Denial Constraints

Tobias Bleifuß

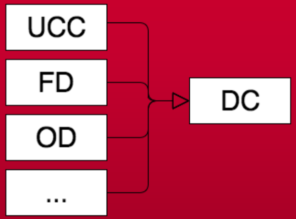
17.07.2017

Some slides provided by Xu Chu.

Motivation
Expressiveness of Denial Constraints

- Functional Dependency ZIP → City
  - $\forall t_1, t_2 \in R: \neg(t_1.zip = t_2.zip \wedge t_1.city \neq t_2.city)$
- Order Dependency
  - $\forall t_1, t_2 \in R: \neg(t_1.date \leq t_2.date \wedge t_1.population > t_2.population)$
- Same state, more income, lower tax rate
  - $\forall t_1, t_2 \in R: \neg(t_1.state = t_2.state \wedge t_1.income > t_2.income \wedge t_1.taxRate < t_2.taxRate)$
- Cross-column predicates
  - $\forall t_1 \in R: \neg(t_1.openingTime > t_1.closingTime)$
- Trump-Rule
  - $\forall t_1 \in R: \neg(t_1.name = „Trump“ \wedge t_1.taxRate = 0)$

Motivation
Why Denial Constraints?



Generalization of many other ICs

Fast DC discovery
+ Classification →

Fast discovery of other ICs

Higher expressiveness

Enables expression of business rules that cannot be expressed with more restrictive ICs

Versatility

Balance
expressive power and complexity

Why not even higher expressive-ness? (e.g. general first order logic)

Search space

Reasoning

Motivation
Unified View for Repairs

- DCs are useful in applications of ICs, i.e. data cleansing (repair):

Repairing IC types in isolation → **Violating IC of another type** / **Non-optimal repair** / **Different result for different order of repairs** — **Unified view through DCs**

- Example
  - FD phone → city
  - UCC zip,city

ZIP	City	Phone
123	Berlin	030
123	Potsdam	030

References
- Chu et al.: "Holistic data cleaning: Putting violations into context." (2013)
- Geerts et al. "The LLUNATIC data-cleaning framework." (2013)

Motivation
Example

TID	FN	LN	GD	AC	PH	CT	ST	ZIP	MS	CH	SAL	TR	STX	MTX	CTX
t_1	Mark	Ballin	M	304	232-7667	Anthony	WV	25813	S	Y	5000	3	2000	0	2000
t_2	Chunho	Black	M	719	154-4816	Denver	CO	80290	M	N	60000	4.63	0	0	0
t_3	Annja	Rebizant	F	636	604-2692	Cyrene	MO	64739	M	N	40000	6	0	4200	0
t_4	Annie	Puerta	F	501	378-7304	West Crossett	AR	72045	M	N	85000	7.22	0	40	0
t_5	Anthony	Landram	M	319	150-3642	Gifford	IA	52404	S	Y	15000	2.48	40	0	40
t_6	Mark	Murro	M	970	190-3324	Denver	CO	80251	S	Y	60000	4.63	0	0	0
t_7	Ruby	Billinghurst	F	501	154-4816	Kremlin	AR	72045	M	Y	70000	7	0	35	1000
t_8	Marcelino	Nuth	F	304	540-4707	Kyle	WV	25813	M	N	10000	4	0	0	0

$Key: \{AC, PH\}$	$\forall t_\alpha, t_\beta \in R, \neg(t_\alpha.AC = t_\beta.AC \wedge t_\alpha.PH = t_\beta.PH)$
$Domain: MS \in \{S, M\}$	$\forall t_\alpha \in R, \neg(t_\alpha.MS \neq S \wedge t_\alpha.MS \neq M)$
$FD: ZIP \rightarrow ST$	$\forall t_\alpha, t_\beta \in R, \neg(t_\alpha.ZIP = t_\beta.ZIP \wedge t_\alpha.ST \neq t_\beta.ST)$
$CFD: CT = Los\ Angeles \rightarrow ST = CA$	$\forall t_\alpha \in R, \neg(t_\alpha.CT = Los\ \text{Angeles} \wedge t_\alpha.ST \neq CA)$
$Check: SAL \geq STX$	$\forall t_\alpha \in R, \neg(t_\alpha.SAL < t_\alpha.STX)$
Business logic	$\forall t_\alpha, t_\beta \in R, \neg(t_\alpha.ST = t_\beta.ST \wedge t_\alpha.SAL < t_\beta.SAL \wedge t_\alpha.TR > t_\beta.TR)$

Denial Constraints (DCs)
Formal Definition

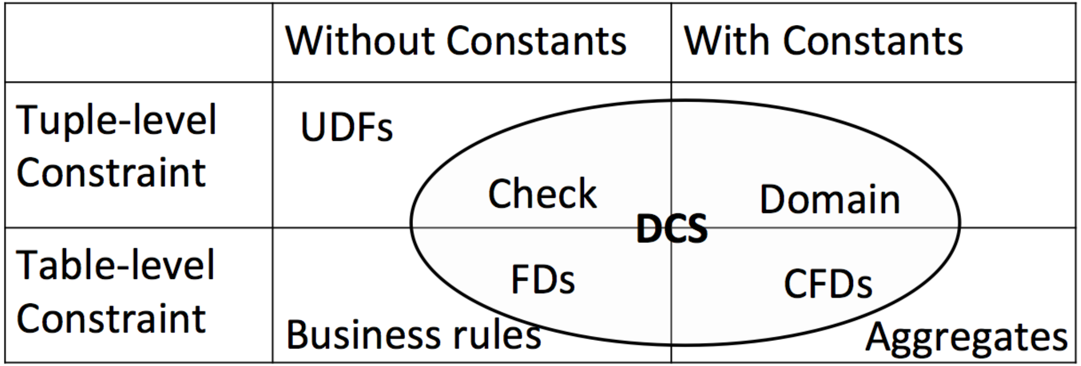
$$\varphi: \forall t_\alpha, t_\beta, \ldots \in R: \ \neg(p_1 \ \wedge \ \ldots \wedge p_m)$$

$$p_i: t_x.A \ \phi \ t_y.B \ or \ t_x.A \ \phi \ c$$

$x, y \in \{\alpha, \beta, \ldots\}$ $A, B \in R$ c is a constant ϕ is a built-in operator (in our case $=, \neq, <, \leq, >, \geq$)

- A DC expresses that a set of predicates cannot be true together for any combination of tuples in a relation
- Each predicate expresses a relationship between two cells, or between a cell and a constant

Motivation
Why Denial Constraints? II



- Easy violation detection using SQL
- Proven useful in:

  Data repairing, consistent query answering, and data currency rules
- A set of sound and powerful inference rules

DC Axioms
Triviality

Rule: $\forall p_i, p_j$: if $\overline{p}_i \in Imp(p_j)$, then $\neg(p_i \wedge p_j)$ is a trivial DC

ϕ	$=$	$\neq$	$>$	$<$	$\geq$	$\leq$
$\overline{\phi}$	$\neq$	$=$	$\leq$	$\geq$	$<$	$>$
$Imp(\phi)$	$=, \geq, \leq$	$\neq$	$>, \geq, \neq$	$<, \leq, \neq$	$\geq$	$\leq$

Example:

p_i p_j

$$\forall t_\alpha, t_\beta \in R, \neg(t_\alpha.SAL = t_\beta.SAL \wedge t_\alpha.SAL > t_\beta.SAL)$$

$$\overline{p}_i: t_\alpha.SAL \neq t_\beta.SAL$$

$$Imp(p_j) = \{t_\alpha.SAL > t_\beta.SAL,\ t_\alpha.SAL \geq t_\beta.SAL,\ t_\alpha.SAL \neq t_\beta.SAL\}$$

DC Axioms
Augmentation

Rule: If $\neg(p_1 \wedge \ldots \wedge p_n)$ is valid, then $\neg(p_1 \wedge \ldots \wedge p_n \wedge q)$ is also valid

Example:

$$\forall t_\alpha, t_\beta \in R, \neg(t_\alpha.ZIP = t_\beta.ZIP \wedge t_\alpha.ST \neq t_\beta.ST)$$

$$\forall t_\alpha, t_\beta \in R, \neg(t_\alpha.ZIP = t_\beta.ZIP \wedge t_\alpha.ST \neq t_\beta.ST \wedge t_\alpha.SAL < t_\beta.SAL)$$

Not Minimal

DC Axioms
Transitivity

Rule: If $\neg(p_1 \wedge \ldots \wedge p_n \wedge q_1)$, and $\neg(r_1 \wedge \ldots \wedge r_n \wedge q_2)$ are valid, and $q_2 \in Imp(\overline{q}_1)$, then $\neg(p_1 \wedge \ldots \wedge p_n \wedge r_1 \wedge \ldots \wedge r_n)$ is valid

