On the Complexity of Mining Itemsets from the Crowd Using Taxonomies

Antoine Amarilli[1,2] Yael Amsterdamer[1] Tova Milo[1]

[1]Tel Aviv University, Tel Aviv, Israel

[2]École normale supérieure, Paris, France

Data mining

Data mining – discovering interesting patterns in large databases

Database – a (multi)set of transactions

Transaction – a set of items (aka. an itemset)

A simple kind of pattern to identify are frequent itemsets.

$$
\begin{aligned}
D = \{ & \\
& \{\texttt{beer}, \texttt{diapers}\}, \\
& \{\texttt{beer}, \texttt{bread}, \texttt{butter}\}, \\
& \{\texttt{beer}, \texttt{bread}, \texttt{diapers}\}, \\
& \{\texttt{salad}, \texttt{tomato}\} \\
\} &
\end{aligned}
$$

- An itemset is frequent if it occurs in at least $\Theta = 50\%$ of transactions.
- $\{\texttt{salad}\}$ is not frequent.
- $\{\texttt{beer}, \texttt{diapers}\}$ is frequent. Thus, $\{\texttt{beer}\}$ is also frequent.

Human knowledge mining

- What if the database doesn't really exist?

Things to do in Athens:

$$D = \{ \{\texttt{icdt}, \texttt{monday}, \texttt{laptop}\}, \{\texttt{acropolis}, \texttt{sunglasses}\}, \ldots \}$$

Traditional medicine:

$$D = \{ \{\texttt{hangover}, \texttt{coffee}\}, \{\texttt{cough}, \texttt{honey}\}, \ldots \}$$

This data only exists in the minds of people!

Harvesting this data

- We cannot collect such data in a centralized database:
  1. It's impractical to ask all users to surrender their data.

     "Everyone please tell us all that you did the last three months."
  2. People do not remember the information.

     "What were you doing on August 23th, 2013?"
- However, people remember summaries that we could access.

  "Do you often play tennis on weekends?"
- We can just ask people if an itemset is frequent.

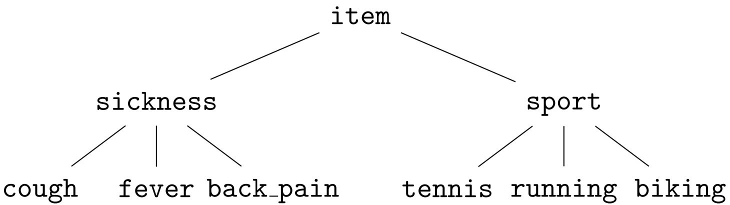
Crowdsourcing

- Crowdsourcing – solving hard problems through elementary queries to a crowd of users.
- Find out if an itemset is frequent with the crowd:
  1. Draw a sample of users from the crowd. *(black box)*
  2. Ask: is this itemset frequent? *("Do you often play tennis?")*
  3. Corroborate the answers to eliminate bad answers. *(black box)*
  4. Reward the users. *(e.g., monetary incentive)*

$\Rightarrow$ An oracle that takes an itemset and finds out if it is frequent or not by asking crowd queries.

Taxonomies

Having a taxonomy over the items can save us work!



- If $\{\texttt{sickness}, \texttt{sport}\}$ is infrequent then all itemsets such as $\{\texttt{cough}, \texttt{biking}\}$ are also infrequent.
- Without the taxonomy, we need to test all combinations!
- Also avoids redundant itemsets like $\{\texttt{sport}, \texttt{tennis}\}$.

Cost

How to evaluate the performance of a strategy to identify the frequent itemsets?

Crowd complexity: The number of itemsets we ask about (monetary cost, latency...)

Computational complexity: The complexity of computing the next question to ask

There is a tradeoff between the two:

- Asking random questions is computationally inexpensive but the crowd complexity is bad.
- Asking clever questions to obtain optimal crowd complexity is computationally expensive.

The problem

We can now describe the problem:

- We have:
  - A known item domain $\mathcal{I}$ (set of items).
  - A known taxonomy Ψ on $\mathcal{I}$ (is-a relation, partial order).
  - A crowd oracle freq to decide if an itemset is frequent or not.
- Choose interactively questions based on past answers.
- Balance crowd complexity and computational complexity.

$\Rightarrow$ Find out the status of all itemsets (learn freq exactly).

What is a good algorithm to solve this problem?

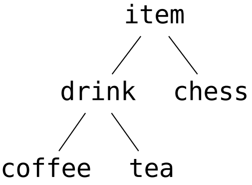
Table of contents

Itemset taxonomy

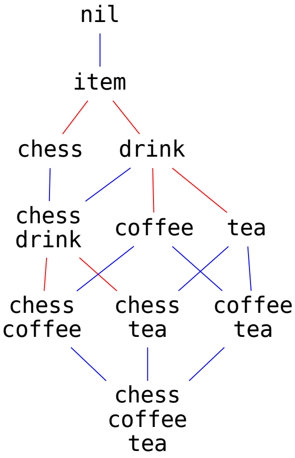
- Itemsets $\mathrm{I}(\Psi)$ – the sets of pairwise incomparable items. (e.g. $\{\texttt{coffee}, \texttt{tennis}\}$ but not $\{\texttt{coffee}, \texttt{drink}\}$)
- If an itemset is frequent then its subsets are also frequent.
- If an itemset is frequent then itemsets with more general items are also frequent.
- We define an order relation $\leqslant$ on itemsets: $A \leqslant B$ for "A is more general than B".
- Formally, $\forall i \in A,\ \exists j \in B$ s.t. i is more general than j.
- freq is monotone: if $A \leqslant B$ and B is frequent then A also is.

