

Structures de données

Master 1 Cyber Sécurité et Intelligence Artificielle
CSIA

Année universitaire : 2025 / 2026

Introduction

- La plupart des bons algorithmes fonctionnent grâce à une méthode astucieuse pour organiser les données.
- quatre grandes classes de structures de données :
 - Les structures de données séquentielles (tableaux) ;
 - Les structures de données linéaires (liste chaînées) ;
 - Les arbres ;
 - Les graphes.

Tableaux

- un tableau est une structure de données basée sur la disposition contiguë d'un certain nombre d'éléments ayant le même type.



Piles

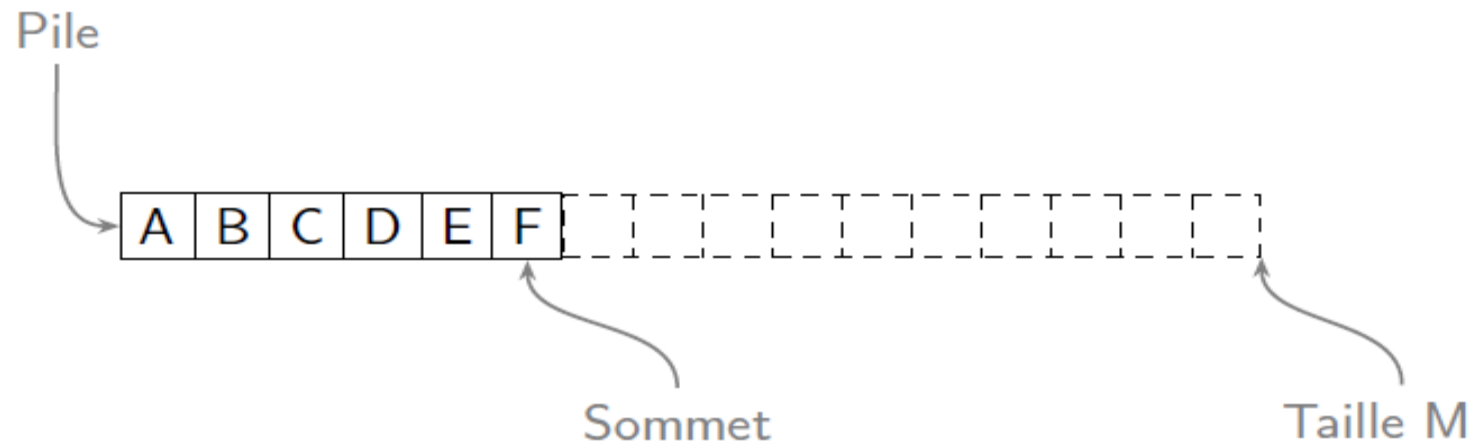
Définition(pile):

- Une pile est une structure de données mettant en œuvre le principe «dernier entré premier sorti»
- LIFO: Last In First Out

Piles

- Une pile P peut être implémentée par un tableau, et elle est caractérisée par:
- Un sommet noté $\text{sommet}(P)$ indiquant l'indice de l'élément le plus récemment inséré dans la pile
- Un caractère spécial, comme \$ ou NIL, initialisant la pile
- Une procédure $\text{EMPILER}(P, x)$
- Une fonction $\text{DEPILER}(P)$
- Une fonction booléenne $\text{PILE-VIDE}(P)$ retournant VRAI si et seulement si P est vide

Piles



Piles

- Applications
 - Dans un navigateur web, une pile sert à mémoriser les pages Web visitées.
 - L'évaluation des expressions mathématiques en notation post-fixée utilise une pile.
 - La fonction « Annuler la frappe » d'un traitement de texte mémorise les modifications apportées au texte dans une pile.
 - Un algorithme de recherche en profondeur utilise une pile pour mémoriser les nœuds visités.

