Example: Two-Way External Merge Sort

Input file

page

(1) data bigger than main memory

how to sort?

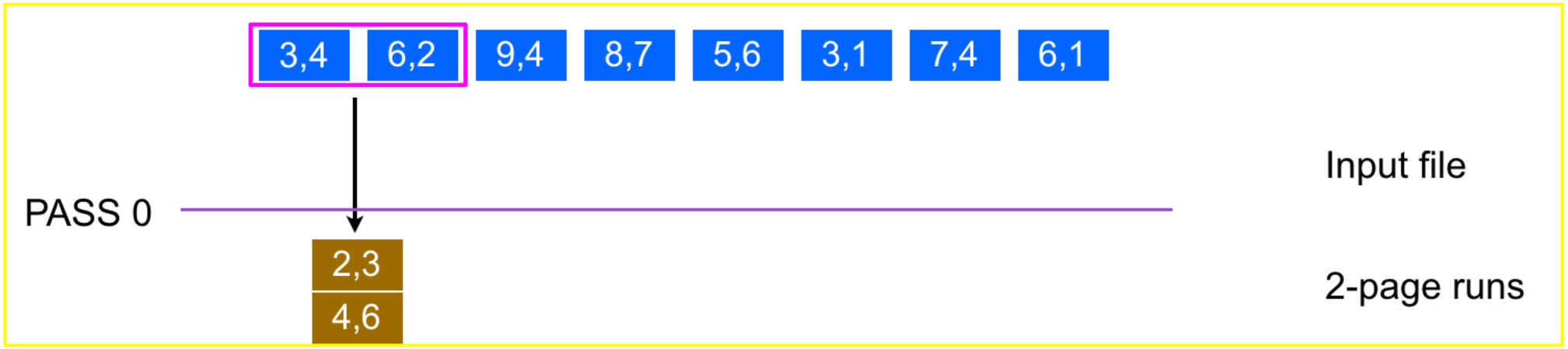
(2) Optimizations

(3) Transition to bigger query plans

