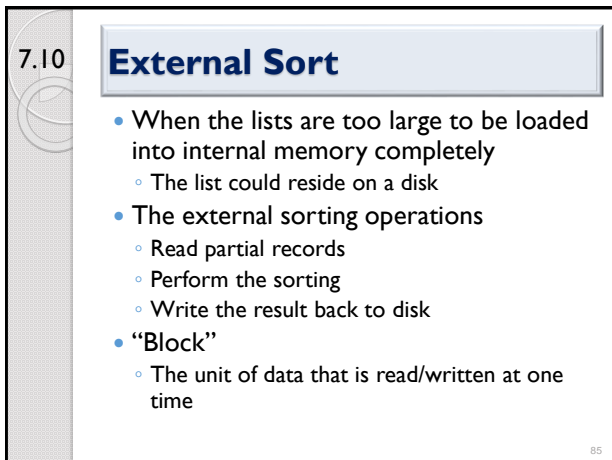


7.10

External Sorting

2018/10/20 © Ren-Song Tsay, NTHU, Taiwan 84

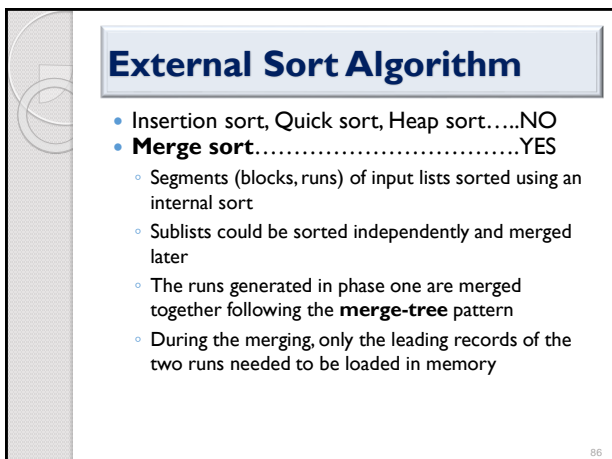


7.10

External Sort

- When the lists are too large to be loaded into internal memory completely
 - The list could reside on a disk
- The external sorting operations
 - Read partial records
 - Perform the sorting
 - Write the result back to disk
- “Block”
 - The unit of data that is read/written at one time

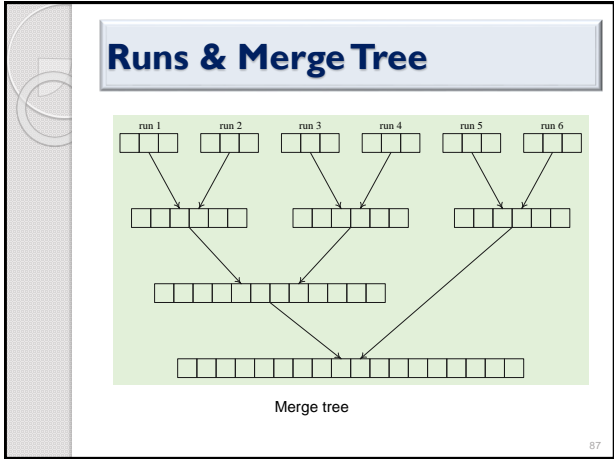
85

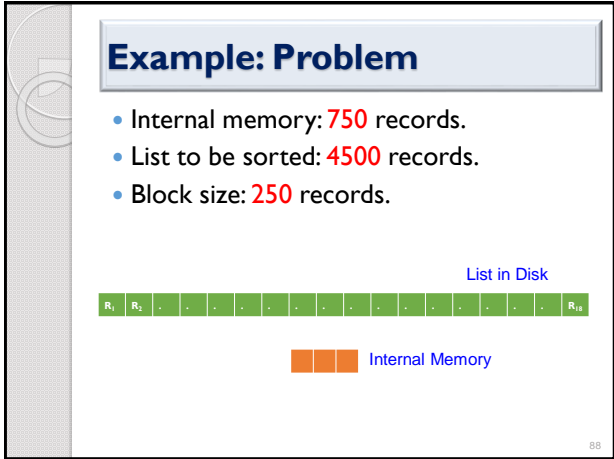


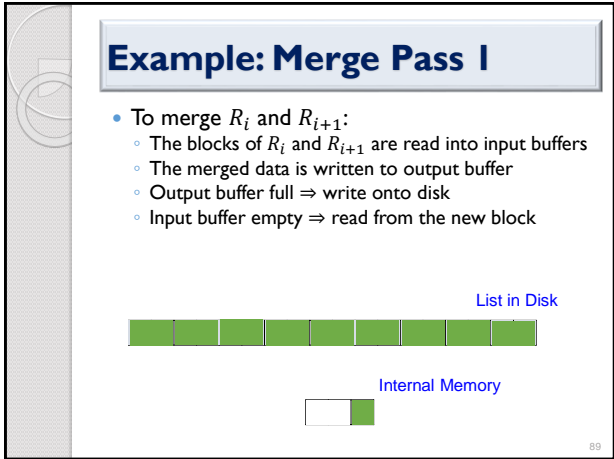
External Sort Algorithm

- Insertion sort, Quick sort, Heap sort.....NO
- **Merge sort**.....YES
 - Segments (blocks, runs) of input lists sorted using an internal sort
 - Sublists could be sorted independently and merged later
 - The runs generated in phase one are merged together following the **merge-tree** pattern
 - During the merging, only the leading records of the two runs needed to be loaded in memory

