External Sorting

Example of Two-Way Sorting:

N = 14, M = 3 (14 records on tape Ta, memory capacity: 3 records.)

Ta: 17, 3, 29, 56, 24, 18, 4, 9, 10, 6, 45, 36, 11, 43

A. Sorting of runs:

1. **Read 3 (M) records in main memory, quick sort them and store them on Tb1:**

   17, 3, 29 -> 3, 17, 29

   Tb1: 3, 17, 29

2. **Read the next 3 records in main memory, sort them and store them on Tb2**

   56, 24, 18 -> 18, 24, 56

   Tb2: 18, 24, 56

3. **Read the next 3 records in main memory, sort them and store them on Tb1**

   4, 9, 10 -> 4, 9, 10

   Tb1: 3, 17, 29, **4, 9, 10**

4. **Read the next 3 records in main memory, sort them and store them on Tb2**

   6, 45, 36 -> 6, 36, 45

   Tb2: 18, 24, 56, **6, 36, 45**

5. **Read the next 3 records in main memory, sort them and store them on Tb1**

   (there are only two records left)
   11, 43 -> 11, 43

   Tb1: 3, 17, 29, 4, 9, 10, **11, 43**

   At the end of this process we will have three runs on Tb1 and two runs on Tb2:

Tb1: 3, 17, 29 | 4, 9, 10 | 11, 43

Tb2: 18, 24, 56 | 6, 36, 45 |

B. Merging of runs

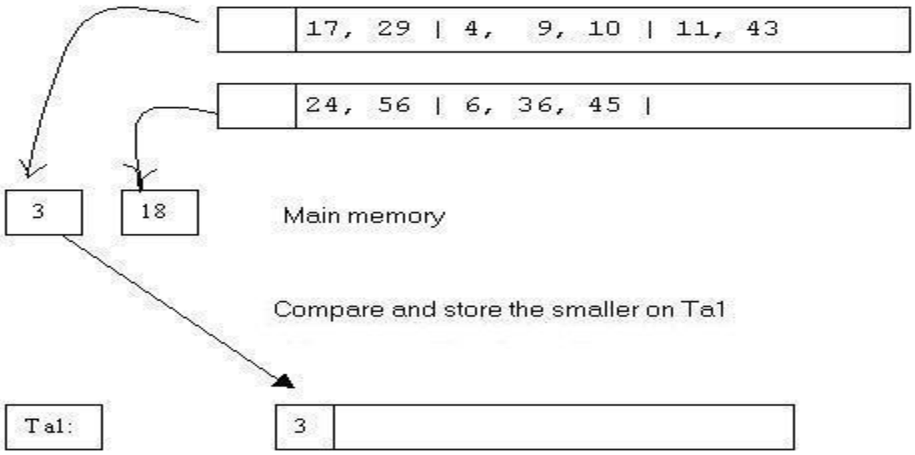
B1. Merging runs of length 3 to obtain runs of length 6.

Source tapes: Tb1 and Tb2, **result** on Ta1 and Ta2.

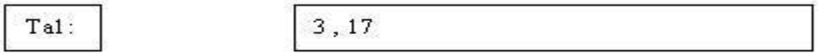
Merge the first two runs (on Tb1 and Tb2) and store the result on Ta1.

