External Sorting

- Adapt fastest internal-sort methods.

✓ Quick sort …best average run time.

- Merge sort … best worst-case run time.

Internal Merge Sort Review

- Phase 1
  - Create initial sorted segments
    - Natural segments
    - Insertion sort
- Phase 2
  - Merge pairs of sorted segments, in merge passes, until only 1 segment remains.

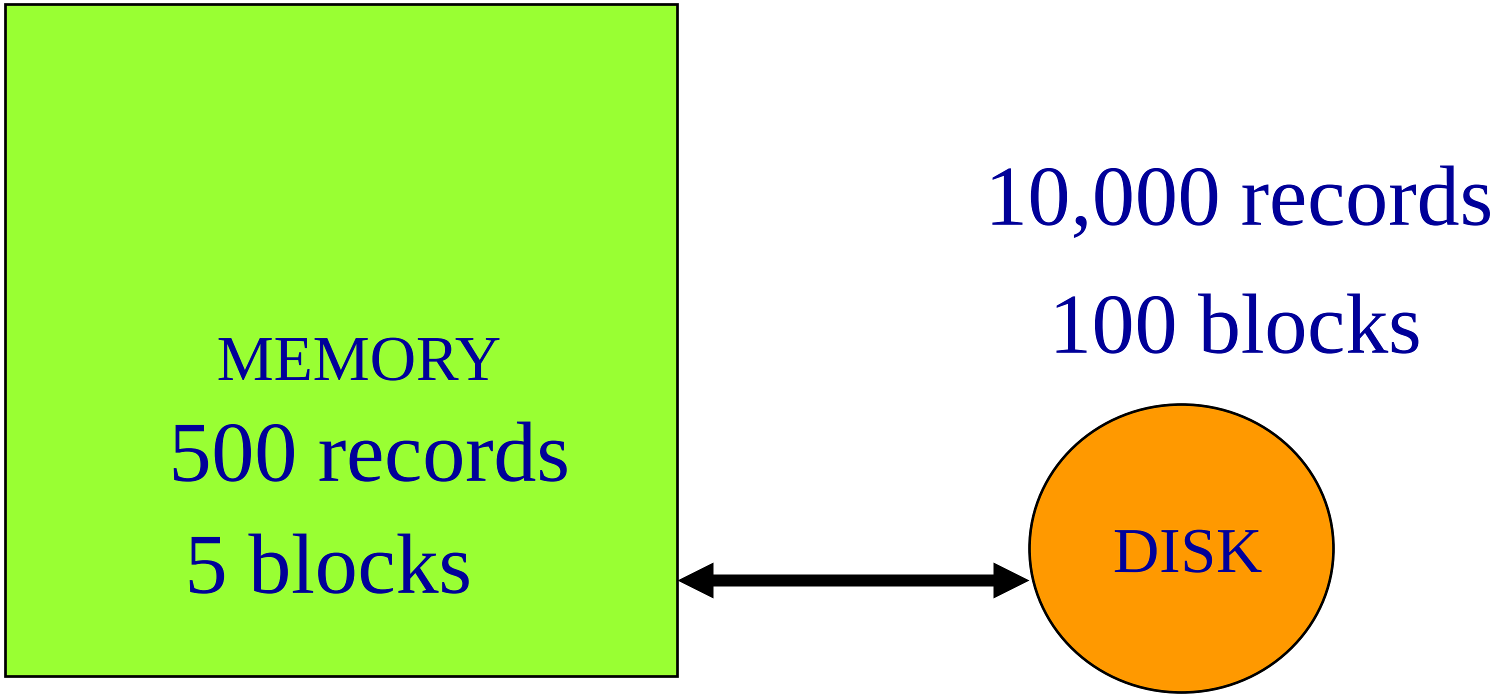
External Merge Sort

- Sort 10,000 records.
- Enough memory for 500 records.
- Block size is 100 records.
- t_{IO} = time to input/output 1 block (includes seek, latency, and transmission times)
- t_{IS} = time to internally sort 1 memory load
- t_{IM} = time to internally merge 1 block load

External Merge Sort

- Two phases.
  - Run generation.
    - A run is a sorted sequence of records.
  - Run merging.

