DBMS Tutorial 07.11.2018

Merge Sort

https://youtu.be/GCae1WNvnZM?t=12

Merge Sort

- Merge = take two sorted lists and repeatedly choose the smaller of the “heads” of the lists (head = first of the unchosen).
  - Example: merge 1,3,4,8 with 2,5,7,9 = 1,2,3,4,5,7,8,9.

- Merge Sort based on recursive algorithm: divide records into two parts; recursively merge-sort the parts, and merge the resulting lists.

Two-Phase, Multiway Merge Sort

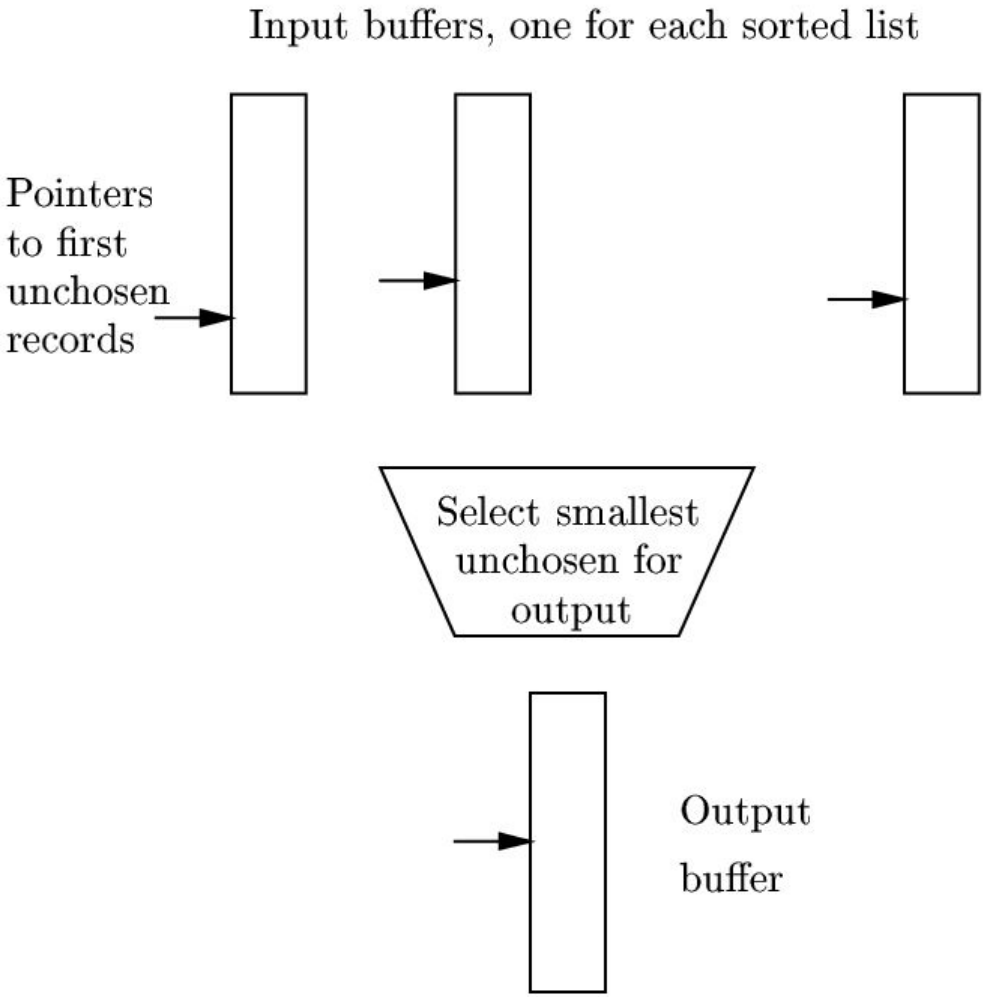
Traditional sort algorithms (e.g., quick sort) requires that the file must fit in the main memory to be sorted

The Two-Pass Multiway Merge Sort (TPMMS) algorithm can be used to sort files that are larger than the main memory

- **Phase 1**
  1. Fill main memory with records.
  2. Sort using favorite main- memory sort.
  3. Write sorted sublist to disk.
  4. Repeat until all records have been put into one of the sorted lists.

