PresQ: Automatic Discovery of Equally-Distributed Attributes

Alejandro Álvarez Ayllón

October 31, 2022

UNIVERSITÉ DE GENÈVE

Outline

Introduction

Imagine you have two or more catalogs you need to cross-match, but they are `CSV` files without headers. How do you do it?

R				S			
A	**B**	**C**	...	**M**	**N**	**O**	...
0.247	5.944	10.451	...	0.850	5.107	10.844	...
0.752	5.846	10.758	...	0.698	5.132	10.429	...

We want to find a *dependency* between two datasets, where attributes are *equally distributed*: **Equally-Distributed Dependencies** ($\overset{d}{=}$)

1EDD

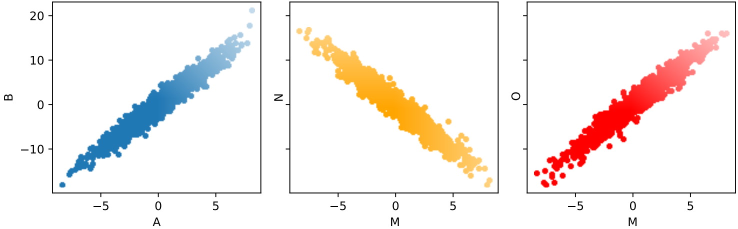
The obvious solution is to run pairwise statistical test between attributes from dataset **R** and **S**.

- **A** vs {**M**, **N**, **O**}
- **B** vs {**M**, **N**, **O**}
- **C** vs {**M**, **N**, **O**}

If R has n attributes, and S has m, $O(n \times m)$

2EDD

This has obvious drawbacks, so we can not stop there:



- If $A \stackrel{d}{=} M$, $B \stackrel{d}{=} N$, $B \stackrel{d}{=} O$, we need to test $(A, B) \stackrel{d}{=} (M, N)$ and $(A, B) \stackrel{d}{=} (M, O)$.
- What about 3EDD?

Inclusion Dependencies

This problem is similar to the *Inclusion-Dependency Search* (hence the name EDD). We need three inference rules to build on existing solutions [DLP02]:

Inference rules

Reflexivity

$$R[X] \stackrel{d}{=} R[X]$$

Permutation and projection

If $R[A_1, \ldots, A_n] \stackrel{d}{=} S[B_1, \ldots, B_n]$ then $R[A_{i_1}, \ldots, A_{i_m}] \stackrel{d}{=} S[B_{i_1}, \ldots, B_{i_m}]$ for each sequence $i_1, \ldots, i_m$ of distinct integers from $\{1, \ldots, n\}$

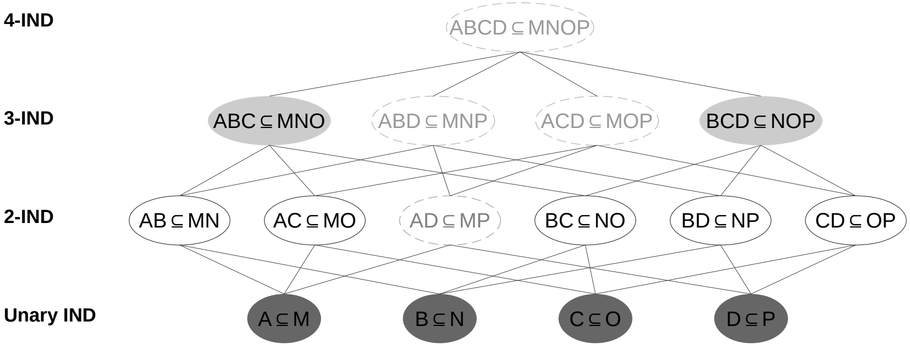
Transitivity

$$R[X] \stackrel{d}{=} S[Y] \wedge S[Y] \stackrel{d}{=} T[Z] \implies R[X] \stackrel{d}{=} T[Z]$$

Luckily, we can use them! [RW79]

Bottom-Up Search

The inference rules are essential to define a *partial order relation*, called specialization, between EDDs



Bottom-Up Search

Complexity

If we have an n-EDD between two datasets, and traverse the lattice bottom-up, we need to test...

$$\sum_{k=0}^{n} \binom{n}{k} = \sum_{k=0}^{n} \frac{n!}{n!(n-k)!}$$

possible combinations of attributes.