Run Generation



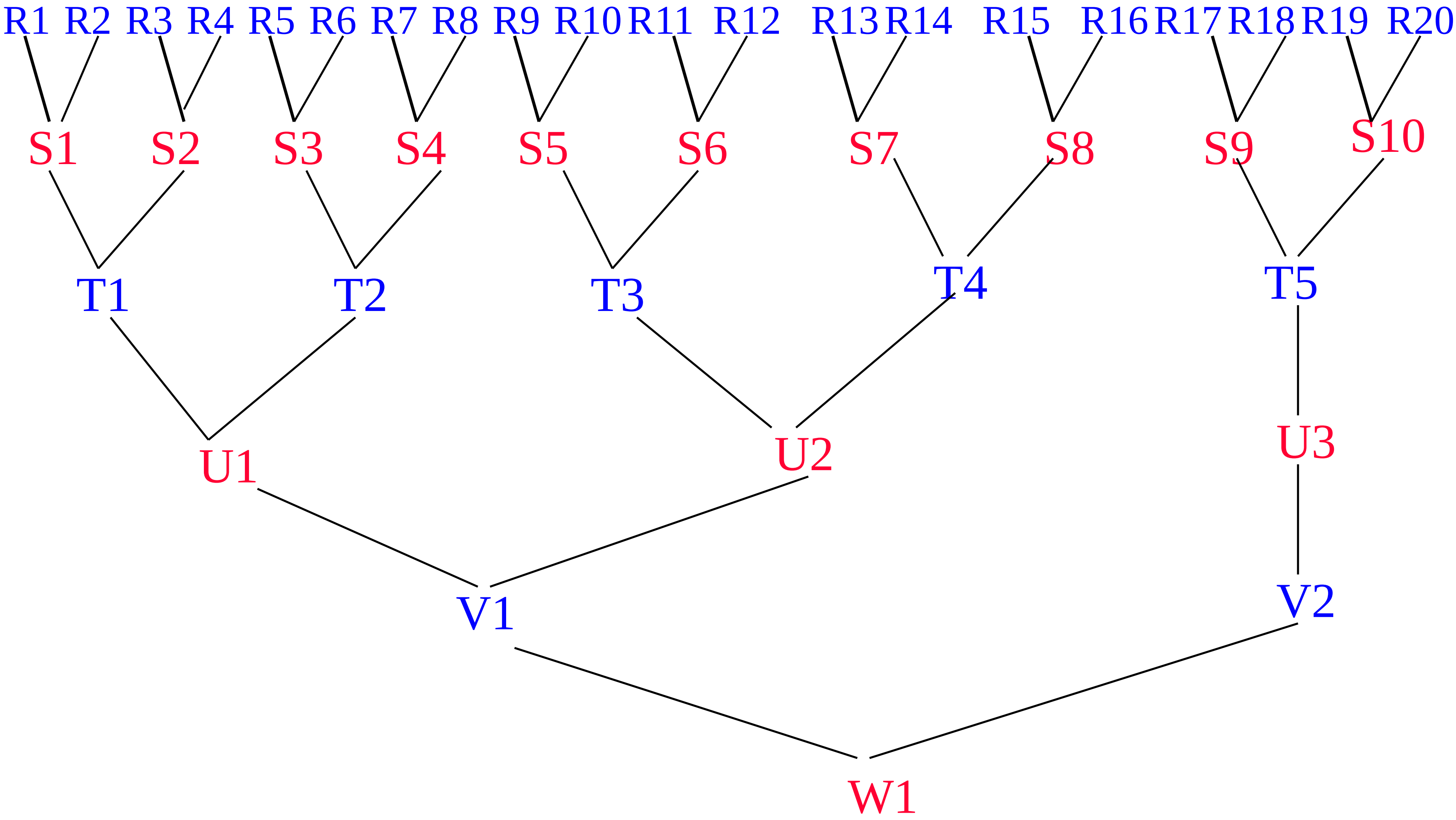
- Input 5 blocks.
- Sort.
- Output as a run.
- Do 20 times.