Tb1: 3, 17, 29 | 4, 9, 10 | 11, 43

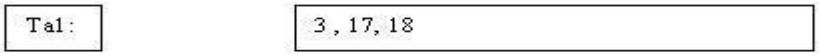
Tb2: 18, 24, 56 | 6, 36, 45 |

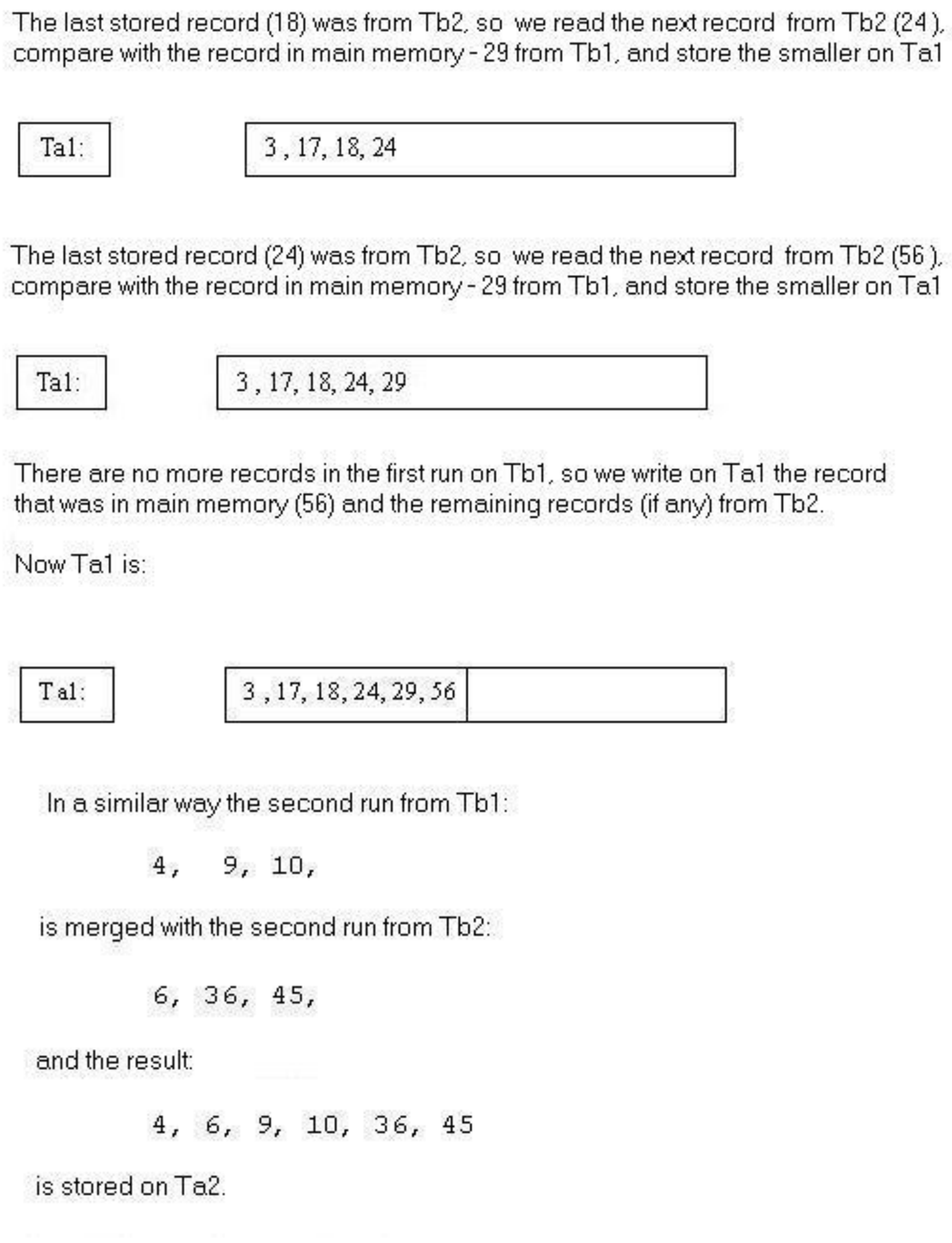


Read the next record (17) from Tb1, compare with 18, and store the smaller on Ta1



Read the next record (29) from Tb1, compare with 18, and store the smaller on Ta1





The last stored record (18) was from Tb2, so we read the next record from Tb2 (24), compare with the record in main memory - 29 from Tb1, and store the smaller on Ta1

Ta1: 3 , 17, 18, 24

The last stored record (24) was from Tb2, so we read the next record from Tb2 (56), compare with the record in main memory - 29 from Tb1, and store the smaller on Ta1

Ta1: 3 , 17, 18, 24, 29

There are no more records in the first run on Tb1, so we write on Ta1 the record that was in main memory (56) and the remaining records (if any) from Tb2.

