Data Structures and Algorithms (11)

Instructor: Ming Zhang
Textbook Authors: Ming Zhang, Tengjiao Wang and Haiyan Zhao
Higher Education Press, 2008.6 (the "Eleventh Five-Year" national planning textbook)

https://courses.edx.org/courses/PekingX/04830050x/2T2014/

Chapter 11 Indexing

- Basic Concepts
- 11.1 Linear Indexing
- 11.2 Static Indexing
- 11.3 Inverted Indexing
- 11.4 Dynamic Indexing
- 11.5 Bit Indexing
- 11.6 Red-Black Tree

Entry-Sequenced File

- **Entry-sequenced file**
  - **Storing the records in the order when they enter the system**
  - **Entry-sequenced file is like an un-ordered linear list on disk**

Thus it does not support efficient access.

Primary Key

- Primary key represents each record in the database uniquely
  - e.g., primary key of a company staff system could be the id card number.
  - If there is only primary key, access would be inflexible.

Secondary Key

- **Secondary key** : keys that could appear repeatedly in the database
- Secondary key indexing associates a secondary key with primary keys of records that contain this secondary key
  - Most indexing is accomplished using secondary key.

Indexing

- Indexing associates a key with the locations where the corresponding data is stored.
  - (key, pointer)
  - Pointers point to full record in the database.
- Index file is the type of files that keeps track of such association.
- Indexing is crucial in constructing large-size database.
  - Efficient access
  - Insertion, update, deletion

Index File

- A main file can have several index files
  - Each index file usually supports one key
  - Do not need to re-sort the main file
- One can access key values in the main file efficiently by using index file.

Dense Indexing vs Sparse Indexing

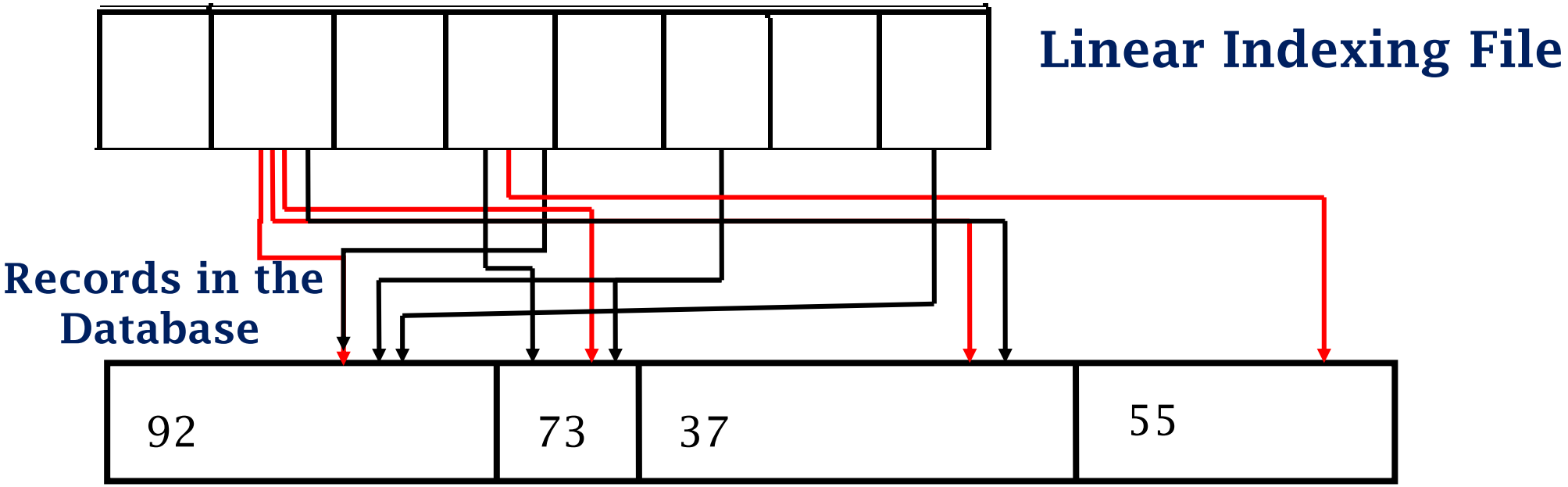
- Dense indexing: construct an entry for each record
  - Main file is not sorted by keys.
- Sparse indexing: construct an entry for each group of records.
  - Records are stored in the order of keys.
  - Records are divided into groups (blocks)
  - ➢Indexing pointer points to the starting location of a group of records (a block).