- $5t_{IO}$
- t_{IS}
- $5t_{IO}$
- $200t_{IO} + 20t_{IS}$

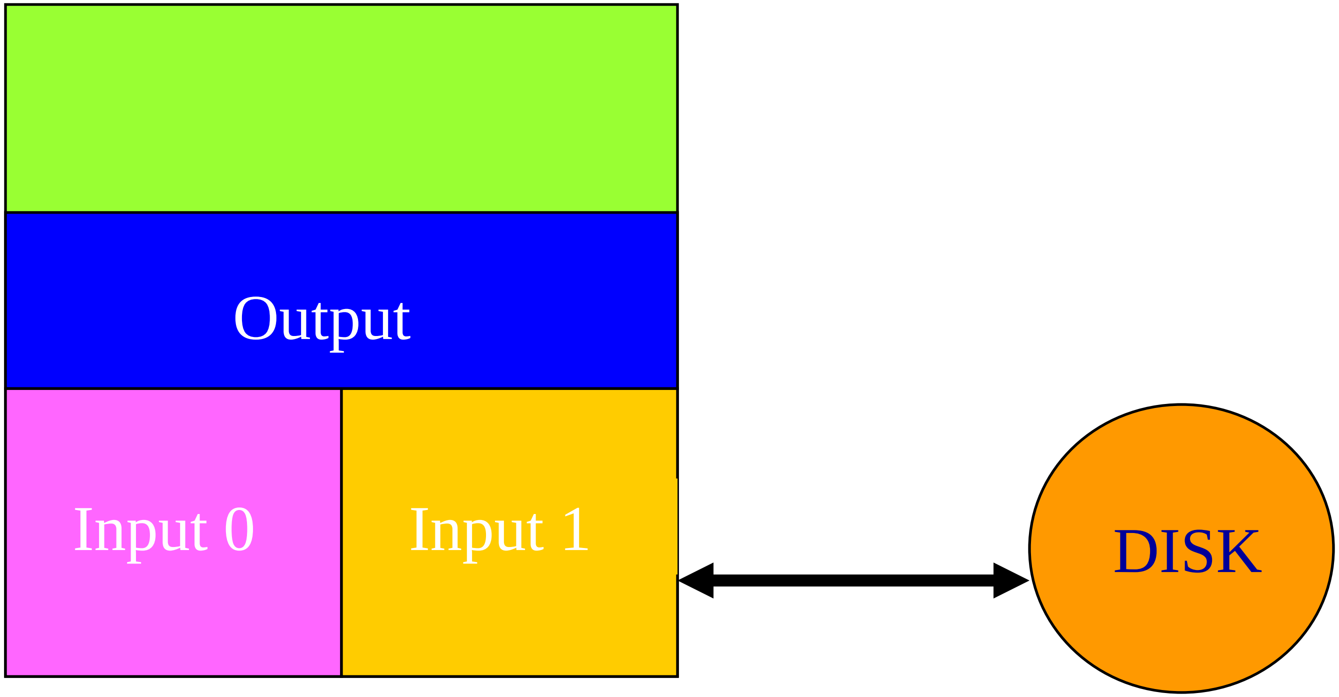
Run Merging

- Merge Pass.
  - Pairwise merge the 20 runs into 10.
  - In a merge pass all runs (except possibly one) are pairwise merged.
- Perform 4 more merge passes, reducing the number of runs to 1.

Merge 20 Runs



Merge R1 and R2



- Fill I0 (Input 0) from R1 and I1 from R2.
- Merge from I0 and I1 to output buffer.
- Write whenever output buffer full.
- Read whenever input buffer empty.

Time To Merge R1 and R2

- Each is 5 blocks long.
- Input time = $10t_{IO}$.
- Write/output time = $10t_{IO}$.
- Merge time = $10t_{IM}$.
- Total time = $20t_{IO} + 10t_{IM}$.

Time For Pass 1 (R → S)

- Time to merge one pair of runs
  $= 20t_{IO} + 10t_{IM}$.
- Time to merge all 10 pairs of runs
  $= 200t_{IO} + 100t_{IM}$.

Time To Merge S1 and S2

- Each is 10 blocks long.
- Input time = $20t_{IO}$.
- Write/output time = $20t_{IO}$.
- Merge time = $20t_{IM}$.
- Total time = $40t_{IO} + 20t_{IM}$.

Time For Pass 2 (S → T)

- Time to merge one pair of runs $= 40t_{IO} + 20t_{IM}$.
- Time to merge all 5 pairs of runs $= 200t_{IO} + 100t_{IM}$.

Time For One Merge Pass

- Time to input all blocks $= 100t_{IO}$.
- Time to output all blocks $= 100t_{IO}$.
- Time to merge all blocks $= 100t_{IM}$.
- Total time for a merge pass $= 200t_{IO} + 100t_{IM}$.

