**Programme de colle - Semaine 13 (colle du 10 décembre)**

*La démonstration des énoncés marqués d'une étoile est exigible*

# 1 Union-Find et algorithme de Kruskal

- Structure Union-Find : implémentation naïve favorisant l'opération `FIND` avec un tableau dans lequel chaque case $i$ contient le représentant de l'élément numéro $i$. Complexités des opérations.
- Structure Union-Find : implémentation à l'aide d'une forêt favorisant l'opération `UNION`.

  1. Optimisation 1 : en plaçant systématiquement l'arbre de plus petite hauteur comme fils lors de l'opération union. **Dans ce cas, la hauteur de chaque arbre ne dépasse pas $\log_2(m)$ avec $m$ la taille de l'arbre considéré (*)**
  2. Optimisation 2 : en compressant les chemins lors de l'opération `FIND`. La complexité amortie obtenue est hors programme.
- Notion d'arbre couvrant d'un graphe non orienté. Algorithme générique de construction d'un arbre couvrant : utilisation de Union-Find dans ce cadre.
- Arbre couvrant de poids minimal d'un graphe non orienté pondéré. Existence. **Algorithme de Kruskal** : savoir décrire l'algorithme en pseudo-code avec la structure Union-Find, savoir l'appliquer sur un exemple, la preuve n'est pas au programme de colle.

# 2 Théorème de Kleene

- La classe des langages réguliers et la classe des langages reconnaissables sont les mêmes.
- Algorithme de Berry-Séthi : passage d'une expression régulière à un automate. Dans ce contexte : langage local (notations utilisées : $P(L)$ pour les premières lettres, $D(L)$ pour les dernières et $T(L)$ pour les paires de lettres valides), automate de Glushkov pour reconnaître un langage local, procédure de Berry-Séthi.
- Algorithme de Thomson : passage d'une expression régulière à un automate. Notion d'automate **normalisé**. Automates de Thomson : construction par induction d'un automate normalisé qui reconnaît le langage dénoté par une expression régulière.
- Passage d'un automate à une expression régulière :

  - Automate généralisé avec des expressions régulières sur les transitions.

  - Algorithme par élimination successive des états en partant d'un automate normalisé.
- Rappels sur les conséquences du théorème de Kleene sur les stabilités (propriétés de fermeture) des langages réguliers=reconnaissables.
- Rappel : Lemme de l'étoile. **Le langage $L = \{a^n b^n, n \in \mathbb{N}\}$ n'est pas régulier (*)**.

## 3 Apprentissage supervisé

- Principe de l'apprentissage supervisé
- Arbres binaires de décision : définition et écriture de la fonction de classification utilisant un ABD.
- Éléments de théorie de l'information : définition de l'entropie, cas du pile ou face traité en cours, calcul de l'entropie dans le cas d'un ensemble d'exemples d'apprentissage, calcul du gain d'information pour un attribut binaire uniquement.
- Algorithme ID3 (*): savoir écrire l'algorithme en pseudo-code et l'appliquer sur un exemple. Seul le cas d'apprentissage d'arbres **binaires** de décision est au programme (attributs binaires)
- Algorithme des $k$-plus proches voisins : savoir l'algorithme et le mettre en œuvre dans des cas simples (par exemple $k$ fois l'extraction d'un minimum dans une liste)
- Arbres $k$-dimensionnels : généralisation des arbres binaires de recherche en dimension $k$. Définition. **Algorithme d'existence d'un point dans un arbre $k$-dimensionnel (*)**. Algorithme d'insertion d'un point dans un arbre $k$-dimensionnel. La recherche du plus proche voisin à l'aide de cette structure a été vue (mais ne pas interroger dessus...).
- Matrice de confusion

## 4 Logique propositionnelle (révisions 1A)

- Syntaxe de la logique propositionnelle : les formules sont définies par induction. Symboles utilisés : $\bot$, $\top$, $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$. Les formules sont des **données** (arbres) sur lesquelles on travaille.
- Sémantique : les valeurs de vérité ont été notées *false* et *true*. Valuations. Valeur de vérité d'une formule $F$ dans le contexte de la valuation $\varphi$ (notée $[\![F]\!]_\varphi$). Table de vérité d'une formule.
- Tautologies. Antilogies. Formules satisfisables. Utilisation de tables de vérité dans ce cadre.
- **Une formule est une tautologie si et seulement si sa négation est non satisfiable (*)**
- Algorithme de Quine pour tester la satisfiabilité d'une formule.
- Équivalence de deux formules propositionnelles (notation $\equiv$). Calculs par équivalents. Équivalences usuelles (distributivité de $\vee$ sur $\wedge$ et réciproquement, lois de De Morgan, $(p \rightarrow q) \equiv (\neg p \vee q)$, ...).
- Formes normales conjonctives et disjonctives. Littéral. Clause. Mise sous forme normale conjonctive (resp. disjonctive) d'une formule.
- Conséquence. $\Gamma \models F$ signifie que $F$ est la conséquence sémantique de l'ensemble de formules $\Gamma$. Exemple : $(p \rightarrow q), (q \rightarrow r) \models (p \rightarrow r)$.