Now Ta1 is:

Ta1: 3 , 17, 18, 24, 29, 56

In a similar way the second run from Tb1:

```
4,   9, 10,
```

is merged with the second run from Tb2:

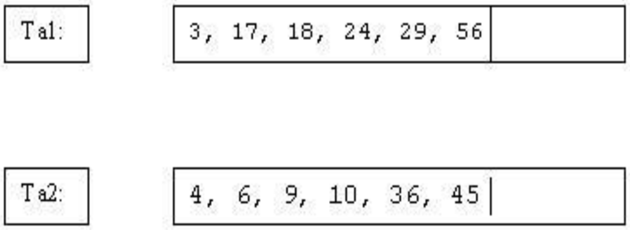
```
6, 36, 45,
```

and the result:

```
4, 6, 9, 10, 36, 45
```

is stored on Ta2.

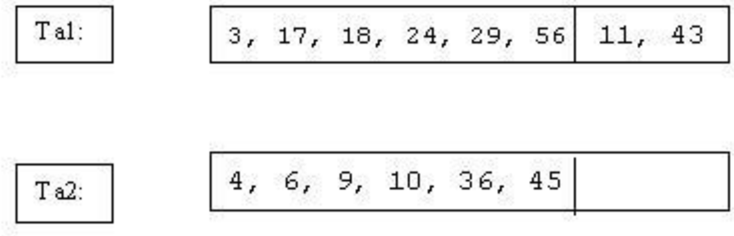
Thus we have the first two runs on Ta1 and Ta2, each twice the size of the original runs:



Ta1: 3, 17, 18, 24, 29, 56 |

Ta2: 4, 6, 9, 10, 36, 45 |

Next we merge the third runs on Tb1 and Tb2 and store the result on Ta1. Since only Tb1 contains a third run, it is copied onto Ta1:



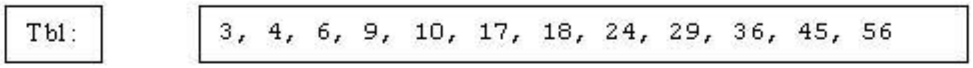
Ta1: 3, 17, 18, 24, 29, 56 | 11, 43

Ta2: 4, 6, 9, 10, 36, 45 |

B2. Merging runs of length 6 to obtain runs of length 12.

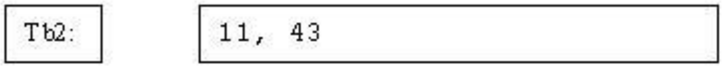
Source tapes: Ta1 and Ta2. **Result** on Tb1 and Tb2:

After merging the first two runs from Ta1 and Ta2, we get a run of length 12, stored on Tb1:



Tb1: 3, 4, 6, 9, 10, 17, 18, 24, 29, 36, 45, 56

The second set of runs is only one run, copied to Tb2

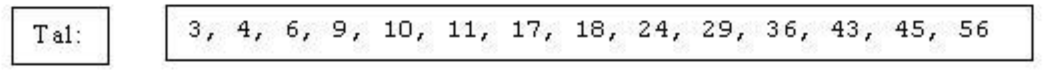


Tb2: 11, 43

Now on each tape there is only one run. The last step is to merge these two runs and to get the entire file sorted.

B3. Merging the last two runs.

The result is:



Ta1:

3, 4, 6, 9, 10, 11, 17, 18, 24, 29, 36, 43, 45, 56

Number of passes: **log(N/M)**

In each pass the size of the runs is doubled, thus we need **[log(N/M)]+1** to get to a run equal in size to the original file. This run would be the entire file sorted.

In the example we needed three passes (B1, B2 and B3) because [Log(14/3)] + 1 = 3.