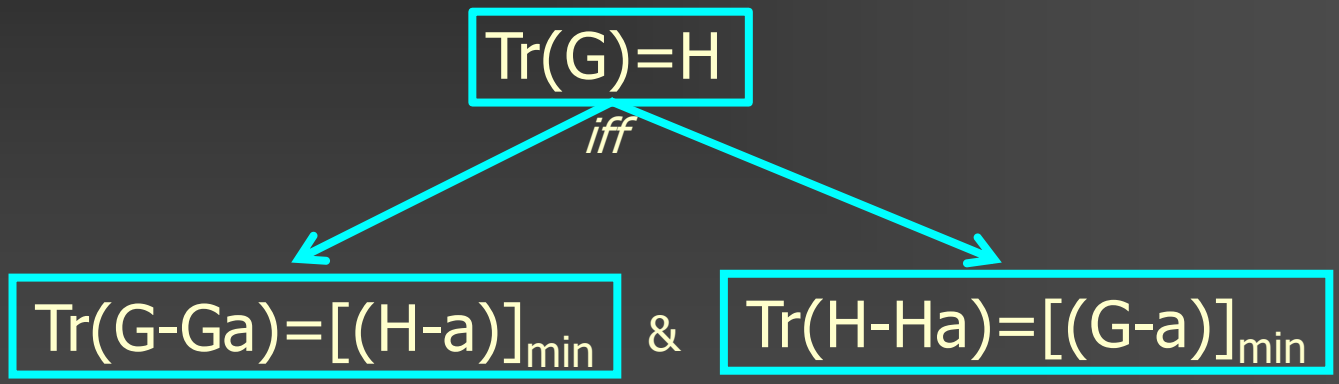
Let $a \in V$, then

G_a := hypergraph formed by all hyperedges of G that contain a

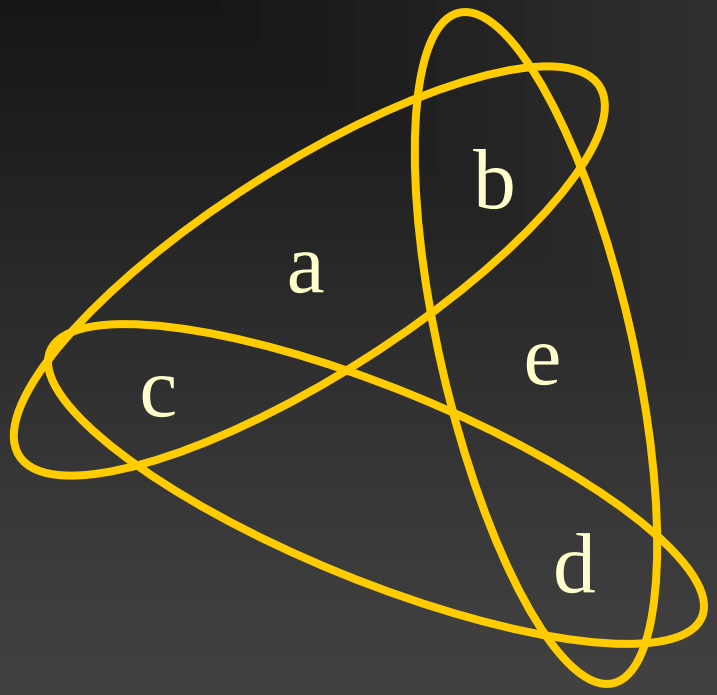
G-a := { e-{a} | $e \in$ edges(G) }

The Basic Self-Reduction

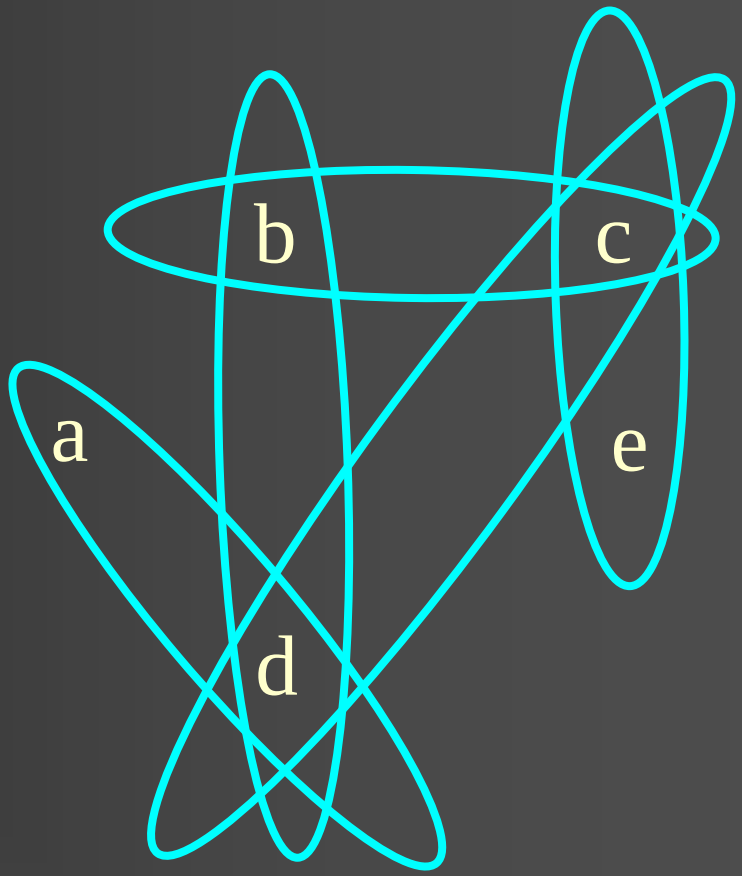
Let E, E′ be sets of edges over vertex set V, and let $a \in V$.