ϕ	$=$	$\neq$	$>$	$<$	$\geq$	$\leq$
$\overline{\phi}$	$\neq$	$=$	$\leq$	$\geq$	$<$	$>$
$Imp(\phi)$	$=, \geq, \leq$	$\neq$	$>, \geq, \neq$	$<, \leq, \neq$	$\geq$	$\leq$

Example:

q_1

$$\forall t_\alpha, t_\beta \in R, \neg(t_\alpha.ST = t_\beta.ST \wedge t_\alpha.SAL < t_\beta.SAL \wedge t_\alpha.TR > t_\beta.TR)$$

$$\forall t_\alpha, t_\beta \in R, \neg(t_\alpha.ZIP = t_\beta.ZIP \wedge t_\alpha.ST \neq t_\beta.ST)$$

q_2

$$\forall t_\alpha, t_\beta \in R, \neg(t_\alpha.ZIP = t_\beta.ZIP \wedge t_\alpha.SAL < t_\beta.SAL \wedge t_\alpha.TR > t_\beta.TR)$$

Problem Statement
Denial Constraint Discovery

- **Given:**

Relational Instance I

A	B
1	3
2	2

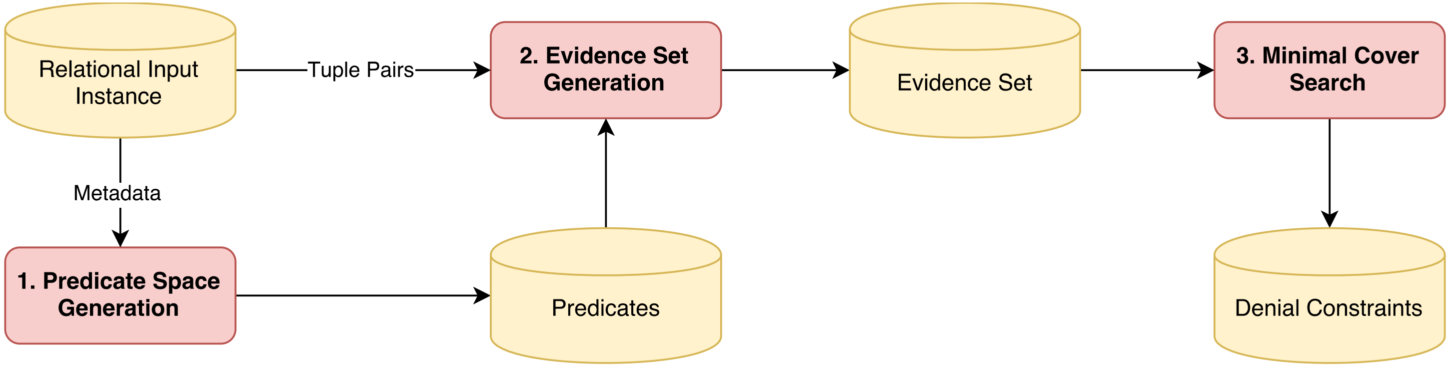
Predicate Space P

- $t_1.A = t_2.A$
- $t_1.A \neq t_2.A$
- $t_1.B = t_2.B$
- $t_1.A < t_1.B$
- …

- No constants
- Operators: $=, \neq, <, \leq, >, \geq$
- At most two tuples (t_1, t_2)
- Must be negation closed

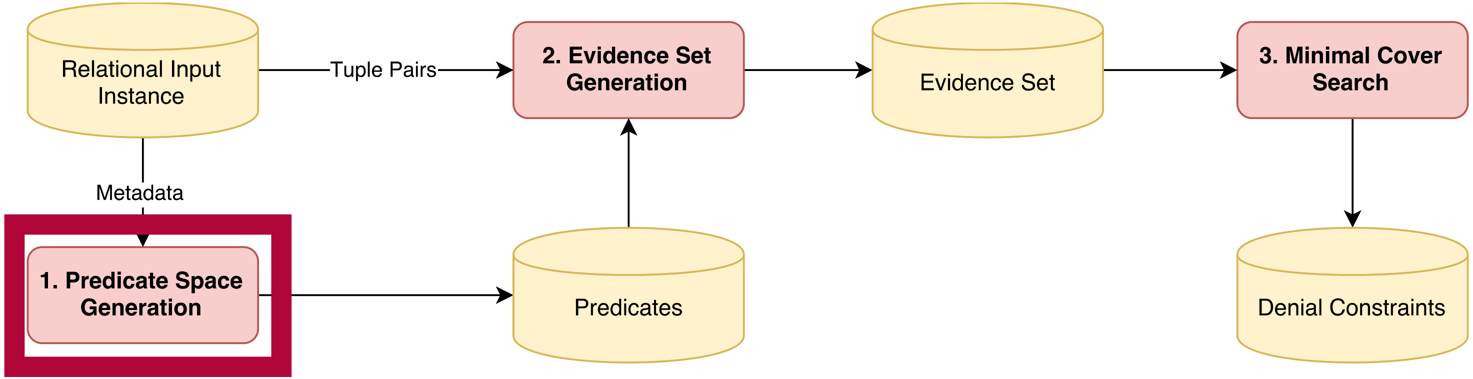
- **Task:** Find all minimal, non-trivial DCs over the given predicate space P valid on I
  - Trivial: $\forall t_1, t_2 : \neg(t_1.A = t_2.A \wedge t_1.A \neq t_2.A)$
  - Minimal: $\forall t_1.t_2 : \neg(t_1.A = t_2.A) \Rightarrow \forall t_1.t_2 : \neg(t_1.A = t_2.A \wedge \ \ldots)$

FastDC
Overview

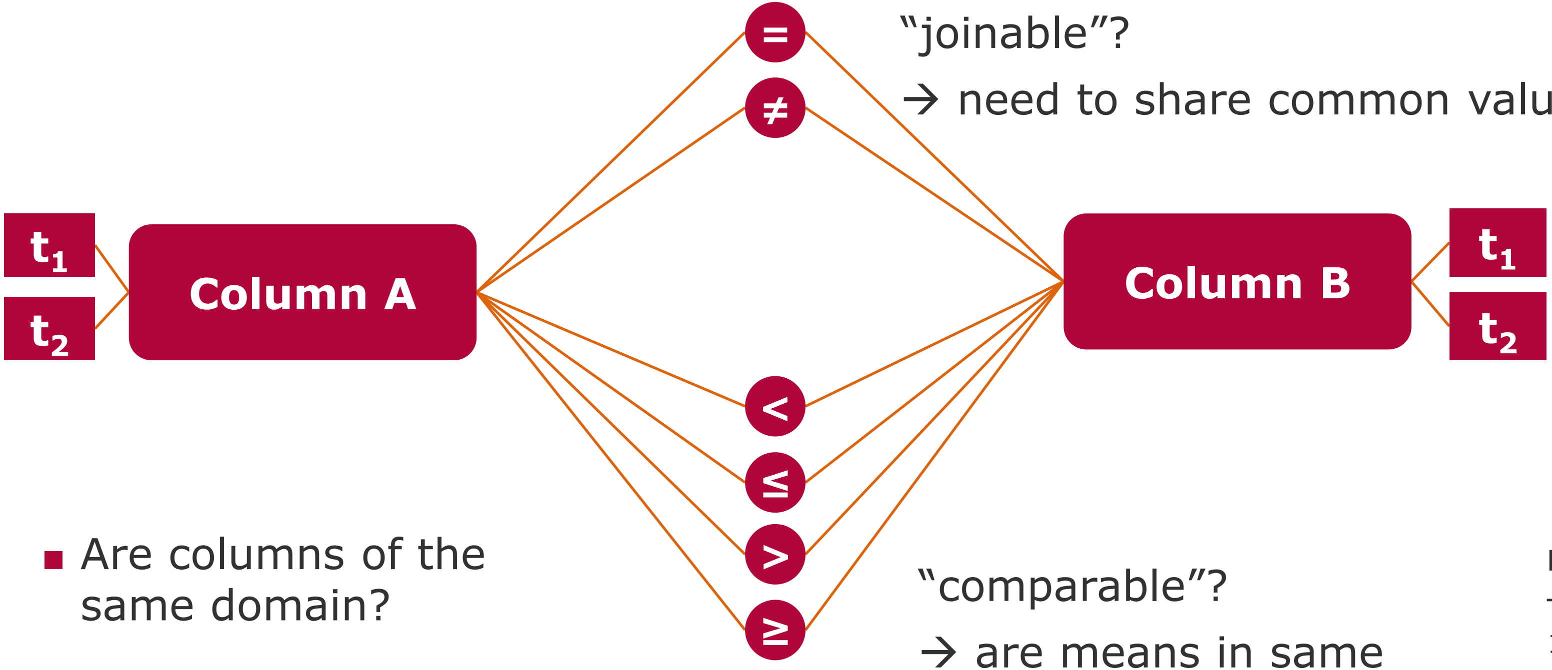


Xu Chu, Ihab F Ilyas, and Paolo Papotti. Discovering Denial Constraints. In Proceedings of the VLDB Endowment, 2013.

