External Sort

Database Systems: The Complete Book
Ch. 15.4

Operator		Memory Needed
π	Project	O(1)
σ	Select	O(1)
$\cup$	Bag Union	O(1)
$\bowtie$	Join	O(1) or O(|R|+|S|)
γ_L	Group	O(|G|)
δ	Distinct	O(|R|)
τ	Sort	O(|R|)

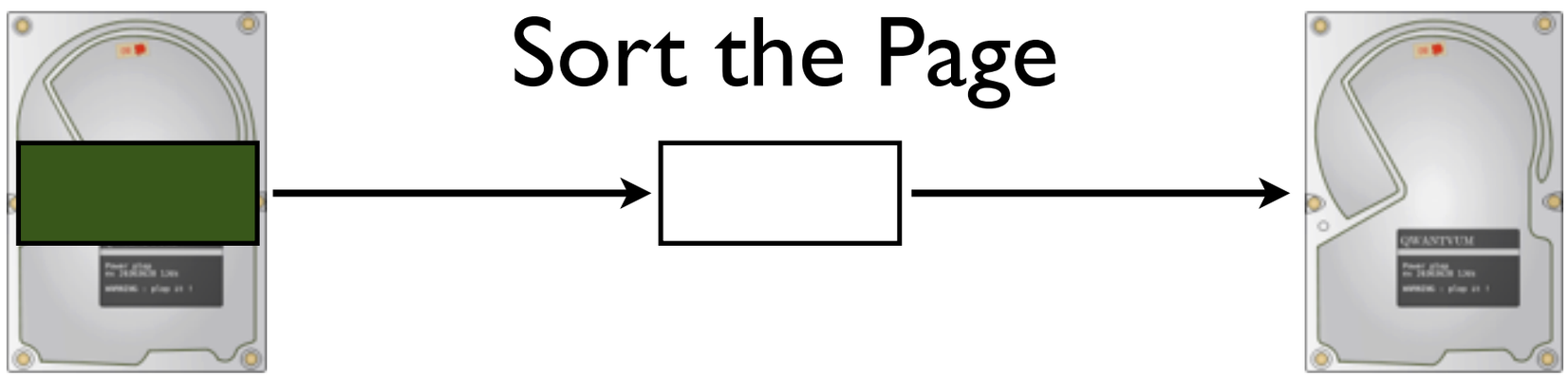
You can get away with almost no disk-based algorithms…

… as long as you have external sort.