Files

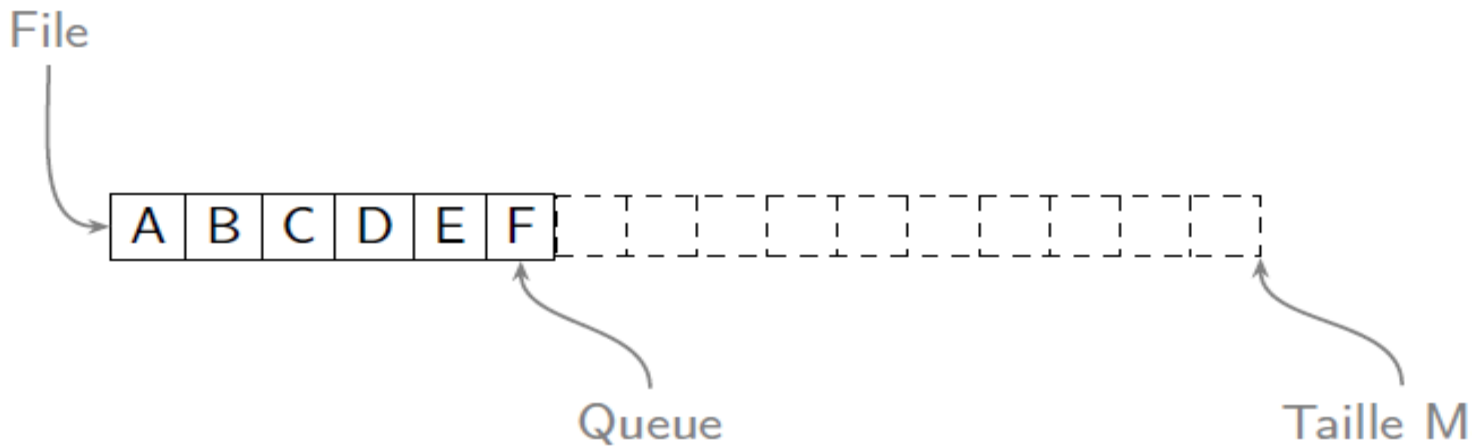
Définition(file):

- Une file est une structure de données mettant en œuvre le principe «premier entré premier sorti»
- FIFO: First In First Out

Files

- Une file F peut être implémentée par un tableau, et elle est caractérisée par :
- Un pointeur tête(F) qui pointe vers la tête de la file (l'élément le plus anciennement inséré)
- Un pointeur queue(F) qui pointe vers la première place libre, où se fera la prochaine insertion éventuelle d'un élément
- Initialement : tête(F)=NIL et queue(F)=1

Files



Files

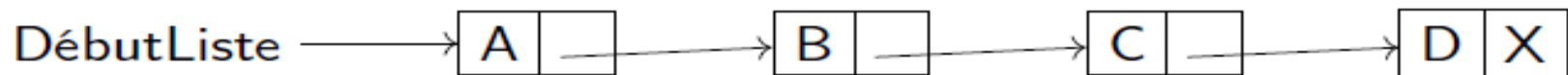
- Applications
 - En général, on utilise des files pour mémoriser temporairement des transactions qui doivent attendre pour être traitées.
 - Les serveurs d'impression, qui doivent traiter les requêtes dans l'ordre dans lequel elles arrivent, et les insèrent dans une file d'attente (ou une queue).
 - Certains moteurs multitâches, dans un système d'exploitation, qui doivent accorder du temps-machine à chaque tâche, sans en privilégier aucune.
 - Un algorithme de parcours en largeur utilise une file pour mémoriser les nœuds visités.

Listes chaînées

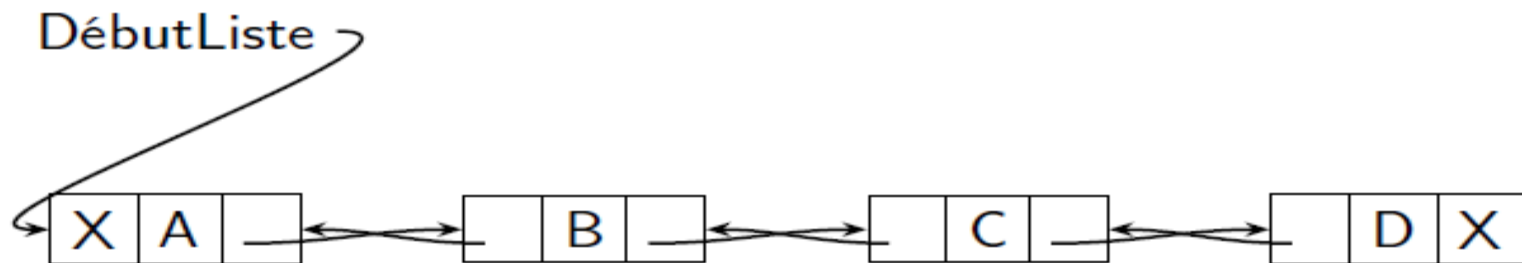
Définition (liste chaînée) :

