TP2 : Manipulation des B-Arbres et B*-Arbres d’ordre 7
TP 2 : MANIPULER LES ARBRES EQUILIBRÉS
Module : Algorithme avoncé
Professeur : 1 Mme AROUSSI (s_aroussi@esi.dz)
Anné : 2025 - 2026

Université Blida 1
Faculté des Sciences
Département d’Informatique
Master IL (Ingénierie Logiciel)
Semestre 1

ESCHROUGUI Yousra
SEKKAL Wassila
OUR GROUPE :
08
Hadj Mohammed douae
Fares yasmine
Galleze nada
Lezoul imene

Environnement utilisé
Structures de données et Opérations
Les bibliothèques
Mode d’emploi
Résultats
PLAN DE PRESENTATION
01
Introduction
02
03
Analyse critique et Conclusion
04
05
06
07

Les arbres équilibrés sont essentiels dans les structures de données et les bases de données. Ils garantissent une gestion efficace des opérations de recherche, d'insertion et de suppression. Les B-arbres et B-arbres* sont parmi les plus utilisés pour optimiser l'accès aux données et sont des arbres de recherche équilibrés (AMR).
Ce TP vise à :
Comprendre la structure et les propriétés des B-arbres et B-arbre*.
Implémenter les opérations : recherche, insertion et suppression.
Visualiser le fonctionnement des arbres à l'aide d'une bibliothèque graphique.

INTRODUCTION
01

Comparaison entre B-arbre et B-arbre* :

INTRODUCTION

01

Comparaison entre B-arbre et B-arbre* :

INTRODUCTION

01

Comparaison entre B-arbre et B-arbre* :

INTRODUCTION

01

ENVIRONNEMENT UTILISÉ
-Processeur : Intel(R)core(TM)i7-5600U CPU @ 2.60GHZ
-RAM: 4.00 GB
-Carte Graphique : Intel(R) HD Graphics 5500
02
ENVIRONNEMENT MATÉRIEL :
ENVIRONNEMENT LOGICIEL :
-Os utilisé : Windows 10 PRO
-Language :Python “3.12.3”
-Environnement de dévloppement: Visual Studio Code
-Bibliothèque : Tkinter , Matpotlib , uuid, typing, math
ASSISTANCE DU CHATBOT IA :

ChatGPT (OpenAI) a contribué à hauteur d'environ 60 % d’assistance (correction, amélioration du code et visualisation graphique)

LES BIBLIOTHÈQUES

Tkinter: Définition et rôle

C’est un bibliothèque standard de Python pour la création d’interfaces graphiques (GUI).

03

Matplotlib: Définition et rôle

C’est un bibliothèque de traçage utilisée pour la visualisation graphique des graphes et arbres.

Typing : Définition et rôle

C’est un bibliothèque pour indiquer les types de données
dans le code (comme List, Optional, etc.).

LES BIBLIOTHÈQUES
03
Uuid : Définition et rôle
C'est un bibliothèque permet de générer
des identifiants uniques pour différencier les objets.
Math : Définition et rôle
C'est un bibliothèque contient des fonctions
mathématiques comme ceil, sqrt, sin, etc.

STRUCTURES DE DONNÉES ET OPÉRATIONS
04
Le principe d’équilibrage de B-arbre*:
Principe d’équilibrage :
Un B-arbre* conserve son équilibre après chaque opération d’insertion ou de suppression.

Contrairement à un B-arbre , où un nœud plein éclate immédiatement, un B-arbre* tente d’abord de redistribuer les clés entre les nœuds frères afin de minimiser les éclatements. Cette redistribution se fait de manière à ce que chaque nœud soit rempli au moins aux 2/3 de sa capacité, ce qui permet d’optimiser l’utilisation de la mémoire et de réduire la hauteur de l’arbre.

Voici un exemple illustratif montrant cette différence dans une série de d’opérations :

Exemple d’équilibrage après insertion
Cas 1 : Si un nœud devient plein → on tente une redistribution 2/3 avec un frère
Données initiales :
L = [25, 60, 35, 10, 5, 20, 65, 45, 70, 40, 50]
Ordre m = 7
Kmax = m-1 = 6
Kmin = 2*m/3 = 4
Le principe d’équilibrage de B-arbre*:

Le principe d’équilibrage de B-arbre*:
Avant redistribution 2/3 :
Après redistribution 2/3 :
Cas 2 : Si la redistribution 2/3 est impossible $\rightarrow$ on effectue un écletement en 3.
Données initiales :
L = [25, 60, 35, 10, 5, 20, 65, 45, 70, 40]
Ordre m = 5
Kmax = m-1 = 4
Kmin = 2*m/3 = 3