Itemset taxonomy example

Taxonomy Ψ



Itemset taxonomy $\mathrm{I}(\Psi)$

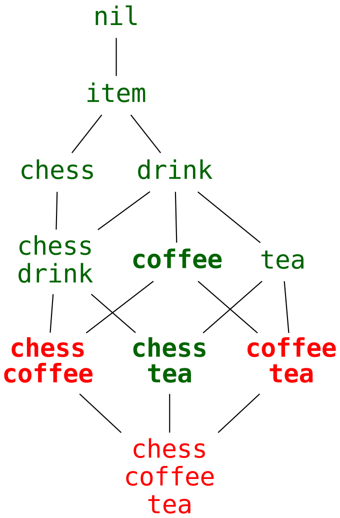


Solution taxonomy $\mathrm{S}(\Psi)$

Maximal frequent itemsets

- Maximal frequent itemset (MFI): a frequent itemset with no frequent descendants.
- Minimal infrequent itemset (MII).
- The MFIs (or MIIs) concisely represent freq.

⇒ We can study complexity as a function of the size of the output.



Solution taxonomy

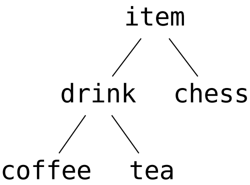
- Conversely, (we can show) any set of pairwise incomparable itemsets is a possible MFI representation.
- Hence, the set of all possible solutions has a similar structure to the "itemsets" over the itemset taxonomy $I(\Psi)$.

$\Rightarrow$ We call this the solution taxonomy $S(\Psi) = I(I(\Psi))$.

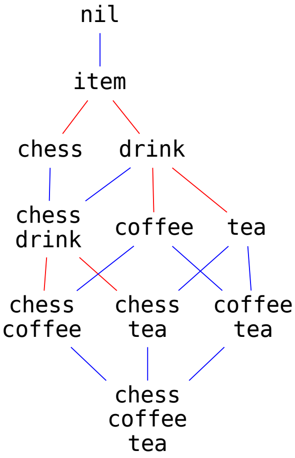
Identifying the freq predicate amounts to finding the correct node in $S(\Psi)$ through itemset frequency queries.

Solution taxonomy example

Taxonomy Ψ



Itemset taxonomy I(Ψ)



Solution taxonomy S(Ψ)

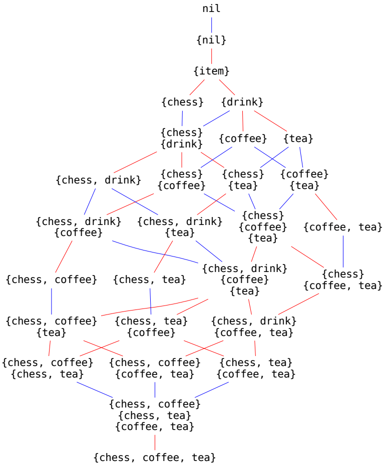


Table of contents

Lower bound

- Each query yields one bit of information.
- Information-theoretic lower bound: we need at least $\Omega(\log |S(\Psi)|)$ queries.
- This is bad in general, because $|S(\Psi)|$ can be doubly exponential in Ψ.
- As a function of the original taxonomy Ψ, we can write: $\Omega\left(2^{\mathrm{width}[\Psi]}/\sqrt{\mathrm{width}[\Psi]}\right)$.

Upper bound

- We can achieve the information-theoretic bound if is there always an unknown itemset that is frequent in about half of the possible solutions.
- A result from order theory shows that there is a constant $\delta_0 \approx 1/5$ such that some element always achieves a split of at least δ_0.
- Hence, the previous bound is tight: we need $\Theta(\log |S(\Psi)|)$ queries.

`nil`	$6/7$
`a1`	$5/7$
`a2`	$4/7$
`a3`	$3/7$
`a4`	$2/7$
`a5`	$1/7$

Lower bound, MFI/MII

- To describe the solution, we need the MFIs or the MIIs.
- However, we need to query both the MFIs and the MIIs to identify the result uniquely: $\Omega(|\mathsf{MFI}| + |\mathsf{MII}|)$ queries.
- We can have $|\mathsf{MFI}| = \Omega\left(2^{|\mathsf{MII}|}\right)$ and vice-versa.
- This bound is not tight (e.g., chain).

nil
|
a1
|
a2
|
a3
|
a4
|
a5

Upper bound, MFI/MII

- There is an explicit algorithm to find a new MFI or MII in $\leqslant |\mathcal{I}|$ queries.
- Intuition: starting with any frequent itemset, add items until you cannot add any more without becoming infrequent.
- The number of queries is thus $O(|\mathcal{I}| \cdot (|\text{MFI}| + |\text{MII}|))$.