Total Run-Merging Time

- (time for one merge pass) * (number of passes)

  = (time for one merge pass)

  * ceil($\log_2$(number of initial runs))

  = ($200t_{IO} + 100t_{IM}$) * ceil($\log_2(20)$)

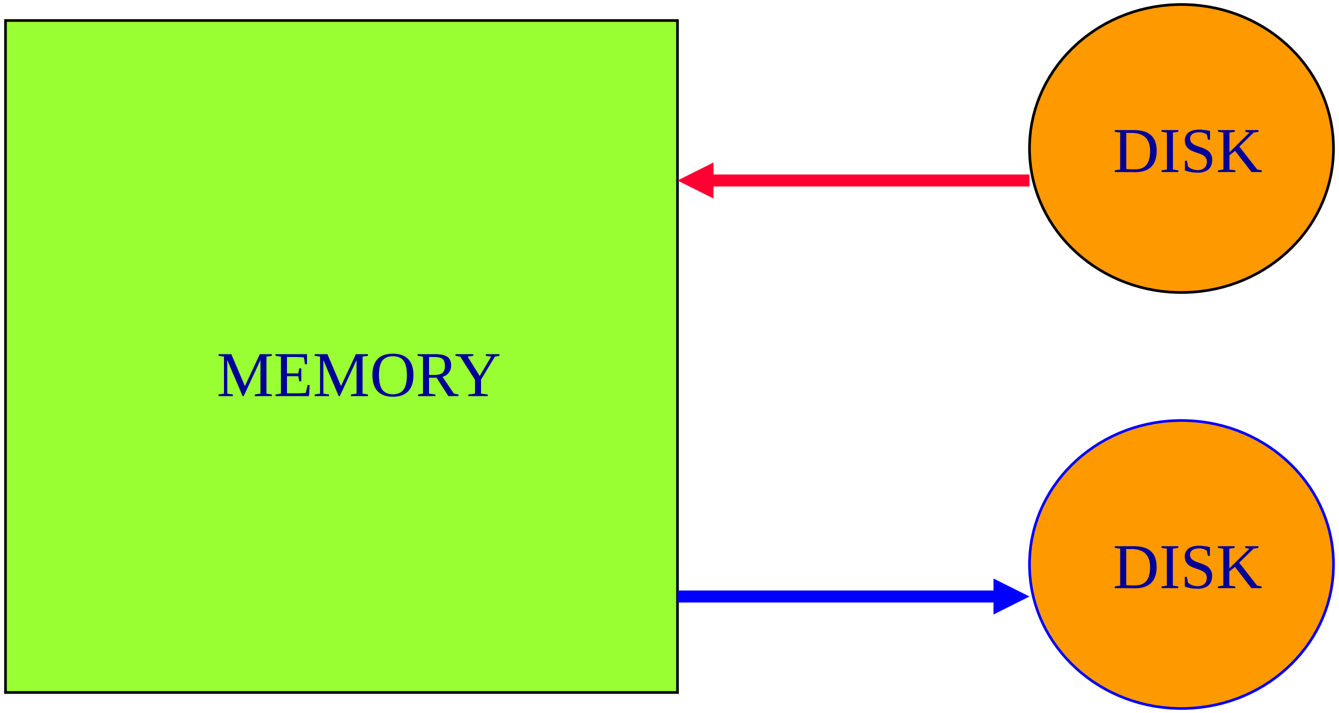
  = ($200t_{IO} + 100t_{IM}$) * 5

Factors In Overall Run Time

- Run generation. $200t_{IO} + 20t_{IS}$
  - Internal sort time.
  - Input and output time.
- Run merging. $(200t_{IO} + 100t_{IM}) * \text{ceil}(\log_2(20))$
  - Internal merge time.
  - Input and output time.
  - Number of initial runs.
  - Merge order (number of merge passes is determined by number of runs and merge order)

Improve Run Generation

- Overlap input, output, and internal sorting.