86

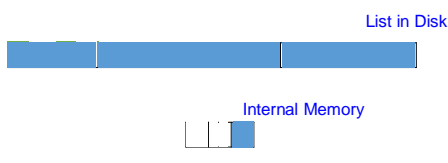






Example: Merge Pass 2

- To merge R_i and R_j :
 - The blocks of R_i and R_j are read into input buffers
 - The merged data is written to output buffer
 - Output buffer full \Rightarrow write onto disk
 - Input buffer empty \Rightarrow read from the new block

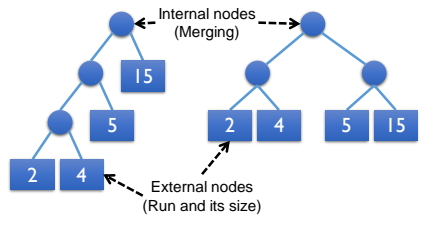


90

7.10.5

Optimal Merging of Runs

- Runs with different sizes.
- Different merge sequence may result in different runtime.

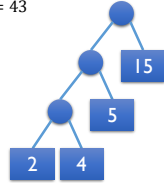


93

Runtime Evaluation

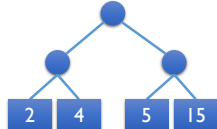
Merge tree A

$$\begin{aligned}
 \text{Cost} &= (2 + 4) + (2 + 4 + 5) + (2 + 4 + 5 + 15) \\
 &= 2 * 3 + 4 * 3 + 5 * 2 + 15 * 1 \\
 &= 43
 \end{aligned}$$



Merge tree B

$$\begin{aligned}
 \text{Cost} &= 2 * 2 + 4 * 2 + 5 * 2 + 15 * 2 = 52
 \end{aligned}$$



94

Weighted External Path Length

- The total number of merge steps is equal to:

$$\sum_{i=1}^n s_i d_i$$

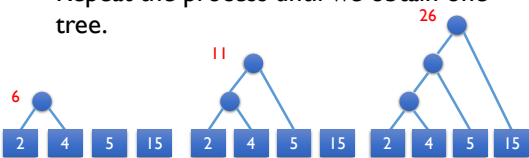
- Where s_i is the size of Run i and d_i is the distance from the node to root.
- How to build a merge tree such that the total cost is minimized?**

95

Sort by Block Size

- Sort runs using its size.
- Take the two runs with **least sizes** and combine them into a tree.
- Repeat the process until we obtain one tree.

2 4 5 15



96

Similar to Message Encoding

- Given a set of messages $\{M_1, M_2, \dots, M_i\}$
- How do we encode each M_i using a binary code such that the total number of message bits is minimum?

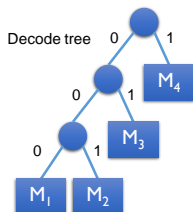
	Encode 1	Encode 2	Encode 3
M_1	0	0001	0001
M_2	1	0010	1
M_3	10	0100	01
M_4	11	1000	001

97

7.10.5
F7.28

Huffman Code

- Using a binary tree, called **decode tree** to encode messages.



Huffman Code	
M_1	000
M_2	001
M_3	01
M_4	1

99

Decoding Cost

- Cost of decoding a code word is proportional to the number of bits of the word.
 - Decoding a code word contain $2 * M_1$ and $1 * M_4$ requires process $2 * 3 + 1 = 7$ bits.
- Assume the message M_i with encoded bit length d_i , occurring frequency is s_i , then the total cost of the code word is:

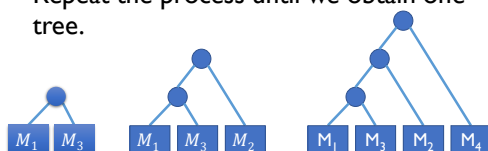
$$\sum_{i=1}^n s_i d_i$$

- How do we construct a decode tree such that the decoding cost is minimized?

100

Optimal Merge Tree


- Follow Huffman Code Method
- Sort the message according to s_i
 - M_1 (2), M_3 (4), M_2 (5), M_4 (15)
- Take two messages with the **least** s_i and combine them into a tree (a new message)
- Repeat the process until we obtain one tree.



101

Self-Study Topics

- 7.8 List and Table Sorts



103