Davis-Putnam-like recursion scheme

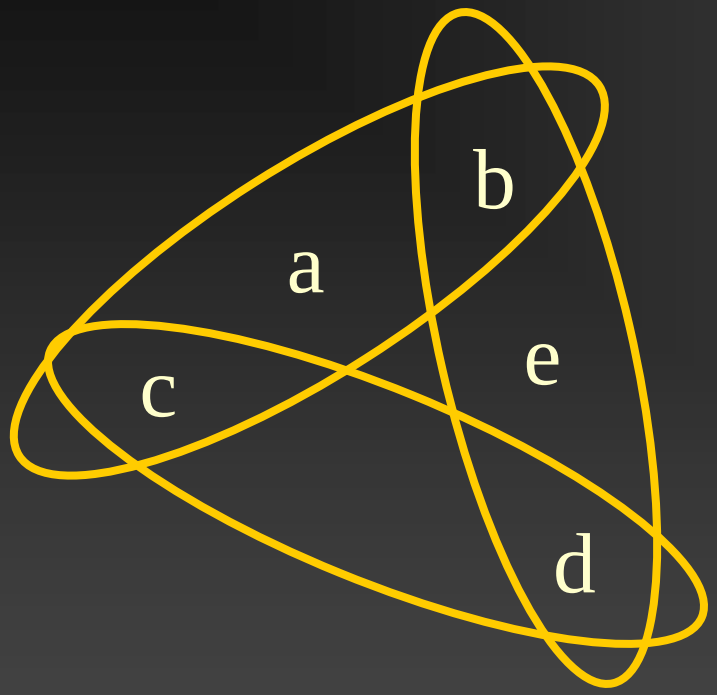


G

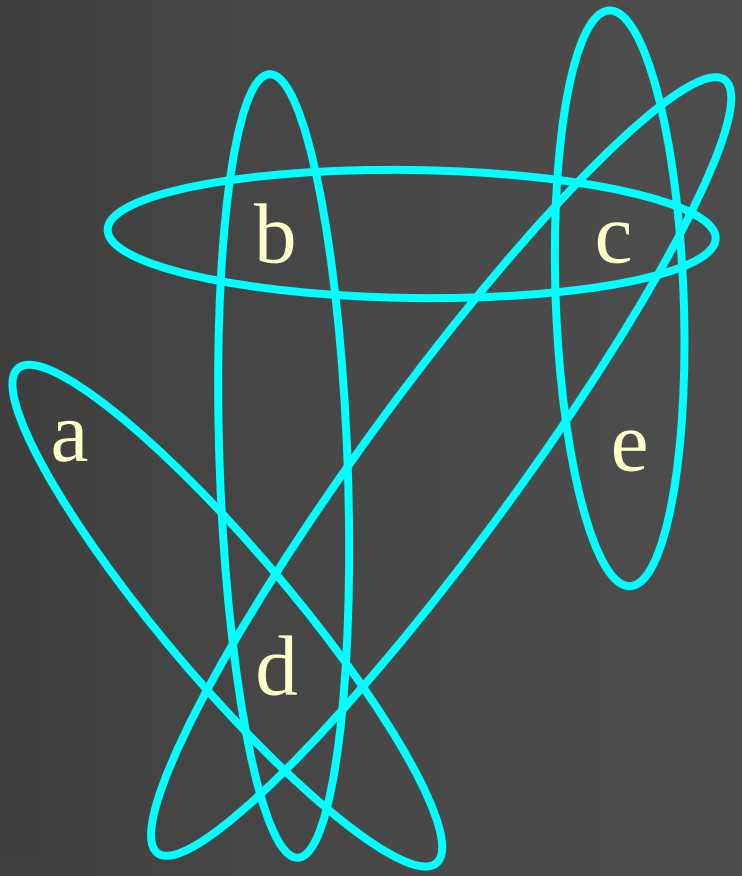


H

$$(c \vee a \vee b) \wedge (b \vee e \vee d) \wedge (c \vee d) \wedge$$
$$(\neg c \vee \neg d) \wedge (\neg a \vee \neg d) \wedge (\neg b \vee \neg d) \wedge (\neg b \vee \neg c) \wedge (\neg c \vee \neg e)$$