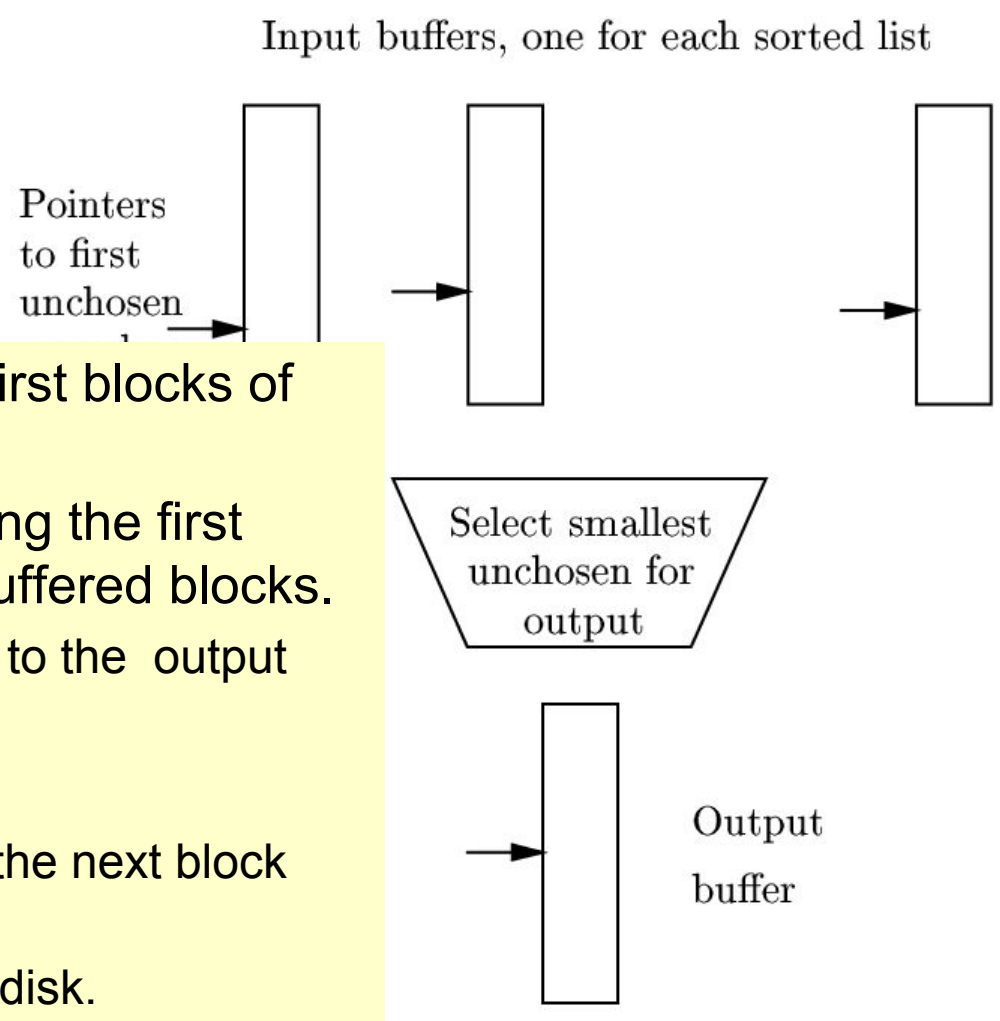
Phase 2

- Use one buffer for each of the sorted sublists and one buffer for an output block.



Phase 2

- Use one buffer for each of the sorted sublists and one buffer for an output block.



- Initially load input buffers with the first blocks of their respective sorted lists.
- Repeatedly run a competition among the first unchosen records of each of the buffered blocks.
  - Move the record with the least key to the output block; it is now “chosen.”
- Manage the buffers as needed:
  - If an input block is exhausted, get the next block from the same file.
  - If the output block is full, write it to disk.

