Lecture 14:

String B-Trees (ctd.)

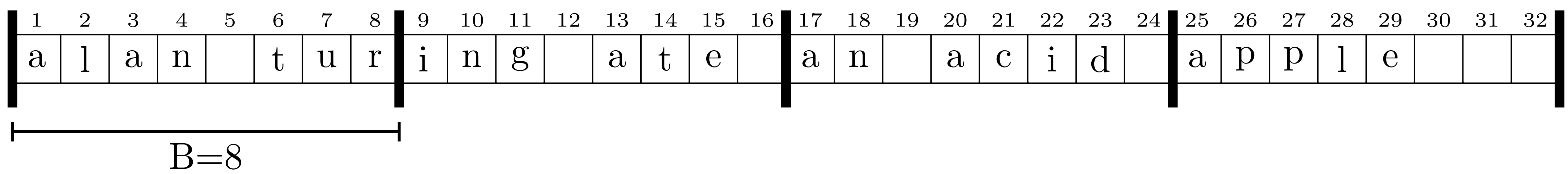
Cache-Oblivious DSs

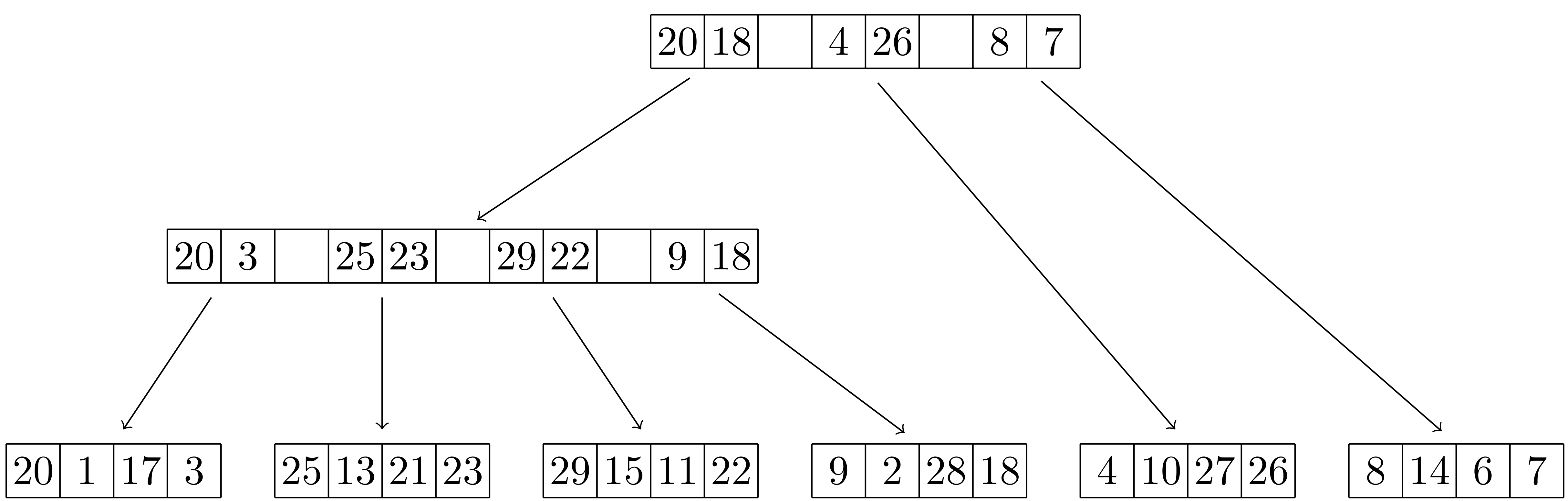
Johannes Fischer

Reminder

- search tree of degree $\Theta(B) \Rightarrow$ height $\lg_B N$
  - **leaves**: pointers to b strings [$b = \Theta(B)$]
  - **internal**: separators $L(v_1),R(v_1),\ldots, L(v_b),R(v_b)$
- search P: at every node with children $v_1,\ldots,v_b$
  - load 1 block containing $L(v_1),\ldots,R(v_b)$: one IO
  - load lg B strings & compare with P (bin. search)
    - $O(|P|/B)$ IOs per comparison
- **total**: $O(\lg_B N \times \lg B \times |P|/B) = O(|P|/B \lg N)$