11.1 Linear Indexing

- Basic Concepts
- Pros & cons
- Secondary Linear Indexing

Linear Indexing File

- Sort by keys.
- Pointers in indexing file point to the starting location in which records are stored, or the starting location in which the primary keys are stored.



Problems of Linear Indexing

- Linear indexing is too large, and needs to be stored on the disks
- One indexing process may involve several disk accesses, thus slows down the indexing.
  - Secondary linear indexing

- Updating linear indexing
  - When inserting or deleting records in the database.

Secondary Linear Indexing

- e.g., each block in the disks is 1024 bytes, each (key, pointer) entry needs 8 bytes.
- 1024 / 8 = 128
- Each block can store 128 such entries.
  - Suppose data file consists of 10000 records.
  - Dense main linear indexing consists of 10000 records.
  - 10000/128 = 78.1
  - Thus main linear indexing takes up 79 blocks.
  - Accordingly, secondary linear indexing file consists of 79 entries.
  - This secondary linear indexing file can be stored on a single block.

An Example of Secondary Linear Indexing

- Key is the same as the one of the first record in the corresponding disk.
- Pointer points to the starting location of the corresponding disk.

secondary index

1	2003	5744	10723	……

main index

1…… 2002	2003 5583	5744 9297	10723 13293
……			

Disk block

pointer with key of 2555

e.g., find record with key of 2555

Secondary Indexing	1	2003	5744	10723	……

Linear Indexing	1 2002	2003 5583	5744 9297	10723 13293	
	……				

Disk block

record with key of 2555

1. Read secondary index file into memory.
2. Find the location of the main index containing the largest key no larger than 2555 by bisection method--------record with key of 2003
3. Find the disk in which the secondary index file is stored according to the pointer of record 2003, and read the corresponding disk into memory.
4. Determine the location by bisection method.
5. Read the record into the memory and thus finish one indexing access.

11.2 Static Indexing

Static Indexing

- constructed when records are read
- is stable once constructed through the whole process of system running (inserting/deleting records)
- is allowed to be altered only when file re-orgainzation

