Exam : 2 hours - 19/02/2025 — *M2 Data Science (24/25)*

N. LAZAAR — *Paris-Saclay University*

Exam : Constraints & Data Mining

Exercise 1 (4pts)

A company has 4 tasks (T_1, T_2, T_3, T_4) that need to be assigned to 3 workers (W_1, W_2, W_3).

Each task must be assigned to exactly one worker, ensuring that no task is left unassigned. Additionally, each worker can be assigned at least ~~most~~ one task, preventing any worker from being overloaded. Moreover, workers can only be assigned tasks for which they are qualified : W_1 is capable of performing T_1 and T_2, W_2 can handle T_2 and T_3, while W_3 is qualified for T_3 and T_4.

The goal is to find a valid assignment of tasks to workers.

1. (3 points) Model this problem as a constraint network $N = \langle X, D, C\rangle$.
2. (1 point) Provide a possible assignment satisfying all constraints.

Exercise 2 (5pts)

Consider a constraint network consisting of three variables : $X = \{X_1, X_2, X_3\}$
Each variable has the domain : $D(X_1) = D(X_2) = D(X_3) = \{1, 2, 3\}$
The constraints between the variables are : $X_1 < X_2$; $X_2 = X_3$; $X_1 > X_3$

1. (1 point) Write the initial propagation queue (list of arcs) that the AC3 algorithm will start with. To simplify the notation, you can, for example, represent $x_1 \neq x_2$, by $\overrightarrow{x_1 \neq x_2}$ and $\overrightarrow{x_2 \neq x_1}$.
2. (3 points) Apply the AC3 algorithm step by step, showing :
   - The arc being checked.
   - Any changes made to the domains.
   - Any new arcs added back to the queue if a domain is reduced.
3. (1 point) What are the final domains of all variables after running AC3?

Exercice 3 (6pts)

Soit $\mathcal{D}_1$ une base de transactions représentée horizontalement :

trans.	Items			
t_1	A		C	D
t_2	A	B		D
t_3	A		C	
t_4	A			D

1. (2 points) Give the ~~set of frequent~~, ~~closed, and~~ ~~maximal~~ itemsets for the dataset $\mathcal{D}_1$ with $\alpha = 1$.
2. (1 point) Give the formulas that allow you to derive the frequent itemsets from the closed and maximal itemsets.

Definition 1 (Apriori Property) *Let P be an itemset. If P is frequent, then all subsets of P must also be frequent. That is, if $freq(P) \geq \alpha$, then $freq(Q) \geq \alpha$ for all $Q \subseteq P$.*

1. (3 points) Write a proof for the Apriori property. √

Exercise 4 (5pts)

Consider a transactional database where the items are numbered from a_1 to a_n, and the transactions are represented as sequences of these item numbers.

You are required to design a constraint network $N = (X, D, C)$ that models the query Q_1 : enumeration of **minimal rare (infrequent) patterns** satisfying the following conditions :

1. **Minimum Support :** The itemsets should have a **support** less than or equal to a given threshold α. $\rightsquigarrow$ support$(I) \leq \alpha$
2. **Minimum Size :** The itemsets should have a size greater than or equal to a specified threshold β (lower bound on itemset size). $\rightarrow$ Size$_\beta(I)$($\Leftrightarrow$ $|I| \geqslant \beta$)
3. **Non-successive Items :** The items in each itemset must **not be consecutive** (i.e., the items in the set must not appear consecutively in the database).
4. **Minimality :** The itemsets must be **minimal**, meaning that no proper subset of the itemset can be frequent. (

1. (3 points) Construct the corresponding constraint network $N = (X, D, C)$ for query Q_1.
2. (2 points) According to the work of Bonchi and Lucchese (2004), does query Q_1 have one or two different interpretations? Explain why there is one interpretation or two, and provide the result for the unique or both possible interpretations on dataset $\mathcal{D}_1$ with $\alpha = 1$ and $\beta = 2$.

Annex : AC-3 Algorithm and the REVISE Function

Algorithm 1: AC3 Algorithm

Input: $\langle X, D, C\rangle$ (Constraint Network)
Output: **true** if arc-consistency is achieved, **false** otherwise
$Q \leftarrow \{(x_i, c) \mid c \in C, x_i \in \text{var}(c)\}$;
while $Q \neq \emptyset$ **do**
  Pick $(x_i, c) \in Q$;
  if REVISE(x_i, c) **then**
    if $D(x_i) = \emptyset$ **then** **return false** ;
    else $Q \leftarrow Q \cup \{(x_j, c') \mid c' \in C, x_i, x_j \in \text{var}(c')\}$;
  end
end
return true;

Algorithm 2: REVISE Function

Input: X_i (variable), c (constraint)
Output: **true** if a revision occurred, **false** otherwise
CHANGE $\leftarrow$ **false**;
foreach $v \in D(x_i)$ **do**
  if *No allowed pair* (v, v_i) *satisfies* c **then**
    Remove v from $D(x_i)$;
    CHANGE $\leftarrow$ **true**;
  end
end
return CHANGE;