FastDC Overview



FastDC
Predicate Space Generation



"joinable"?

→ need to share common values

- Are columns of the same domain?

"comparable"?

→ are means in same order of magnitude?

FastDC
Predicate Space Generation

	pId	$partner$	$hInc$
t_1	n1	n2	60000
t_2	n2	n1	60000
t_3	n3	n5	40000
t_4	n6	n7	40000

String

- $p_1: t_1.pId = t_2.pId$
- $p_2: t_1.pId \neq t_2.pId$
- $p_3: t_1.partner = t_2.partner$
- $p_4: t_1.partner \neq t_2.partner$

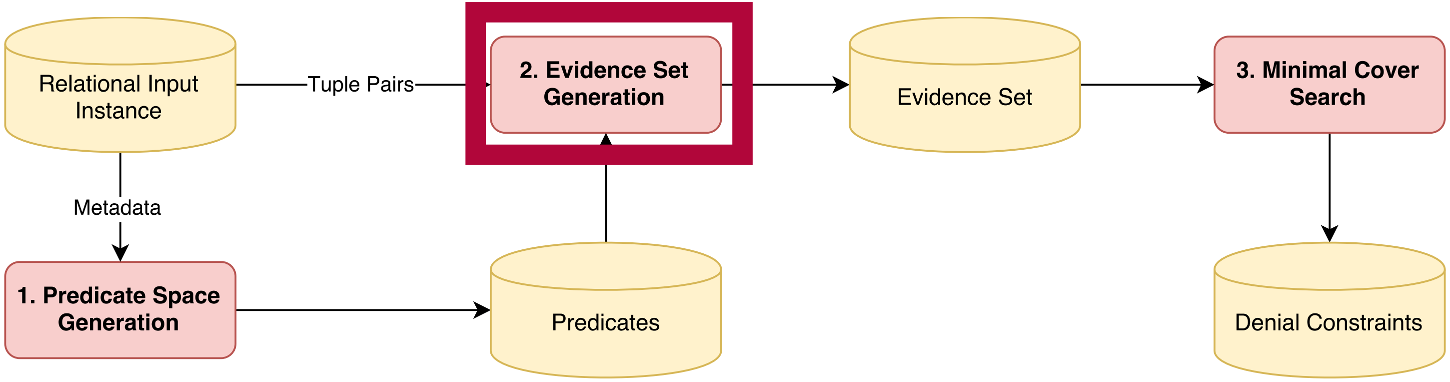
Numeric

- $p_5: t_1.hInc = t_2.hInc$
- $p_6: t_1.hInc \neq t_2.hInc$
- $p_7: t_1.hInc < t_2.hInc$
- $p_8: t_1.hInc \geq t_2.hInc$
- $p_9: t_1.hInc > t_2.hInc$
- $p_{10}: t_1.hInc \leq t_2.hInc$

Cross-column

- $p_{11}: t_1.pId = t_2.partner$
- $p_{12}: t_1.pId \neq t_2.partner$
- $p_{13}: t_1.pId = t_1.partner$
- $p_{14}: t_1.pId \neq t_1.partner$

FastDC
Overview



FastDC
Evidence Set Generation

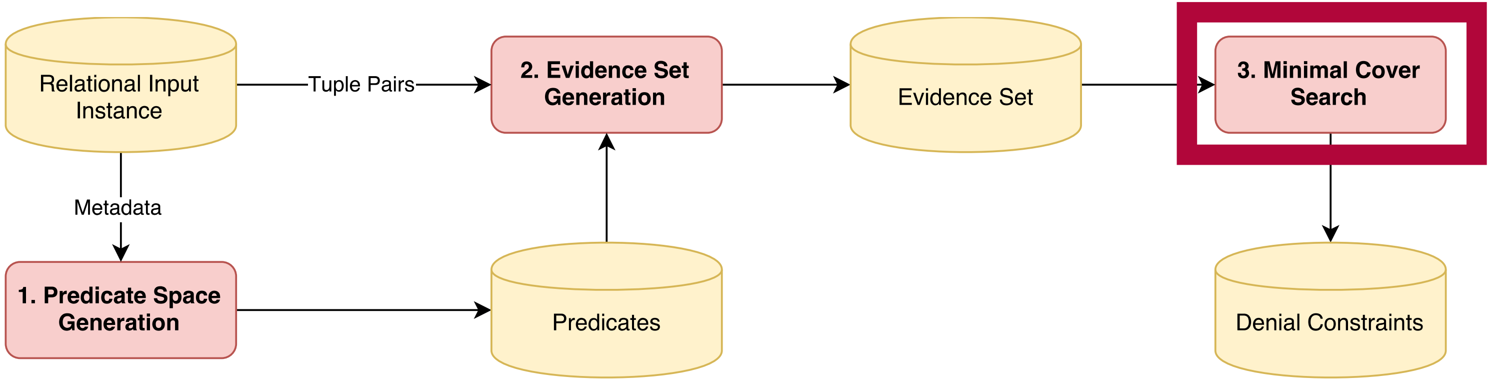
	pId	$partner$	$hInc$
t_1	n1	n2	60000
t_2	n2	n1	60000
t_3	n3	n5	40000
t_4	n6	n7	40000

- For every tuple pair calculate set of satisfied predicates
- **Result:** set of predicate sets ("evidence set")

- $t_1.pId \neq t_2.pId$
- $t_1.partner \neq t_2.partner$
- $t_1.hInc = t_2.hInc$
- $t_1.hInc \geq t_2.hInc$
- $t_1.hInc \leq t_2.hInc$
- $t_1.pId = t_2.partner$
- $t_1.pId \neq t_1.partner$

Compares each tuple pair
→ quadratic complexity in the number of tuples ☹

FastDC Overview



FastDC
Minimal Cover Search

- **Definition:** $X = \{p_1, \dots, p_n\}$ is a minimal set cover for the evidence set Evi if $\forall E \in Evi: X \cap E \neq \emptyset$, and $\nexists Y \subset X: \forall E \in Evi: Y \cap E \neq \emptyset$.
- **Theorem:** $\neg(\overline{p}_1 \wedge \dots \wedge \overline{p}_n)$ is a valid minimal DC iff $X = \{p_1, \dots, p_n\}$ is a minimal set cover for the evidence set.

- X is a cover for $Evi \Rightarrow \neg(\overline{p}_1 \wedge \dots \wedge \overline{p}_n)$ is a valid DC: The elements of Evi represent all possible violations of a DC, for every $E \in Evi$ there exists one p_i such that $p_i \in E$ and therefore $\overline{p}_i \notin E$
- $\neg(\overline{p}_1 \wedge \dots \wedge \overline{p}_n)$ is a valid DC $\Rightarrow X$ is a cover for Evi: No tuple pairs fulfills $\overline{p}_1$ to $\overline{p}_n$ together, so for every tuple pair there is one $\overline{p}_i$ that is not part of the corresponding evidence. Thus p_i is part of the evidence and the evidence is covered by X.

FastDC
Minimal Cover Search

- Theorem: $\neg(\overline{p}_1 \wedge \cdots \wedge \overline{p}_n)$ is a valid minimal DC iff $X = \{p_1, \ldots, p_n\}$ is a minimal set cover for the evidence set.

p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	p_9	p_{10}	p_{11}	p_{12}	p_{13}	p_{14}
-	+	-	+	+	-	-	+	-	+	+	-	-	+
-	+	-	+	+	-	-	+	-	+	-	+	-	+
-	+	-	+	-	+	+	-	-	+	+	-	-	+
-	+	-	+	-	+	-	+	+	-	+	-	-	+

- $\{p_2\} = \{t_1.pId \neq t_2.pId\}$ → $\forall t_1, t_2 : \neg(t_1.pId = t_2.pId)$
- $\{p_5, p_{11,}\}$ $=\{t_1.pId \neq t_2.partner, t_1.hInc = t_2.hInc\}$
  → $\forall t_1, t_2 : \neg(t_1.pId = t_2.partner \wedge t_1.hInc \neq t_2.hInc)$

FastDC
Minimal Cover Search

- Theorem: $\neg(\overline{p}_1 \wedge \cdots \wedge \overline{p}_n)$ is a valid minimal DC iff $X = \{p_1, \ldots, p_n\}$ is a minimal set cover for the evidence set.

p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	p_9	p_{10}	p_{11}	p_{12}	p_{13}	p_{14}
-	+	-	+	+	-	-	+	-	+	+	-	-	+
-	+	-	+	+	-	-	+	-	+	-	+	-	+
-	+	-	+	-	+	+	-	-	+	+	-	-	+
-	+	-	+	-	+	-	+	+	-	+	-	-	+
0	**4**	**0**	**4**	**2**	**2**	**1**	**3**	**1**	**3**	**3**	**1**	**0**	**4**