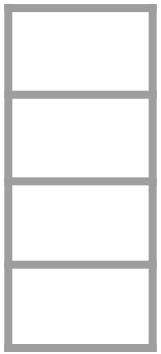
Toy Example

- 1 block can hold 2 tuples.
- A file/relation with 23 tuples (only the sorting key shown here).
- Suppose main memory (M) can hold 4 blocks (/buffers) i.e. 8 tuples.

File on Disk:

	12,10	25,20	40,30	27,29	14,18	45,23	70,65	35,11	49, 47	22,21	46,34	31	

Available Buffers in Main Memory



Toy Example

Original File:

12,10	25,20	40,30	27,29	14,18	45,23	70,65	35,11	49, 47	22,21	46,34	31

The main file can be divided into the following sublists to fit main memory to be sorted there (one sublist at a time)

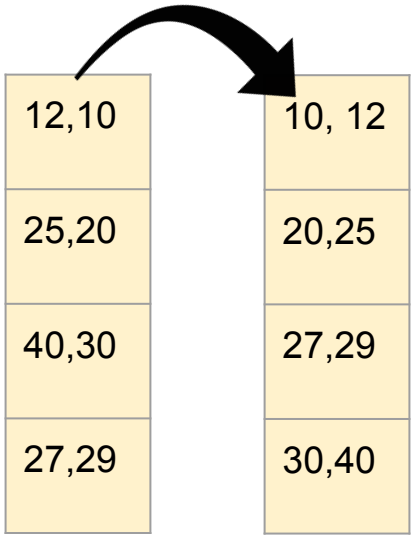
12,10	14,18	49,47
25,20	45,23	22,21
40,30	70,65	46,34
27,29	35,11	31

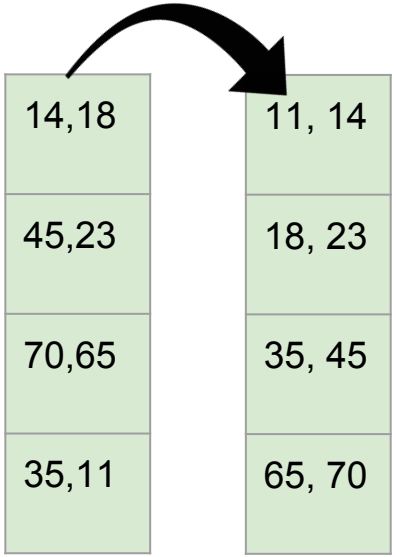
Toy Example : Phase 1

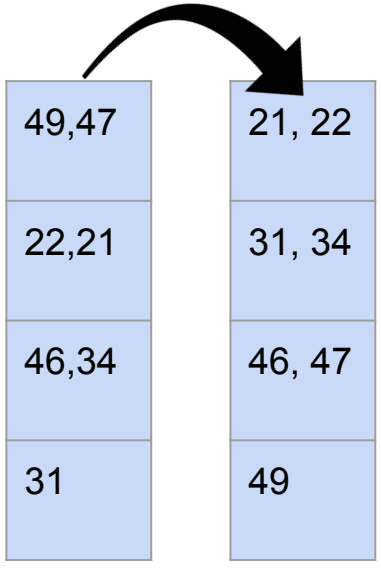
Original File on Disk:

12,10	25,20	40,30	27,29	14,18	45,23	70,65	35,11	49, 47	22,21	46,34	31

In Main Memory (one sublist at a time)







Toy Example : Phase 1

Each sublist is read (or loaded) into main memory once, sorted in the main memory and then written back to disk.

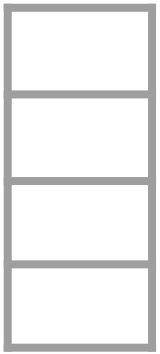
So at the end of phase 1 we are left with sorted sublists. The next phase (or phases) deal with merging these individual sorted sublists into one sorted relation/file.

