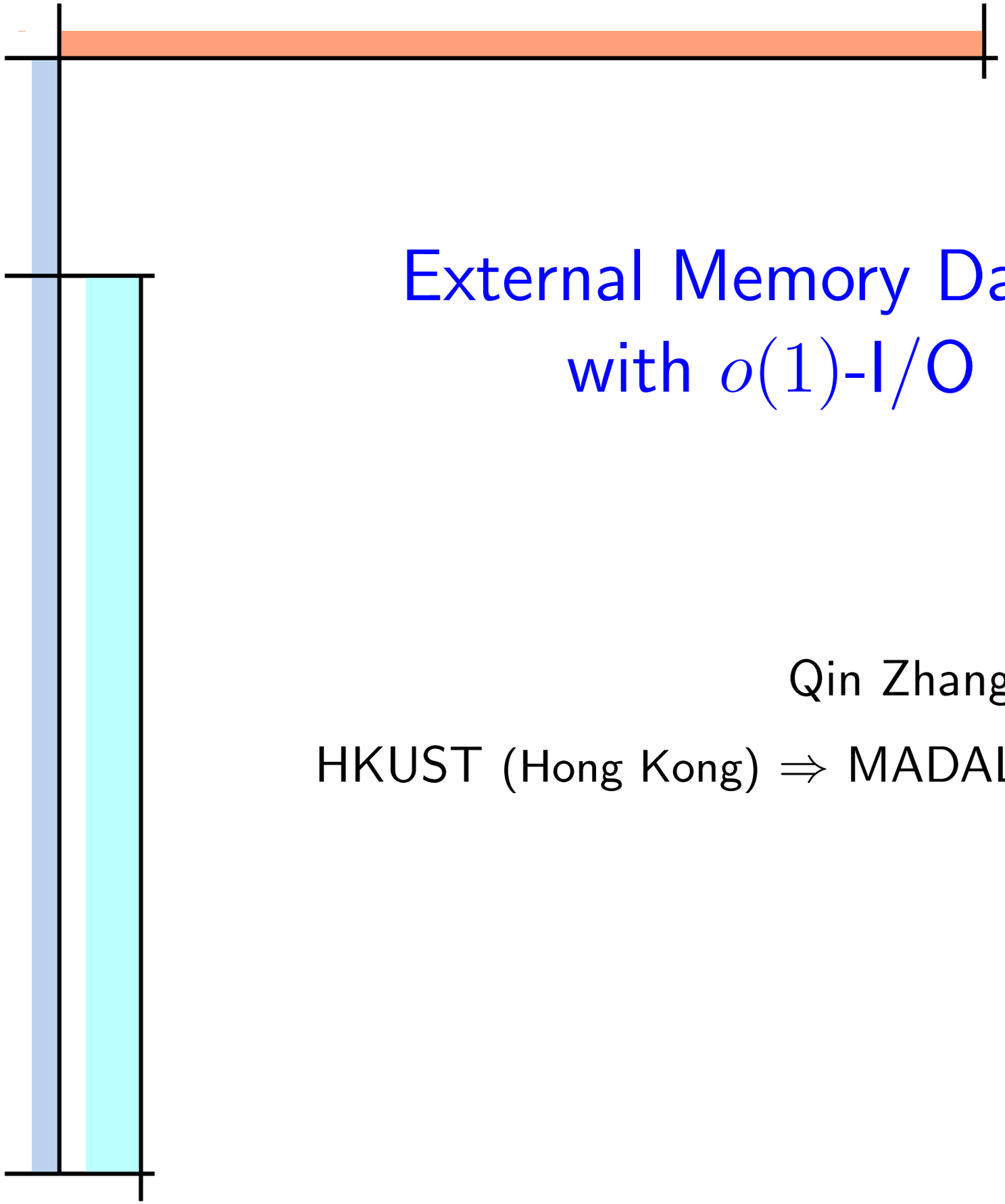


External Memory Data Structures with $o(1)$-I/O Updates

Qin Zhang

HKUST (Hong Kong) $\Rightarrow$ MADALGO (Aarhus, Denmark)