$D = \{\texttt{alan},\texttt{turing},\texttt{ate},\texttt{an},\texttt{acid},\texttt{apple}\}$, $B = 8$





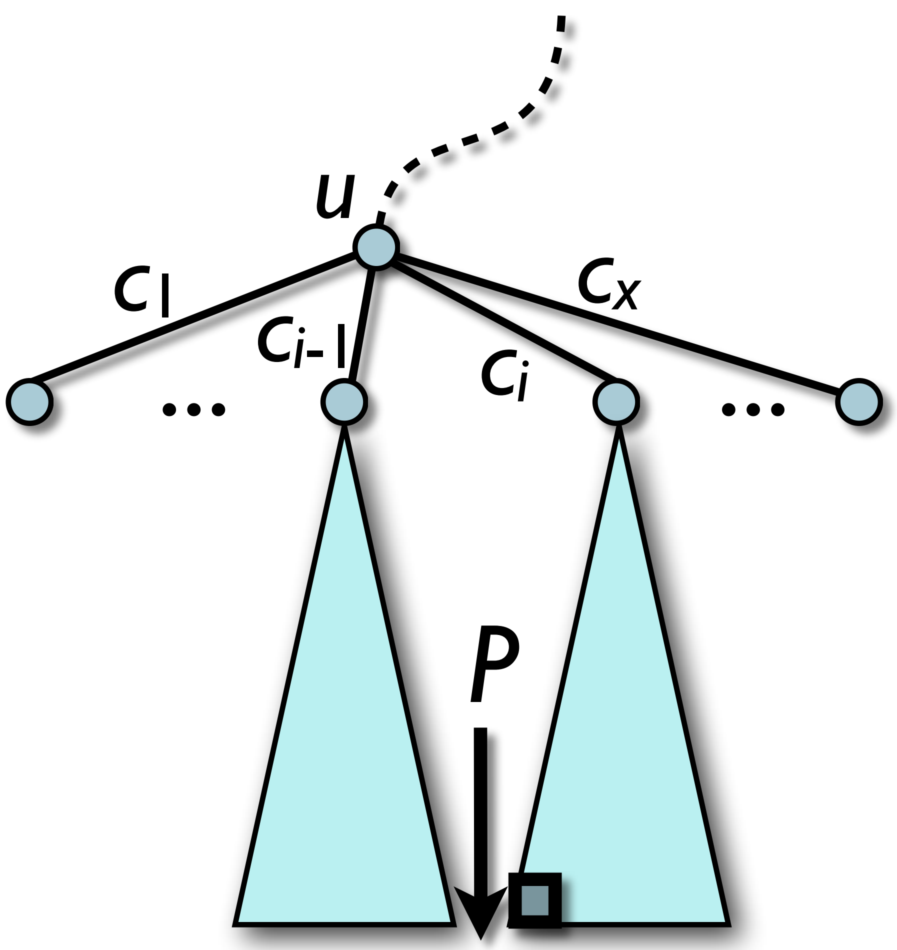
First Improvement

- add **Patricia Tries** (PT) to B-tree nodes
- PT_S for string set $S=\{S_1,...,S_k\}$:
  - compact trie over S (cf. suffix tree)
  - edges: store 1st (branching) character & length
  - size: $O(k)$ [**NOT** $O(|\sum|S_i||)$!!!]
- **blind search**: skip characters not stored
  - ⇝ false matches