- Une liste chaînée est une structure de données dont les éléments sont arrangés linéairement, l'ordre linéaire étant donné par des pointeurs sur les éléments
- Un élément d'une liste chaînée est un enregistrement contenant un champ *clé*, un champ *successeur* consistant en un pointeur sur l'élément suivant dans la liste
- Si le champ *successeur* d'un élément vaut *NIL*, l'élément est le dernier élément de la liste, appelé également queue de la liste
- Un pointeur TETE(L) est associé à une liste chaînée L : il pointe sur le premier élément de la liste
- Si une liste chaînée L est telle que TETE(L)=NIL alors la liste est vide

Listes chaînées



listes simplement chaînées

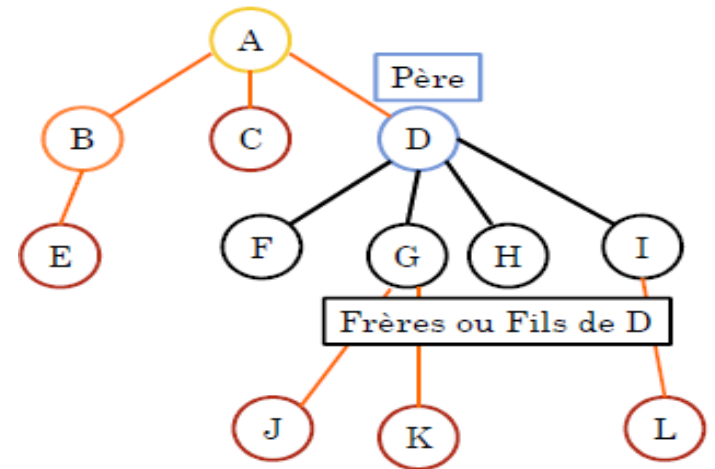
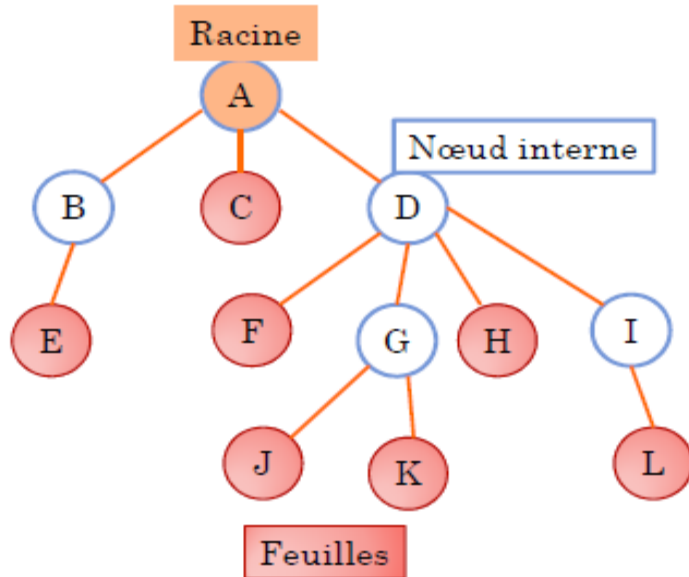
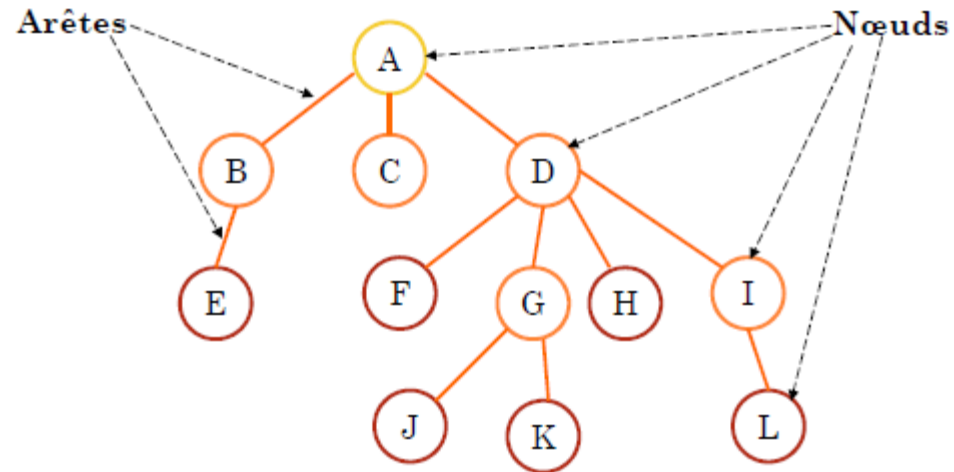


liste doublement chaînée

Arbres

- L'unique nœud sans père : la racine
- Nœuds avec fils : nœuds internes
- Sous-arbre de racine x : l'arbre composé des descendants de x .
- Degré d'un nœud : nombre de fils
- Profondeur d'un nœud x : longueur du chemin entre la racine et le nœud
- Hauteur d'un arbre
- Nœuds sans fils : feuilles
- Exemple : Arbre binaire, Arbre n -aire, Tas ...
- Forêt : est un ensemble d'arbres.

