2.8. *B*-дерева

Структуру даних «дерево» можна організовувати також на жорсткому диску, якщо об'єм оперативної пам'яті недостатній. Різниця складається

у тому, що посилання на вузли означають адреси на диску, а не в оперативній пам'яті.

Якщо ми маємо бінарне дерево пошуку із кількістю вузлів 10^6, то для локалізації певного вузла нам буде потрібно виконати не більше $log_2 10^6 = 20$ звернень, а це вже достатньо багато, враховуючи час на доступ до диску.

Якщо відбувається звернення до одиночного елемента, який розташований на зовнішньому носії, то майже без додаткових затрат часу можна звертатися до групи елементів при умові, що вони розташовані на одній доріжці жорсткого диску. Звідки висновок: все дерево слід розділити на піддерева, рахуючи, що вони одночасно всі доступні і назвемо їх сторінками. Зменшення кількості звернень до диску може бути вельми суттєвим. Припустимо, що кожна сторінка буде мати 100 вузлів, тоді пошук елемента у згаданому дереві потребує всього $log_{100} 10^6 = 3$ звернень до диску.

У реальному випадку формується дерево із сторінками, кожна з яких містить від n до $2n$ вузлів (де n – постійна величина, наприклад 100). Тоді у дереві з N елементами і максимальним розміром сторінки $2n$ найгірший випадок пошуку потребує $log_n N$ звернень до сторінок (див. рис. 2.31).

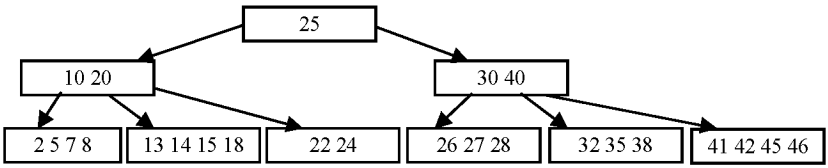

Рис. 2.31. Приклад зображення *B*-дерева. Кожна сторінка містить по 4 вузла

Коли знайдена потрібна сторінка, її дані зчитуються у оперативну пам'ять, де відбувається остаточний пошук. Якщо сторінка невелика, то організується послідовний пошук (при цьому дані на сторінках можна не сортувати), у протилежному випадку – бінарний. Який підхід вибрати залежить від розміру сторінки та вимоги до частоти звернень.

Така схема організації даних потребує відносно простих алгоритмів пошуку, включення та видалення вузлів. Якщо якась сторінка переповнюється, то організується нова, яка буде у подальшому поступово заповнюватись.