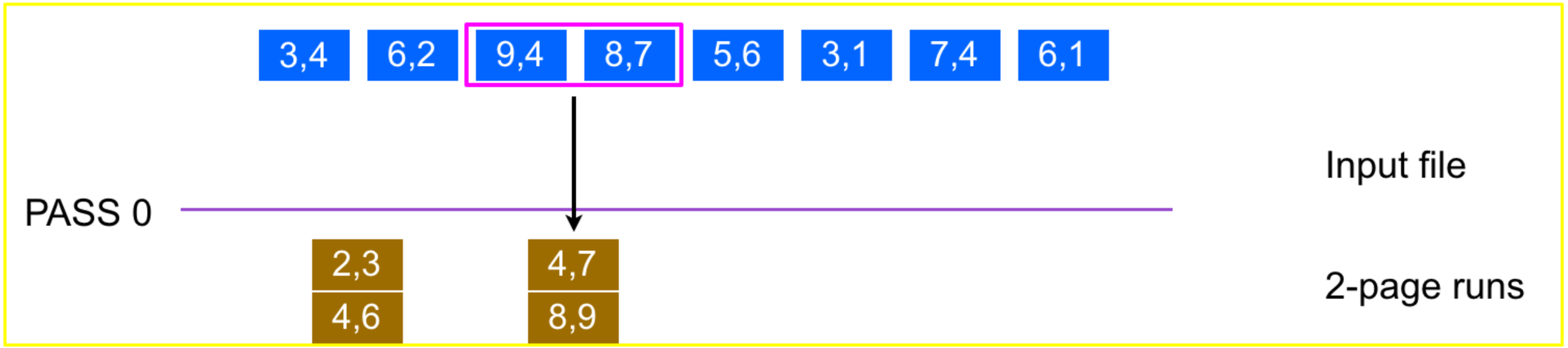
EM 5

single algo ⟶ multiple algos

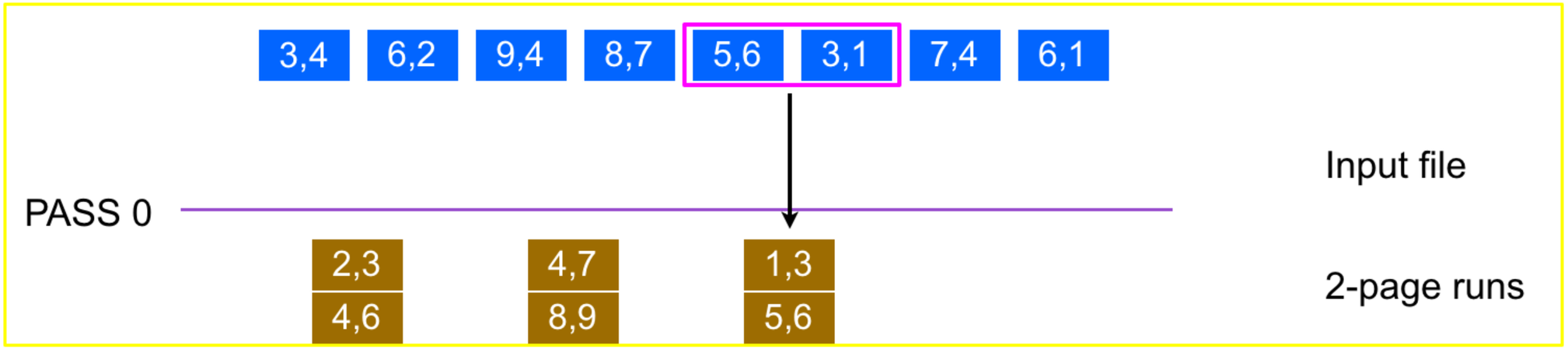
Run Generation: First Run



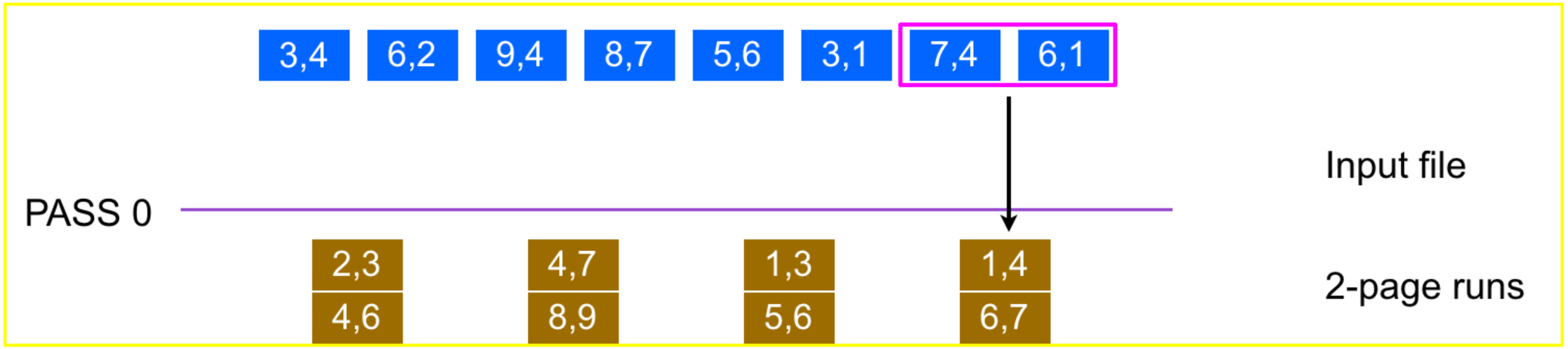
