**Theoretical aspects**

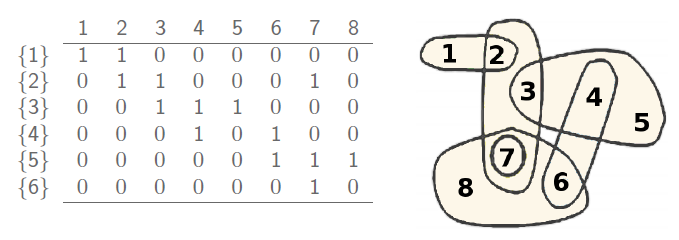
A hypergraph could be represented through the triplet K = (ξ, X, R) where ξ and X denote, respectively, finite set of **hyperedges** (or transactions) and **vertex** (or items), and R ⊆ ξ x X is the binary relation linking vertex to hyperedges

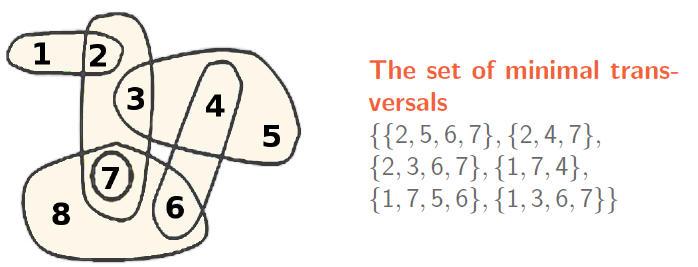
T is a minimal transversal if :

1. Supp(vT) > max{Supp(vT \ x) | ∀x ∈T} (The essentiality condition) and
2. Supp(vT) = | ξ |.

**Illustrative example of SBI algorithm**

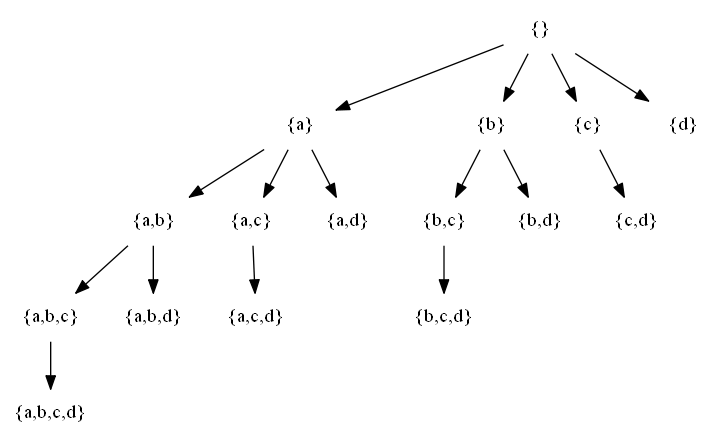
Let us consider the following hypergraph represented as an extraction context

The set of the Minimal transversals is the following one:



Two important remarks have to be underscored:

**Remark1** : the search space by means as an enumeration tree: each element is met ONLY ONCE. An example of a search space of {1,2,3,4}. Please note that the as far as we explore the list, the smaller becomes the search space. For example, the element 1 stands on top of 2^3 elements, while the node stands on 2^2, etc.



**Remark 2**

The MT-miner algorithm operates as follows **at each level**:

-1 compute the DIJUNCTIVE SUPPORT of each candidate.

2- if it exists a candidate sucht that Dis-supp(X)=|N|, then add to MT-list.

3-determine MAX-CLIQUE-OF-NON-GENERATORS

**Execution track of MT-miner algorithm**

**Iteration1**

At first, we have to compute the disjunctive support of the elements of the ground set, to wit

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Item** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** |
| Dij Supp | 1 | 2 | 2 | 2 | 1 | 2 | 3 | 1 |

**MT-list={}**

Now, we have to create the maximal clique of non-transversals, i.e., the largest space in which is not possible to find minimal transversals. The lager this maximal clique, the higher the pruning of the search space is.