Toy Example : End of Phase 1

Sorted Sublists On Disk:

	10,12	20,25	27, 29	30,40	11,14	18,23	35, 45	65,70	21,22	31,34	46,47	49

In Main Memory:



(Phase 2)

Sublist 1:	10,12	20,25	27, 29	30,40
Sublist 2:	11,14	18,23	35, 45	65,70
Sublist 3:	21,22	31,34	46,47	49

Disk

Sorted List

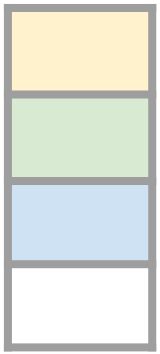
Main Memory (4 buffers)

Input Buffer1:

Input Buffer2:

Input Buffer3:

Output Buffer:



(Phase 2)

Sublist 1:	10,12	20,25	27, 29	30,40
Sublist 2:	11,14	18,23	35, 45	65,70
Sublist 3:	21,22	31,34	46,47	49

Disk

Sorted List			

Main Memory (4 buffers)

Input Buffer1:	10,12
Input Buffer2:	11,14
Input Buffer3:	21,22
Output Buffer:	

(Phase 2)

Sublist 1:	10,12	20,25	27, 29	30,40
Sublist 2:	11,14	18,23	35, 45	65,70
Sublist 3:	21,22	31,34	46,47	49

Disk

Sorted List			

Main Memory (4 buffers)

Input Buffer1:	12
Input Buffer2:	11,14
Input Buffer3:	21,22
Output Buffer:	10,

(Phase 2)

Sublist 1:	10,12	20,25	27, 29	30,40
Sublist 2:	11,14	18,23	35, 45	65,70
Sublist 3:	21,22	31,34	46,47	49

Disk

Sorted List

Main Memory (4 buffers)

Input Buffer1:	12
Input Buffer2:	14
Input Buffer3:	21,22
Output Buffer:	10,11

Output Buffer Full, need to write the block to disk!

(Phase 2)

Sublist 1:	10,12	20,25	27, 29	30,40
Sublist 2:	11,14	18,23	35, 45	65,70
Sublist 3:	21,22	31,34	46,47	49

Disk

Sorted List	10,11		

Main Memory (4 buffers)

Input Buffer1:	12
Input Buffer2:	14
Input Buffer3:	21,22
Output Buffer:	

(Phase 2)

Sublist 1:	10,12	20,25	27, 29	30,40
Sublist 2:	11,14	18,23	35, 45	65,70
Sublist 3:	21,22	31,34	46,47	49

Disk

Sorted List	10,11		

Main Memory (4 buffers)

Input Buffer1:	
Input Buffer2:	14
Input Buffer3:	21,22
Output Buffer:	12,

(Phase 2)

Sublist 1:	10,12	20,25	27, 29	30,40
Sublist 2:	11,14	18,23	35, 45	65,70
Sublist 3:	21,22	31,34	46,47	49

Disk

Sorted List	10,11		

Main Memory (4 buffers)

Input Buffer1:	20,25
Input Buffer2:	14
Input Buffer3:	21,22
Output Buffer:	12,

(Phase 2)

Sublist 1:	10,12	20,25	27, 29	30,40
Sublist 2:	11,14	18,23	35, 45	65,70
Sublist 3:	21,22	31,34	46,47	49

Disk

Sorted List	10,11		

Main Memory (4 buffers)

Input Buffer1:	20,25
Input Buffer2:	
Input Buffer3:	21,22
Output Buffer:	12,14

Output Buffer Full, need to write the block to disk!

(Phase 2)

Sublist 1:	10,12	20,25	27, 29	30,40
Sublist 2:	11,14	18,23	35, 45	65,70
Sublist 3:	21,22	31,34	46,47	49

Disk

Sorted List	10,11	12,14	

Main Memory (4 buffers)

Input Buffer1:	20,25
Input Buffer2:	18,23
Input Buffer3:	21,22
Output Buffer:	

(Phase 2)

Sublist 1:	10,12	20,25	27, 29	30,40
Sublist 2:	11,14	18,23	35, 45	65,70
Sublist 3:	21,22	31,34	46,47	49

Disk

Sorted List	10,11	12,14	

Main Memory (4 buffers)

Input Buffer1:	20,25
Input Buffer2:	23
Input Buffer3:	21,22
Output Buffer:	18,

(Phase 2)

Sublist 1:	10,12	20,25	27, 29	30,40
Sublist 2:	11,14	18,23	35, 45	65,70
Sublist 3:	21,22	31,34	46,47	49

Disk

Sorted List	10,11	12,14	

Main Memory (4 buffers)

Input Buffer1:	25
Input Buffer2:	23
Input Buffer3:	21,22
Output Buffer:	18,20

Output Buffer Full, need to write the block to disk!