2-Way Sort

Pass 1

Load a Page



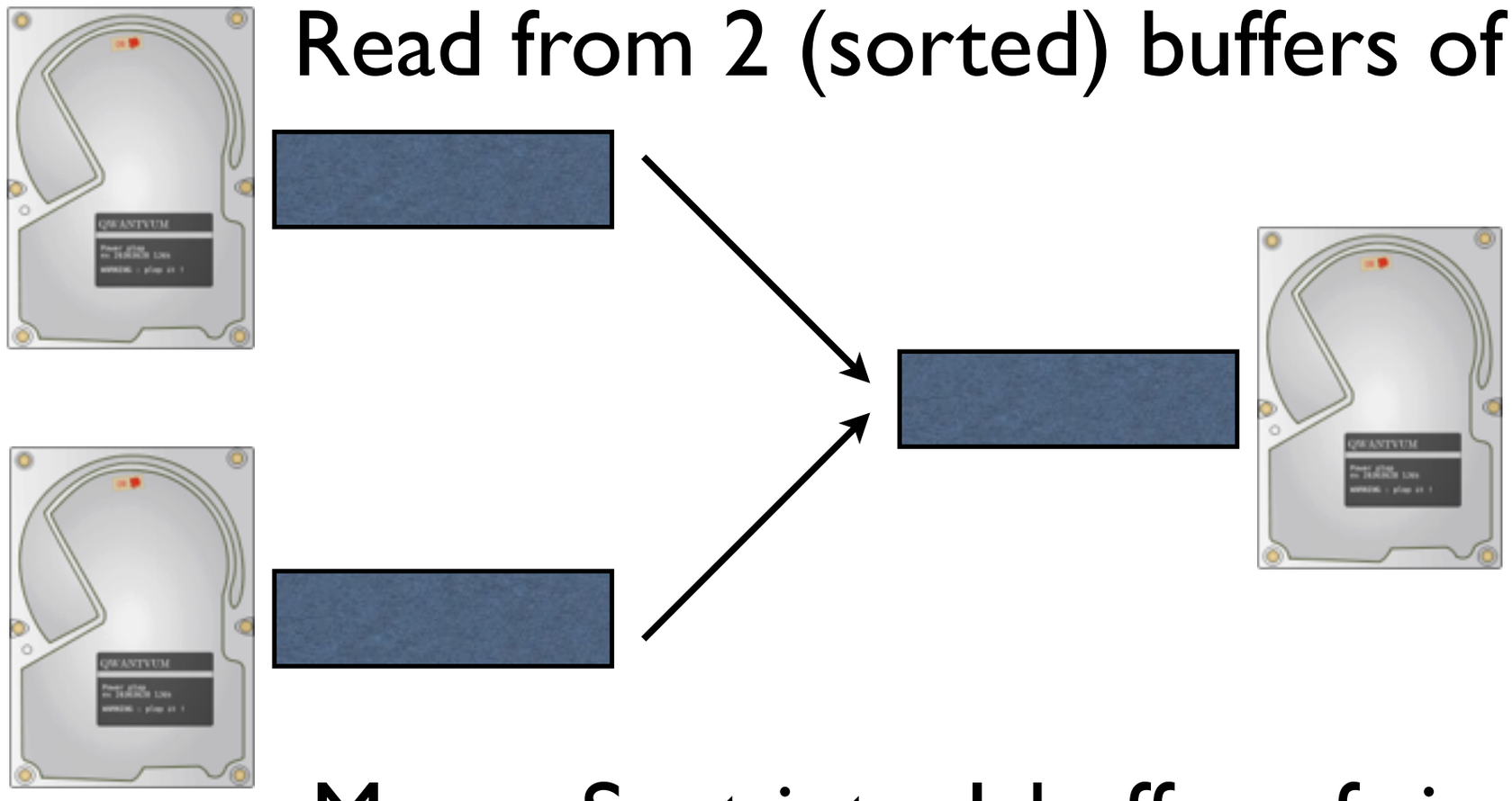
Flush the Page

image credit: openclipart.org

2-Way Sort

Pass 2 and beyond

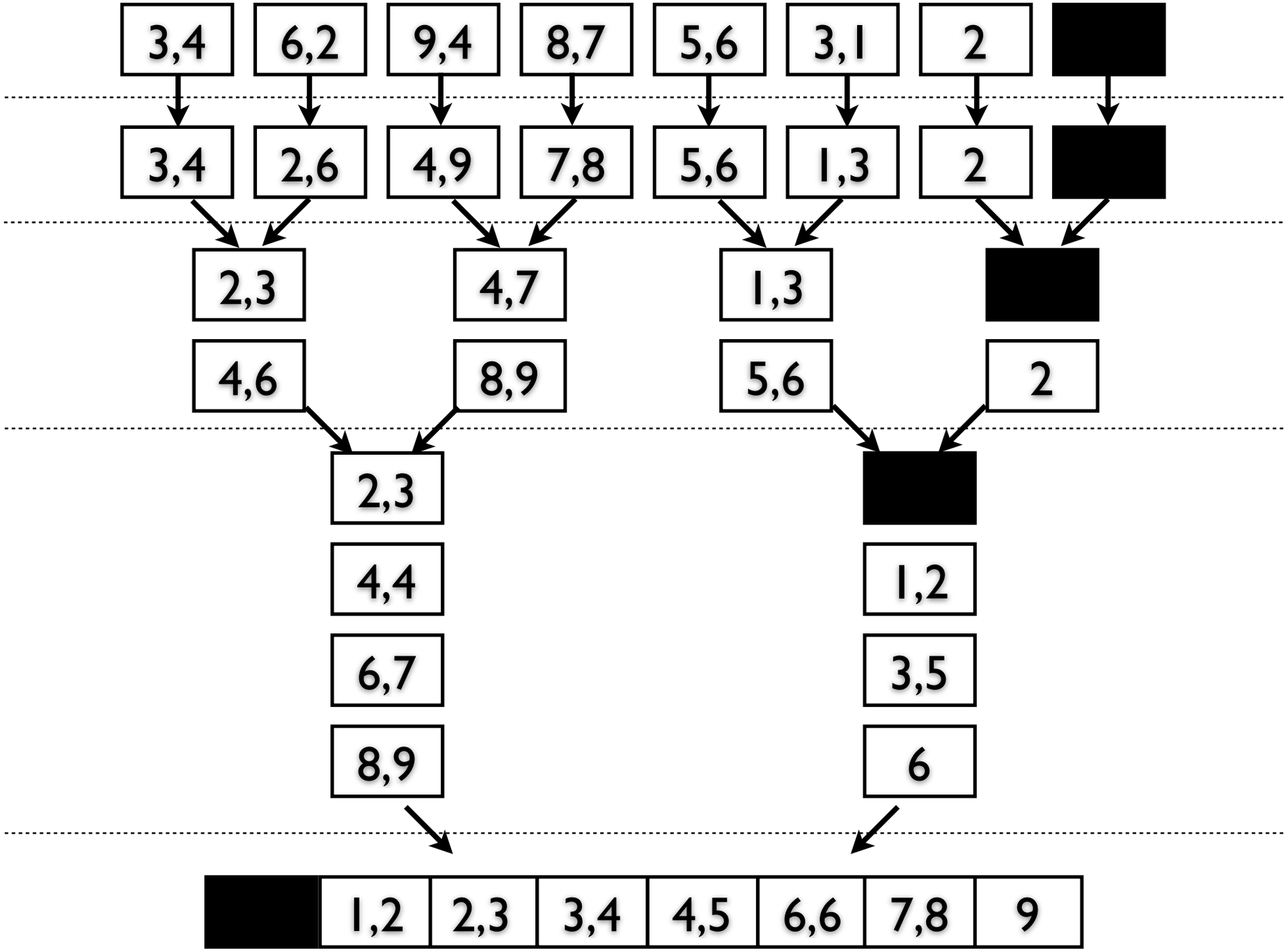
Read from 2 (sorted) buffers of size K



Merge Sort into 1 buffer of size 2K

Repeat (how many times?)

2-Way Sort


3,4
6,2
9,4
8,7
5,6
3,1
2
3,4
2,6
4,9
7,8
5,6
1,3
2
2,3
4,7
1,3
4,6
8,9
5,6
2
2,3
4,4
6,7
8,9
1,2
3,5
6
1,2
2,3
3,4
4,5
6,6
7,8
9

Generalized External Sort

How can we use N buffer frames?

For Pass 1? Sort Bigger Initial Buffers

For Pass 2 onwards? Merge-sort Multiple Streams

How many passes do we make over the full data?

For data of size N, a K-way sort requires $\lceil \log_K(N) \rceil + 1$ passes

How many IOs do we use?

$2 \cdot \#\texttt{pages} \cdot \#\texttt{passes}$

Pass 1 is memory-limited

If we have N pages of memory,
can we create more than N pages of sorted data?

Replacement Sort

- General idea: Create “runs” of sorted data
- Keep a very large “working set” of data.
- Keep appending data in ascending order to an output buffer.
- As you flush sorted data to the output, keep loading new tuples into the working set.
  - If you get new tuples useful for the current buffer, great!
  - Otherwise, they’ll go into the next run
- When you run out of valid tuples to append, start a new run!

Replacement Sort

Repeat until k is bigger than all values in the working set

Step 3: Insert a tuple from the input buffer into the working set and re-sort the working set

Finish the "run" and start a new one

Input Buffer

12

Working Set

2

8

10

...

k=8 k=5

5 3

Output Buffer

Replacement Sort

E[k] = avg(k)

On average, half of the tuples you read in will be useful for the current stream.

If you have N pages of memory, how many pages of sorted data will you make?

How do we use sorted data to implement other memory-bound operators?

