

# THESE

présentée à

**L'UNIVERSITE PARIS IX DAUPHINE  
U.E.R. SCIENCES DES ORGANISATIONS**

pour obtenir

**LE DIPLOME DE DOCTEUR EN INFORMATIQUE**

par

**Djamel Eddine ZEGOUR**

Directeur de recherche  
**Gérard LEVY**

Sujet de la thèse :

**EXTENSIONS DU HACHAGE DIGITAL :  
HACHAGE DIGITAL MULTINIVEAUX  
HACHAGE DIGITAL AVEC REPRESENTATIONS  
SEQUENTIELLES**

Soutenue le 21 juin 1988 devant la commission composée de :

**MM C. BERTHET**

**Président**

**C. DELOBEL**

**E. DIDAY**

**G. LEVY**

**W. LITWIN**

**E. PICHAT**

**Examineurs**

## Résumé

Le hachage dynamique a pris naissance dès l'année 1978 avec les premières propositions de Litwin et Larson. Contrairement au hachage classique, la fonction d'accès générée par la méthode varie dynamiquement de telle sorte que l'espace des adresses possibles ne soit plus limité. Depuis, plusieurs techniques d'accès basées sur ce principe ont été développées principalement pour gérer des fichiers dynamiques. Certaines d'entre elles préservent en plus l'ordre des clés et/ou permettent de gérer des données multidimensionnelles.

Le hachage digital , en anglais "trie hashing " conçu en 1981, est une technique d'accès aux fichiers monoclés dynamiques et ordonnés basée sur le concept du hachage dynamique. Son facteur de chargement est de l'ordre de 70%. En plus, un accès disque au plus suffit pour retrouver tout article pouvant appartenir à un fichier d'une contenance moyenne de quelques millions d'articles.

Cette thèse propose une extension du hachage digital aux fichiers volumineux. La nouvelle technique a été baptisée hachage digital multiniveaux. Sa formulation consiste à segmenter l'arbre représentant la fonction d'accès, ne pouvant résider longtemps en mémoire principale, en pages sur le disque, L'organisation correspondante est alors similaire à celle des arbres B. L'analyse des performances d'accès montre que deux accès par recherche de clé suffisent pour des fichiers de plus de 500 000 000 d'articles. Ainsi le hachage digital multiniveaux se situe parmi les techniques d'accès aux fichiers les plus efficaces connues actuellement.

Cette thèse propose également des représentations très concises de la fonction d'accès engendrée par le hachage digital. Avec ces nouvelles représentations, on peut doubler le contenu du fichier pour le même espace alloué pour la fonction d'accès que si une représentation normale, appelée représentation standard, est utilisée. En contrepartie, l'algorithmique est plus complexe et le temps de calcul d'une adresse est plus long. Cependant, elles peuvent être avantageuses pour certaines applications.

## Mots clés

Structures de données - Algorithmes - Méthodes d'accès - Arbres B - Hachage classique - Hachage dynamique - Hachage digital - Hachage digital multiniveaux - Représentations séquentielles.

# SOMMAIRE

**Première partie**

## **Introduction**

- 1. Introduction**
  - 1.1 Hachage classique**
  - 1.2 Hachage dynamique**
  - 1.3 Objectifs de la thèse**
  - 1.4 Démarche de la thèse**
  - 1.5 Organisation de la thèse**

## Deuxième partie

# Les méthodes d'accès

- 2.1 Introduction**
  - 2.1.1 Généralités**
  - 2.1.2 Objectifs**
- 2.2 Le hachage**
  - 2.2.1 Fonctions de hachage**
    - 2.2.1.1 Fonction de division**
  - 2.2.2 Méthodes de résolution des collisions**
    - 2.2.2.1 Chaînage séparé**
    - 2.2.2.2 Essai linéaire**
- 2.3 Les arbres**
  - 2.3.1 Arbres m-aires simples**
  - 2.3.2 Arbres B**
    - 2.3.2.1 Définition**
    - 2.3.2.2 Opérations**
      - 2.3.2.2.1 Recherche**
      - 2.3.2.2.2 Insertion**
      - 2.3.2.2.3 Suppression**
  - 2.3.3 Autres variantes**
    - 2.3.3.1 ISAM**
    - 2.3.3.2 VSAM**
    - 2.3.3.3 Arbres B+**
    - 2.3.3.4 Arbres B\***
    - 2.3.3.5 Arbres B préfixés**
    - 2.3.3.6 Arbres B virtuels**
- 2.4 Comparaison**
- 2.5 Conclusion**