CTW 2010

September 2010, Tsinghua

Motivations for $o(1)$-I/O updates

for sufficiently large B

- Practical motivations:
  - If we can make updates faster without affecting the query time, why not?
  - There are situations where we desire even faster updates at the cost of slower queries

Archival data

Data streams

Motivations for $o(1)$-I/O updates

for sufficiently large B

- Practical motivations:
  - If we can make updates faster without affecting the query time, why not?
  - There are situations where we desire even faster updates at the cost of slower queries

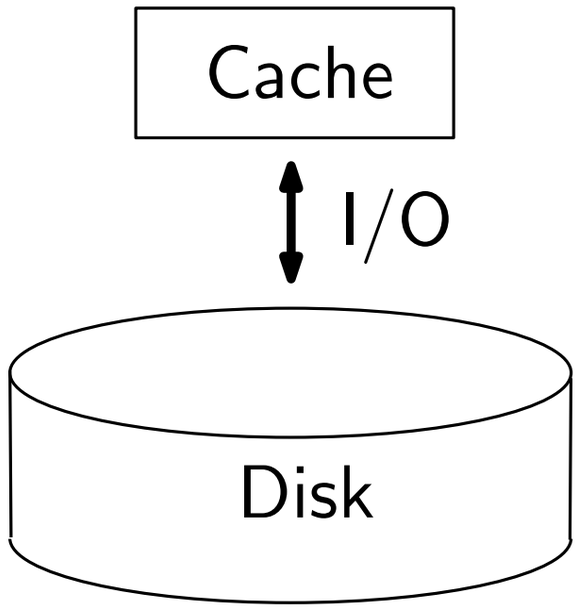
Archival data 

Data streams 

- Theoretical motivations
  - The update vs. query tradeoff
  - A more fundamental reason concerning the EM model (later)

The external memory model

- External memory (EM) model (or I/O model) (Aggarwal and Vitter 1988):



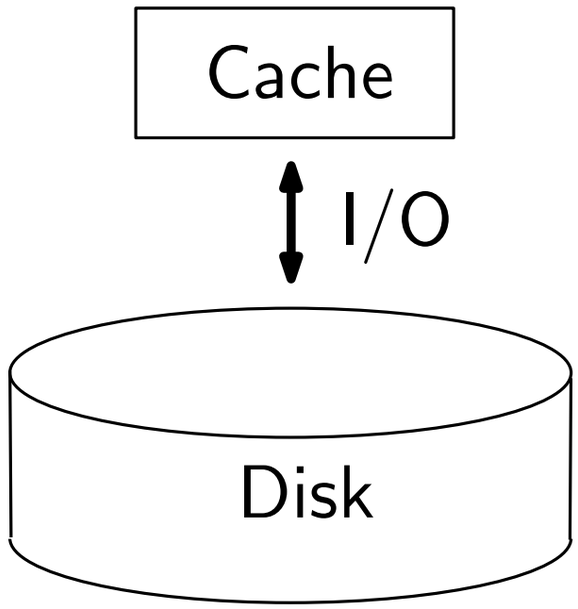
Cache of size $M = \Omega(B)$

Each I/O reads/writes a cell

Disk partitioned into cells of size B

The external memory model

- External memory (EM) model (or I/O model) (Aggarwal and Vitter 1988):