Le principe d’équilibrage de B-arbre*:
Avant écletement en 3 :
Après écletement en 3 :
Exemple d’équilibrage après suppression
Cas 1 : Redistribution 2/3 après suppression
Données initiales :
L 1= [25, 60, 35, 10, 5, 20, 65, 45, 70, 40, 50, 55, 30, 15, 22, 62, 64, 4, 8]
L2 =[15, 20, 10]
ordre = 7
Kmax = 6
Kmin = 4

Le principe d’équilibrage de B-arbre*:
Avant suppression de 10 :
Après redistribution 2/3:
Cas 2 : Fusion après suppression
Données initiales :
L 1= [25, 60, 35, 10, 5, 20, 65, 45, 70, 40, 50, 55]
L2 =[20]
ordre = 5
Kmax = 4
Kmin = 3

Le principe d’équilibrage de B-arbre*:
Avant suppression de 20 :
Après Fusion:

STRUCTURES DE DONNÉES ET OPÉRATIONS
04
La structure des données utilisée pour les types d’arbres :
Structure du B-arbre : La structure principale est définie par deux classes :
La structure d’un nœud :
La classe principale :

STRUCTURES DE DONNÉES ET OPÉRATIONS
04
La structure des données utilisée pour les types d’arbres :
Structure du B-arbre* : La structure principale est définie par deux classes :
La structure d’un nœud :
La classe principale :
Chaque nœud du B-arbre* contient entre (2/3)*m et m-1 clés, assurant un remplissage minimal de 66 %

STRUCTURES DE DONNÉES ET OPÉRATIONS
04
Les algorithmes des opérations principales de B-arbre :
Recherche :

STRUCTURES DE DONNÉES ET OPÉRATIONS
04
Les algorithmes des opérations principales de B-arbre :
Insertion : Insertion dans la racine

STRUCTURES DE DONNÉES ET OPÉRATIONS
04
Les algorithmes des opérations principales de B-arbre :
Insertion : Insertion dans un nœud non plein

STRUCTURES DE DONNÉES ET OPÉRATIONS
04
Les algorithmes des opérations principales de B-arbre :
Insertion : Eclatement d’un nœud plein

STRUCTURES DE DONNÉES ET OPÉRATIONS
04
Les algorithmes des opérations principales de B-arbre :
Suppression : Fonction principale

STRUCTURES DE DONNÉES ET OPÉRATIONS
04
Les algorithmes des opérations principales de B-arbre :
Suppression : Fonction interne

STRUCTURES DE DONNÉES ET OPÉRATIONS
04
Les algorithmes des opérations principales de B-arbre :
Suppression : Descente dans le sous-arbre

STRUCTURES DE DONNÉES ET OPÉRATIONS
04
Les algorithmes des opérations principales de B-arbre :
Suppression : Outils de suppression : Elles permettent de trouver la position, le prédécesseur ou le successeur d’une clé

STRUCTURES DE DONNÉES ET OPÉRATIONS
04
Les algorithmes des opérations principales de B-arbre :
Suppression : Rééquilibrage : Emprunt et Fusion

STRUCTURES DE DONNÉES ET OPÉRATIONS
04
Les algorithmes des opérations principales de B-arbre* :
Recherche : même principe d’un B-arbre

STRUCTURES DE DONNÉES ET OPÉRATIONS
04
Les algorithmes des opérations principales de B-arbre* :
Insertion : Insertion dans la racine

STRUCTURES DE DONNÉES ET OPÉRATIONS
04
Les algorithmes des opérations principales de B-arbre* :
Insertion : Insertion dans un nœud non plein

STRUCTURES DE DONNÉES ET OPÉRATIONS
04
Les algorithmes des opérations principales de B-arbre* :
Insertion : Redistribution 2/3

STRUCTURES DE DONNÉES ET OPÉRATIONS
04
Les algorithmes des opérations principales de B-arbre* :
Insertion : Eclatement en 3

STRUCTURES DE DONNÉES ET OPÉRATIONS
04
Les algorithmes des opérations principales de B-arbre* :
Suppression : Fonction principale

STRUCTURES DE DONNÉES ET OPÉRATIONS
04
Les algorithmes des opérations principales de B-arbre* :
Suppression : Fonction interne

STRUCTURES DE DONNÉES ET OPÉRATIONS
04
Les algorithmes des opérations principales de B-arbre* :
Suppression : Rééquilibrage (redistribution + fusion)

STRUCTURES DE DONNÉES ET OPÉRATIONS
04
Les algorithmes des opérations principales de B-arbre* :
Suppression : Fusion

MODE D’EMPLOI
Language de dévloppement : Python (version 3.13.9)
-lien :https://www.python.org/downloads/
Environnement de dévloppement : Visual Studio Code (version 1.90)
-lien: https://code.visualstudio.com/
Bibliothèque de graphes :
Matplotlib (version 3.10.0)
-lien :https://matplotlib.org/
Tkinter
-lien : https://docs.python.org/3/library/tkinter.html
Exécution
-python tp2_btree_app.py

05
Les installations requises:

LE FORMAT DES DONNÉES D’ENTRÉE :
Liste de clés séparées par des virgules : Par exemple L= [25, 60, 35, 10, 5, 20, 65, 45, 70, 40, 50, 55, 30, 15, 22, 62, 64, 4, 8]

Actions disponibles dans l’interface :
- Choisir entre B-arbre et B-arbre* .
- Créer l’arbre.
- Insérer, Supprimer, Rechercher.
- Affichage graphique automatique .