G

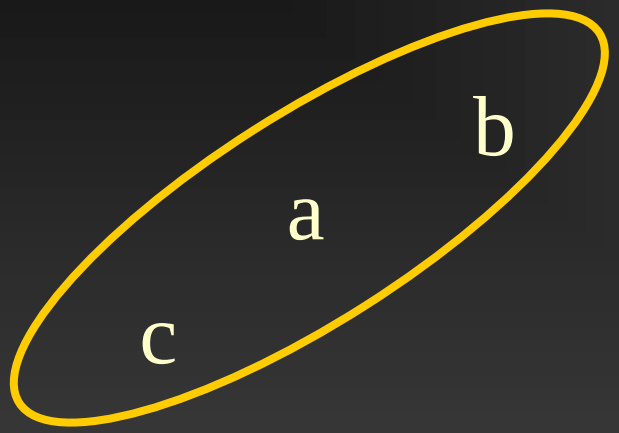


H

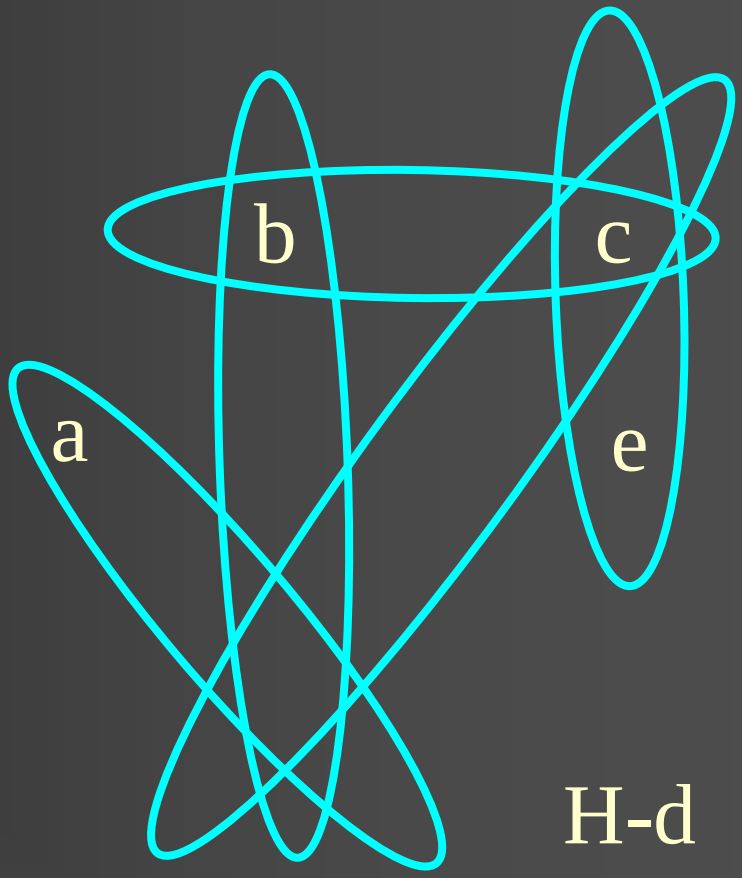
d=true

$(c \vee a \vee b) \wedge (b \vee e \vee d) \wedge (c \vee d) \wedge$

$(\neg c \vee \neg d) \wedge (\neg a \vee \neg d) \wedge (\neg b \vee \neg d) \wedge (\neg b \vee \neg c) \wedge (\neg c \vee \neg e)$