D'OH!

Complexity

It turns out that finding INDs/EDDs is one of the hardest computer science problems [BFS17].

- It is **NP-complete**.
- The **number of solutions** can be **exponential** in the input size.
- It is **non approximable** (NP-complete even if we accept an error margin).

However, the run-time can be reasonable if we are optimistic: i.e. if we assume n 1EDD are derived from a single nEDD, we only need one test!: top-down traversal.

But we risk being too optimistic...

Clique Finding

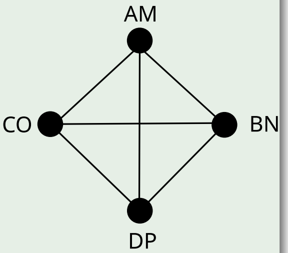
The problem can be mapped to finding cliques on hypergraphs (FIND2 [KR03]):

1. Pairwise matches (1EDD) are mapped to **nodes**.
2. *k*-combinations of 1EDD are mapped to *k*-**edges**.
3. Cliques of size *n may* correspond to *n*EDD.
4. If they are not, we *break* these cliques into a set of $k+1$ edges and validate them.

Clique Finding

Example

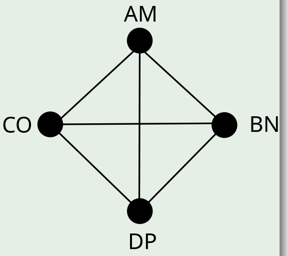
1. $A \stackrel{d}{=} M, B \stackrel{d}{=} N, C \stackrel{d}{=} O, D \stackrel{d}{=} P$
2. All 6 possible 2-EDD are true
3. Clique with 4 nodes $ABCD \stackrel{d}{=} MNOP$
4. If false, we break it into
    1. $ABC \stackrel{d}{=} MNO$
    2. $BCD \stackrel{d}{=} NOP$
    3. $ACD \stackrel{d}{=} MOP$
    4. ...



Clique Finding

Example

1. $A \stackrel{d}{=} M, B \stackrel{d}{=} N, C \stackrel{d}{=} O, D \stackrel{d}{=} P$
2. All 6 possible 2-EDD are true
3. Clique with 4 nodes $ABCD \stackrel{d}{=} MNOP$
4. If false, we break it into
   1. $ABC \stackrel{d}{=} MNO$
   2. $BCD \stackrel{d}{=} NOP$
   3. $ACD \stackrel{d}{=} MOP$
   4. ...



But we are using statistical tests.

- We may falsely refuse an EDD (type I error)
- Or falsely accept it (type II error).

Quasi-Clique Finding

Is a clique with missing edges, which can be limited as a ratio of the total (1) or as a ratio of the degree of a node (2).

Definition

Given a k-uniform hypergraph (V, E), and two parameters $\lambda, \gamma \in [0, 1]$, the sub-graph $H' = (V', E')$ induced by a subset $V' \subseteq V$ is a $(\lambda - \gamma)$ quasi-clique iff:

$$|E'| \geq \gamma \cdot \binom{|V'|}{k} \tag{1}$$

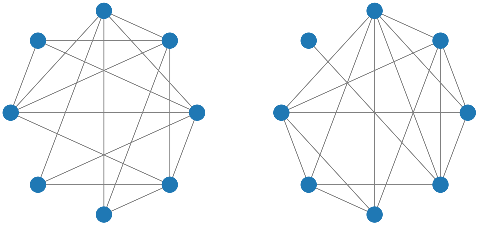
$$\forall v \in V' : deg_{V'}(v) \geq \lambda \cdot \binom{|V'| - 1}{k - 1} \tag{2}$$

Where $deg_{V'}(v)$ represents the degree of v, and E' is a subset of E such that $\forall e \in E' : e \subseteq V'$

We can approximate $\gamma \approx 1 - \alpha$. How do we bound λ ?