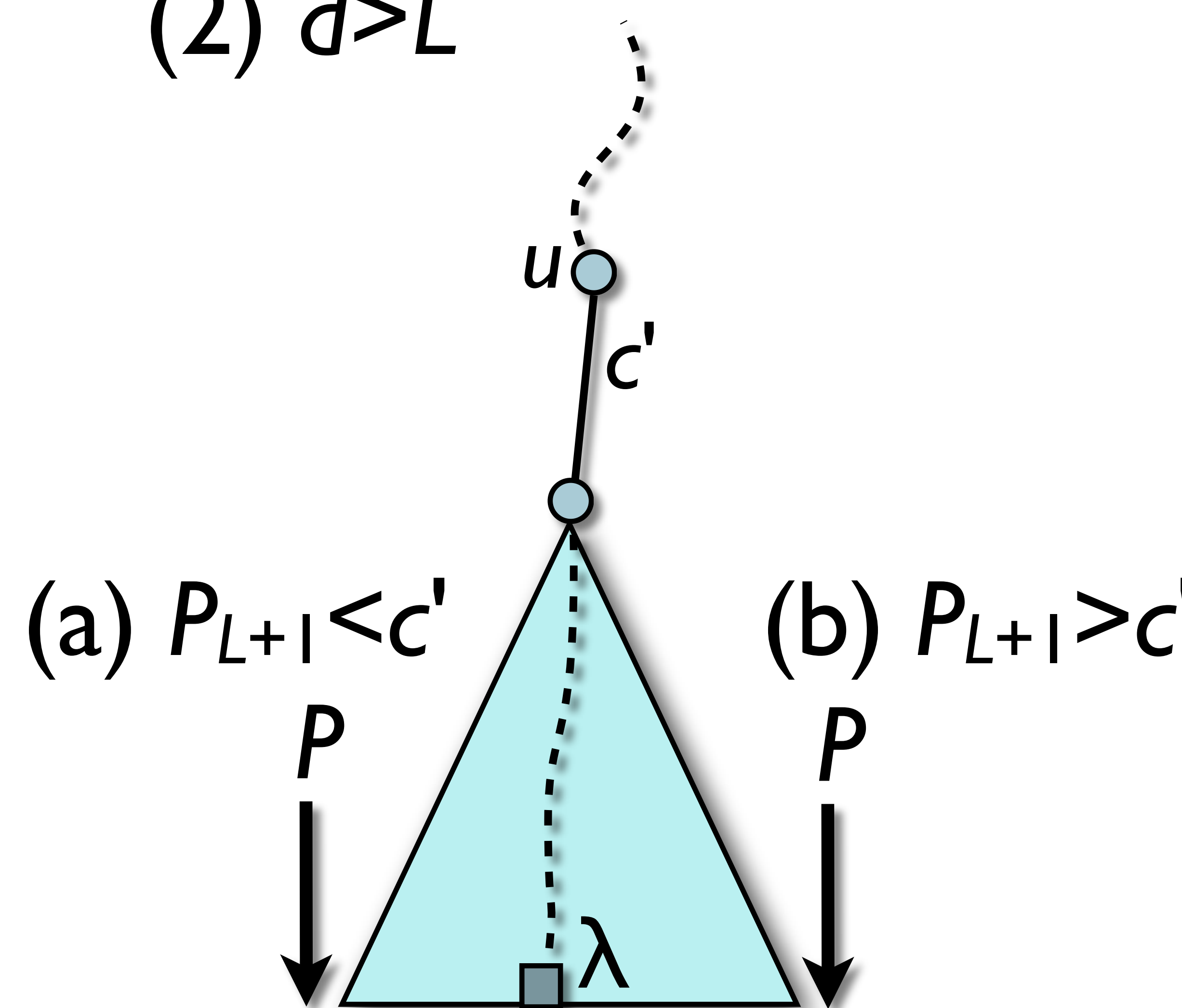
Correct Insertion Point

- say blind search ends at leaf λ
  - compute L=LCP(P, λ)
  - u: 1[st] node on root-to-λ path with $d \geq L$ chars