Arbres



Arbres

- Le parcours d'un arbre consiste à passer par tous ses nœuds.
- Les parcours permettent d'effectuer tout un ensemble de traitement sur les arbres.
- On distingue deux types de parcours :
- Des parcours en **profondeur** explorent l'arbre branche par branche. Parmi lesquels:
 - **Préfixe** : ou on affiche la racine avant ses fils,
 - **Postfixe** : ou on affiche les fils avant leur racine.
 - **Infixe** : ou, dans le cas d'un arbre binaire, on affiche le fils gauche avant sa racine et son frère droit
- Des parcours en **largeur** explorent l'arbre niveau par niveau

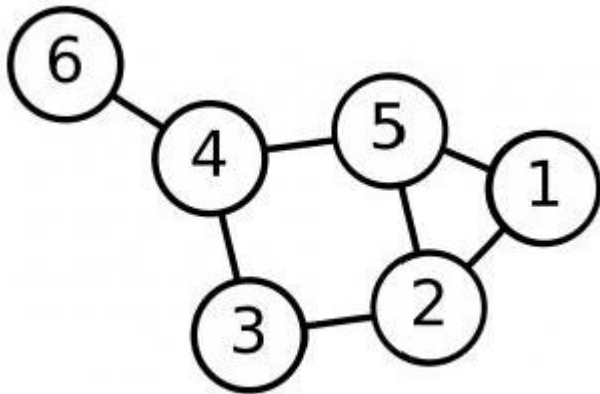
Graphes

- **Graphe orienté** : couple (S,A) , S ensemble fini (sommets) et A relation binaire sur S (arcs)
 - Un arc (u,v) part du sommet u et arrive au sommet v
 - Boucle (arc reliant un sommet à lui-même)
 - Degré sortant d'un sommet: nombre d'arcs en partant
 - Degré entrant d'un sommet: nombre d'arcs y arrivant
 - Degré = degré sortant + degré entrant
 - Chemin élémentaire
 - Circuit et circuit élémentaire

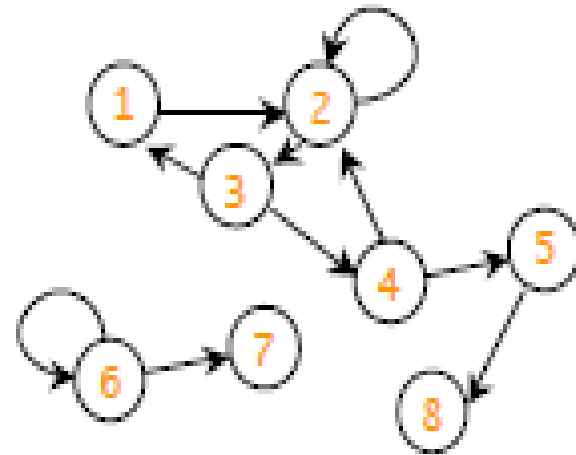
Graphes

- **Graphe non orienté:**
 - arêtes
 - **Boucles interdites**
 - Une arête (u,v) est incidente aux sommets u et v
 - Degré d'un sommet: nombre d'arêtes qui lui sont incidentes
 - Chaîne et chaîne élémentaire
 - Cycle et cycle élémentaire