Quasi-Clique Finding

There is no reason to think that any particular subset of the edges has a higher probability of having missing members. If a given node has an unexpectedly low degree, it is most likely connected by spurious edges.
The degree of the nodes should follow a hypergeometric distribution:



PresQ [ÁPD22] is an algorithm for finding quasi-cliques on uniform k-hypergraphs.

1. Finds "seeds" using a modified version of Hyperclique [KR03].
2. Grows the "seeds" following a tree-shaped, depth-first traversal [Uno10].

Complexity

The number of maximal cliques is bound in general by $\Omega(a^{|V|/b})$, where a, b are two constants that depend on the rank of the hypergraph [Tom81].
The worst-case is always going to be exponential for this problem.

PresQ

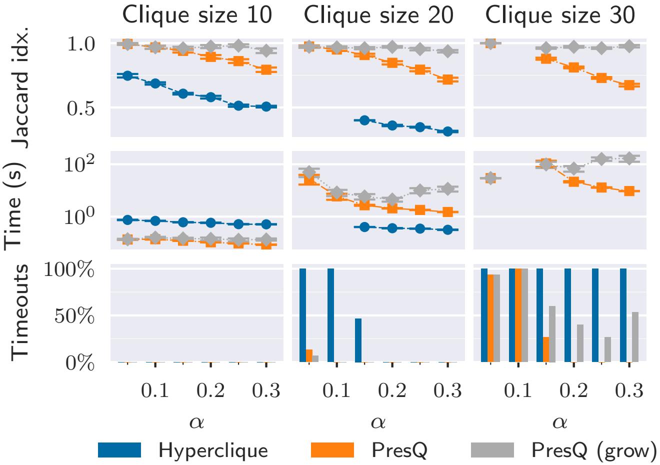


Figure: Robustness wrt missing edges (3-hypergraph)

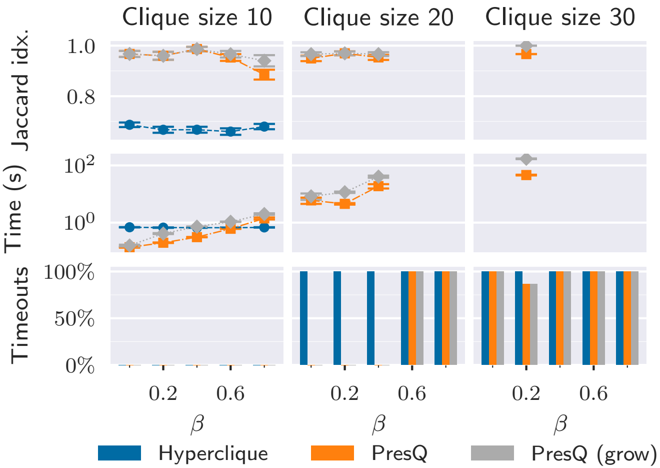
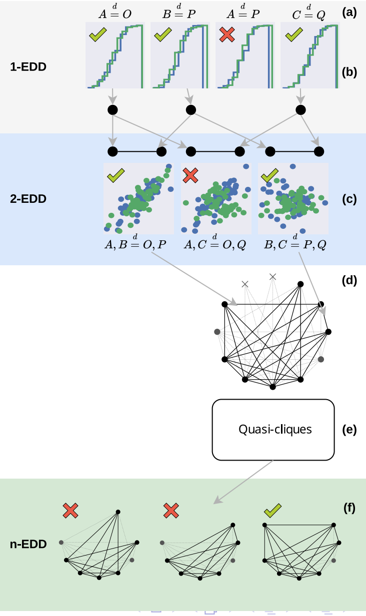


Figure: Robustness wrt spurious edges (3-hypergraph)