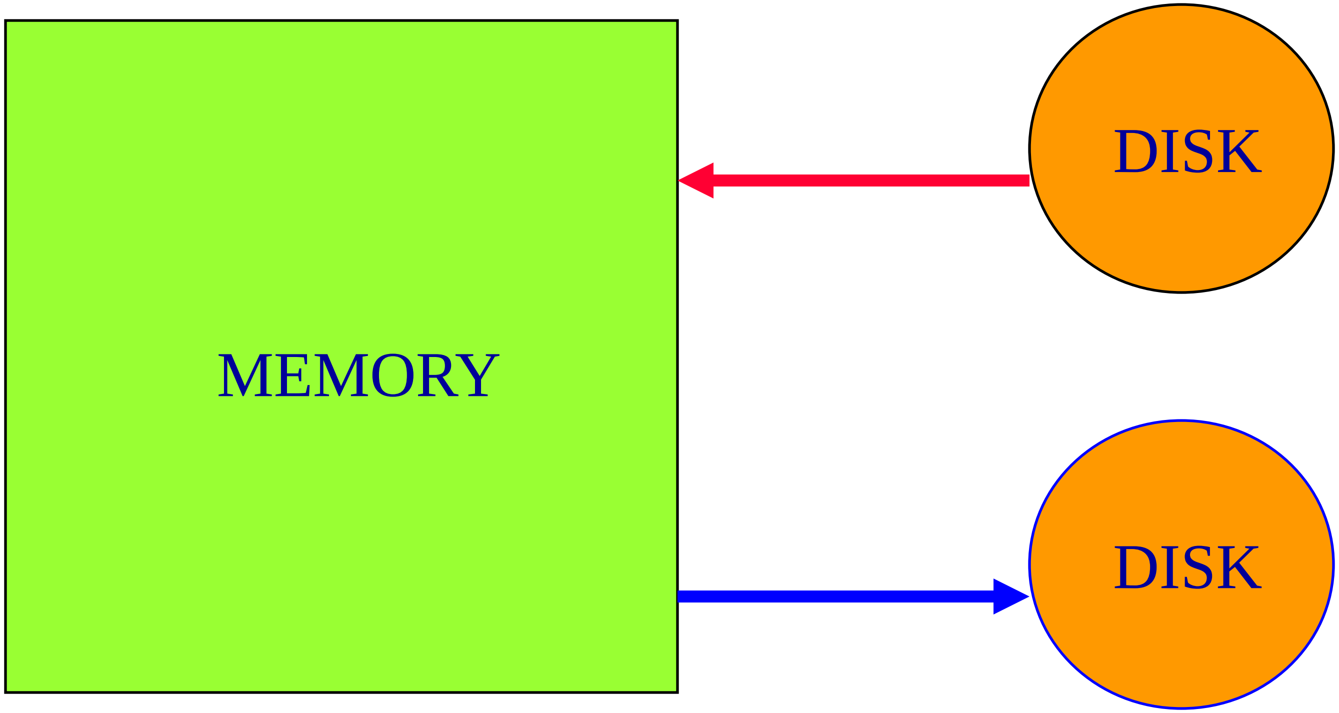


Improve Run Generation

- Generate runs whose length (on average) exceeds memory size.
- Equivalent to reducing number of runs generated.

Improve Run Merging

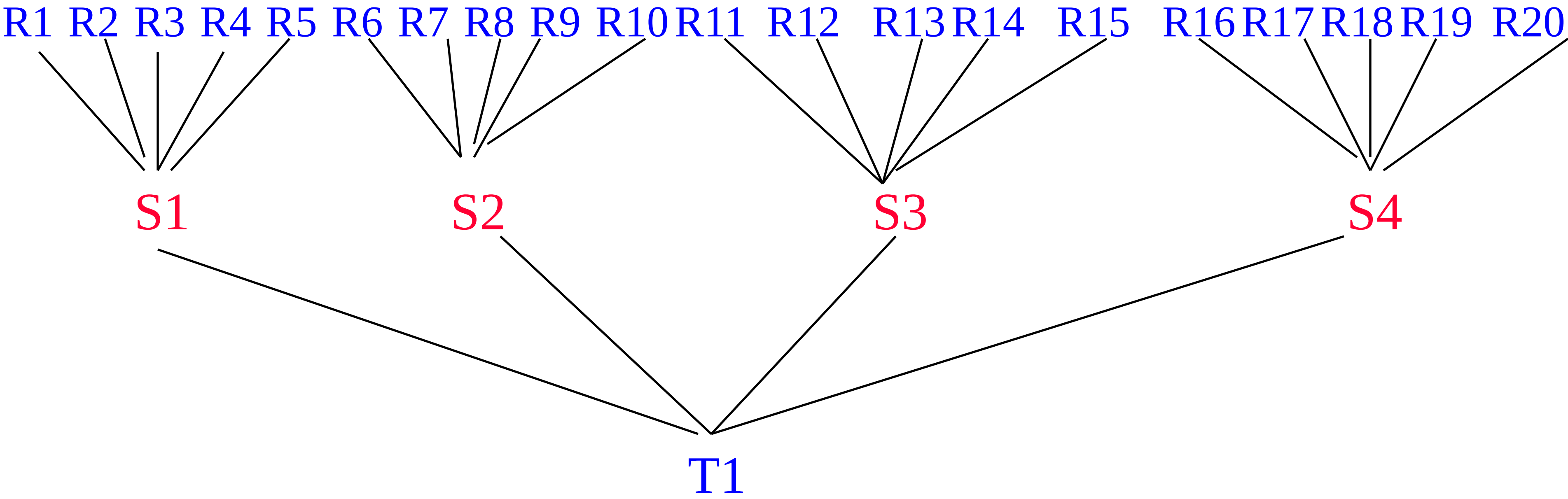
- Overlap input, output, and internal merging.