Graphes



Graphe non orienté



Graphe orienté

Graphes

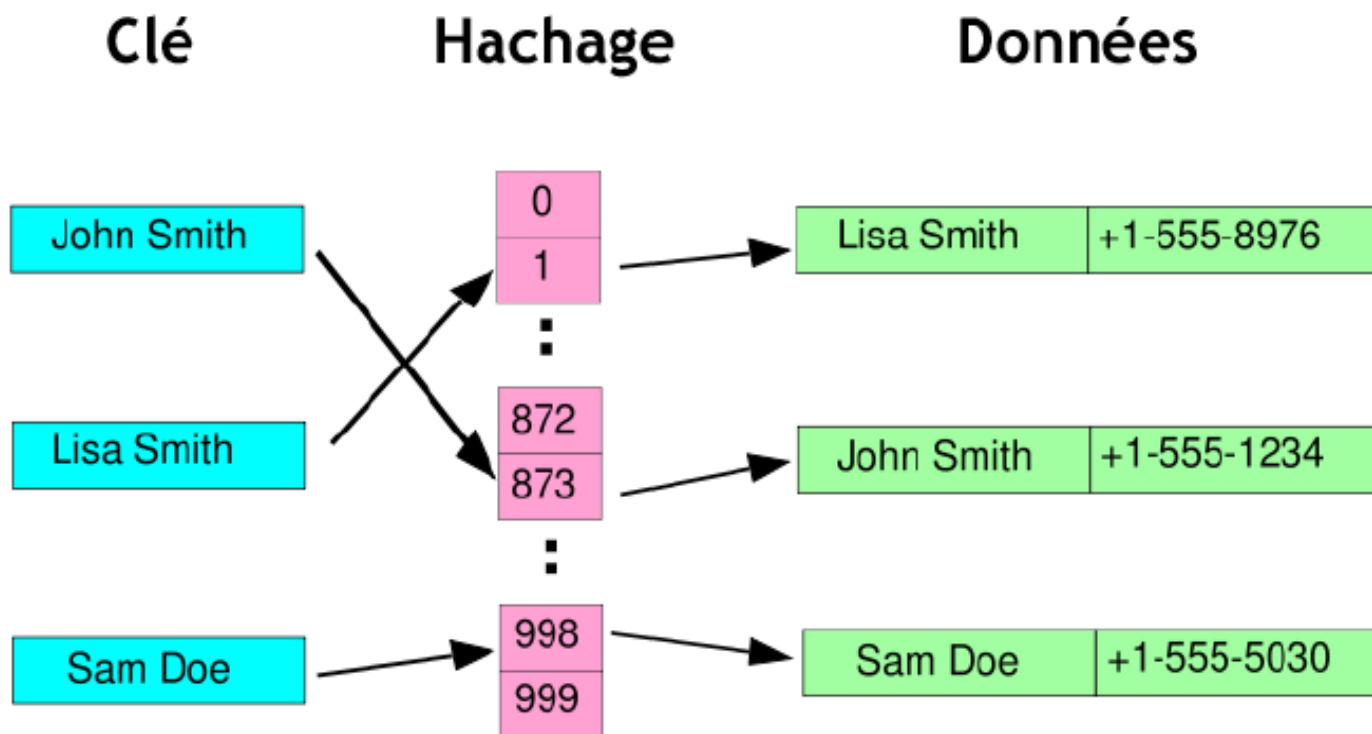
- Les réseaux de transport (routiers, ferrés, aériens . . .),
- Les réseaux téléphoniques, électriques, de gaz, . . . etc,
- Les réseaux d'ordinateurs,
- Ordonnancement des tâches,
- Circuits électroniques,

Les arbres et les listes linéaires chaînées ne sont que des cas particuliers des graphes

Table de hachage

- Une table de hachage est une structure de donnée permettant d'associer une valeur à une clé.
- Il s'agit d'un tableau ne comportant pas d'ordre (un tableau est indexé par des entiers).
- L'accès à un élément se fait en transformant la clé en une valeur de hachage (ou simplement hachage) par l'intermédiaire d'une fonction de hachage.
- Le hachage est un nombre qui permet la localisation des éléments dans le tableau, typiquement le hachage est l'indice de l'élément dans le tableau
- La fonction de hachage doit donner des valeurs entières dans l'intervalle des indices de la tables : $0 \leq h(\text{clé}) \leq N$.

Table de hachage



Une table de hachage implémentant un annuaire téléphonique

Fonctions de hachage

- On trouve plusieurs types de fonctions utilisées en Hashcoding :
 - $h(\text{clé}) = \text{CodeASCII}(\text{1er car}) + \text{Code ASCII}(\text{2`eme car}) \bmod N$
Si la clé est une chaine de caractères (nom)
 - $h(\text{clé}) = \text{clé} \bmod N$
Si clé est une valeur numérique.
 - Méthode du milieu du carré
 - Clé = 453
 - $(\text{Cl})^2 = (453)^2 = 205209$
 - $h(453) = 52$ Si $n > 100$, on prend 3 chiffres.
 - Hachage de Fibonacci : C'est une fonction de hachage fréquemment utilisée : $h(\text{clé}) = |N \times (\text{clé} \times r - |\text{clé} \times r|)|$ où $||$ donne la partie entière, avec $r = \frac{\sqrt{5}-1}{2}$
- En plus, cette fonction doit être la plus distribuée possible sur cet intervalle, pour que les informations ne se concentrent pas dans une partie de la table.