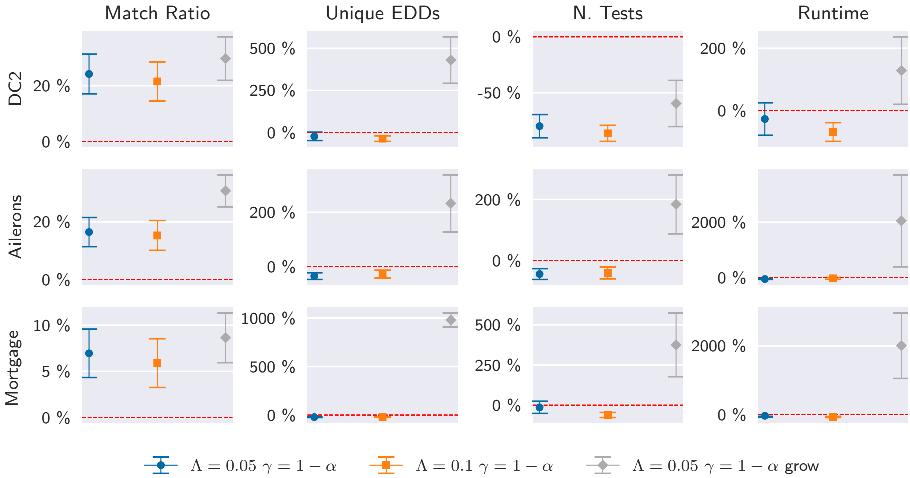
PresQ for EDD Finding

Similarly to the IND algorithm:

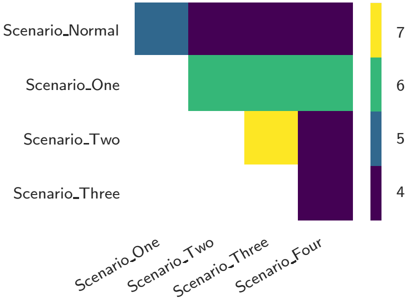
- (a) Generate candidate 1EDD.
- (b) If $\stackrel{d}{=}$ can not be rejected, map to nodes.
- (c) Test all pairwise combinations (2EDD).
- (d) If $\stackrel{d}{=}$ can not be rejected, map to edges on a k-hypergraph.
- (e) Search for quasi-cliques.
- (f) Validate quasi-cliques.
- (g) Rejected quasi-cliques are used to generate a $k+1$-hypergraph.
- (h) Go-to e.



PresQ for EDD Finding

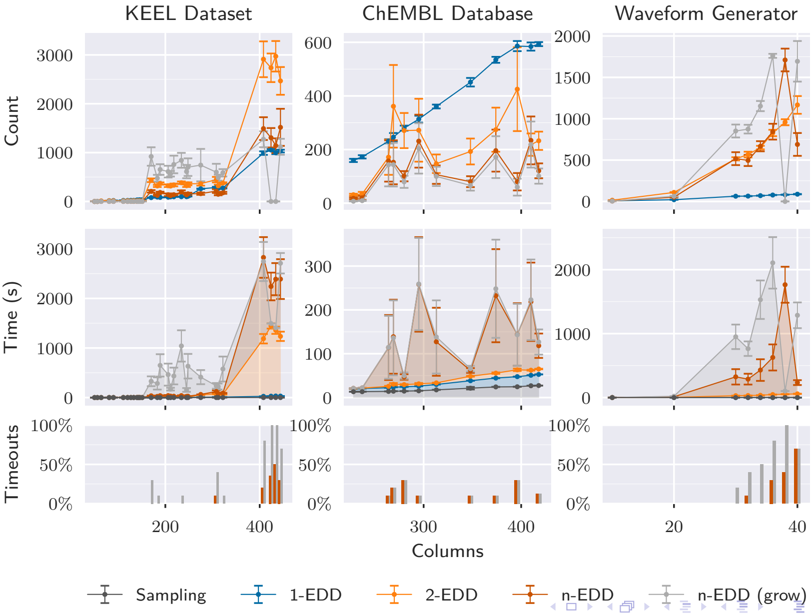


PresQ for EDD Finding



Aircraft Fuel Distribution System (AFDS) comprises five different files. They all can be processed at once. Without even knowing the structure or content of the files, we can see that scenarios two and three are the most similar, which we can confirm on the original paper [Ghe+19].