Run Generation: Second Run



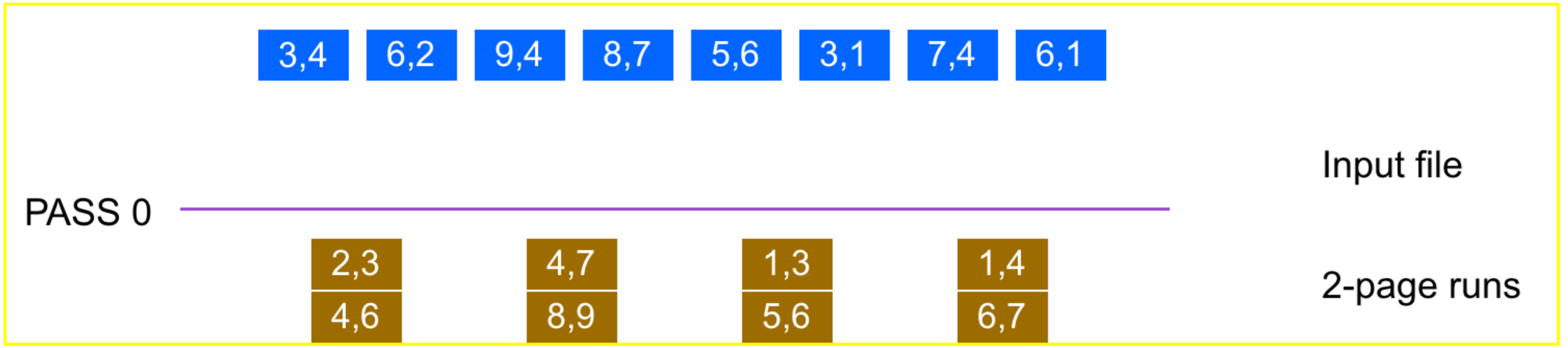
Run Generation: Third Run



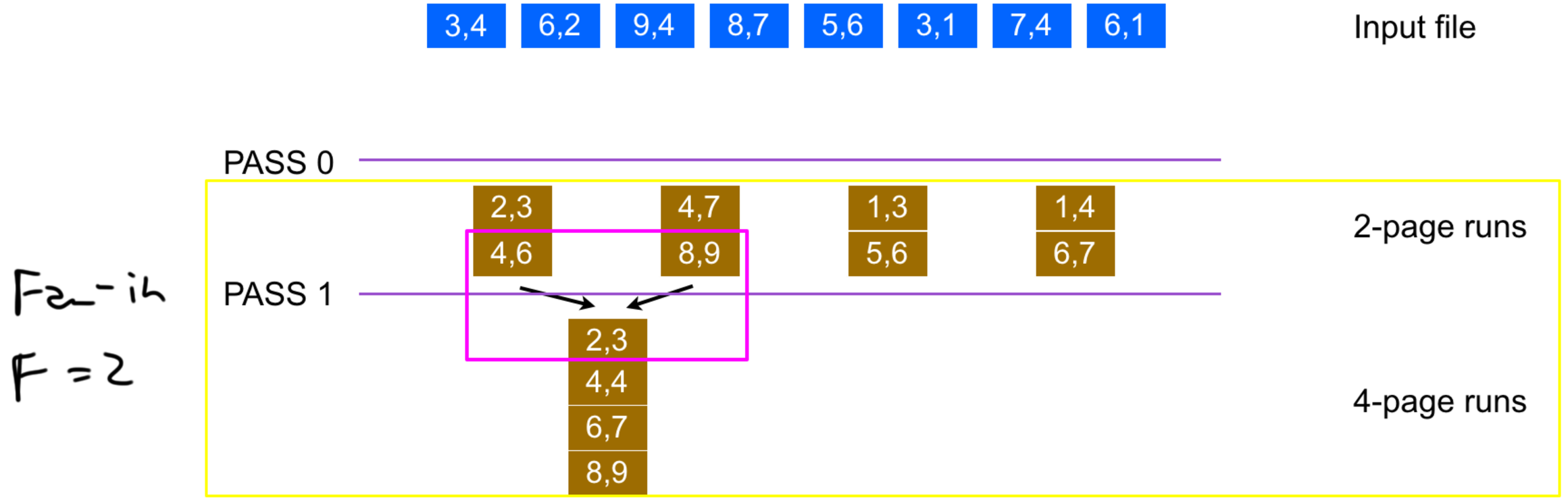
Run Generation: Fourth Run



Run Generation Done.