Problème de collisions

- Un problème se pose avec les fonctions de hachage si deux clés différentes donnent lieu à une même adresse lorsqu'on leur applique la fonction de hachage, c-à-d $h(\text{clé1}) = h(\text{clé2})$.
- Une telle situation est appelée « collision » et plusieurs solutions existent pour sa résolution.

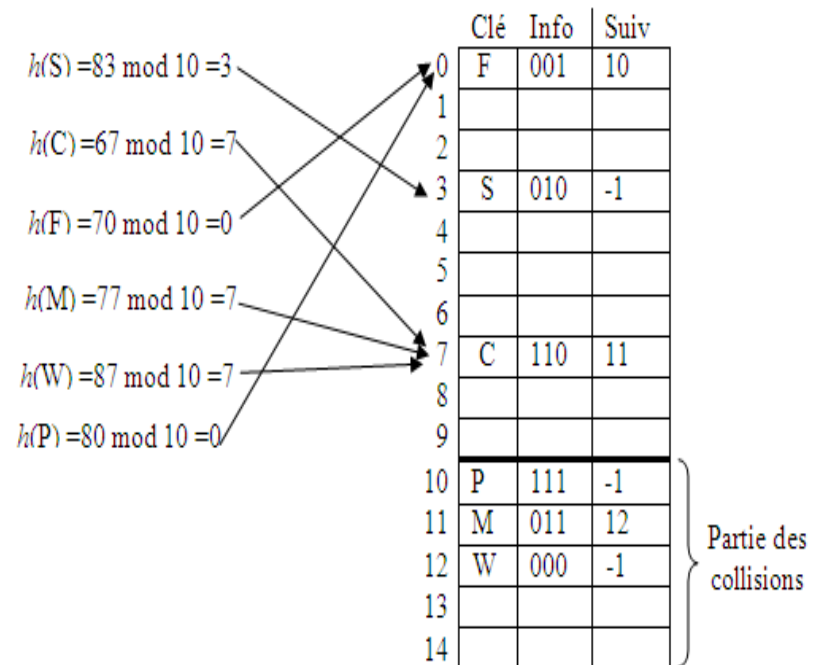
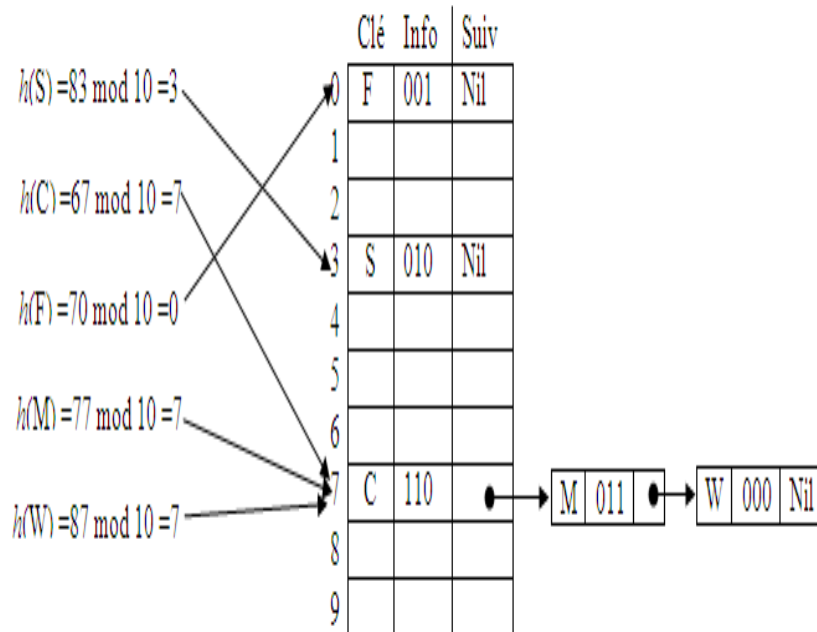


Table de hachage

Applications :

- plusieurs applications réseaux comme le traitement du routage, le contrôle du réseau, la détection d'intrusion (IDS) ou la prévention d'intrusion (IPS) pour la sécurité des réseaux.
- Les applications possibles peuvent concerner différents types de réseaux, comme par exemple : les réseaux haut débit, les réseaux mobiles, les réseaux de capteurs etc.