- DFS + branch pruning + search heuristic:
  sort predicates by descending frequency in the evidence set

FastDC
Minimal Cover Search

DFS

	count
p_2	4
p_4	4
p_{14}	4
p_{11}	3
...	...

p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	p_9	p_{10}	p_{11}	p_{12}	p_{13}	p_{14}
-	+	-	+	+	-	-	+	-	+	+	-	-	+
-	+	-	+	+	-	-	+	-	+	-	+	-	+
-	+	-	+	-	+	+	-	-	+	+	-	-	+
-	+	-	+	-	+	-	+	+	-	+	-	-	+

- $\{p_2\} = \{t_1.pId \neq t_2.pId\}$ → $\forall t_1, t_2 : \neg(t_1.pId = t_2.pId)$

FastDC
Minimal Cover Search

DFS

	count
p_2	4
p_4	4
p_{14}	4
p_{11}	3
…	…

p_1	p_3	p_4	p_5	p_6	p_7	p_8	p_9	p_{10}	p_{11}	p_{12}	p_{13}	p_{14}
-	-	+	+	-	-	+	-	+	+	-	-	+
-	-	+	+	-	-	+	-	+	-	+	-	+
-	-	+	-	+	+	-	-	+	+	-	-	+
-	-	+	-	+	-	+	+	-	+	-	-	+

- $\{p_4\} = \{t_1.partner \neq t_2.partner\}$
  → $\forall t_1, t_2: \neg(t_1.partner = t_2.partner)$
- $\{p_{14}\} = \{t_1.pId \neq t_1.partner\}$
  → $\forall t_1,: \neg(t_1.pId = t_1.parnter)$

FastDC
Minimal Cover Search

DFS

	count
p_2	4
p_4	4
p_{14}	4
p_{11}	3
…	…

p_1	p_3	p_5	p_6	p_7	p_8	p_9	p_{10}	p_{11}	p_{12}	p_{13}
-	-	+	-	-	+	-	+	+	-	-
-	-	+	-	-	+	-	+	-	+	-
-	-	-	+	+	-	-	+	+	-	-
-	-	-	+	-	+	+	-	+	-	-

Filter

p_1	p_3	p_5	p_6	p_7	p_8	p_9	p_{10}	p_{11}	p_{12}	p_{13}
-	-	+	-	-	+	-	+	-	+	-

- $\{p_{11}, p_{12}\} \rightarrow \forall t_1, t_2 : \neg(t_1.pId = t_2.partner \wedge t_1.pId \neq t_2.partner)$
- $\{p_{11,}, p_5\} \rightarrow \forall t_1, t_2 : \neg(t_1.pId = t_2.partner \wedge t_1.hInc \neq t_2.hInc)$

FastDC
Minimal Cover Search

Algorithm 4 SEARCH MINIMAL COVERS

Input: 1. Input Evidence set, Evi_I
2. Evidence set not covered so far, Evi_{curr}
3. The current path in the search tree, $\mathbf{X} \subseteq \mathbf{P}$
4. The current partial ordering of the predicates, $>_{curr}$
5. The DCs discovered so far, Σ

Output: A set of minimal covers for Evi, denoted as ***MC***

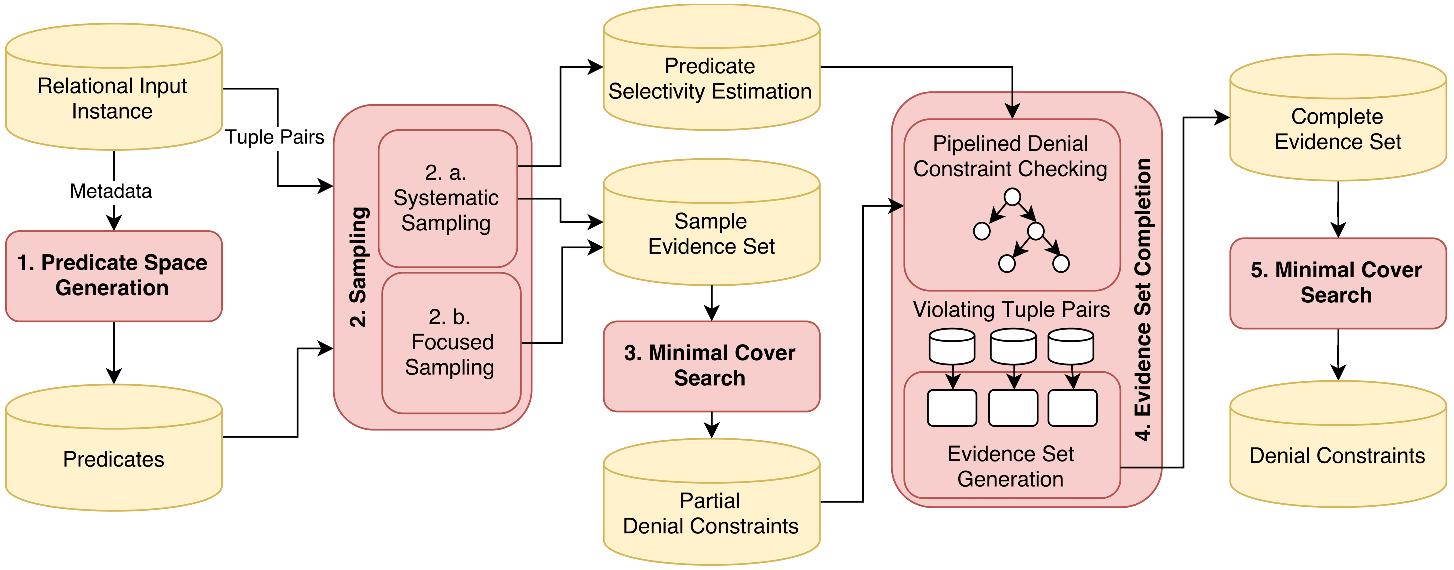
<u>*Branch Pruning*</u>
$P \leftarrow \mathbf{X}.last$ // Last Predicate added into the path
if $\exists Q \in \mathbf{X} - P$, s.t. $P \in Imp(Q)$ **then**
  return //Triviality pruning
if $\exists \mathbf{Y} \in \boldsymbol{MC}$, s.t. $\mathbf{X} \supseteq \mathbf{Y}$ **then**
  return //Subset pruning based on ***MC***
if $\exists \mathbf{Y} = \{Y_1, \dots, Y_n\} \in \boldsymbol{MC}$, and $\exists i \in [1, n]$, and $\exists Q \in Imp(Y_i)$, s.t. $\mathbf{Z} = \mathbf{Y}_{-i} \cup \overline{Q}$ and $\mathbf{X} \supseteq \mathbf{Z}$ **then**
  return //Transitive pruning based on ***MC***
if $\exists \varphi \in \Sigma$, s.t. $\overline{\mathbf{X}} \supseteq \varphi.Pres$ **then**
  return //Subset pruning based on previous discovered DCs
if $Inter(\varphi) < t, \forall \varphi$ of the form $\neg(\overline{\mathbf{X}} \wedge \mathbf{W})$ **then**
  return //Pruning based on $Inter$ score
<u>*Base cases*</u>
if $>_{curr} = \emptyset$ and $Evi_{curr} \neq \emptyset$ **then**
  return //No DCs in this branch
if $Evi_{curr} = \emptyset$ **then**
  if no subset of size $|\mathbf{X}| - 1$ covers Evi_{curr} **then**
    $\boldsymbol{MC} \leftarrow \boldsymbol{MC} + \mathbf{X}$
  return //Got a cover
<u>*Recursive cases*</u>
for all Predicate $P \in >_{curr}$ **do**
  $\mathbf{X} \leftarrow \mathbf{X} + P$
  $Evi_{next} \leftarrow$ evidence sets in Evi_{curr} not yet covered by P
  $>_{next} \leftarrow$ total ordering of $\{P' | P >_{curr} P'\}$ wrt Evi_{next}
  SEARCH MINIMAL COVERS($Evi_I, Evi_{next}, \mathbf{X}, >_{next}, \Sigma$)
  $\mathbf{X} \leftarrow \mathbf{X} - P$

- In FastDC paper: more advanced search strategy that divides the space of DCs in multiple smaller subspaces

