Algorithm – IV

Sorting disk Files:

disk files are too large to be fitted in internal memory and, therefore, not fit for sorting by array sorting or internal sorting techniques. There is no general algorithm though. We provide here a scheme, most external sorting techniques being proprietary.

data from the file are filled in some buffers, where the buffer-heads play a tournament sort. The buffers are FIFO. Winners go up.

Outputs from the tournament form a sorted buffer. That is why the corresponding tree is called the tree of winners.

The process goes on recursively, outputs becoming inputs and obviously larger.

The input buffers are replenished with a second level of input buffers when empty or by a (few) floating input buffers.

An example is given here:

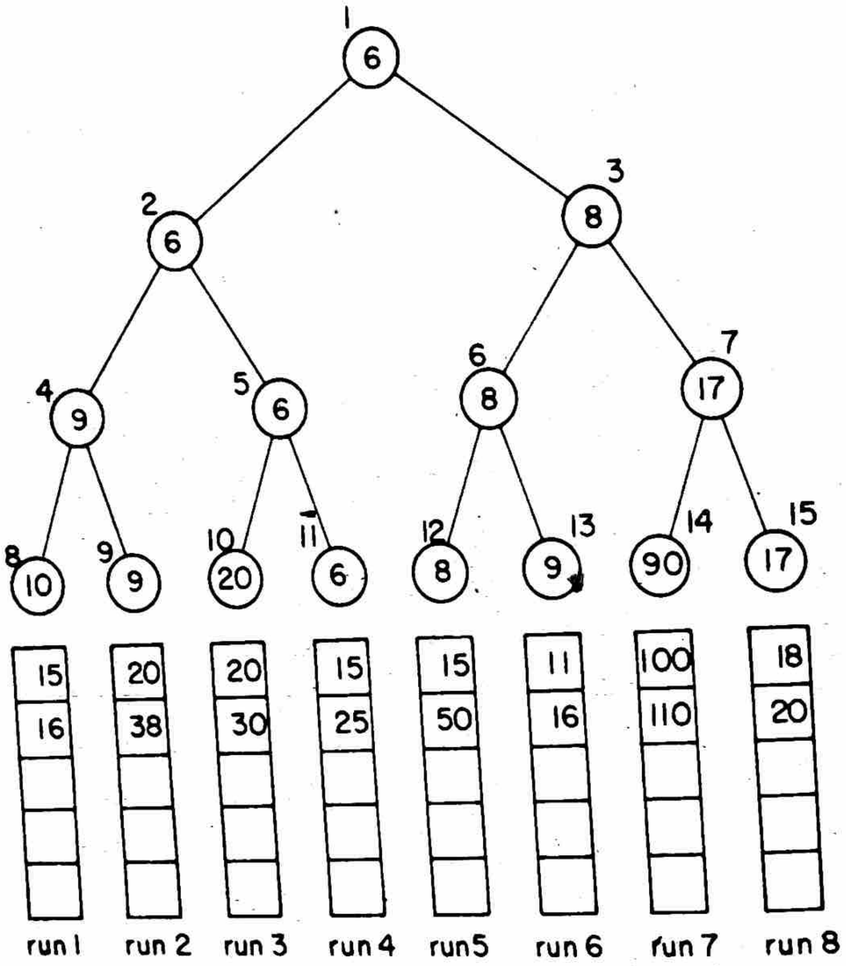

Figure 8.9 Selection tree for 8-way merge showing the first three keys in each of the 8 runs.

the asymptotic internal processing time becomes $O(n \log_2 k \log_k m) = O(n \log_2 m)$. The internal processing time is independent of k.