Cache of size $M = \Omega(B)$

Each I/O reads/writes a cell

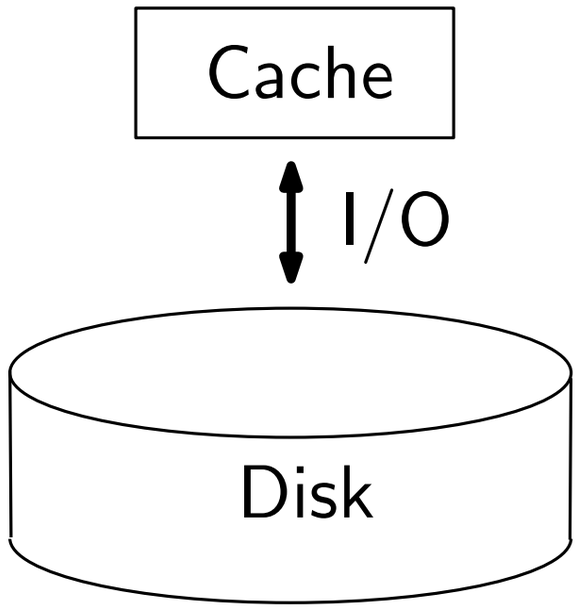
Disk partitioned into cells of size B

Cost of an operation: # of cells read/changed;

accessing the cache is free of charge.

The external memory model

- External memory (EM) model (or I/O model) (Aggarwal and Vitter 1988):



Cache of size $M = \Omega(B)$

Each I/O reads/writes a cell

Disk partitioned into cells of size B

Cost of an operation: # of cells read/changed;

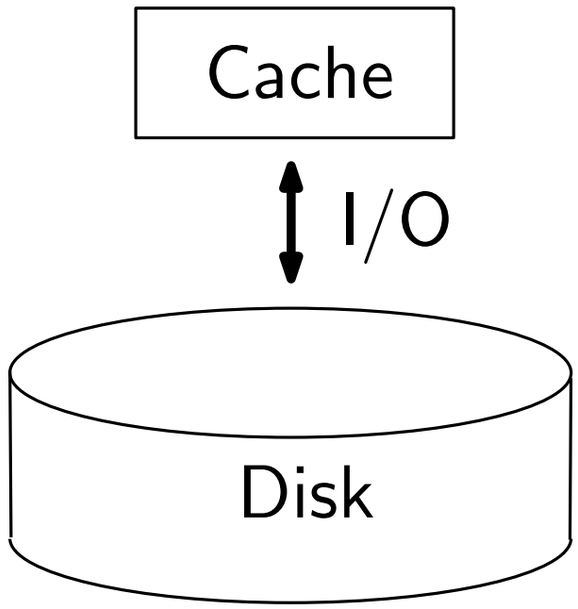
accessing the cache is free of charge.

- Motivated by the real-world applications:

  accessing the cache is much faster than accessing higher level memory hierarchies.

The external memory model

- External memory (EM) model (or I/O model) (Aggarwal and Vitter 1988):



Cache of size $M = \Omega(B)$

Each I/O reads/writes a cell

Disk partitioned into cells of size B

Cost of an operation: # of cells read/changed;
accessing the cache is free of charge.

- Similar to Yao's *cell probe* model, only that in the EM model (1) cell size is large; (2) has an explicit cache.

The central issue in the EM model

Can we buffer the updates ($t_u = o(1)$)?

What is the limit of buffering?

The central issue in the EM model

Can we buffer the updates ($t_u = o(1)$)?

What is the limit of buffering?

Think about *stack*, *queue*, . . .

Dynamic data structure problems

(i.e. support insertion / deletion)

1. Membership: Maintain a set $S \subseteq U$ with $|S| \leq n$.
   Given an $x \in U$, is $x \in S$? Yes or No.
2. Dictionary: Similar. If $x \in S$, return associated info.
3. Predecessor: Maintain a set $S \subseteq U$ with $|S| \leq n$.
   Given an $x \in U$, return $\max\{y \mid y \leq x, y \in S\}$.
4. 1D range reporting: Maintain a set $S \subseteq U$ with $|S| \leq n$.
   Given a range $[a, b]$, return $\{x \mid a \leq x \leq b, x \in S\}$.
   Cost is written as $f(n) + k/B$.
5. Partial sum: Given an array $A = \{a_1, a_1, \ldots, a_n\}$,
   Update(i, j): set $a_i = j$. Query(i): $\sum_{k=1}^{i} a_k$.
6. Priority queue: Support insert, delete and deletemin.
7. Union-find: Support union and find. ...