PASS 1: First Merge



For-in

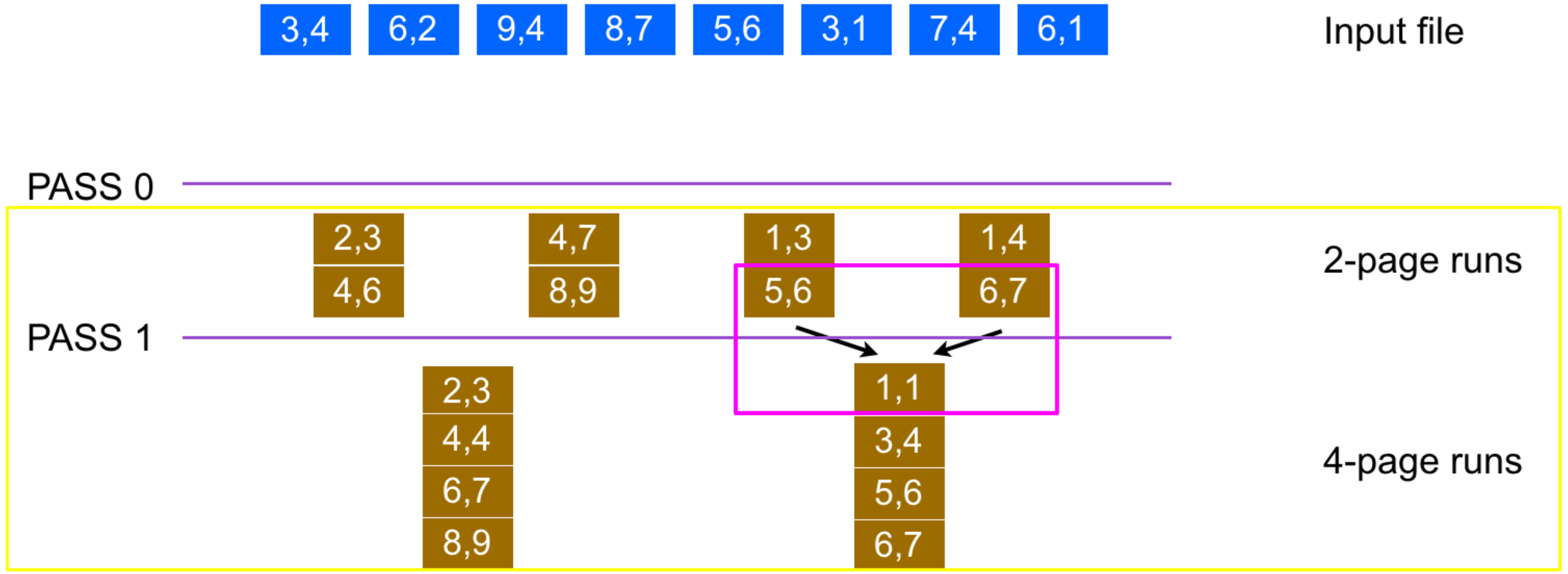
F = 2