To do, we have to stick element by element and test whether the disjunctive support of the obtained element is equal to the cardinality of the object set. If it is the case, then the last topped-up element would lead to a transversal (not necessary element) and it is placed the to-explore list. Otherwise, the last topped-up item is added to the maximal clique.

Considering the sorted list of items, the max-clique=empty set

After adding {1}, max-clique ={1}

Considering, 2, the support of {1,2} is equal to 2<6, MAX-CLIQUE-OF-NON-GENERATORS ={1,2}

Considering 3, the support of {1,2,3} is equal to 3<6, MAX-CLIQUE-OF-NON-GENERATORS ={1,2,3}

We continue so on, until topping-up 7 which leads to breaking the max-clique, so it is inserted in the **to-explore-list**. So, after handling the ground elements of the ground set, we have the following lists:

To-explore-list ={7}

MAX-CLIQUE-OF-NON-GENERATORS = {1, 2,3,4,5,6,8 }=> All the SUBSETS of {1, 2,3,4,5,6,8 }, even some of them being minimal, will NEVER lead to MT

**Remark**: as you can remark, none of the subsets of the {1, 2,3,4,5,6,8 } is a minimal transversals. This means that we can simply ignore exploring non-negligible search space. It simply spits the search space by half. Instead of the exploring 2^8, we are about to explore in the worst case only 2^7.

**Iteration 2**: We have to iterate on the elements of To-explore-list, here equal to the singleton {7} and generate 2-items (i.e., of size 2). Please note that in the following tables, we use a separator-free form, e.g., **78** stands -for **{7,8}.**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Item** | **71** | **72** | **73** | **74** | **75** | **76** | **78** |
| Disj Supp | 4 | 4 | 4 | 5 | 4 | 4 | 3 |

Clearly, none of the 2-itemses is a minimal transversal (since their respective disjunctive supports is less than 6). Please note the 2-itemset 78 has to be removed since it does not fulfill the first condition of MT (called the essentiality condition), i.e. DIsj-support(78)=Disj-support(7)=3.

Constructing the maximal clique at the level 2, leads to

To-explore-list ={74, 76}

Max-clique-list = {71,72,73,75}

Remark: check that it does exist a minima transversal such that it is a subset of {7,1, 2,3, 5}.

**Iteration 3**: we consider the exploration of the first element of To-explore-list, to wit 74 and we generate the 3-items

Level 3

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Item** | **746** | **741** | **742** | **743** | **745** |
| Dij Supp | 5 | 6 | 6 | 5 | 5 |

We have 2 minimal transversals, so MT-List ={**741**, **742**}. The 3-itemset 745 has to be removed, since it does not fulfill the essentiality condition (its disjunctive support is equal to that of its direct subset 74).

If we build the Max-clique, we find it equal to : {7,1,5.8.2,3}.

And the exploration of the branch of “**74 comes to an end here**”.

We back track to explore the remaining element of the To-explore-list, i.e., **76**

Level 3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item** | **761** | **762** | **763** | **765** |
| Dij Supp | 5 | 5 | 5 | 5 |

None of them is a minima generator, so we have to build our lists

To-explore-list ={763, 765}

Max-clique-list = {761,762}

**Level 4**

We start by exploring 763

|  |  |  |  |
| --- | --- | --- | --- |
| **Item** | **7635** | **7631** | **7632** |
| Dij Supp | 5 | 6 | 6 |

.

We have 2 minimal generators, then MT-List ={**741**, **742, 7631, 7632**}. We stop exploring this branch since To-explore-list ={}.

Now, we have just to explore, the 3-itemset : 765

|  |  |  |
| --- | --- | --- |
| **Item** | **7651** | **7652** |
| Dij Supp | 6 | 6 |

We have 2 minimal generators, then MT-List ={**741**, **742, 7631, 7632, 7651, 7652**}. We stop exploring this branch and the overall execution comes to an end.

The output : MT-List ={**741**, **742, 7631, 7632, 7651, 7652**}