G-Gd

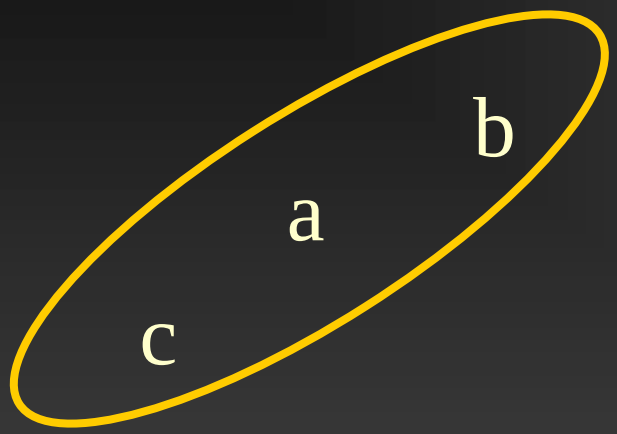


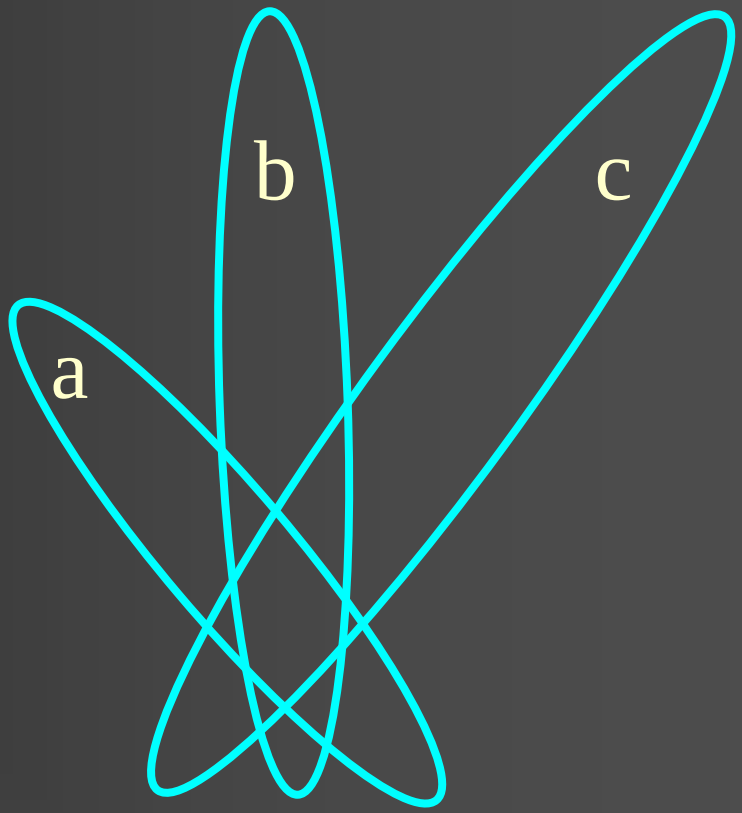
H-d

d=true

(c ∨ a ∨ b) ∧ ~~(b ∨ c ∨ d)~~ ∧ ~~(c ∨ d)~~ ∧

(¬c ∨ ¬~~d~~) ∧(¬a ∨ ¬~~d~~) ∧ (¬b ∨ ¬~~d~~) ∧ (¬b ∨ ¬c) ∧ (¬c ∨ ¬e)





