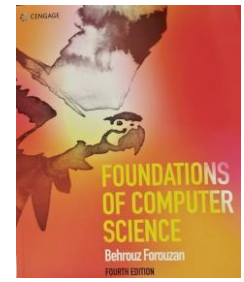
Chapter 13



File Structure

檔案結構

Objectives 學習目標

After studying this chapter, students should be able to:

- **Define two categories of access methods: sequential access and random access. 檔案存取分順序存取與隨機存取**
- **Understand the structure of sequential files and how they are updated.**
- **Understand the structure of indexed files and the relation between the index and the data file. 索引檔案的結構**
- **Understand the idea behind hashed files and describe some hashing methods. 混雜檔案的結構與存取方法**
- **Describe address collisions and how they can be resolved.**
- **Define directories and how they can be used to organize files.**
- **Distinguish between text and binary files.文字檔與二進制檔的異同**

ACCESS METHODS 存取方法

When we design a file, the important issue is how we will **retrieve information** (a specific record) from the file. Sometimes we need to process records one after another, whereas sometimes we need to access a specific record quickly without retrieving the preceding records. The **access method** determines how records can be retrieved: **sequentially** or **randomly**. 從檔案中存取一筆紀錄資料做調查用，分為順序存取與隨機存取

13.3

Sequential access 順序存取

If we need to access a file sequentially—that is, one record after another, from beginning to end —we use a **sequential file structure**. 從順序檔案中從頭到尾存取一筆紀錄資料接一筆紀錄資料

Random access 隨機存取

If we need to access a specific record without having to retrieve all records before it, we use a file structure that allows random access. **Two file structures allow this: indexed files and hashed files**. 隨機存取分索引與混雜等兩種檔案結構

13.4

SEQUENTIAL FILES 順序檔案

A sequential file is one in which records can only be accessed one after another from **beginning to end**. Figure 13.2 shows the layout of a sequential file. Records are stored one after another in auxiliary storage, such as tape or disk, and **there is an EOF (end-of-file) marker after the last record**. The operating system has no information about the record addresses, it only knows where the whole file is stored. The only thing known to the operating system is that the records are sequential. 檔案的尾端有一個的特別標誌**EOF**

13.5

INDEXED FILES 索引檔案

To **access a record** in a file randomly, we need to **know the address** of the record.

經由索引表的對映，間接存取隨機檔案的紀錄資料

13.6

HASHED FILES 混雜檔案

A **hashed file** uses a mathematical function to accomplish this mapping. The user gives the key, the function maps the key to the address and passes it to the operating system, and the record is retrieved. 經由混雜數學函數的對映，間接存取隨機檔案的紀錄資料

13.7

Hashing methods 各種混雜存取方法

For **key-address mapping**, we can select one of several hashing methods. We discuss a few of them here.

Direct hashing 直接混雜存取方法

In direct hashing, **the key is the data file address without any algorithmic manipulation.** 沒有混雜數學函數，金鎖就是紀錄資料的位址。 The file must therefore contain a record for every possible key. Although situations suitable for direct hashing are limited, it can be very powerful, because **it guarantees that there are no synonyms** or **collisions** 保證不會有同名或碰撞, as with other methods.

13.8

Modulo division hashing 除法混雜存取方法

Also known as **division remainder** hashing, the modulo division method **divides the key by the file size** and uses the remainder plus 1 for the address. This gives the simple hashing algorithm that follows, where list_size is the number of elements in the file. The reason for adding a 1 to the mod operation result is that our list starts with 1 instead of 0.
金鎖除以檔案大小取 餘數+1，轉換為紀錄資料的位址

13.9

Collision 碰撞

Generally, the **population of keys** for a hashed list is **greater than the number of records** in the data file.
For example, if we have a file of 50 students for a class in which the students are identified by the last four digits of their social security number, then there are 200 possible keys for each element in the file (10,000/50). Because there are many keys for each address in the file, there is a possibility that **more than one key will hash to the same address** in the file. 碰撞：金鎖經由混雜函數轉換為紀錄資料的位址，可能產生不同的金鎖而轉換為相同的位址。We call the set of keys that hash to the same address in our list synonyms. The collision concept is illustrated in Figure 13.10.

13.10

Collision resolution 碰撞的解決之道

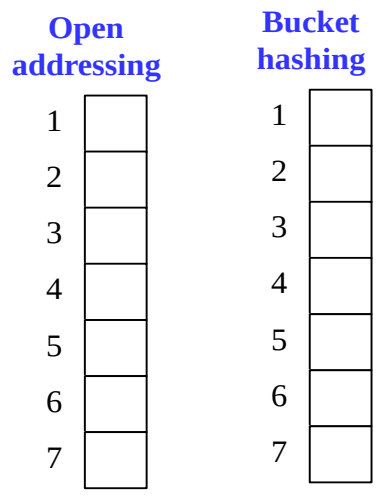
With the exception of the **direct method**, none of the methods we have discussed for hashing creates one-to-one mappings. This means that when we hash a new key to an address, we may create a collision. There are several methods for handling collisions, each of them independent of the hashing algorithm. That is, any hashing method can be used with any collision resolution method. In this section, we discuss some of these methods.

碰撞的解決之道：下列三種方法

1. **Open addressing resolution** 開放式位址之解決方法
2. **Linked list resolution** 鍊結列之解決方法
3. **Bucket hashing resolution** 吊桶混雜之解決方法

13.11

Exercise: Using the hash function h(i) = Xi mod 7 + 1 for the keys of *Xi*, 99, 55, 39, 67, 105, 201, and 158, please show the hash table using open addressing and bucket hashing resolutions if collision.請顯示混雜表內容(使用不同碰撞解決方法)



13.12

Text files 文字檔

A text file is a file of characters. It cannot contain integers, floating-point numbers, or any other data structures in their internal memory format. To store these data types, they **must be converted to their character equivalent formats**. 文字檔不包含整數、浮點數或其他的資料結構，且其資料必須可以轉換為其對應的文字格式
Some files can only use character data types. Most notable are file streams (input/output objects in some object-oriented language like C++) for keyboards, monitors and printers. This is why we need special functions to format data that is input from or output to these devices.

13.13

Binary files 二進制檔

A binary file is a collection of data stored in the internal format of the computer. In this definition, **data can be an integer** (including other data types represented as unsigned integers, **such as image, audio, or video**), **a floating-point number or any other structured data** (except a file). 二進制檔為電腦內部格式之資料儲存，可包含整數(如聲音、影像等)、浮點數或其他的資料結構。
Unlike text files, **binary files contain data that is meaningful only if it is properly interpreted by a program**. 二進制檔資料必須經由其對應的程式來轉換才有意義 If the data is textual, one byte is used to represent one character (in ASCII encoding). But if the data is numeric, two or more bytes are considered a data item.

13.14

Review Questions

- Please show the **major differences** between **sequential file** and **random access file.**
- Please state **address collisions in hash-based access** and **how they can be resolved**.
- Please introduce the **major difference** of **text** and **binary files**.
- What is the **ended character** or **symbol** in a **text file**?
- Using the hash function of **h(i) = Xi mod 7 + 1** for the sequential keys of *Xi*, 99, 55, 39, 67, 105, 201, and 158, please show the hash table with the collision solutions of **open addressing** and **linked list**.

13.15