(1) $d=L$, $c_i < P_{L+1} < c_{i+1}$



(2) $d>L$



Blind Search: IOs

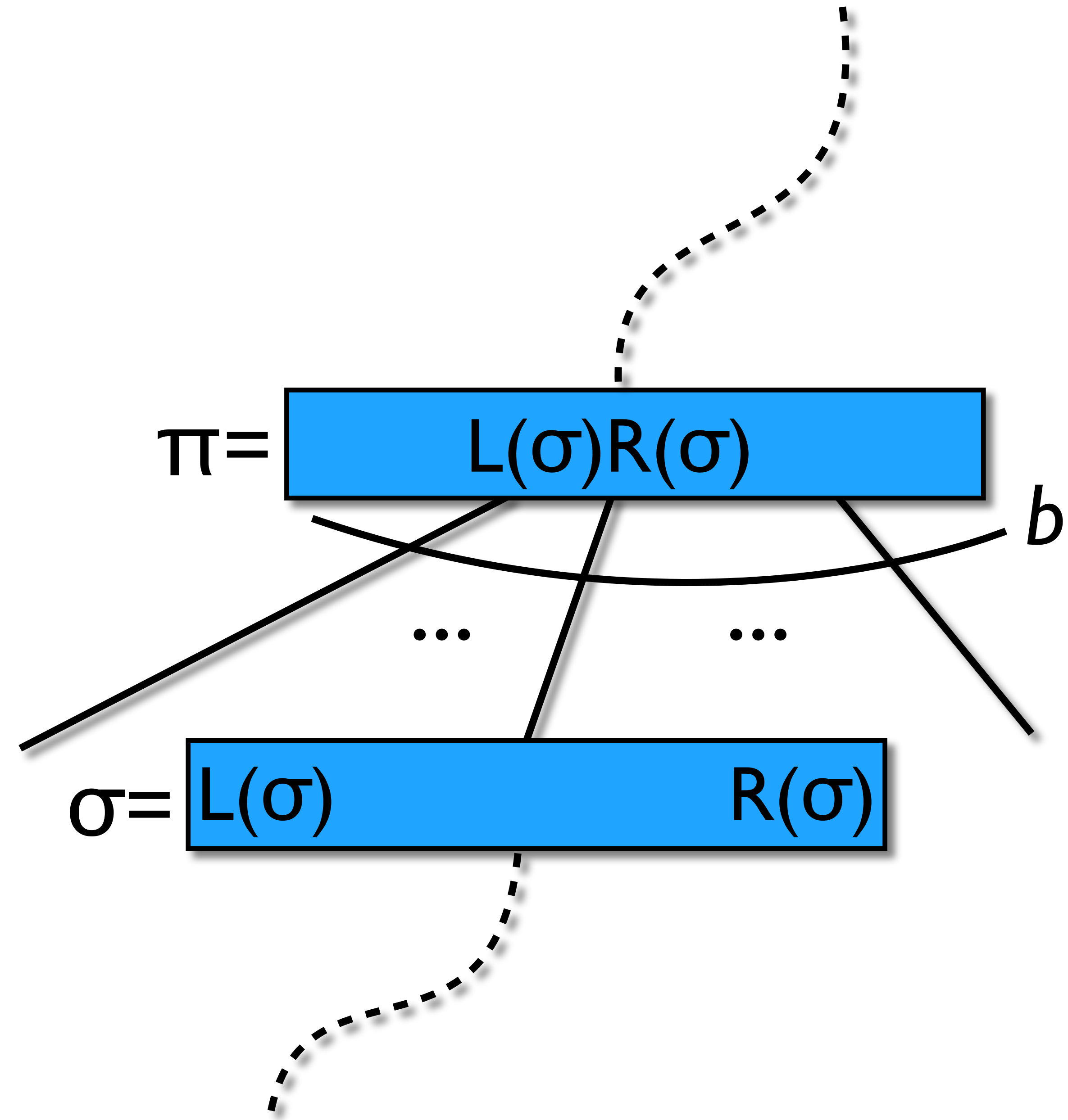
- at every node with children $v_1,...,v_b$:
  - load PT_S: **one** IO with $S=L(v_1),...,R(v_b)$
  - search PT_S for λ: **no IOs**
  - load **one** string and compare with P: $O(|P|/B)$ IOs
  - identification of insertion point: **no IOs**
- **total**: $O(|P|/B \lg_B N)$ IOs

Second Improvement

- search for *P*:
  - ...→ π → σ →...
- in PT_π:
  - compute *L*=LCP(P, λ)
- all strings in σ begin with *L*

⇒ in PT_σ:

  - compute *L*'=LCP(P, λ') **starting at** *P*[*L*+1]