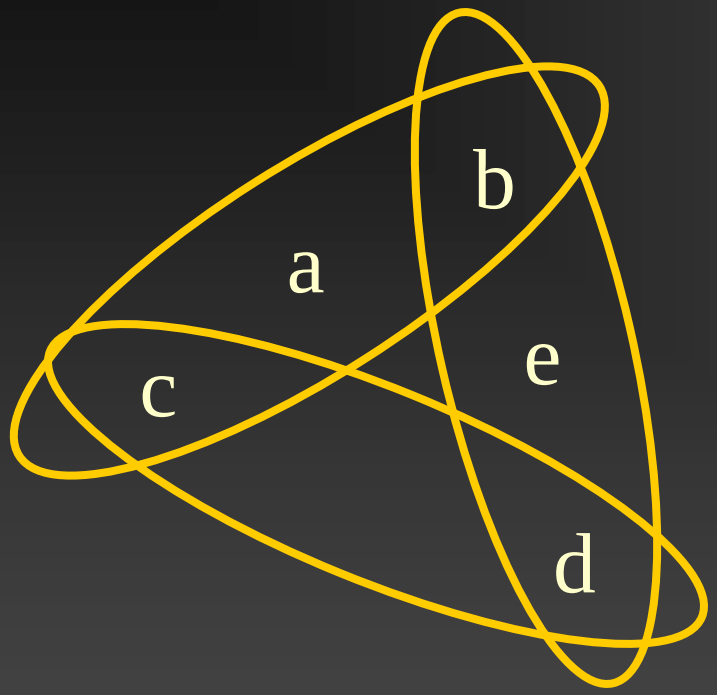
G-Gd

$[H\text{-}d]_{min}$

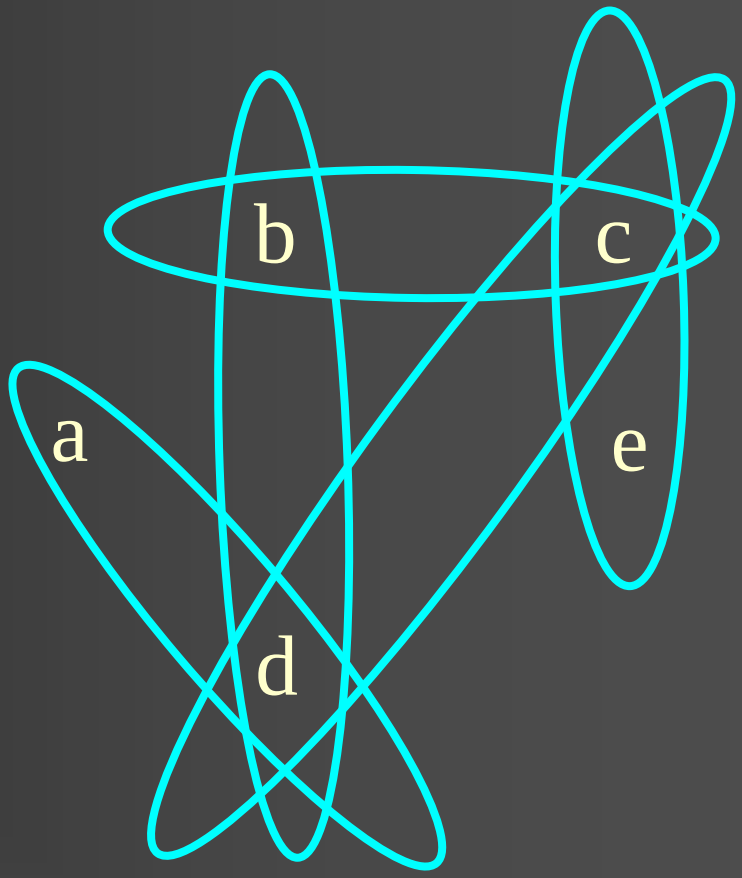
d=true

(c ∨ a ∨ b) ∧ (b ∨ ~~c ∨ d)~~ ∧ ~~(c ∨ d)~~ ∧

(¬c ∨ ¬~~d~~) ∧(¬a ∨ ¬~~d~~) ∧ (¬b ∨ ¬~~d~~) ∧ ~~(¬b ∨ ¬c)~~ ∧ ~~(¬c ∨ ¬e)~~



G

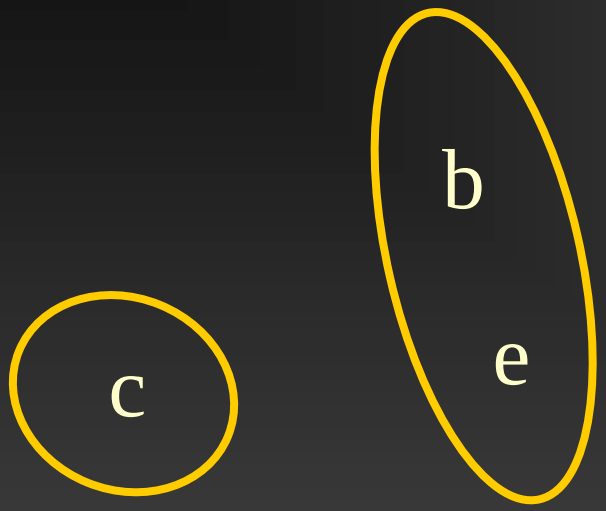


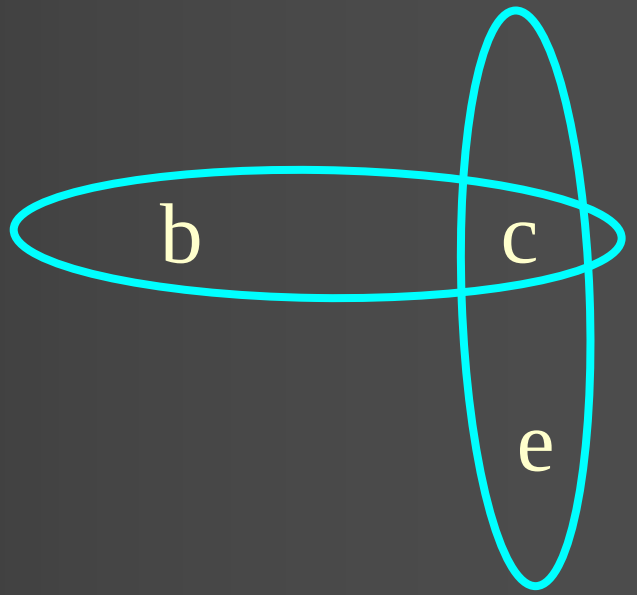
H

d=false

(c ∨ a ∨ b) ∧ (b ∨ e ∨ d) ∧ (c ∨ d) ∧

(¬c ∨ ¬d) ∧ (¬a ∨ ¬d) ∧ (¬b ∨ ¬d) ∧ (¬b ∨ ¬c) ∧ (¬c ∨ ¬e)





$[G\text{-}d]_{min}$

H-Hd

d=false

(c ∨ a ∨ b) ∧ (b ∨ e ∨ d) ∧ (c ∨ d) ∧

(¬c ∨ ¬d) ∧ (¬a ∨ ¬d) ∧ (¬b ∨ ¬d) ∧ (¬b ∨ ¬c) ∧ (¬c ∨ ¬e)

Useful Facts and Notions:

LEMMA [Boros et al.]: Let G be a hypergraph and t a minimal transversal of G. For each $v \in t$, there exists an edge e of G, called *critical*, such that $e \cap t = \{v\}$.