(Phase 2)

Sublist 1:	10,12	20,25	27, 29	30,40
Sublist 2:	11,14	18,23	35, 45	65,70
Sublist 3:	21,22	31,34	46,47	49

Disk

Sorted List	10,11	12,14	18,20

Main Memory (4 buffers)

Input Buffer1:	25
Input Buffer2:	23
Input Buffer3:	21,22
Output Buffer:	

(Phase 2)

Sublist 1:	10,12	20,25	27, 29	30,40
Sublist 2:	11,14	18,23	35, 45	65,70
Sublist 3:	21,22	31,34	46,47	49

Disk

Sorted List	10,11	12,14	18,20		

Main Memory (4 buffers)

Input Buffer1:	25
Input Buffer2:	23
Input Buffer3:	
Output Buffer:	21,22

Output Buffer Full, need to write the block to disk!

(Phase 2)

Sublist 1:	10,12	20,25	27, 29	30,40
Sublist 2:	11,14	18,23	35, 45	65,70
Sublist 3:	21,22	31,34	46,47	49

Disk

Sorted List	10,11	12,14	18,20	21,22	

Main Memory (4 buffers)

Input Buffer1:	25
Input Buffer2:	23
Input Buffer3:	31,34
Output Buffer:	

(Phase 2)

Sublist 1:	10,12	20,25	27, 29	30,40
Sublist 2:	11,14	18,23	35, 45	65,70
Sublist 3:	21,22	31,34	46,47	49

Disk

Sorted List	10,11	12,14	18,20	21,22	

Main Memory (4 buffers)

Input Buffer1:	25
Input Buffer2:	
Input Buffer3:	31,34
Output Buffer:	23

(Phase 2)

Sublist 1:	10,12	20,25	27, 29	30,40
Sublist 2:	11,14	18,23	35, 45	65,70
Sublist 3:	21,22	31,34	46,47	49

Disk

Sorted List	10,11	12,14	18,20	21,22		

Main Memory (4 buffers)

Input Buffer1:	25
Input Buffer2:	35,45
Input Buffer3:	31,34
Output Buffer:	23

(Phase 2)

Sublist 1:	10,12	20,25	27, 29	30,40
Sublist 2:	11,14	18,23	35, 45	65,70
Sublist 3:	21,22	31,34	46,47	49

Disk

Sorted List	10,11	12,14	18,20	21,22		

Main Memory (4 buffers)

Input Buffer1:	
Input Buffer2:	35,45
Input Buffer3:	31,34
Output Buffer:	23,25

Output Buffer Full, need to write the block to disk!

(Phase 2)

Sublist 1:	10,12	20,25	27, 29	30,40
Sublist 2:	11,14	18,23	35, 45	65,70
Sublist 3:	21,22	31,34	46,47	49

Disk

Sorted List	10,11	12,14	18,20	21,22	23,25	

Main Memory (4 buffers)

Input Buffer1:	27,29
Input Buffer2:	35,45
Input Buffer3:	31,34
Output Buffer:	

(Phase 2)

Sublist 1:	10,12	20,25	27, 29	30,40
Sublist 2:	11,14	18,23	35, 45	65,70
Sublist 3:	21,22	31,34	46,47	49

Disk

Sorted List	10,11	12,14	18,20	21,22	23,25	

Main Memory (4 buffers)

Input Buffer1:	
Input Buffer2:	35,45
Input Buffer3:	31,34
Output Buffer:	27, 29

Output Buffer Full, need to write the block to disk!