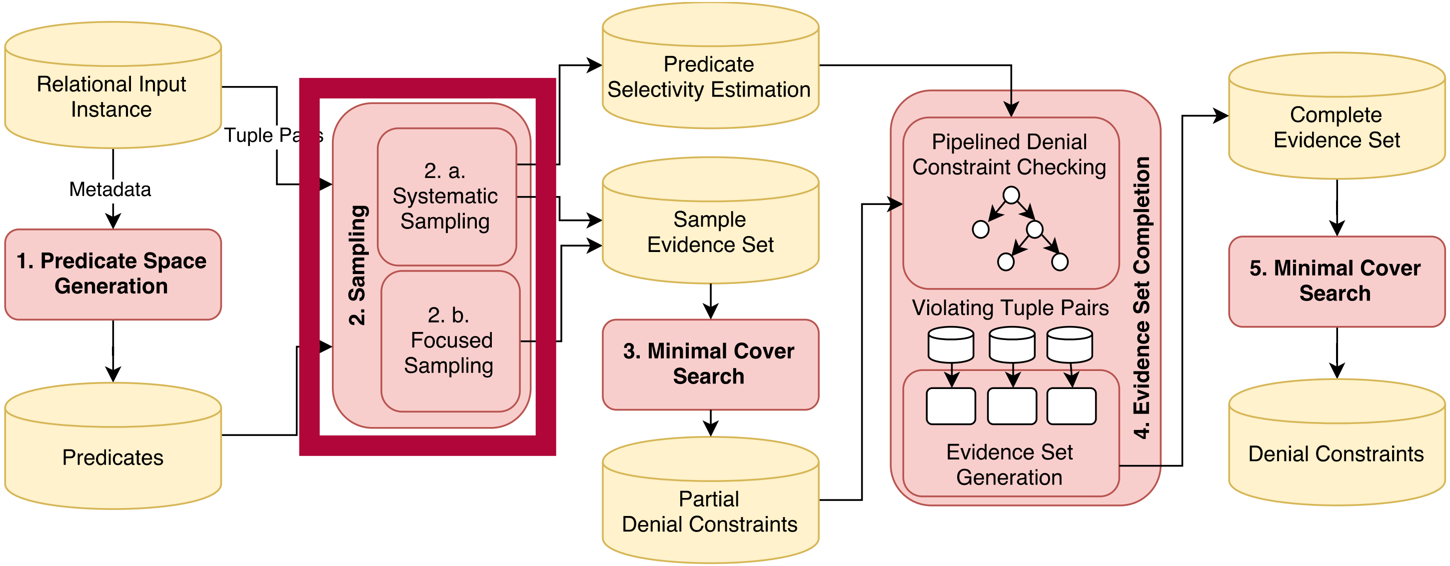
FastDC
A-FastDC and C-FastDC

- **Approximate DCs: A-FASTDC**
  - Why? Overfitting and data errors
  - What? Consider a DC valid even if a small percentage of tuple pairs violates the DC
  - How? Count number of occurrences per evidence, small adjustments to the minimal cover search to allow small error threshold
- **Constant DCs: C-FASTDC**
  - Why? DC might not hold on the entire dataset
  - What? Introduce predicates that compare attribute values to constants
  - How? Allow frequent constants in predicates

Hydra
Overview



Hydra Overview



Hydra
Focused Sampling

Problem:

Quadratic complexity of evidence set generation in FastDC

Suggested solution:

Sampling of tuple pairs

Aim:

Complete as possible evidence set in a short amount of time

Remember:

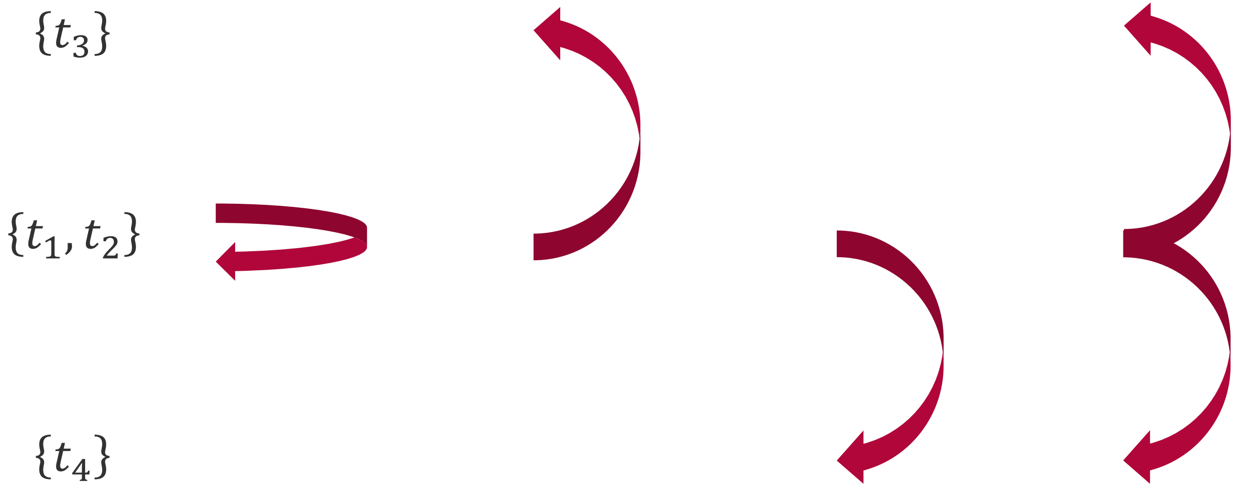
In FD discovery: only compare tuples that share at least one value

Hydra
Focused Sampling

Column A WITHIN LESS GREATER OTHER

A
2.0
2.0
1.4
2.1

Sorted PLI



■ $t_1.A = t_2.A$ ■ $t_1.A < t_2.A$ ■ $t_1.A > t_2.A$ ■ $t_1.A \neq t_2.A$

Hydra
Focused Sampling

	A	B
t_1	2.0	A
t_2	2.0	C
t_3	1.4	A
t_4	2.1	D

(Sorted) PLI

- A: $[\{t_3\}, \{t_1, t_2\}, \{t_4\}]$
- B: $\{\{t_2\}, \{t_4\}, \{t_1, t_3\}\}$

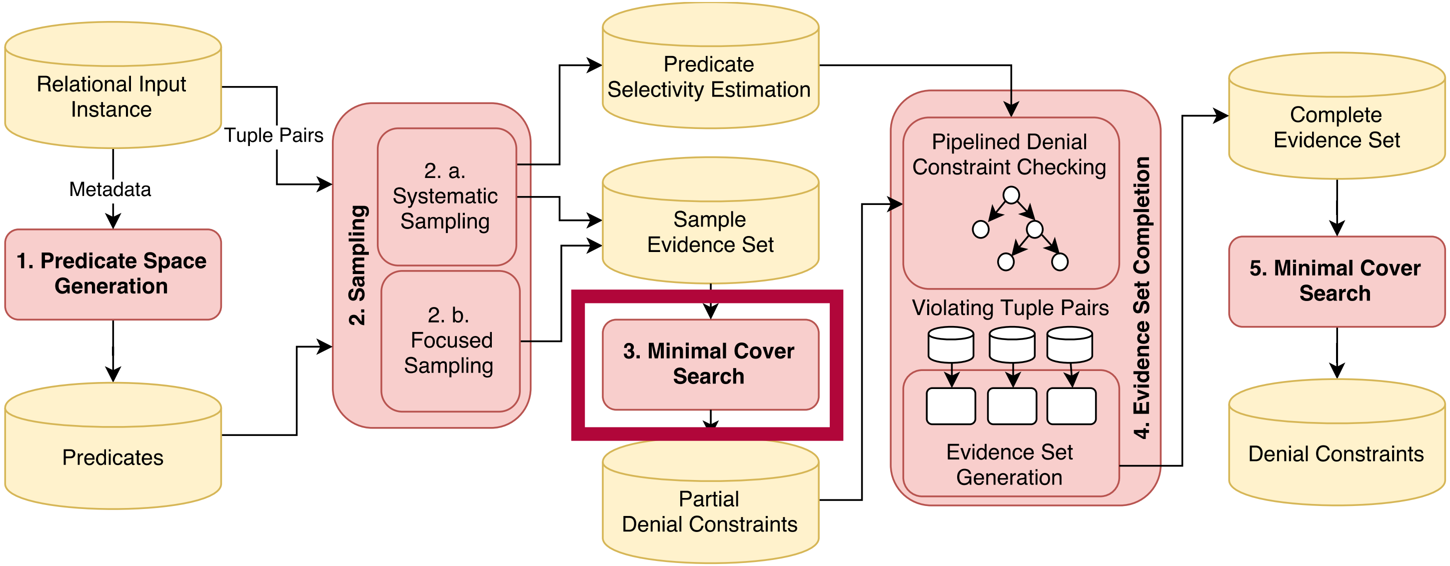
Strategies

- A: WITHIN, LESS, GREATER
- B: WITHIN, OTHER

General procedure:

- Execute all strategies once
- Measure efficiency (number of new elements in the evidence set)
- Store strategies in heap
- Execute best strategies until efficiency drops below threshold

Hydra Overview



Hydra
Minimal Cover Search

- Start with most general DCs (consisting of only one predicate)
- For each element in the evidence set:
  1. Get subsets of the evidence element
  2. Add predicate that is not present in the evidence element
  3. If still minimal add to the set of DCs again