We then also say that (v,e) is a *critical pair* wrt t.

Let (G,H) be a pair of hypergraphs where $H \subseteq tr(G)$ and $G \subseteq tr(H)$.

Vertex v is *frequent* if it occurs in at least half of the edges of H.

Otherwise v is *infrequent*.

Method for Constructing a New Transversal

Let (G,H) be a pair of hypergraphs over a vertex set V, where $H \subseteq tr(G)$ and $G \subseteq tr(H)$.

Assume there exists a new minimal transversal t of G, $t \notin H$.

Successively guess and construct *assignments* (IN,EX) such that at each step: $IN \subseteq t$ and $EX \subseteq V - t$

Initially, (IN,EX)=({},{}), at the end (IN,EX)=(t,V-{t}), which can be easily verified.

We use three types of operations to extend (IN,EX):

INCLUSION, EXCLUSION, COMPLETION.

After each INCLUSION/EXCLUSION step, (G,H) is reduced to a smaller instance according to the standard self-reduction.

EXCLUSION OPERATION

Assume a current instance (G,H) and a current set (IN,EX).

Guess a frequent vertex v such that $v \notin t$.

We will *exclude* v as follows

Perform: $(IN,EX) := (IN, EX \cup \{v\})$ and

$(G,H) := ([G\text{-}v]_{min}, H\text{-}Hv)$

EXCLUSION OPERATION

Assume a current instance (G,H) and a current set (IN,EX).

Choose a frequent vertex v to be excluded from t.

We will *exclude* v as follows

Perform: (IN,EX):=(IN,EX$\cup${v}) and

$$(G,H) := ([G\text{-}v]_{min}, H\text{-}Hv)$$

Because v is frequent, at least half of the edges of H disappear

INCLUSION OPERATION

Assume a current instance (G,H) and a current set (IN,EX).

Guess an infrequent vertex v to be included in t.

Choose e such that (v,e) is a critical pair.

Include v and exclude all nodes in e-{v} as they are not in t.

Perform: $(IN,EX):=(IN\cup\{v\},EX\cup(e-\{v\}))$ and

$$(G,H) := ([(G-Gv)-e]_{min} , [(H-H_{e-\{v\}}) -v]_{min})$$

INCLUSION OPERATION

Assume a current instance (G,H) and a current set (IN,EX).

Guess an infrequent vertex v such that $v \in t$.

Choose e such that (v,e) is a critical pair.

Include v and exclude all nodes in e-{v} as they are not in t.

Perform: $(IN,EX) := (IN \cup \{v\}, EX \cup (e-\{v\}))$ and

$$(G,H) := ([(G-Gv)-e]_{min} , [(H-H_{e-\{v\}}) -v]_{min})$$

At least half of the edges of H disappear: given that v is infrequent, e-{v} intersects more than half of H's edges.

COMPLETION OPERATION

Assume a current instance (G,H) and a current set (IN,EX).

Assume we have done a sequence of insertion and deletion operations and guess that no further such operation is possible.

$\Rightarrow$ There is thus no frequent vertex in V-{t} and no infrequent vertex in {t} left.