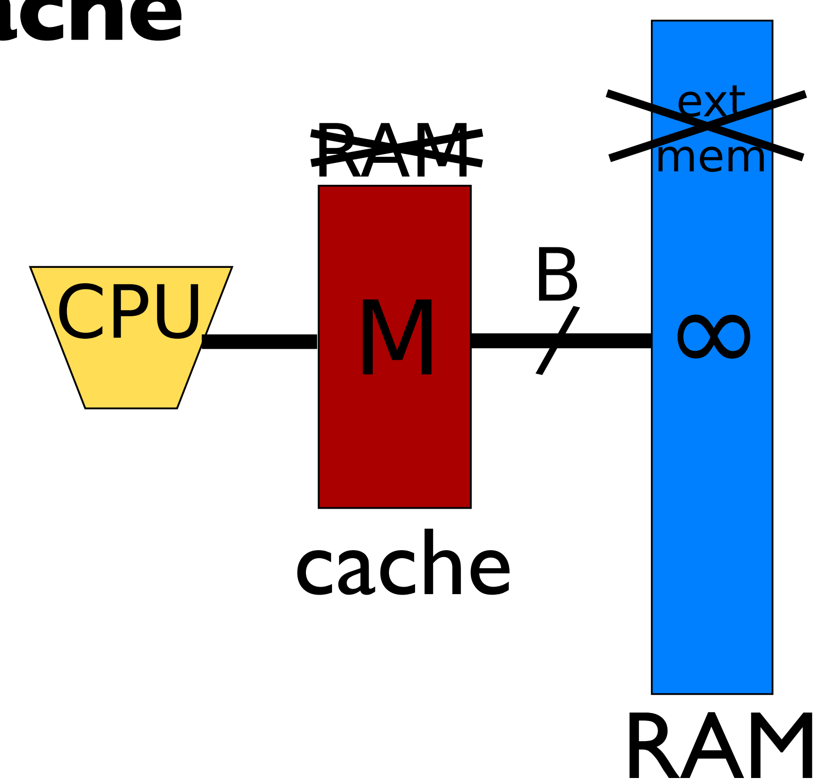
Final Complexity

- pass matched LCPs down the B-tree
- telescoping sum $\sum_{i \leq h} \frac{L_i - L_{i-1}}{B}$ IOs
  - height of B-tree $h = \lg_B N$
  - L_i = LCP-value on level i of String B-tree
- with $L_0 = 0$ and $L_h \leq |P|$:
  - $O(|P|/B + \lg_B N)$ IOs
- **inserting** P to D possible in $O(|P| \cdot h)$ IOs

Outlook on **Cache Oblivious** Data Structures

The Model

- Like EM:
  - *M*: size of internal memory ≜ **cache**
  - external memory ≜ **RAM**
  - *B*: block **transfer** size
- Now: *M* & *B* **unknown**
  - analysis over **all** values of *M*,*B*
- cache oblivious algorithm:
  - achieves EM lower bound **for all** values of *M*,*B*



Thoughts on CO-Model

- Example: **Scanning** $N \gg M$ items
  - optimal $O(N/B)$ in EM
  - no need to know $B \Rightarrow$ cache oblivious
- assumes **optimal** cache replacement
  - otherwise always next block evicted $\rightsquigarrow M=1$
  - LRU is 2-competitive
- **tall cache assumption**: $M = \Omega(B^2)$

Funnelsort

- *k*-funnel: black box for **merging** COly
  - merge k sorted lists of total size k^3
  - $O(k^3/B \lg_{M/B}(k^3/B)+k)$ IO's
  - space k^2