(Phase 2)

Sublist 1:	10,12	20,25	27, 29	30,40
Sublist 2:	11,14	18,23	35, 45	65,70
Sublist 3:	21,22	31,34	46,47	49

Disk

Sorted List	10,11	12,14	18,20	21,22	23,25	27, 29		

Main Memory (4 buffers)

Input Buffer1:	30,40
Input Buffer2:	35,45
Input Buffer3:	31,34
Output Buffer:	

(Phase 2)

Sublist 1:	10,12	20,25	27, 29	30,40
Sublist 2:	11,14	18,23	35, 45	65,70
Sublist 3:	21,22	31,34	46,47	49

Disk

Sorted List	10,11	12,14	18,20	21,22	23,25	27, 29		

Main Memory (4 buffers)

Input Buffer1:	40
Input Buffer2:	35,45
Input Buffer3:	34
Output Buffer:	30,31

Output Buffer Full, need to write the block to disk!

(Phase 2)

Sublist 1:	10,12	20,25	27, 29	30,40
Sublist 2:	11,14	18,23	35, 45	65,70
Sublist 3:	21,22	31,34	46,47	49

Disk

Sorted List	10,11	12,14	18,20	21,22	23,25	27, 29	30,31	

Main Memory (4 buffers)

Input Buffer1:	40
Input Buffer2:	35,45
Input Buffer3:	34
Output Buffer:	

(Phase 2)

Sublist 1:	10,12	20,25	27, 29	30,40
Sublist 2:	11,14	18,23	35, 45	65,70
Sublist 3:	21,22	31,34	46,47	49

Disk

Sorted List

10,11	12,14	18,20	21,22	23,25	27, 29	30,31	

Main Memory (4 buffers)

Input Buffer1:	40
Input Buffer2:	35,45
Input Buffer3:	
Output Buffer:	34

(Phase 2)

Sublist 1:	10,12	20,25	27, 29	30,40
Sublist 2:	11,14	18,23	35, 45	65,70
Sublist 3:	21,22	31,34	46,47	49

Disk

Sorted List	10,11	12,14	18,20	21,22	23,25	27, 29	30,31	

Main Memory (4 buffers)

Input Buffer1:	40
Input Buffer2:	35,45
Input Buffer3:	46,47
Output Buffer:	34

(Phase 2)

Sublist 1:	10,12	20,25	27, 29	30,40
Sublist 2:	11,14	18,23	35, 45	65,70
Sublist 3:	21,22	31,34	46,47	49

Disk

Sorted List	10,11	12,14	18,20	21,22	23,25	27, 29	30,31	

Main Memory (4 buffers)

Input Buffer1:	40
Input Buffer2:	45
Input Buffer3:	46,47
Output Buffer:	34,35

Output Buffer Full, need to write the block to disk!

(Phase 2)

Sublist 1:	10,12	20,25	27, 29	30,40
Sublist 2:	11,14	18,23	35, 45	65,70
Sublist 3:	21,22	31,34	46,47	49

Disk

Sorted List

10,11	12,14	18,20	21,22	23,25	27, 29	30,31	34,35

Main Memory (4 buffers)

Input Buffer1:	40
Input Buffer2:	45
Input Buffer3:	46,47
Output Buffer:	

(Phase 2)

Sublist 1:	10,12	20,25	27, 29	30,40
Sublist 2:	11,14	18,23	35, 45	65,70
Sublist 3:	21,22	31,34	46,47	49

Disk

Sorted List	10,11	12,14	18,20	21,22	23,25	27, 29	30,31	34,35			

Main Memory (4 buffers)

Input Buffer1:	
Input Buffer2:	45
Input Buffer3:	46,47
Output Buffer:	40

(Phase 2)

Sublist 1:	10,12	20,25	27, 29	30,40
Sublist 2:	11,14	18,23	35, 45	65,70
Sublist 3:	21,22	31,34	46,47	49

Disk

Sorted List	10,11	12,14	18,20	21,22	23,25	27, 29	30,31	34,35			

Main Memory (4 buffers)

Input Buffer1:	
Input Buffer2:	
Input Buffer3:	46,47
Output Buffer:	40,45

Output Buffer Full, need to write the block to disk!