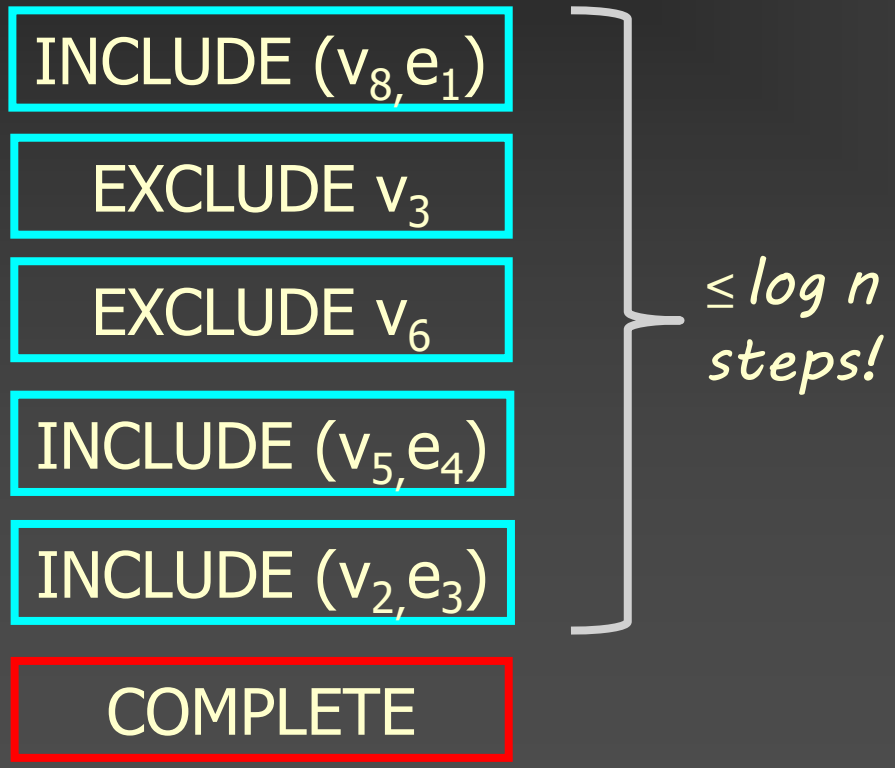
$\Rightarrow$ All remaining frequent vertices are in t and all remaining infrequent vertices are in V-{t}.

Let Freq and Infr denote the frequent/infrequent vertices

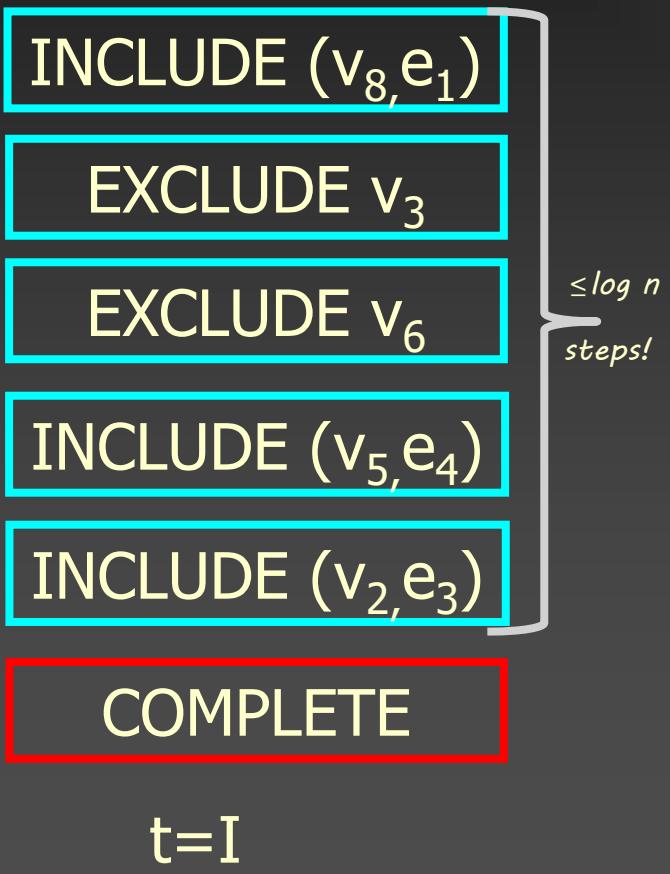
PERFORM (IN,EX):= (IN$\cup$Freq, EX$\cup$Infreq) = (t , V-{t})

WITNESSING A NEW TRANSVERSAL

Every new minimal transversal can be witnessed as follows



New Nondeterministic Algorithm



Nondeterministic Algorithm GUESSTRANS

1. Guess all INCLUDE/EXCLUDE steps simultaneously in form of a pair of relations (I^2, E^1) having ≤ log n tuples in total.

2. Compute reduced Hypergraph (G*,H*) at once.

3. (I,E) := COMPLETION(I,E)

4. Check that I is indeed a new transversal of G:
   (a) I intersects every edge of G
   (b) I does not cover any edge of H

5. If so, output I.

Logical Analysis of GuessTrans

1. Guess all INCLUDE/EXCLUDE steps simultaneously in form of a pair of relations (I^2, E^1) having $\leq \log n$ tuples in total.

$$(\exists^{\leq \log n}\ I)\ (\exists^{\leq \log n}\ E)\quad \phi_{check}\ (H,G,\ I,E)$$

2. Compute reduced Hypergraph (G*,H*) at once.

$(G^*,H^*) := ([(G-G_I)-E]_{min} , [(H-H_E)-I]_{min})$ *Relational algebra, thus in* FO