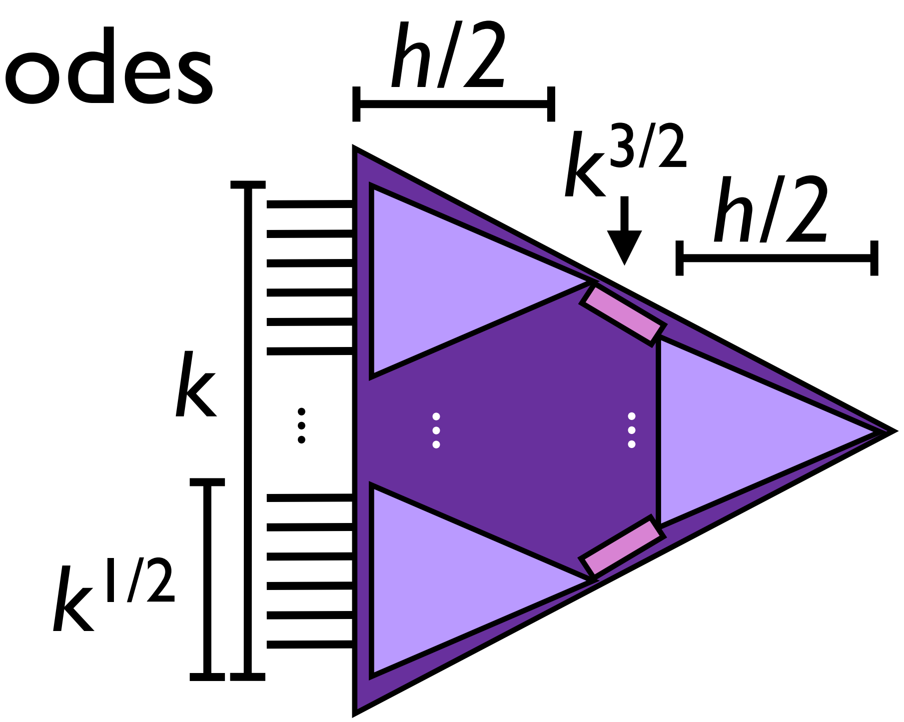
$\Rightarrow$ **Funnelsort** array $A[1,N]$:

1. split A into $N^{1/3}$ segments (size $N^{2/3}$)
2. sort each segment recursively
3. merge parts with $N^{1/3}$-funnels

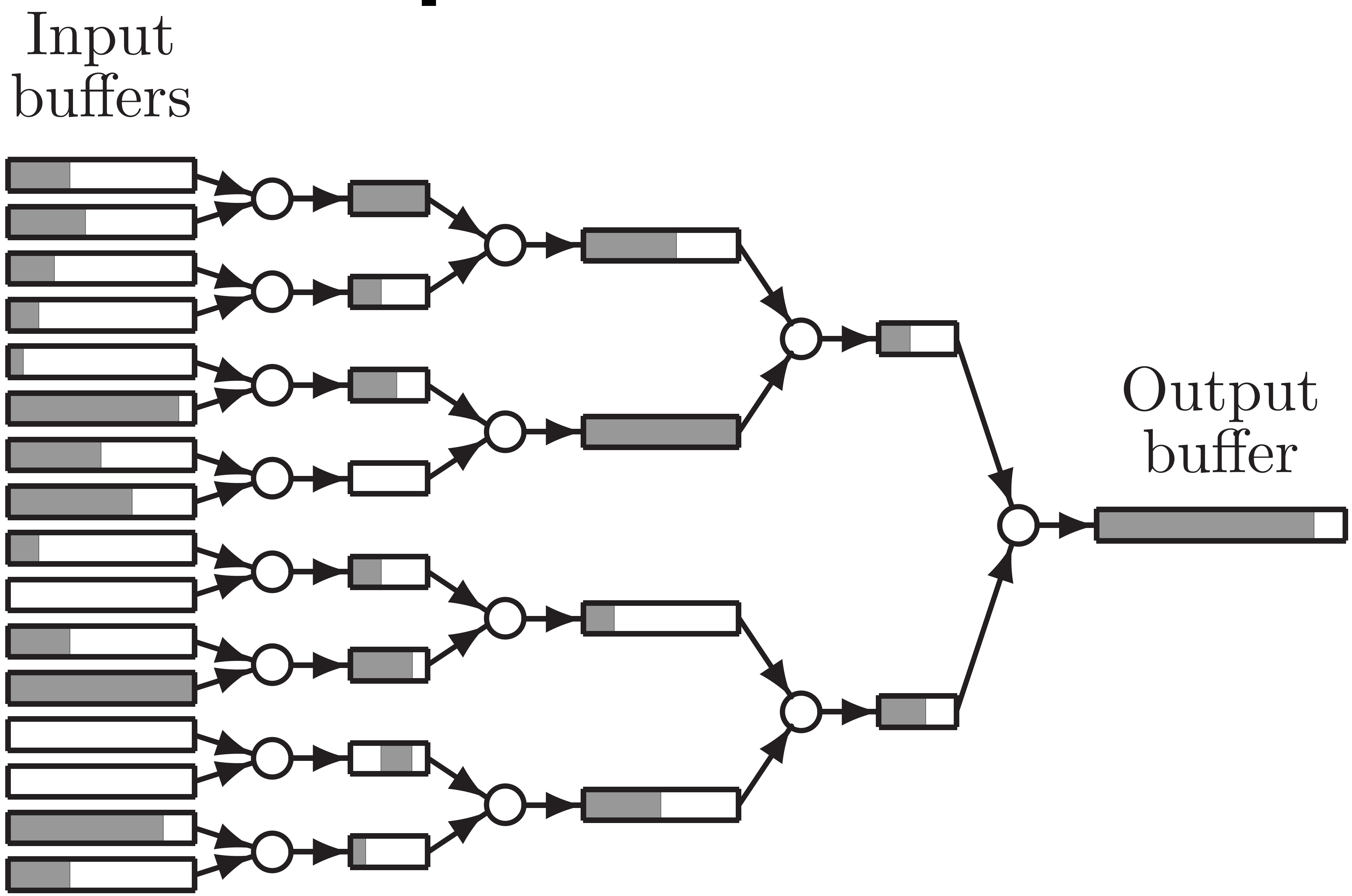
- IO: $T(N) = N^{1/3}T(N^{2/3}) + O(N/B \lg_{M/B}N/B + N^{1/3})$
  $= O(N/B \lg_{M/B}N/B)$ [see blackboard]