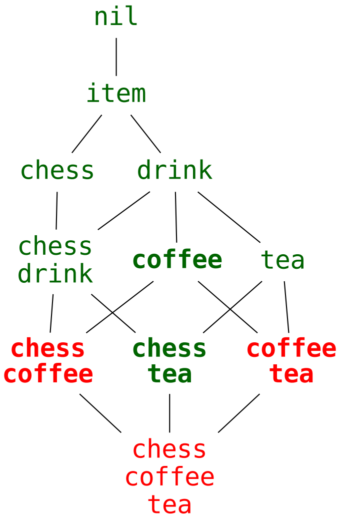


Table of contents

Hardness for standard (input) complexity

- We want an unknown itemset of $\mathrm{I}(\Psi)$ that is frequent for about half of the possible solutions of $\mathrm{S}(\Psi)$.
- We can count over $\mathrm{S}(\Psi)$ but it may be exponential in $|\mathrm{I}(\Psi)|$.
- Counting the antichains of $\mathrm{I}(\Psi)$ is $\mathrm{FP}^{\#\mathrm{P}}$-complete.
- Finding the best-split element in $\mathrm{I}(\Psi)$ is $\mathrm{FP}^{\#\mathrm{P}}$-hard in $|\mathrm{I}(\Psi)|$?
- Problem: $\mathrm{I}(\Psi)$ is not a general DAG, so we only show hardness in $|\Psi|$ for restricted (fixed-size) itemsets.
- Intuition: count antichains by comparing to a known poset; use a best-split oracle to compare; perform a binary search.

Hardness for output complexity

- In the incremental algorithm, materializing $I(\Psi)$ is expensive. Do we need to?
- Actually, how to decide if we can stop with our MFIs and MIIs?
- Proved EQ-hardness for problem EQ (exact complexity open).

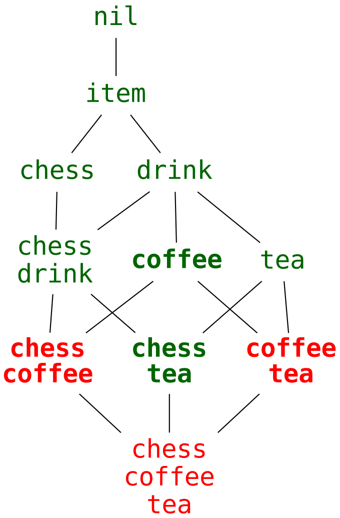


Table of contents

Summary and further work

We have studied the crowd and computational complexity of crowd mining under a taxonomy. What now?

- Improve the bounds and close gaps.
- Benchmark heuristics (chain partitioning, random, etc.).
- Integrate prior knowledge.
- Manage uncertainty (black box for now).
- Guide exploration with a query (under review).
- Work with numerical values for support.
- Mine more expressive patterns.
- Focus on top-k itemsets (work in progress).

Summary and further work

We have studied the crowd and computational complexity of crowd mining under a taxonomy. What now?

- Improve the bounds and close gaps.
- Benchmark heuristics (chain partitioning, random, etc.).
- Integrate prior knowledge.
- Manage uncertainty (black box for now).
- Guide exploration with a query (under review).
- Work with numerical values for support.
- Mine more expressive patterns.
- Focus on top-k itemsets (work in progress).

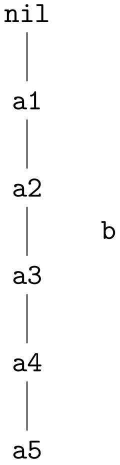
Thanks for your attention!

Greedy algorithms

- Querying an element of the chain may remove $< 1/2$ possible solutions.
- Querying the isolated element b will remove exactly $1/2$ solution.
- However, querying b classifies far less itemsets.

$\Rightarrow$ Classifying many itemsets isn't the same as eliminating many solutions.

Finding the greedy-best-split item is $\mathrm{FP}^{\#\mathrm{P}}$-hard.



Restricted itemsets

- Asking about large itemsets is irrelevant.

  "Do you often go cycling and running while drinking coffee and having lunch with orange juice on alternate Wednesdays?"

- If the itemset size is bounded by a constant, $\mathsf{I}(\Psi)$ is tractable.

$\Rightarrow$ The crowd complexity $\Theta(\log|\mathsf{S}(\Psi)|)$ is tractable too.

Chain partitioning

- Optimal strategy for chain taxonomies: binary search.
- We can determine a chain decomposition of the itemset taxonomy and perform binary searches on the chains.
- Optimal crowd complexity for a chain, performance in general is unclear.
- Computational complexity is polynomial in the size of $I(\Psi)$ (which is still exponential in Ψ).