Note, however, that the internal processing time will be increased slightly because of the increased overhead associated with maintaining the tree. This overhead may be reduced somewhat if each node represents the loser of the tournament rather than the winner. After the record with smallest key is output, the selection tree of figure 8.9 is to be restructured. Since the record with the smallest key value is in run 4, this restructuring involves inserting the next record from this run into the tree. The next record has key value 15. Tournaments are played between sibling nodes along the path from node 11 to the root. Since these sibling nodes represent the losers of tournaments played earlier, we would simplify the restructuring process by placing in each nonleaf node a pointer to the record that loses the tourna-

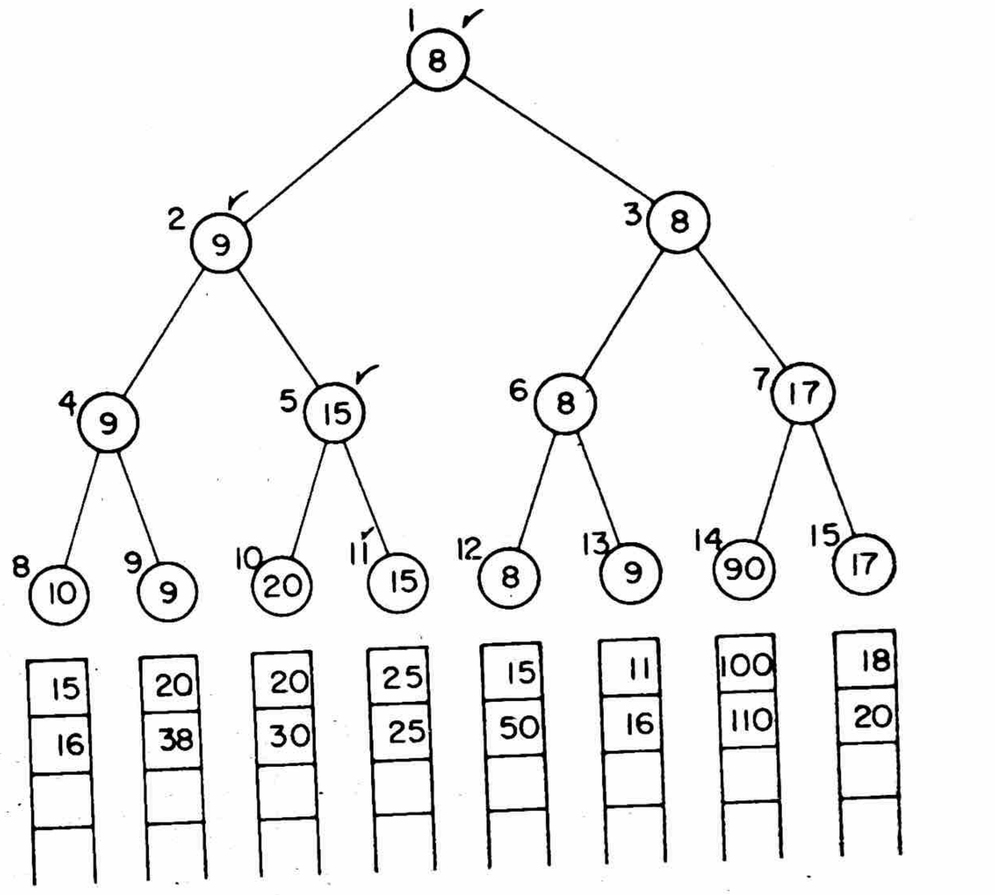

Figure 8.10 Selection tree of Figure 8.9 after one record has been output and tree restructured. Nodes that were changed are marked by ✓

ment rather than to the winner of the tournament. A tournament tree in which each nonleaf node retains a pointer to the loser is called a *tree of losers*. Figure 8.11 shows the tree of losers corresponding to the selection tree of figure 8.9. For convenience, each node contains the key value of a record rather than a pointer to the record represented. The leaf nodes represent the first record in each run. An additional node, node 0, has been added to represent the overall winner of the tournament. Following the output of the overall winner, the tree is restructured by playing tournaments along the path from node 11 to node 1. The records with which these tournaments are to be played are readily available from the parent nodes. We shall see more of loser trees when we study run generation in section 8.2.3.

In going to a higher order merge, we save on the amount of input/output being carried out. There is no significant loss in internal processing speed. Even though the internal processing time is relatively insensitive to the order of the merge, the decrease in input/output time is not as much as