k-Funnels

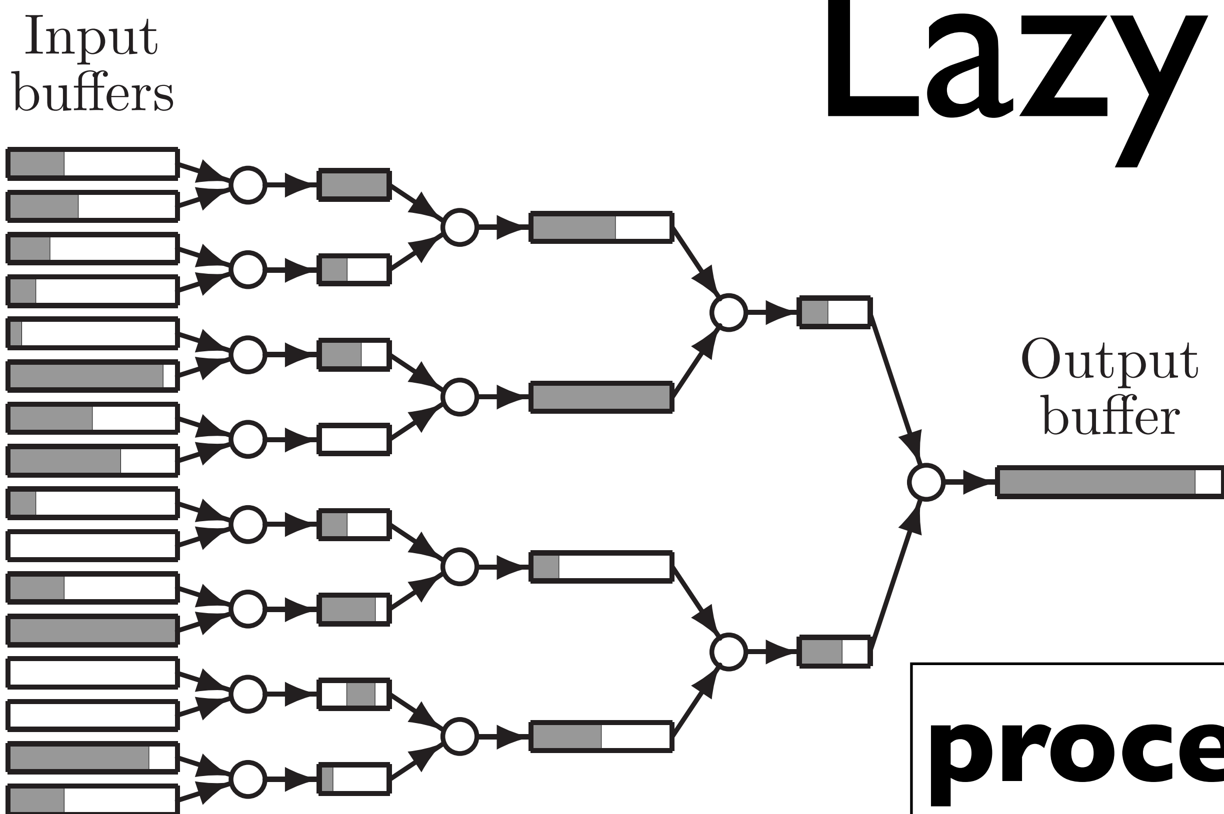
- binary tree
  - k **leaves**: input streams
  - **internal** nodes: mergers
  - output stream at root ($\triangleq$merged input streams)
- **buffers** between merge nodes
  - h=lg k levels with buffers
- size of buffers:
  - on level $h/2$: $k^{3/2}$
  - 1 upper and $k^{1/2}$ lower $k^{1/2}$-funnels: recursively