Dynamic data structure problems

(i.e. support insertion / deletion)

1. Membership: Maintain a set $S \subseteq U$ with $|S| \leq n$.
   Given an $x \in U$, is $x \in S$? Yes or No.
2. Dictionary: Similar. If $x \in S$, return associated info.
3. Predecessor: Maintain a set $S \subseteq U$ with $|S| \leq n$.
   Given an $x \in U$, return $\max\{y \mid y \leq x, y \in S\}$.
4. 1D range reporting: Maintain a set $S \subseteq U$ with $|S| \leq n$.
   Given a range $[a, b]$, return $\{x \mid a \leq x \leq b, x \in S\}$.
   Cost is written as $f(n) + k/B$.
5. Partial sum: Given an array $A = \{a_1, a_1, \ldots, a_n\}$,
   Update(i, j): set $a_i = j$. Query(i): $\sum_{k=1}^{i} a_k$.
6. Priority queue: Support insert, delete and deletemin.
7. Union-find: Support union and find. ...

Objective: Tradeoff between update cost t_u and query cost t_q

Upper bounds: Surprisingly few and simple

- Priority queue: $O(\frac{1}{B}\log_{M/B}\frac{N}{M})$ insertion and delete-min

Upper bounds: Surprisingly few and simple

- Priority queue: $O(\frac{1}{B}\log_{M/B}\frac{N}{M})$ insertion and delete-min

$$sort(N) = O(\frac{N}{B}\log_{M/B}\frac{N}{M})$$

Upper bounds: Surprisingly few and simple

- Priority queue: $O(\frac{1}{B}\log_{M/B}\frac{N}{M})$ insertion and delete-min
- Predecessor
  - B-tree: $O(\log_B \frac{N}{M})$ insertion and query
  - Buffer-tree (buffered-repository tree) [Arge, WADS'95; Buchsbaum, Goldwasser, Venkatasubramanian, Westbrook, SODA'00]

Update: $O(\frac{\ell}{B}\log_\ell \frac{N}{M})$

Query: $O(\log_\ell \frac{N}{M})$

$2 < \ell < B$

or

Update: $O(\frac{1}{B}\log_\ell \frac{N}{M})$

Query: $O(\ell \log_\ell \frac{N}{M})$

$2 < \ell < M/B$

Upper bounds: Surprisingly few and simple

- Priority queue: $O(\frac{1}{B}\log_{M/B}\frac{N}{M})$ insertion and delete-min
- Predecessor
  - B-tree: $O(\log_B \frac{N}{M})$ insertion and query
  - Buffer-tree (buffered-repository tree) [Arge, WADS'95; Buchsbaum, Goldwasser, Venkatasubramanian, Westbrook, SODA'00]

    Update: $O(\frac{\ell}{B}\log_\ell \frac{N}{M})$

    Query: $O(\log_\ell \frac{N}{M})$

    $2 < \ell < B$

    or

    Update: $O(\frac{1}{B}\log_\ell \frac{N}{M})$

    Query: $O(\ell \log_\ell \frac{N}{M})$

    $2 < \ell < M/B$
- Range reporting: the same

Upper bounds: Surprisingly few and simple

- Priority queue: $O(\frac{1}{B}\log_{M/B}\frac{N}{M})$ insertion and delete-min
- Predecessor
  - B-tree: $O(\log_B \frac{N}{M})$ insertion and query
  - Buffer-tree (buffered-repository tree) [Arge, WADS'95; Buchsbaum, Goldwasser, Venkatasubramanian, Westbrook, SODA'00]