- $\neg\{p_1\}$
- $\neg\{p_2\}$
- $\neg\{p_3\}$

$\{p_1, p_3\}$ →

- $\neg\{p_2\}$

$\{p_2, p_3\}$ →

- $\neg\{p_1 p_2\}$

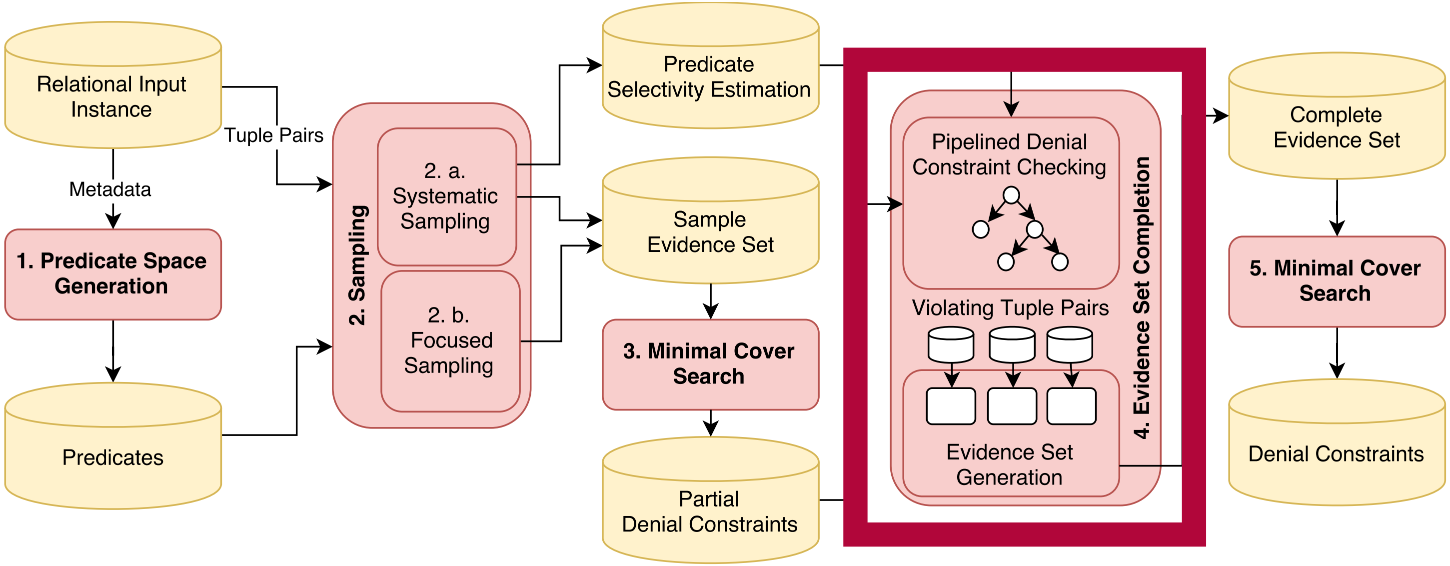
Hydra
Minimal Cover Search

Data: evidence set E, predicate space P
Result: the set of minimal, non-trivial DCs Ψ

$\Psi \leftarrow \{\{p\} \mid p \in P\}$
for $e \in E$ **do**
    `handleEvidence` (e, Ψ, P)
return Ψ
Function `handleEvidence`(e, Ψ, P)
    $\Psi^- \leftarrow \{\psi^- \in \Psi \mid \psi^- \subseteq e\}$
    $\Psi \leftarrow \Psi \setminus \Psi^-$
    for $\psi^- \in \Psi^-$ **do**
        for $p \in (P \setminus e)$ **do**
            if $\nexists \psi \in \Psi : \psi \subseteq (\psi^- \cup \{p\})$ **then**
                $\Psi \leftarrow \Psi \cup \{\psi^- \cup \{p\}\}$

- Start with most general DCs (consisting of only one predicate)
- For each element in the evidence set:
    1. Get subsets of the evidence element
    2. Add predicate that is not present in the evidence element
    3. If still minimal add to the set of DCs again

Hydra Overview



Hydra
Evidence Set Completion

- Sampling of tuple pairs
  → may not have found all evidence elements
- Discovered denial constraints might be violated
- Search for **all violating tuple pairs** of found denial constraints
  - Compute evidence set on those tuple pairs
  - Result: complete evidence set

- Final step: another cover inversion
  - Result: complete set of correct, minimal DCs

Hydra
Evidence Set Completion

- **Why does this result in a complete evidence set?**
  - Each element E in the evidence set contradicts one (minimal or not) DC x that consists of all predicates in E
    - $\{t_1.A < t_1.B, t_1.A = t_2.B\}$ → x: $\forall t_1, t_2: \neg(t_1.A < t_1.B \wedge t_1.A = t_2.B)$
  - Assume one element was not found during the sampling
    → Incomplete evidence set
  - Cover inversion is complete
    → Intermediate result must contain a DC y that implies x
    - E.g. y: $\forall t_1, t_2: \neg(t_1.A = t_2.B)$
  - Checking y returns tuple pair that yields the missing element

Hydra
Evidence Set Completion

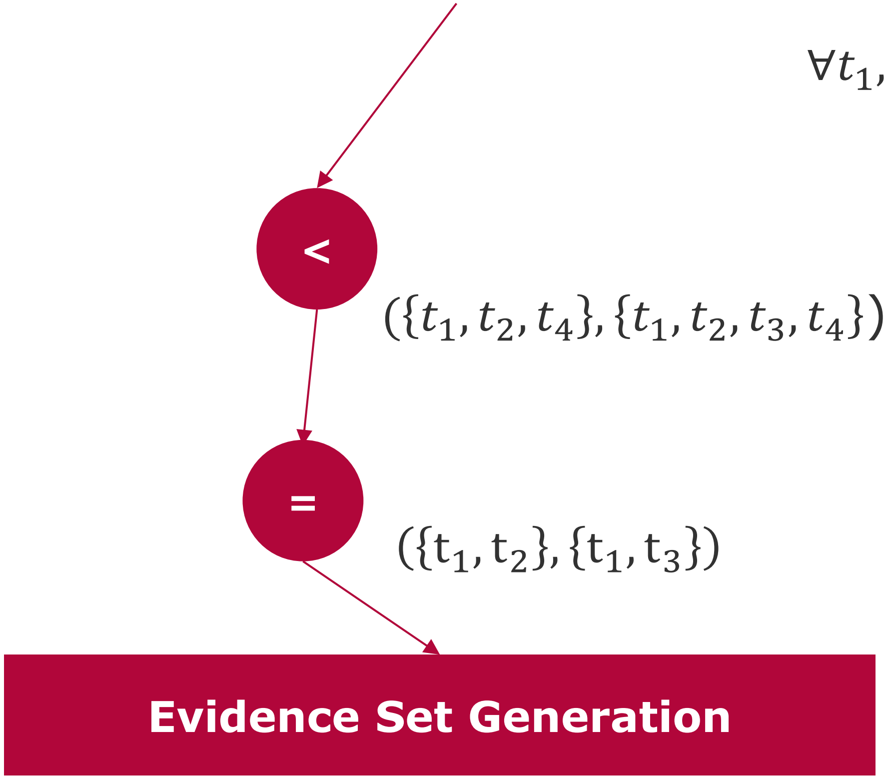
- **Data structures**
  - Cluster: set of tuples $\{t_1, t_2\}$
  - Cluster Pair: combination of two clusters that represents cross-product
  - $(\{t_1, t_2\}, \{t_3, t_4\})$ represents $\{(t_1, t_3), (t_2, t_3), (t_1, t_4), (t_2, t_4)\}$

- Start with “complete” cluster pair of clusters containing all tuples
- Partition refinement: only keep tuple pairs that fulfill the current predicate
- **Specialized algorithms for:**
  - Filters $t_1.A = t_1.B$
  - Equi-joins $t_1.A = t_2.B$
  - Anti-joins $t_1.A \neq t_2.B$
  - Inequality joins $t_1.A < t_2.B$
  - Pairs of inequality joins → IEJoin

Hydra
Evidence Set Completion

$(\{t_1, t_2, t_3, t_4\}, \{t_1, t_2, t_3, t_4\})$

$\forall t_1, t_2 \colon \neg (t_1.A < t_1.B \wedge t_1.A = t_2.B)$



$t_1.A < t_1.B$

$(\{t_1, t_2, t_4\}, \{t_1, t_2, t_3, t_4\})$

$t_1.A = t_2.B$

$(\{t_1, t_2\}, \{t_1, t_3\})$

Hydra
Example: Equi-join Predicates

- $t_1.A = t_2.B$
- **Input:** $(\{t_1, t_2, t_4\}, \{t_1, t_2, t_3, t_4\})$

	A	B
t_1	1	1
t_2	1	3
t_3	2	1
t_4	2	2

1. Index LHS according to column A:
   - 1: $\{t_1, t_2\}$, 2: $\{t_4\}$
2. Index RHS values of column B (if present in index of A):
   - 1: $\{t_1, t_3\}$, 2: $\{t_4\}$