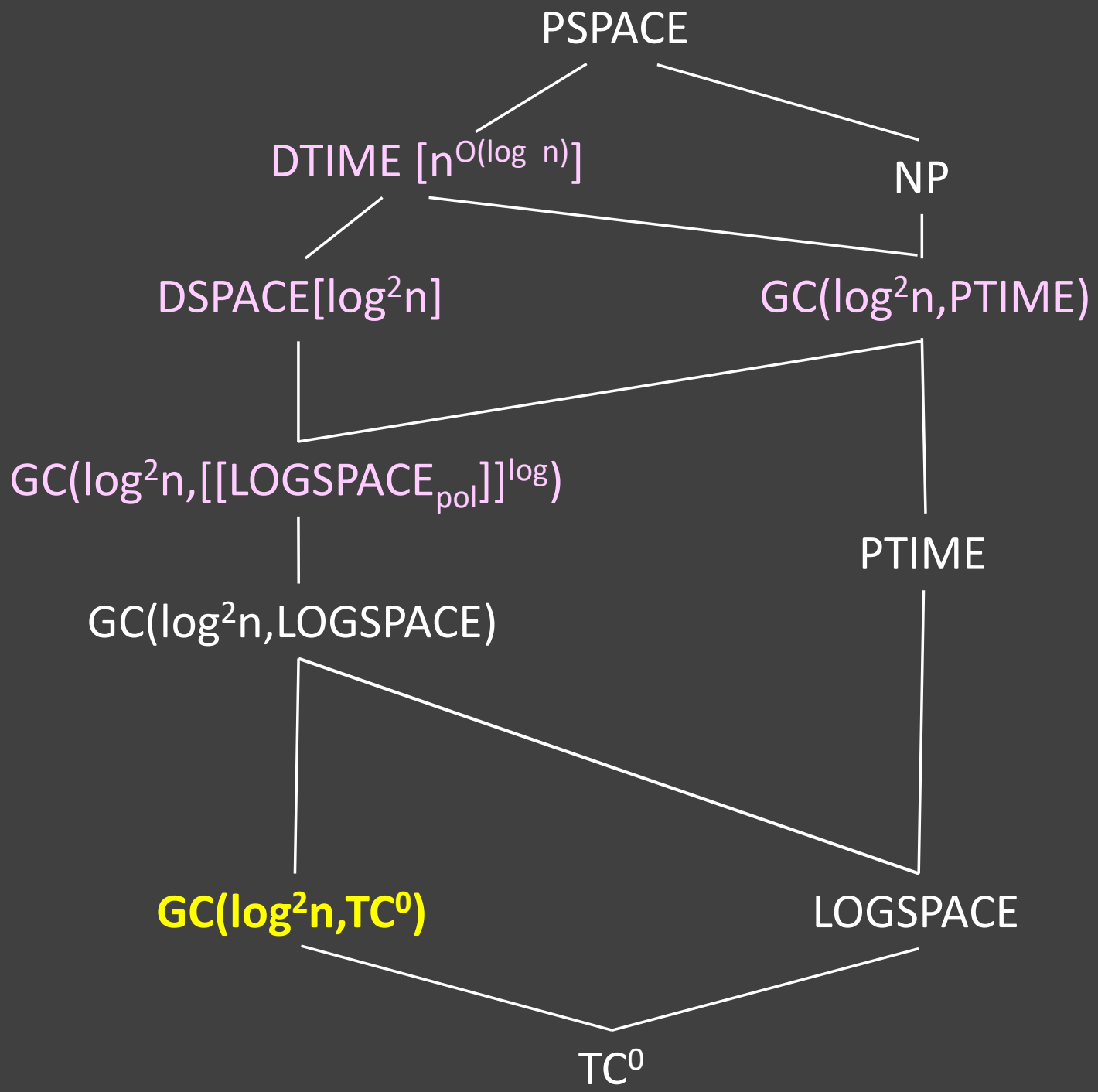
3. (I,E) := COMPLETION(I,E)

(IN,EX):= (IN∪Freq, EX∪Infreq) *Clearly in* FO[COUNT]

4. Check that I is indeed a new transversal of G:
(a) I intersects every edge of G; (b) I does not cover any edge of H.

Clearly in FO

Summary: *Overall problem can be solved in* $\exists^{\leq \log n}$ (FO[COUNT])


PSPACE
DTIME [n^O(log n)]
NP
DSPACE[log^2n]
GC(log^2n,PTIME)
GC(log^2n,[[LOGSPACE_pol]]^log)
PTIME
GC(log^2n,LOGSPACE)
GC(log^2n,TC^0)
LOGSPACE
TC^0

Conclusion

- New bound for TRANS-HYP
- New structural bound → indications where to dig further
- Evidence that transversal might not be LOGSPACE-hard
- STILL OPEN: TRANS-HYP $\in$ PTIME ?

  co-TRANS-HYP $\in$ GC($\log^2 n$,FO) ?

- Practical usefulness?

  We have a very simple guess-and-check algorithm (→ SICOMP paper); adding smart pruning and selection heuristics may be rewarding.