## Le hachage dynamique

- 3.1 Introduction**
- 3.2 Hachage virtuel**
  - 3.2.1 Notion de fonction de division**
  - 3.2.2 HV1**
  - 3.2.3 HV2**
  - 3.2.4 HV0**
  - 3.2.5 HVL**
- 3.3 Désordre borné**
- 3.4 Fichiers en grilles**
- 3.5 Hachage par interpolation**
  - 3.5.1 Éléments de base**
  - 3.5.2 La méthode**
  - 3.5.3 Le choix des fonctions de division**
- 3.6 Hachage digital**
  - 3.6.1 La méthode par des exemples**
  - 3.6.2 Définitions et propriétés**
  - 3.6.3 Opérations**
    - 3.6.3.1 Recherche**
    - 3.6.3.2 Insertion**
    - 3.6.3.3 Suppression**
    - 3.6.3.4 Requête à intervalle**
  - 3.6.4 Représentations mémoire**
    - 3.6.4.1 Représentation standard**
    - 3.6.4.2 Raffinement de la représentation standard**
    - 3.6.4.3 Autres représentations**
  - 3.6.5 Analyse des performances**
    - 3.6.5.1 Facteur de chargement**
    - 3.6.5.2 Taille de la fonction d'accès**
    - 3.6.5.3 Accès**
  - 3.6.6 Résultats de simulations**
  - 3.6.7 Comparaison avec les autres méthodes**
  - 3.6.8 Travaux sur le hachage**
  - 3.6.9 Conclusion**
- 3.7 Conclusion**

## Hachage digital multiniveaux

- 4.1 Introduction
- 4.2 Algorithme de base
  - 4.2.1 Principe
  - 4.2.2 Procédés de segmentation
  - 4.2.3 Algorithme complet d'éclatement
    - 4.2.3.1 Eclatement de case
    - 4.2.3.2 Eclatement de page
- 4.3 Algorithme de transformation clé-adresse
- 4.4 Division de l'arbre digital
  - 4.4.1 Le problème
  - 4.4.2 Le choix du noeud de division
    - 4.2.2.1 Condition de division
      - 4.2.2.1.1 Algorithme 1
      - 4.2.2.1.2 Algorithme 2
      - 4.2.2.1.3 Algorithme 3
    - 4.2.2.2 Descendance logique
      - 4.2.2.2.1 Définitions
      - 4.2.2.2.2 Test de la descendance logique
        - 4.2.2.2.2.1 Solution 1
        - 4.2.2.2.2.2 Solution 2
    - 4.2.2.3 Algorithme de choix de la nouvelle racine
  - 4.4.3 Procédés de division de l'arbre digital
    - 4.4.3.1 Rotation de l'arbre digital
      - 4.4.3.1.1 Algorithme 1
      - 4.4.3.1.2 Algorithme 2
    - 4.4.3.2 Eclatement direct de l'arbre digital
      - 4.4.3.2.1 Algorithme inordre
      - 4.4.3.2.1 Algorithme d'éclatement
  - 4.4.4 Equilibrage récursif de l'arbre digital
- 4.5 Opérations
  - 4.5.1 Recherche
  - 4.5.2 Insertion
  - 4.5.3 Suppression
  - 4.5.4 Requête à intervalle
- 4.6 Analyse des performances
  - 4.6.1 Chargement des cases
  - 4.6.2 Chargement des pages
  - 4.6.3 Fannout

- 4.7 Simulations
  - 4.7.1 Introduction
  - 4.7.2 Objectifs et choix du modèle de simulation
  - 4.7.3 Insertions aléatoires
  - 4.7.4 Insertions ordonnées
    - 4.7.4.1 Insertions ascendantes
    - 4.7.4.2 Insertions descendantes
  - 4.7.5 Conclusion
- 4.8 Comparaison
  - 4.8.1 Arbre B
    - 4.8.1.1 Taille du noeud
    - 4.8.1.2 Facteur de chargement
    - 4.8.1.3 Fannout
    - 4.8.1.4 Accès
  - 4.8.2 Hachage dynamique
    - 4.8.2.1 Hachage par interpolation
    - 4.8.2.2 Fichiers en grilles
    - 4.8.2.3 Désordre borné
- 4.9 Application réelle : dictionnaire anglais-français
- 4.10 Conclusion

## Cinquième partie

# Hachage digital avec représentations séquentielles

- 5.1 Introduction
- 5.2 Représentations séquentielles
  - 5.2.1 Nouveaux graphes
  - 5.2.2 Représentations mémoire
- 5.3 Représentation LL-TB
  - 5.3.1 Construction
  - 5.3.2 Insertion
  - 5.3.3 Recherche
  - 5.3.4 Requête à intervalle
- 5.4 Représentation PP-LR'
  - 5.4.1 Construction
  - 5.4.2 Insertion
  - 5.4.3 Recherche
  - 5.4.4 Requête à intervalle

- 5.5 Analyse des performances
  - 5.5.1 Représentation standard
  - 5.5.2 Représentation LL-TB
  - 5.5.3 Représentation PP-LR'
- 5.6 Implémentation
- 5.7 Conclusion

## Sixieme partie

# Conclusion

- 6.1 Objectifs
- 6.2 Réalisations
- 6.3 Recherches futures

## Annexes

- Annexe 1 : Quelques notations
- Annexe 2 : Modèle de sortie de résultats de simulation.