m = # hm pages

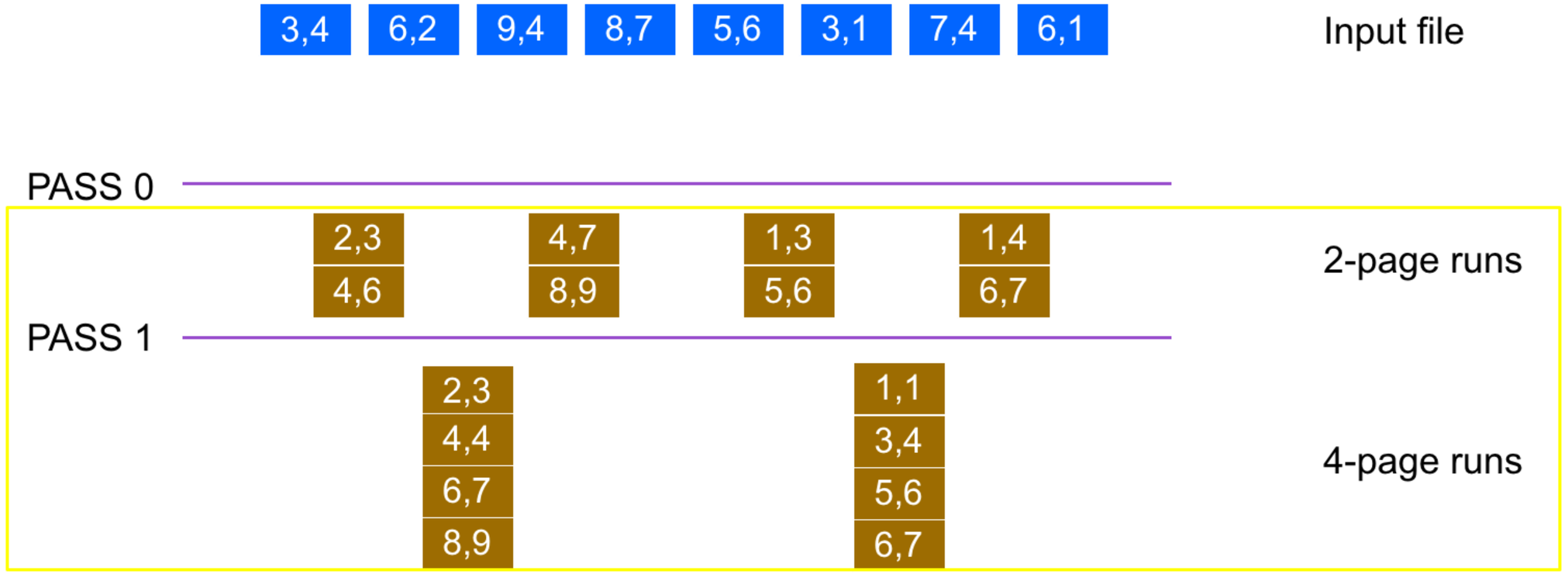
F = m - 1

F ∈ [2 : m-1]