PresQ for EDD Finding



Summary

Finding sets of *Equally-Distributed Dependencies*

- Can be mapped to finding Quasi-Cliques on Hypergraphs.
- It is an NP-Complete problem.

- The run-time depends on the *output size* (number of cliques and their size) which can be *exponential* on the input size.

- The run-time is acceptable for moderate output sizes.

Thank you!

a.alvarezayllon@gmail.com

Bibliography

[ÁPD22] Alejandro Álvarez-Ayllón, Manuel Palomo-Duarte, and Juan-Manuel Dodero. "PresQ: Discovery of Multidimensional Equally-Distributed Dependencies Via Quasi-Cliques on Hypergraphs". In: (2022). DOI: 10.1109/TETC.2022.3198252.

[BFS17] Thomas Bläsius, Tobias Friedrich, and Martin Schirneck. "The Parameterized Complexity of Dependency Detection in Relational Databases". In: 2017. DOI: 10.4230/LIPIcs.IPEC.2016.6.

[DLP02] Fabien De Marchi, Stéphane Lopes, and Jean-Marc Petit. "Efficient algorithms for mining inclusion dependencies". In: 2002. DOI: 10.1007/3-540-45876-X_30.

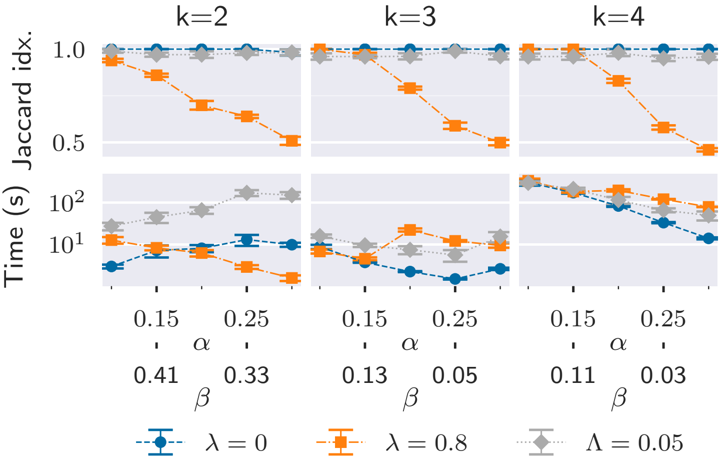
[Ghe+19] Youcef Gheraibia et al. "Safety+AI: a novel approach to update safety models using artificial intelligence". In: (2019). DOI: 10.1109/ACCESS.2019.2941566.

[KR03] Andreas Koeller and Elke A Rundensteiner. "Discovery of high-dimensional inclusion dependencies". In: 2003. DOI: 10.1109/ICDE.2003.1260834.

[RW79] Ronald H Randles and Douglas A Wolfe. *Introduction to the theory of nonparametric statistics*. Tech. rep. John Wiley, 1979.

Backup Slides

Quasi-clique for both α, β



DC2

DC2

	Λ	γ	Time (s)		Match		Unique		Prec.	N	Timeouts
FIND2			74.94	805.71	0.60	0.71	73	150	0.90	53	34.0%
PRESQ	**0.00**	0.9	681.51	1536.19	0.68	0.69	102	200	0.01	16	87.5%
PRESQ	0.05	**0.0**	40.07	189.45	0.80	0.93	46	115	0.10	21	47.6%
PRESQ	0.05	0.9	25.57	214.27	0.76	0.89	46	113	0.14	53	13.2%
PRESQ	0.10	0.9	18.61	144.98	0.76	0.87	42	98	0.18	52	23.1%
PRESQ(G)	0.05	0.9	458.26	1881.02	0.81	0.93	518	798	0.23	52	50.0%

Scalability wrt Sample Size