(Phase 2)

Sublist 1:	10,12	20,25	27, 29	30,40
Sublist 2:	11,14	18,23	35, 45	65,70
Sublist 3:	21,22	31,34	46,47	49

Disk

Sorted List	10,11	12,14	18,20	21,22	23,25	27, 29	30,31	34,35	40,45	

Main Memory (4 buffers)

Input Buffer1:	
Input Buffer2:	65,70
Input Buffer3:	46,47
Output Buffer:	

(Phase 2)

Sublist 1:	10,12	20,25	27, 29	30,40
Sublist 2:	11,14	18,23	35, 45	65,70
Sublist 3:	21,22	31,34	46,47	49

Disk

Sorted List	10,11	12,14	18,20	21,22	23,25	27, 29	30,31	34,35	40,45	

Main Memory (4 buffers)

Input Buffer1:	
Input Buffer2:	65,70
Input Buffer3:	
Output Buffer:	46,47

Output Buffer Full, need to write the block to disk!

(Phase 2)

Sublist 1:	10,12	20,25	27, 29	30,40
Sublist 2:	11,14	18,23	35, 45	65,70
Sublist 3:	21,22	31,34	46,47	49

Disk

Sorted List	10,11	12,14	18,20	21,22	23,25	27, 29	30,31	34,35	40,45	46,47	

Main Memory (4 buffers)

Input Buffer1:	
Input Buffer2:	65,70
Input Buffer3:	49
Output Buffer:	

(Phase 2)

Sublist 1:	10,12	20,25	27, 29	30,40
Sublist 2:	11,14	18,23	35, 45	65,70
Sublist 3:	21,22	31,34	46,47	49

Disk

Sorted List

10,11	12,14	18,20	21,22	23,25	27, 29	30,31	34,35	40,45	46,47	

Main Memory (4 buffers)

Input Buffer1:	
Input Buffer2:	70
Input Buffer3:	
Output Buffer:	49, 56

Output Buffer Full, need to write the block to disk!c

(Phase 2)

Sublist 1:	10,12	20,25	27, 29	30,40
Sublist 2:	11,14	18,23	35, 45	65,70
Sublist 3:	21,22	31,34	46,47	49

Disk

Sorted List	10,11	12,14	18,20	21,22	23,25	27, 29	30,31	34,35	40,45	46,47	49, 56	

Main Memory (4 buffers)

Input Buffer1:	
Input Buffer2:	
Input Buffer3:	
Output Buffer:	70

Last element, will write this to disk as well!

(Phase 2)

Sublist 1:	10,12	20,25	27, 29	30,40
Sublist 2:	11,14	18,23	35, 45	65,70
Sublist 3:	21,22	31,34	46,47	49

Disk

Sorted List	10,11	12,14	18,20	21,22	23,25	27, 29	30,31	34,35	40,45	46,47	49, 56	70

Main Memory (4 buffers)

Input Buffer1:	
Input Buffer2:	
Input Buffer3:	
Output Buffer:	

(Phase 2)

Sublist 1:	10,12	20,25	27, 29	30,40
Sublist 2:	11,14	18,23	35, 45	65,70
Sublist 3:	21,22	31,34	46,47	49

Disk

Sorted List	10,11	12,14	18,20	21,22	23,25	27, 29	30,31	34,35	40,45	46,47	49, 56	70

Main Memory (4 buffers)

Input Buffer1:

Input Buffer2:

Input Buffer3:

Output Buffer:

Sample Question

- 10,000,000 tuples of 160 bytes each = 1.6GB file.
  - Block size 16K (2^{14}) blocks, each holding 100 tuples
  - Entire file takes 100,000 blocks
  - 100MB available main memory

How many phases and how many I/Os would this need if we sort the file using Multiway Merge Sort?

Solution to Sample Question

- 10,000,000 tuples of 160 bytes each = 1.6GB file.
  - Block size 16K (2^{14}) blocks, each holding 100 tuples
  - Entire file takes 100,000 blocks

- 100MB available main memory
  - The number of blocks that can fit in 100MB of memory (which is really 100 x 2^{20} bytes), is

    100 x 2^{20} / 2^{14}, or **6400** blocks ≈1/16th of file.

  (ie we have 6400 blocks available for merge sort)

Solution : Analysis – Phase 1

- 6400 of the 100,000 blocks will fill main memory.
- We thus fill memory $\lceil 100{,}000/6{,}400 \rceil = 16$ times, sort the records in main memory, and write the sorted sublists out to disk. **At the end of phase 1 we get 16 sorted sublists**.