Improve Run Merging

- Reduce number of merge passes.
  - Use higher-order merge.
  - Number of passes
    $= \text{ceil}(\log_k(\text{number of initial runs}))$
    where k is the merge order.

Merge 20 Runs Using 5-Way Merging



R1 R2 R3 R4 R5 R6 R7 R8 R9 R10 R11 R12 R13 R14 R15 R16 R17 R18 R19 R20

S1 S2 S3 S4

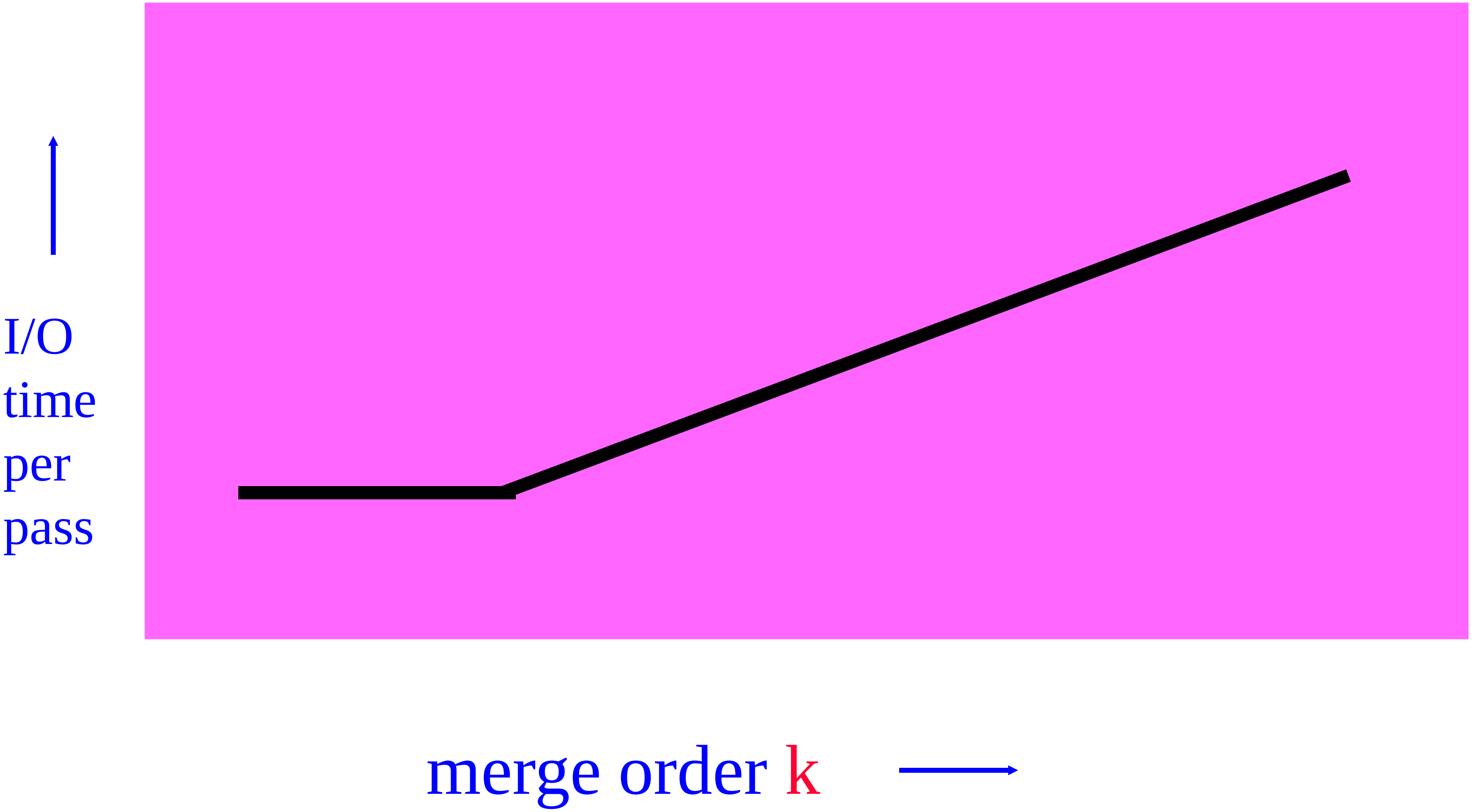