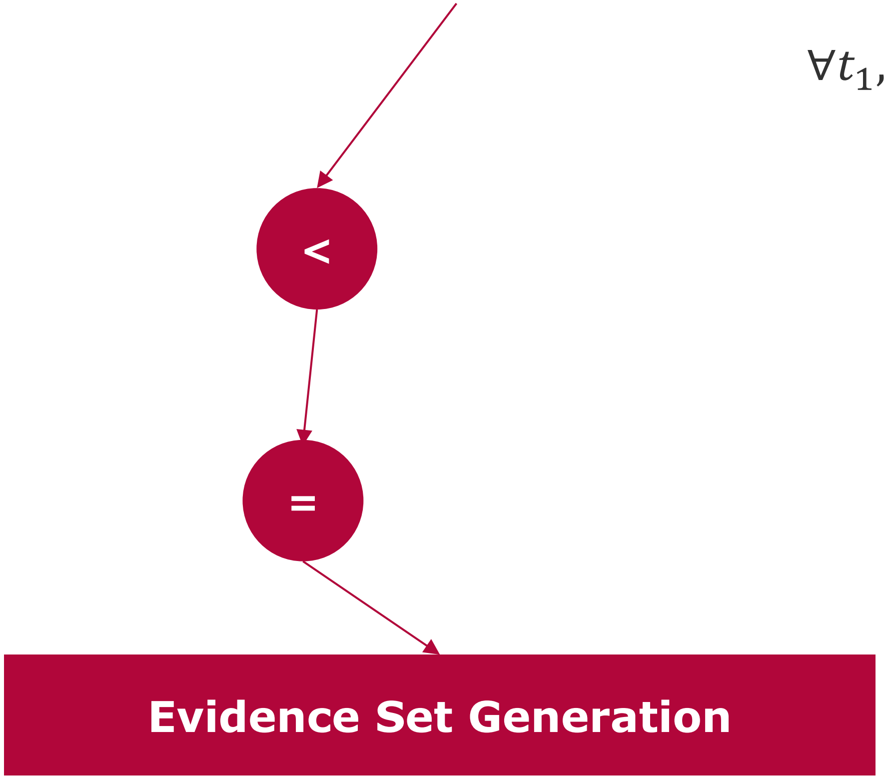
Result:

- $(\{t_1, t_2\}, \{t_1, t_3\})$
- $(\{t_4\}, \{t_4\})$ ← can be skipped!

Hydra
Evidence Set Completion

$$(\{t_1, t_2, t_3, t_4\}, \{t_1, t_2, t_3, t_4\})$$

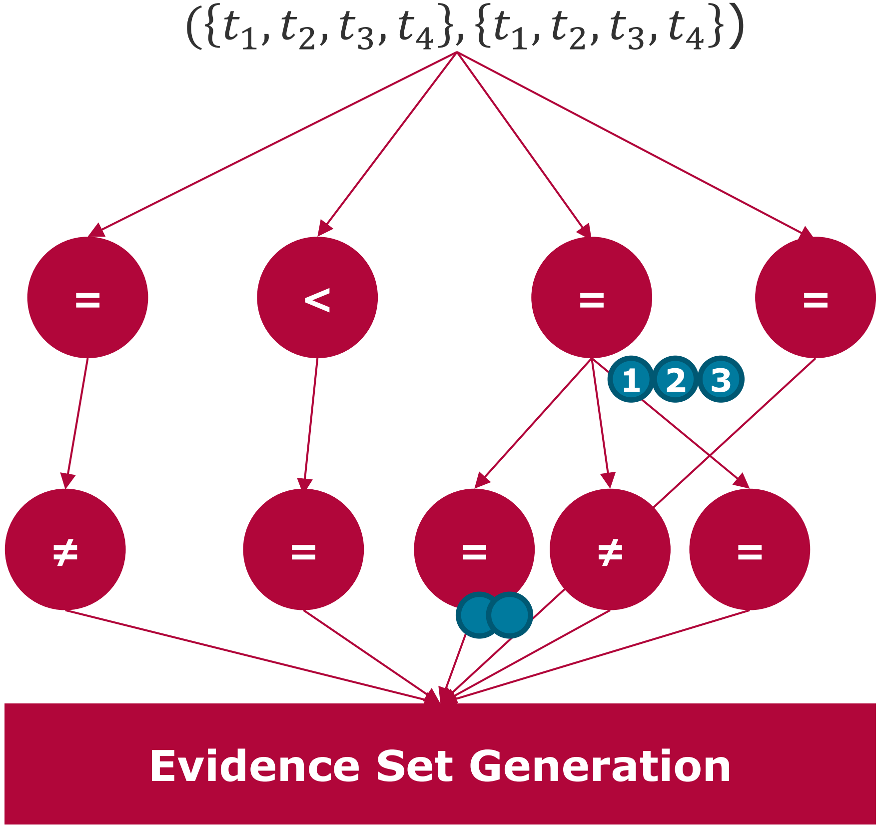
$$\forall t_1, t_2 : \neg(t_1.A < t_1.B \land t_1.A = t_2.B)$$