Example: 16-Funnel

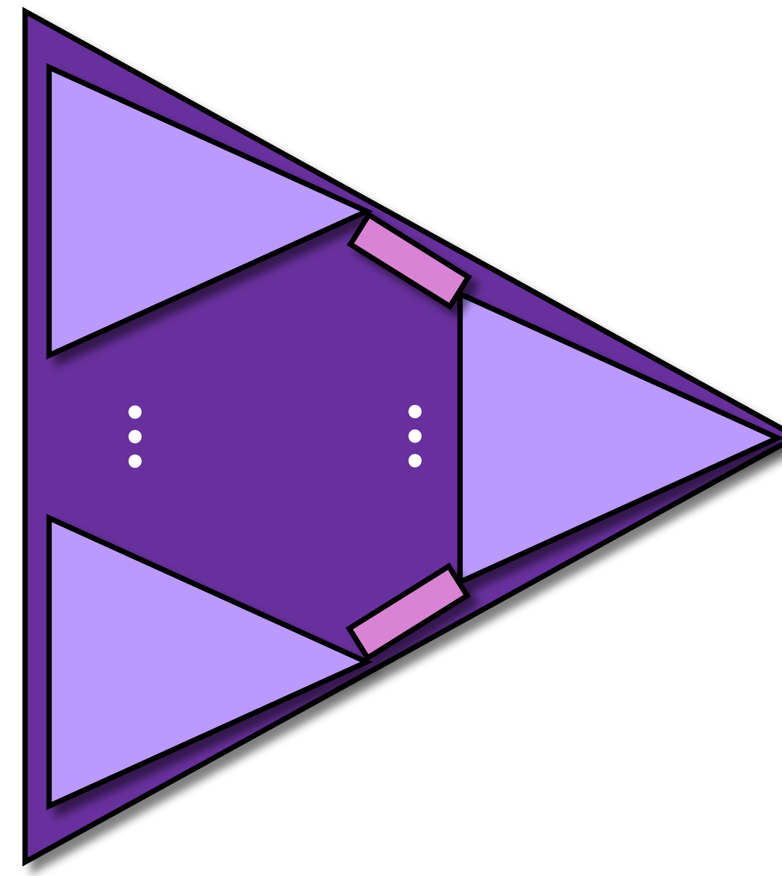


Lazy Filling




```
procedure FILL(v):
    while (v's output buffer not full)
        if (left input buffer empty)
            FILL(left child of v)
        if (right input buffer empty)
            FILL(right child of v)
        perform one merge step
```

Size of k-Funnel



- recall: size of buffers:
  - on level $h/2$: $k^{3/2}$
  - upper and lower $k^{1/2}$-funnels: recursively

$\Rightarrow S(k) = k^{1/2}\, k^{3/2} + (k^{3/2} + 1) S(k^{1/2})$
$= \Theta(k^2)$

IOs of *k*-Funnels (Idea)

- consider 1st recursive level where *j*-mergers have size $\leq M/3$ (**coarsest** level of detail)
- even though recursion continues, on level *j* all work in cache $\Rightarrow j^3/B+j$ IO's for j^3 elt.s
  - only when input buffer empty: evict, fill j^3 elements in input buffer, reload→no extra IOs
  - on path: only $O(\lg_j k)$ such *j*-funnels, $j=\Omega(M^{1/4})$

$\Longrightarrow O(k^3/B \lg_M(k)+k) \rightsquigarrow O(k^3/B \lg_{M/B}(k^3/B)+k)$