2-OUVRIR LE TERMINAL ET SE PLACER DANS LE DOSSIER CONTENANT LE SCRIPT PRINCIPAL
1- INSTALLER LES BIBLIOTHÈQUES NÉCESSAIRES
3. EXÉCUTER LE SCRIPT PRINCIPAL :PYTHON APPLICATION_GRAPHE.PY
4. UTILISER L’APPLICATION
les commandes ou scripts à exécuter :

LES RÉSULTATS ATTENDUS :
Comprendre la structure et les propriétés des B-arbres et B-arbres*.
Implémenter les opérations recherche, insertion et suppression.
Visualiser le fonctionnement des arbres à l’aide d’une bibliothèque graphique.

RÉSULTATS
Créez l'arbre R: à partir des valeurs L1 = [25, 60, 35, 10, 5, 20, 65, 45, 70, 40, 50, 55, 30, 15, 22, 62, 64, 4, 8]
06
Exécution des exercices de la série 1 du TD :
B-arbre d’ordre 7 :
Nous créons les clés de L1 jusqu'à la clé 65, lorsque le nœud est plein Kmax = 6, puis nous appliquons split
Nous continuons à insérer les clés jusqu’à la clé 50 du côté droit.
Lorsque le nœud devient plein, une division (éclatement en 2) est effectuée :
la clé 50 remonte à la racine, et les autres clés sont réparties de gauche à droite entre les nœuds enfants.

RÉSULTATS
06
Exécution des exercices de la série 1 du TD :
Après l’insertion successive des clés restantes, le nœud gauche devient plein.
Une division classique est alors effectuée : la clé 10 remonte à la racine,
ce qui donne une nouvelle racine [10, 25, 50] avec cinq sous-arbres équilibrés.

RÉSULTATS
06
Exécution des exercices de la série 1 du TD :
B-arbre* d’ordre 7 :
Après plusieurs insertions successives (5, 10, 20, 25, 35, 60, 65),
le nœud racine devient plein.
Une division classique est effectuée avec la clé médiane 25,
qui remonte pour devenir la nouvelle racine.
L’arbre est alors scindé en deux sous-arbres équilibrés
Après l’insertion de plusieurs clés supplémentaires (70, puis 40, 50),
le nœud de droite devient surchargé.
Une redistribution 2/3 vers la gauche

RÉSULTATS
06
Exécution des exercices de la série 1 du TD :
Après l’insertion des nouvelles clés (50, puis 30), nous appliquons redistribution .
Après l’insertion de la clé 15 nous appliquons écletement en 3 .

RÉSULTATS
Suppressions de l’arbre R : à partir des valeurs L2 = [15, 70, 50, 35, 60, 25]
06
Exécution des exercices de la série 1 du TD :
B-arbre d’ordre 7 :
suppression de 15 : Action : suppression directe → la feuille devient [20, 22].
suppression de 70 :
Action : suppression directe → [55, 60, 62, 64, 65].

RÉSULTATS
06
Exécution des exercices de la série 1 du TD :
Suppression de 50 : Action : on remplace 50 par son successeur trouvé dans la feuille suivante [55, 60, 62, 64, 65], le successeur est 55, Ensuite, on supprime 50 de la feuille.
Résultat : pas de fusion ni d'emprunt.
Cas remplacement par le successeur au niveau d'une feuille
suppression de 35 :
Action : suppression directe $\rightarrow$ [40, 45].

RÉSULTATS
06
Exécution des exercices de la série 1 du TD :
Suppression de 50 : Action : on remplace 50 par son successeur trouvé dans la feuille suivante [55, 60, 62, 64, 65], le successeur est 55, Ensuite, on supprime 50 de la feuille.
Résultat : pas de fusion ni d'emprunt.
Cas remplacement par le successeur au niveau d'une feuille
suppression de 35 :
Action : suppression directe $\rightarrow$ [40, 45].

ANALYSE CRITIQUE ET CONCLUSION
LES difficultés rencontrées
Limites
Pas d’animation (insertions instantanées)
Redistribution simplifiée (2/3 exacte non visualisée)

Gestion correcte de la redistribution dans le B-arbre* .
Calcul automatique des positions pour l’affichage centré.
07
Points positifs de la solution
Interface graphique intuitive.
Visualisation claire et automatique.
Code structuré, réutilisable et extensible.

MERCI
POUR
VOTRE ATTENTION