$$t_1.A < t_1.B$$

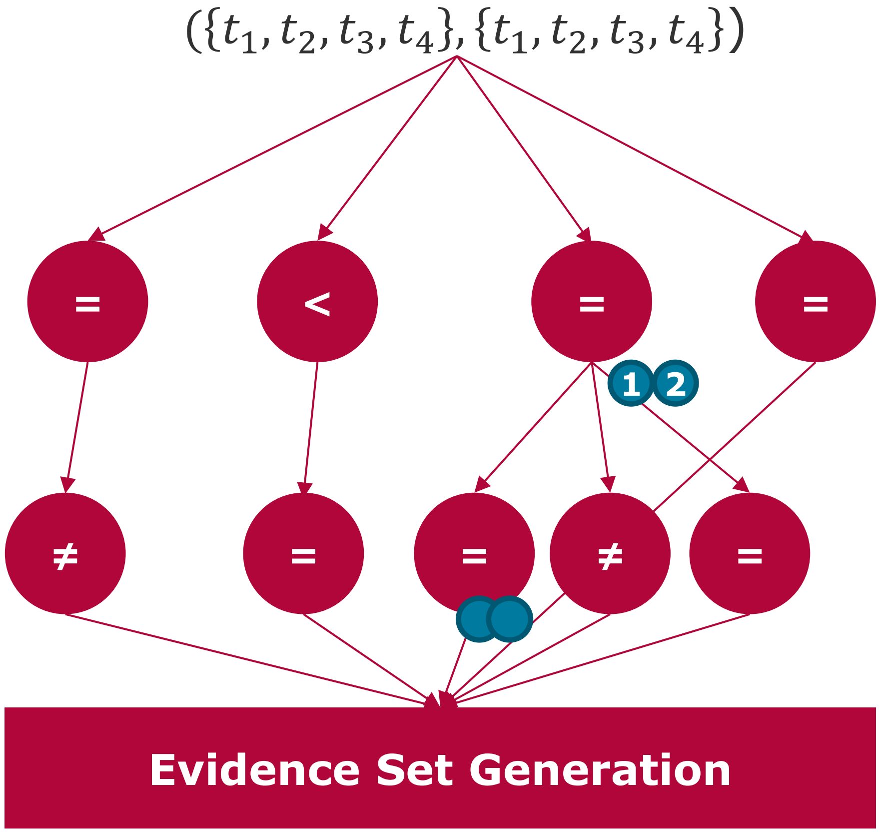
$$t_1.A = t_2.B$$

Hydra
Evidence Set Completion



Combine common prefix of predicates

Hydra
Evidence Set Completion

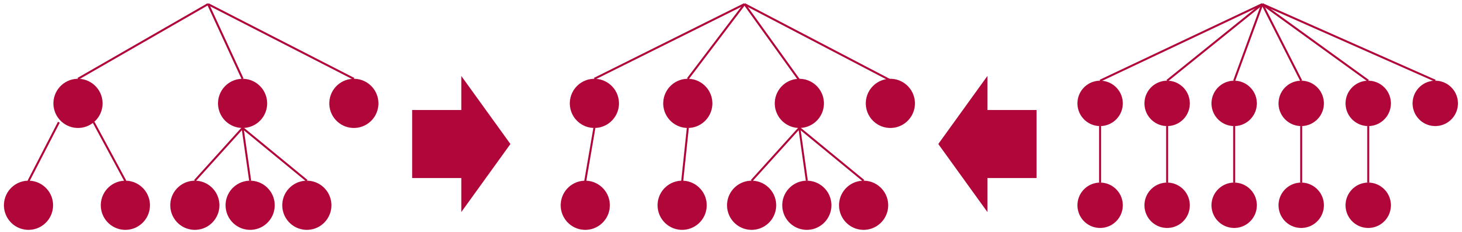


Avoid materialization of intermediate results

→ **Pipelining**

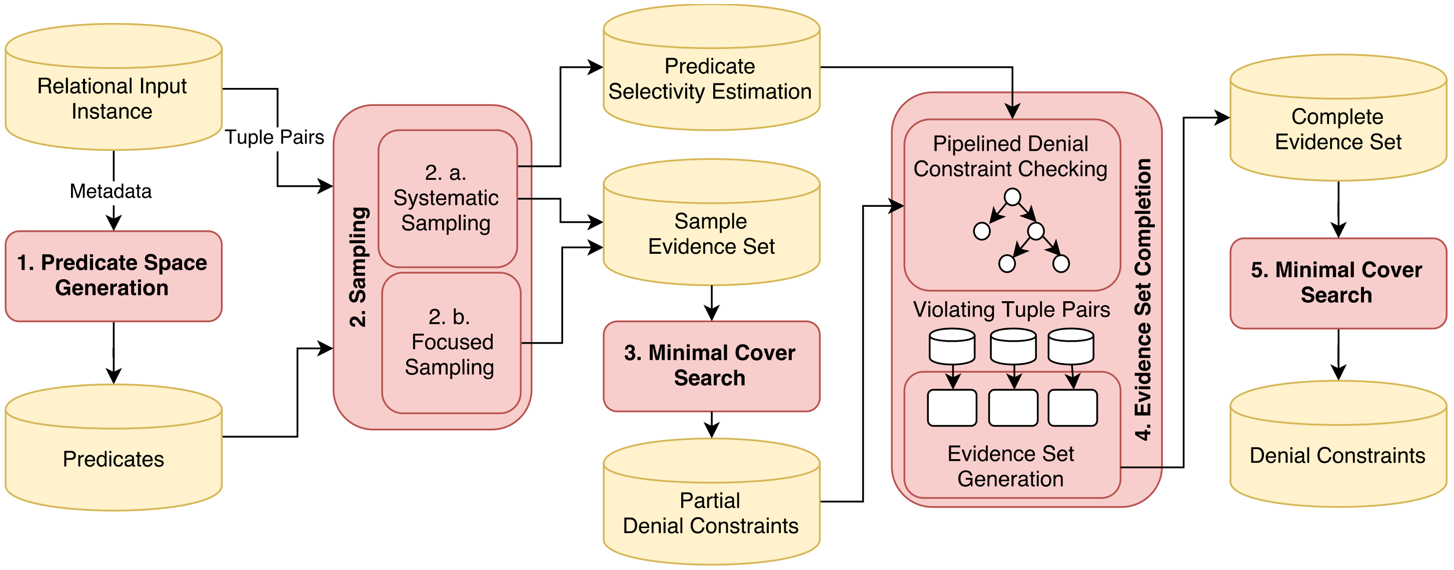
Hydra
Evidence Set Completion

Frequency only	Combined Frequency and Selectivity	Selectivity only

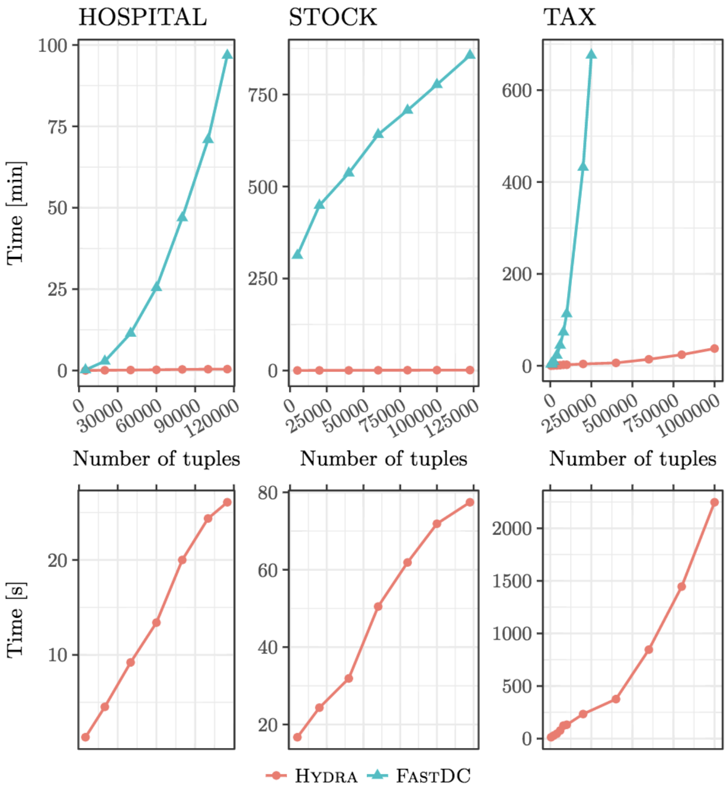


▪ Smaller tree ▪ But: lower nodes get evaluated more often	▪ PROFIT!	▪ Bigger tree ▪ But: smaller number of evaluations for lower nodes

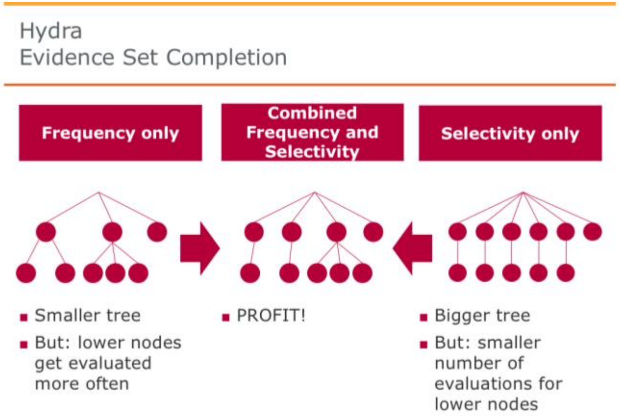
Hydra
Overview



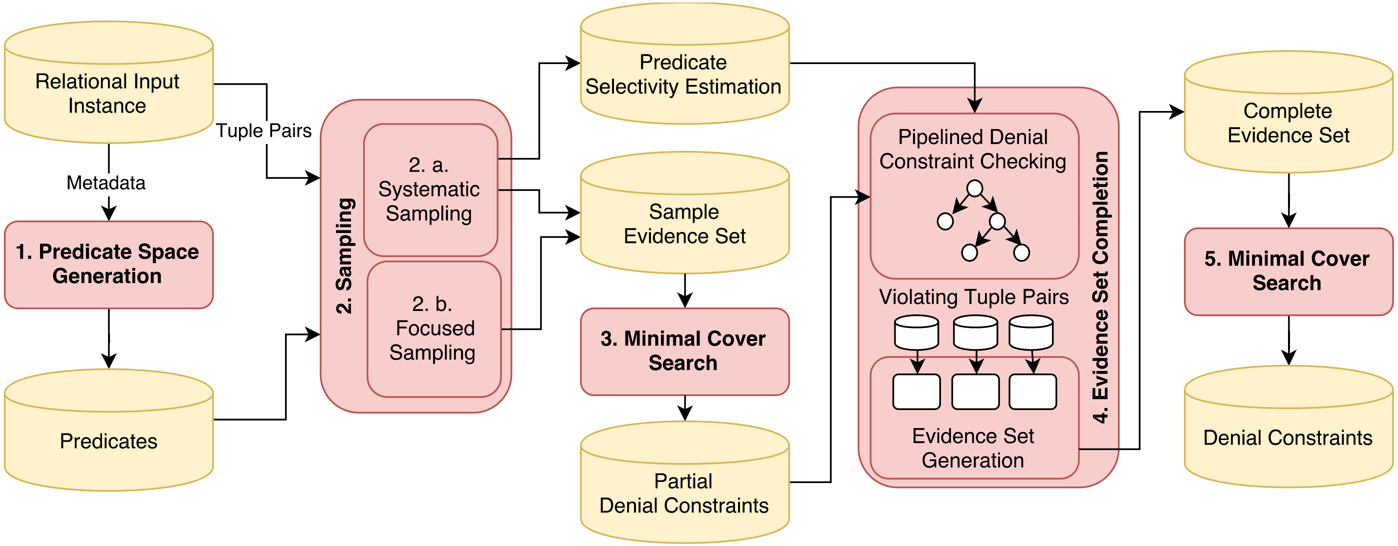
FastDC and Hydra
Scalability in the number of tuples



Evaluation
Evidence Set Completion



	HOSPITAL	STOCK	TAX
Selectivity Only	7.5s	422.8s	$> 2h$
Selectivity / Frequency	**6.1s**	**36.5s**	**38m**
Frequency Only	81.6s	$> 2h$	$> 2h$



Hasso Plattner Institut

IT Systems Engineering | Universität Potsdam

Denial Constraints

Tobias Bleifuß

17.07.2017