- How long does this phase take?
- We read each of the 100,000 blocks once, and we write 100,000 new blocks. Thus, there are 200,000 disk I/Os.

Solution : Analysis – Phase 2

- We have 16 sorted sublists (and 6400 main memory blocks so this can be merged in phase 2 very easily)
- Every block holding records from one of the sorted lists is read from disk exactly once.
  - Thus, the total number of block reads is 100,000 in the second phase, (same as for the first phase).
- Likewise, each record is placed once in an output block, and each of these blocks is written to disk.
  - Thus, the number of block writes in the second phase is also 100,000.
- We conclude that the second phase takes another 200,000 disk I/Os.

Solution: Analysis – Phase 1 + 2

- **Sorting can be done in two phases**
- **Total I/Os: Phase 1 + Phase 2 = 400,000I/Os (or 2I/O per block per phase)**

NOTE: In this example a third phase will only be needed only if the size of the data is more than 100 Mb * (6400-1) = 624.9 Gb

COST in general of Multiway Merge Sort

1 read + 1 write per phase per block per phase, ie,

2 I/Os per block per phase

How Big Should Blocks Be?

- We have assumed a 16K byte block in our analysis. However, there are arguments that a larger block size would be advantageous.

- If we doubled the size of blocks, we would halve the number of disk I/O's. But, how much a disk I/O would cost in such a case (change in transfer time)?
- Assuming it took 0.13 ms for transfer time of a 16K block and 10.63 milliseconds for average seek time and rotational latency.

- Now, the only change in the time to access a block would be that the transfer time increases to **0.13*2=0.26** ms, i.e. only slightly more than before.
- **We would thus approximately halve the time the sort takes.**

Reasons to limit the block size

1. First, we cannot use blocks that cover several tracks effectively.
2. Second, small relations would occupy only a fraction of a block, so large blocks would waste space on the disk.
3. Third, the larger the blocks are, the fewer records we can sort by 2PMMS (see next slide).

(Nevertheless, as machines get more memory and disks more capacious, there is a tendency for block sizes to grow.)

How many records can we sort with 2PMMS?

1. Block size = B bytes.
2. Main memory = M bytes (available for buffering)
3. Record size = R bytes.
4. Number of records = n (Therefore file/relation size = n*R)

- Number of main memory buffers = M/B blocks
- Number of sublists = (M/B)-1 (= input buffers for 2 phase sorting).
- Size of the sublist = M

File/Relation size = *Size of the sublist* X *Number of the sublists*

OR

n*R = (M/R)*[(M/B)-1]

- Hence, n = (M/R)*[(M/B)-1] or ~ M^2/RB.

(Alternatively you can think each time we fill in the memory with M/R records and we do it (M/B)-1 times)

If we use the parameters in the example about 2PMMS we have:
M=100MB = 100,000,000 Bytes = 10^8 Bytes
B = 16,384 Bytes
R = 160 Bytes
So, $M^2/RB = (10^8)^2 / (160 * 16{,}384)$ = 4.2 billion records.

Sorting Example

Setup:

- 10^7 records of 100 bytes = 10^9 bytes file.
  - Stored on disk 1, with 4Kb blocks, each holding 40 records + header information.
  - Entire file takes 250,000 blocks ≈ 1000 cylinders.
- 50Mb available main memory = 12,800 blocks ≈ 1/20th of file.
- Sort by primary key field.

M= 50MB

B = 4KB

R = 100 Bytes

$M^2/RB = (50*2^{20})^2 / (100 * 4*2^{10}) \sim 6.7$ billion records

ie. we can increase the number of records to 6.7 billion and still sort in 2 phases.

Table of units of measurement

Name	Equivalent	Number of Bytes	power of 2
byte	8 bits	1	2^0
Kilobyte (KB)	1024 bytes	1024	2^{10}
Megabyte (MB)	1024 KB	1,048,576	2^{20}
Gigabyte (GB)	1024 MB	1,073,741,824	2^{30}
Terabyte	1024 GB	1,099,511,627,776	2^{40}