T1

Number of passes = 2

I/O Time Per Merge Pass

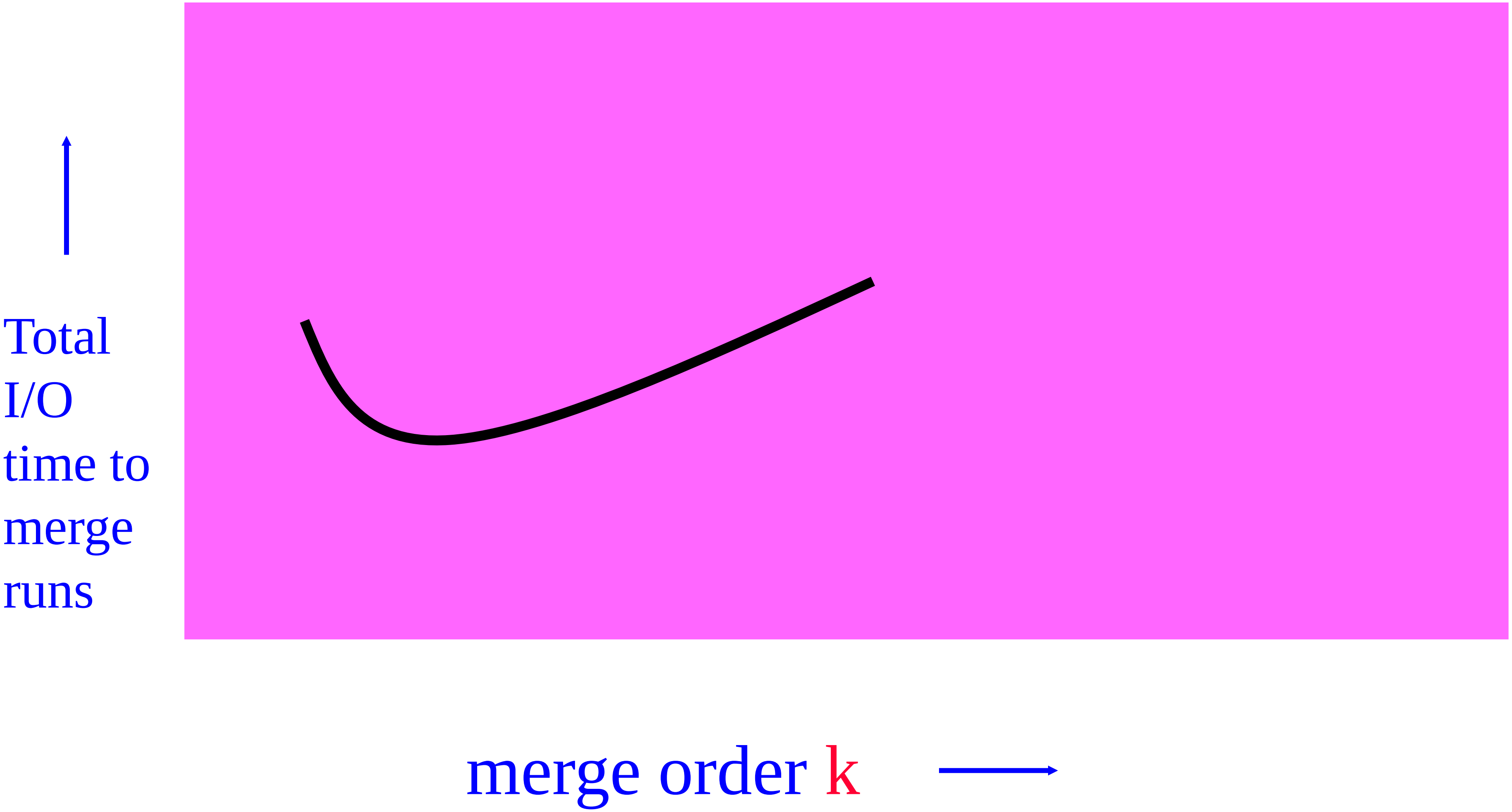
- Number of input buffers needed is linear in merge order k.
- Since memory size is fixed, block size decreases as k increases (after a certain k).
- So, number of blocks increases.
- So, number of seek and latency delays per pass increases.