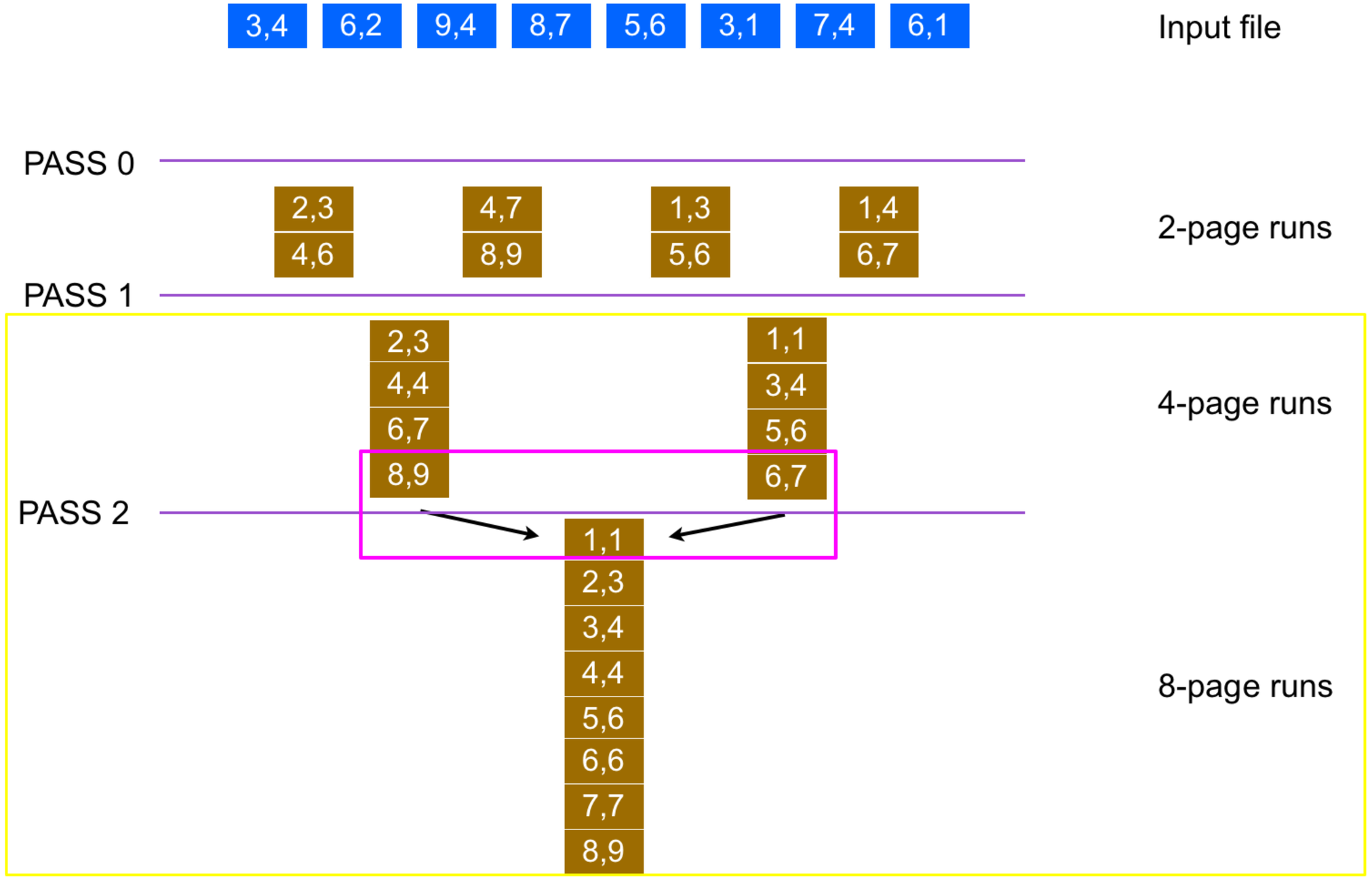
PASS 1: Second Merge



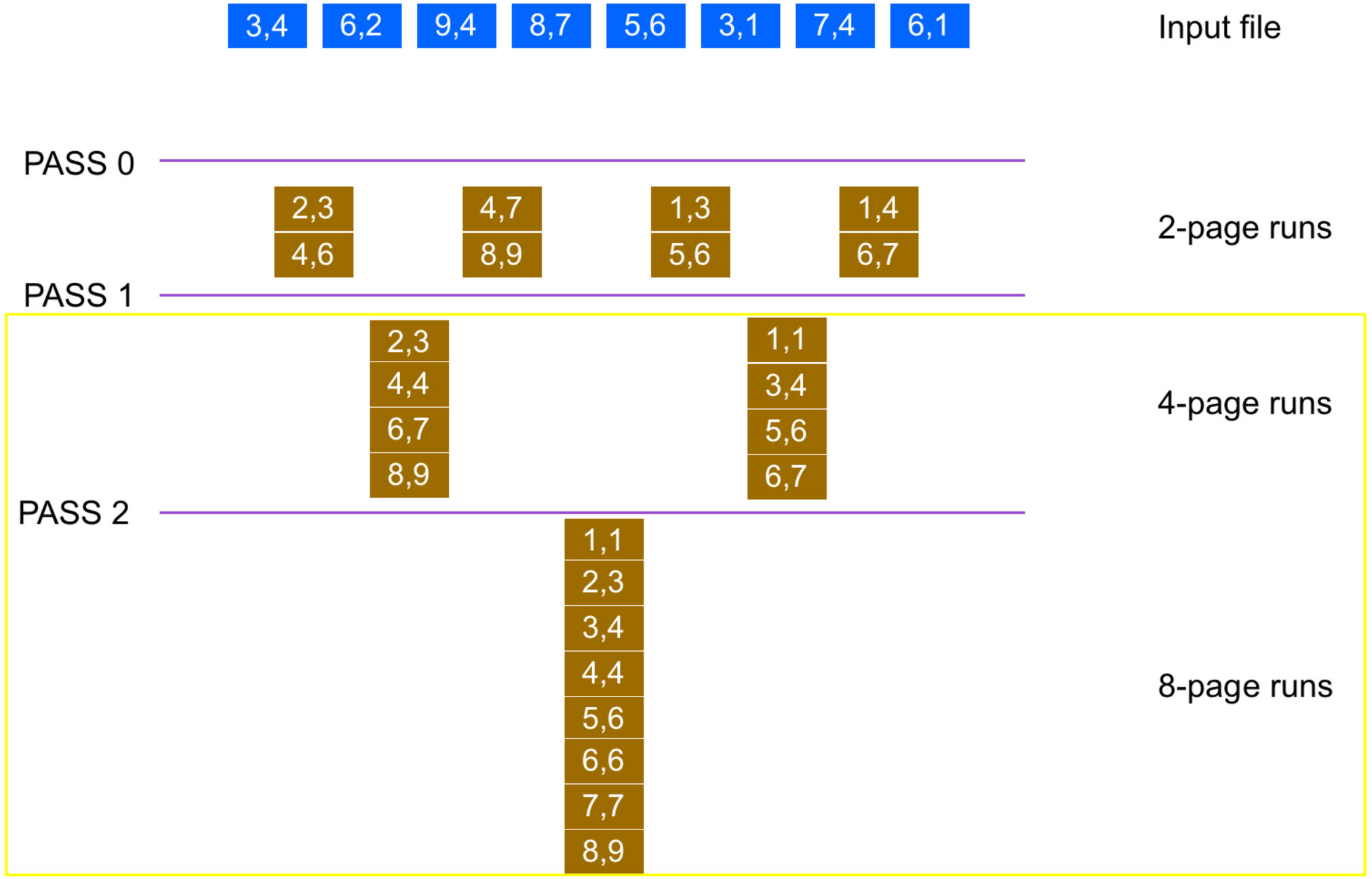
PASS 1 Done.


3,4
6,2
9,4
8,7
5,6
3,1
7,4
6,1
Input file
PASS 0
2,3
4,6
4,7
8,9
1,3
5,6
1,4
6,7
2-page runs
PASS 1
2,3
4,4
6,7
8,9
1,1
3,4
5,6
6,7
4-page runs

PASS 2: First (and only) Merge



PASS 2: First (and only) Merge Done.



External Merge Sort

F := fan-in of the merge-phase
m := # pages available for a run (typically = available main memory)

ExternalSorting(R):

```
Heap<Run> runs;                                  //heap of runs to consider
While R not empty:                               //Pass 0:
    Run run = runGenerate( R, m );                   //read m pages of input from R and sort
    runs.add( run );                                 //add reference to this run to heap
```

size of the run

small -> highest priority

External Merge Sort

F := fan-in of the merge-phase
m := # pages available for a run (typically = available main memory)

ExternalSorting(R):

```
Heap<Run> runs;                          //heap of runs to consider
While R not empty:                       //Pass 0:
    Run run = runGenerate( R, m );           //read m pages of input from R and sort
    runs.add( run );                         //add reference to this run to heap
While runs.size() > 1:                   //Passes 1 and following:
    List<Run> inputs;                        //list of inputs to merge (in a single merge)
    inputs = runs.popK( F );                 //remove next F inputs from the heap
    Run run = mergeRuns( inputs );           //merge runs into one output run
    runs.add( run );                         //add reference to merged run to runs
```

Very simple version of it!