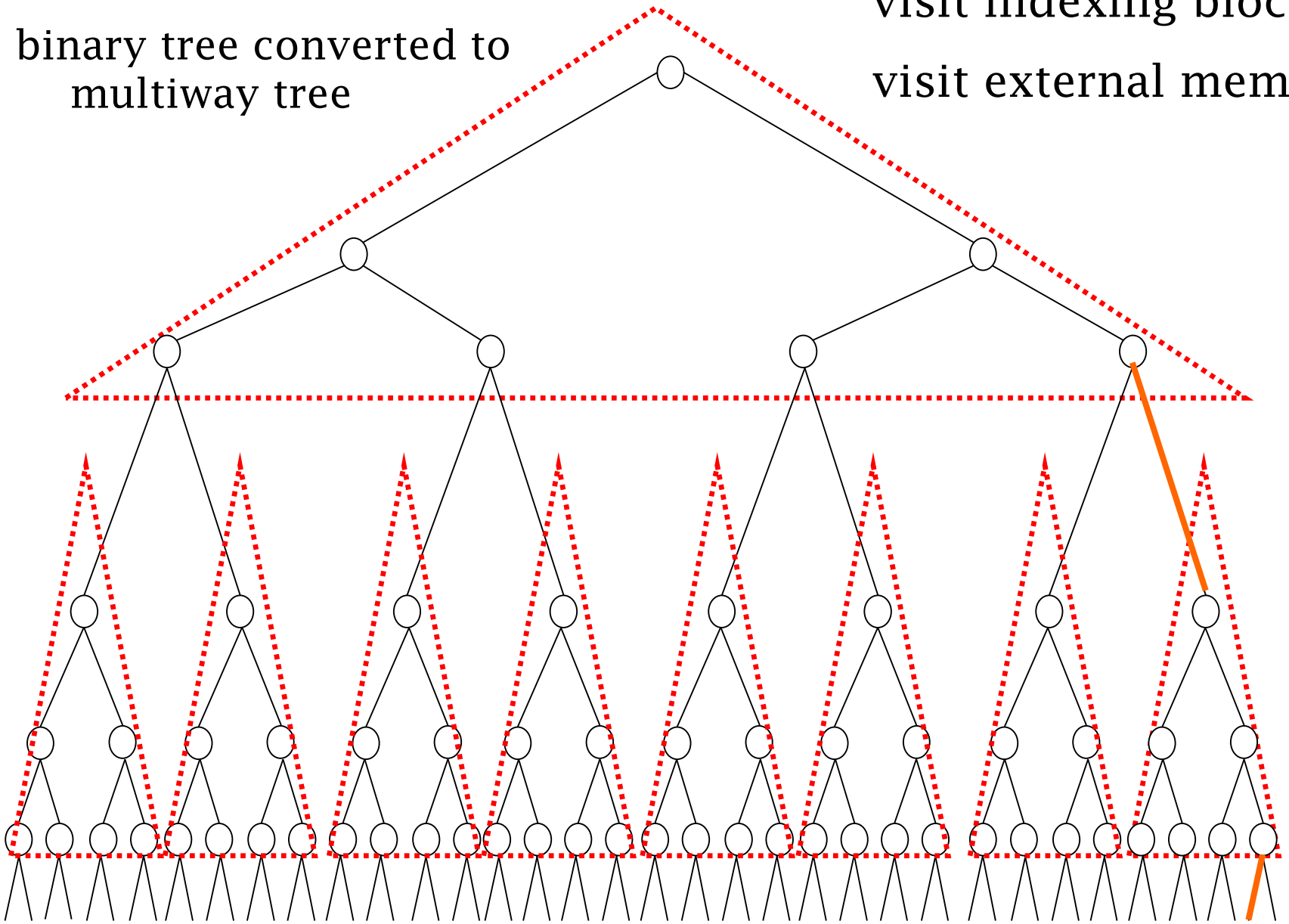
Multiway Tree

- We usually use multiway tree instead of binary tree when organizing indexing
- reduce visit time of external memory significantly

binary tree converted to multiway tree

visit indexing block twice

visit external memory once



ISAM

- Index Sequential Access Method
  - designed for disk accessing
  - use multi-level indexing
    - main index
    - cylinder index
    - track index
- Before using B^+ tree based VSAM (Virtual Storage Access Method), IBM used to use ISAM generally.

11.2 Static Indexing

C_0

T_0	400	T_1	625	T_2	1000	T_3

main index

T_1	80	C_1T_0	200	C_2T_0	400	C_3T_0
T_2					625	C_6T_0
T_3					1000	C_9T_0
	⋮					

cylinder index

C_1

T_0	40 T_1	40 T_1	80 T_2	80 T_2	…
T_1	R_{10}	R_{20}	R_{30}	R_{40}	
T_2	R_{50}	R_{60}	R_{70}	R_{80}	
⋮					
T_7		⋮			

track index

basic area

spilling area

11.2 Static Indexing

C_2

T_0	150 T_1	150 T_1	200 T_2	200 T_2	…
T_1	R_{90}	R_{110}	R_{120}	R_{150}	
T_2	R_{160}	R_{175}	R_{190}	R_{200}	
⋮		⋮			
T_7					

track index

basic area

spilling area

⋮

C_9

T_0	890 T_1	890 T_1	1000 T_2	1000 T_2	…
T_1	R_{830}	R_{840}	R_{880}	R_{890}	
T_2	R_{920}	R_{930}	R_{980}	R_{1000}	
⋮		⋮			
T_7					

track index

basic area

spilling area

Discussion

- Under what circumstances is secondary indexing needed?
- How to set the size of multiway tree?

Data Structures and Algorithms

Thanks

the National Elaborate Course (Only available for IPs in China)
http://www.jpk.pku.edu.cn/pkujpk/course/sjjg/
Ming Zhang, Tengjiao Wang and Haiyan Zhao
Higher Education Press, 2008.6 (awarded as the "Eleventh Five-Year" national planning textbook)