I/O Time Per Merge Pass

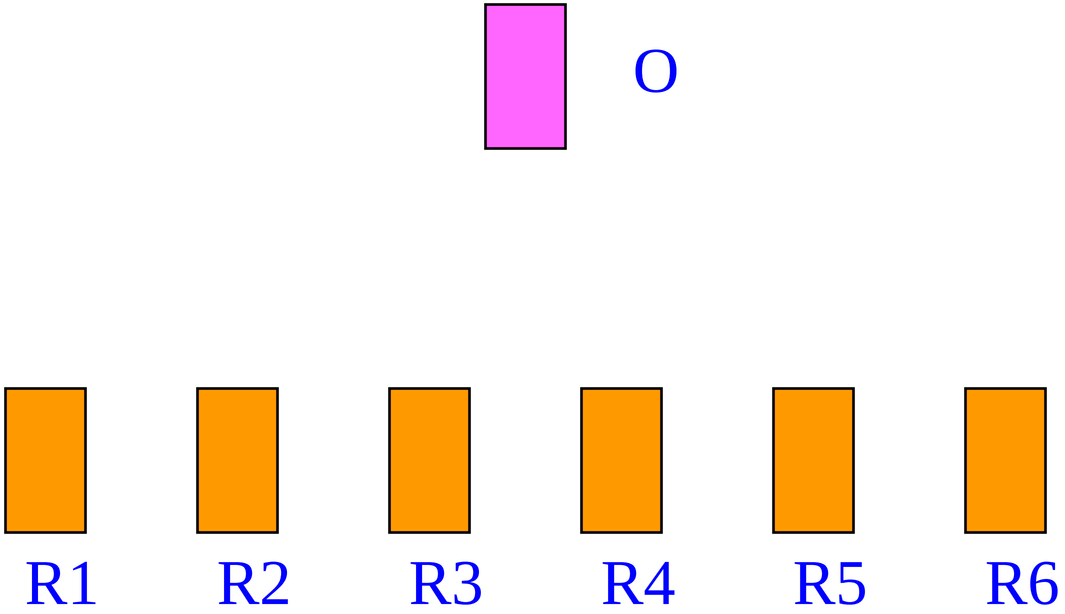


Total I/O Time To Merge Runs

- (I/O time for one merge pass) * ceil($\log_k$(number of initial runs))

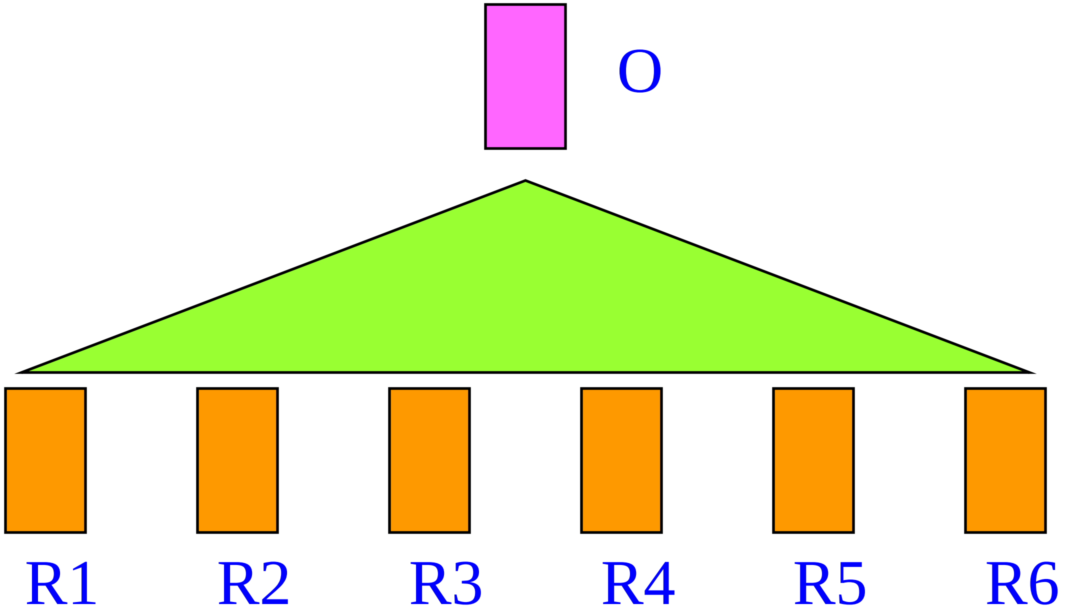


Internal Merge Time



- Naïve way => $k – 1$ compares to determine next record to move to the output buffer.
- Time to merge n records is $c(k – 1)n$, where c is a constant.
- Merge time per pass is $c(k – 1)n$.
- Total merge time is $c(k – 1)n\log_k r \sim cn(k/\log_2 k) \log_2 r$.

Merge Time Using A Selection Tree



- Time to merge n records is $dn\log_2 k$, where d is a constant.
- Merge time per pass is $dn\log_2 k$.
- Total merge time is $(dn\log_2 k)\ \log_k r\ = dn\log_2 r$.