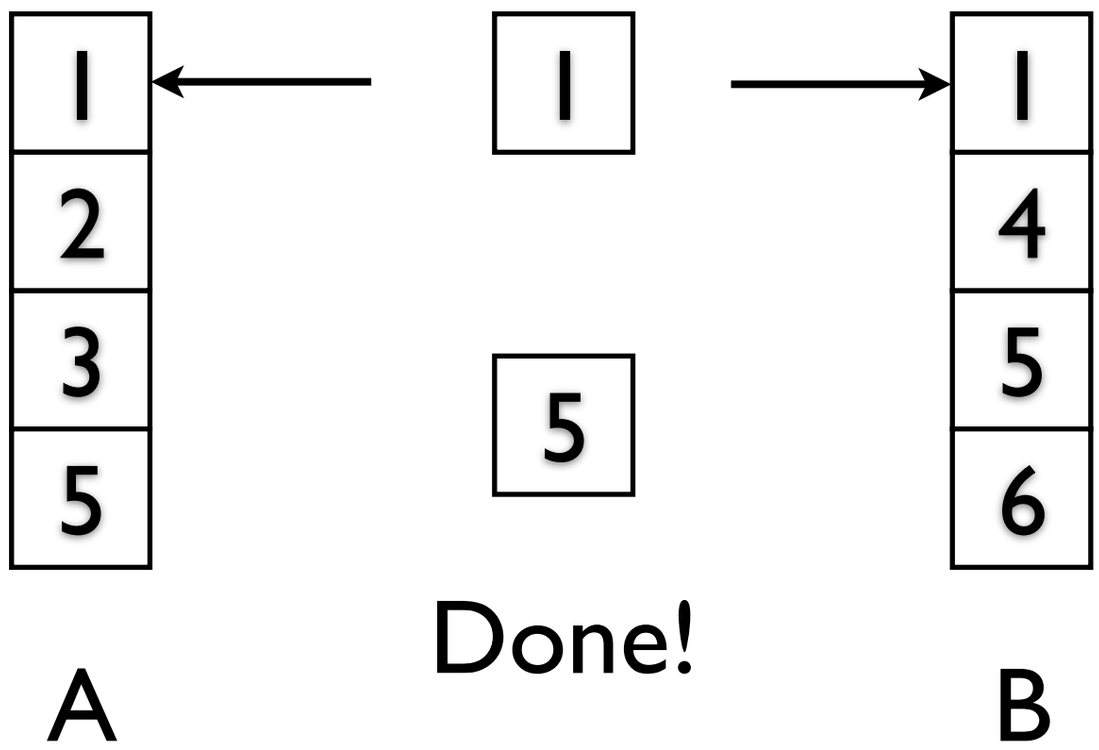
Joins

Implementing: Joins

Solution 3 (Sort-Merge Join)

Keep iterating on the set with the lowest value.

When you hit two that match, emit, then iterate both

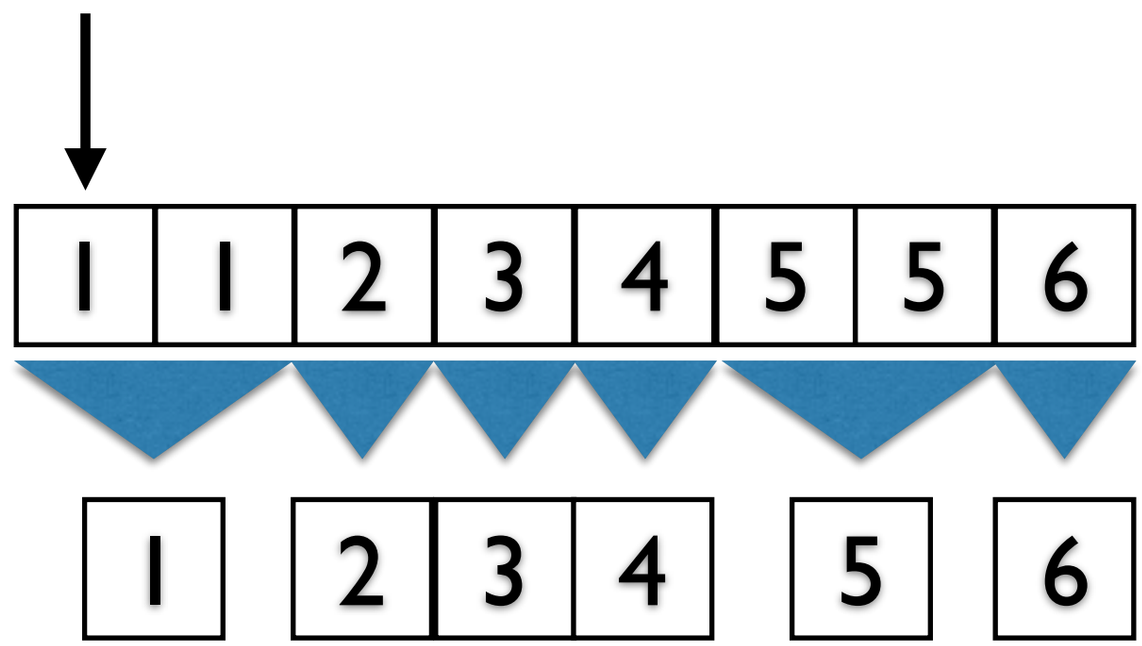


Distinct

Sort-By Distinct

1	2	4	5	1	6	3	5

Sort-By Distinct



Group-By

Sort-By Grouping

1,1	2,2	4,3	5,4	1,5	6,7	3,8	5,9

Sort-By Grouping