    Update: $O(\frac{\ell}{B}\log_\ell \frac{N}{M})$

    Query: $O(\log_\ell \frac{N}{M})$

    $2 < \ell < B$

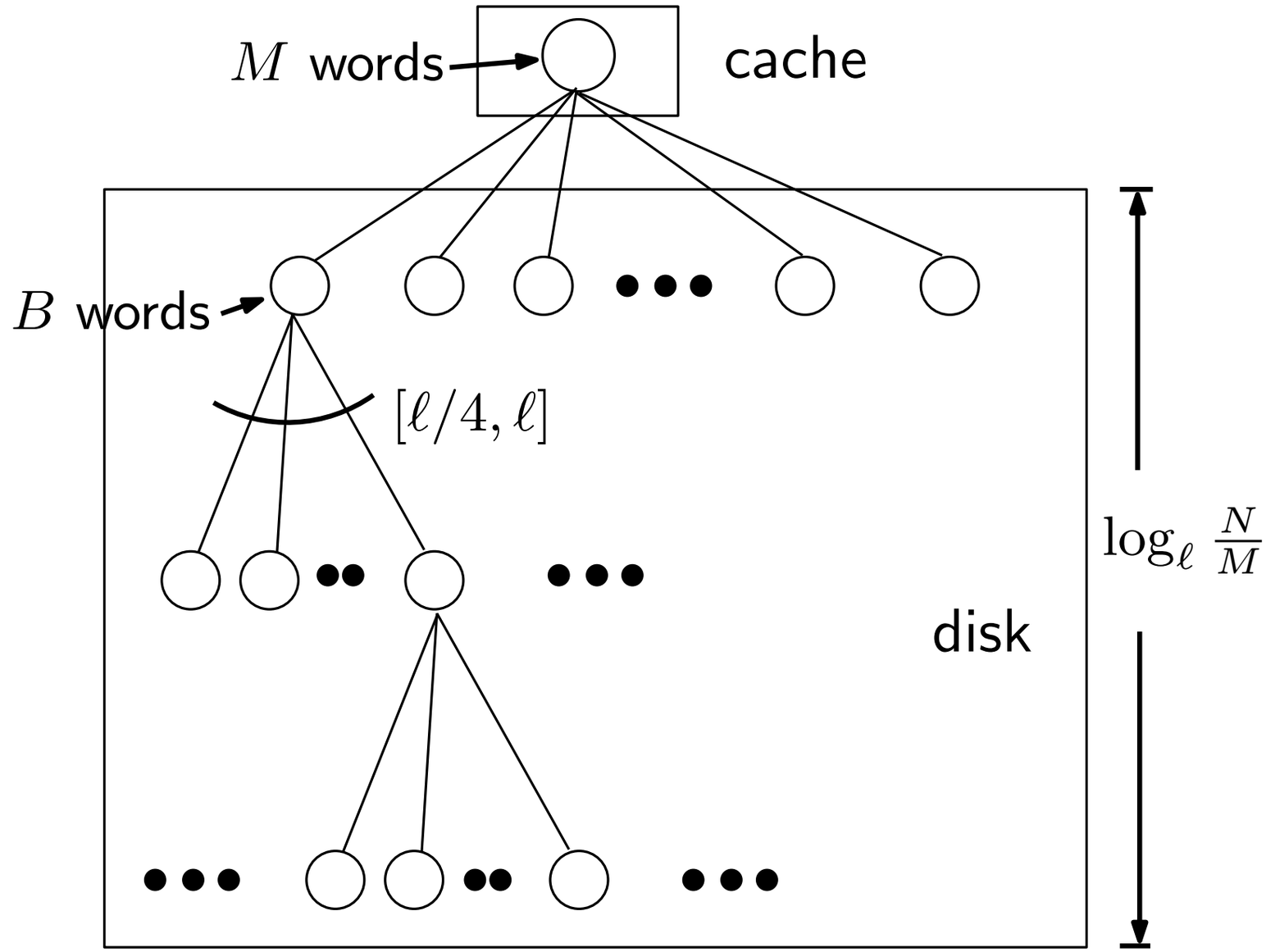
    or

    Update: $O(\frac{1}{B}\log_\ell \frac{N}{M})$

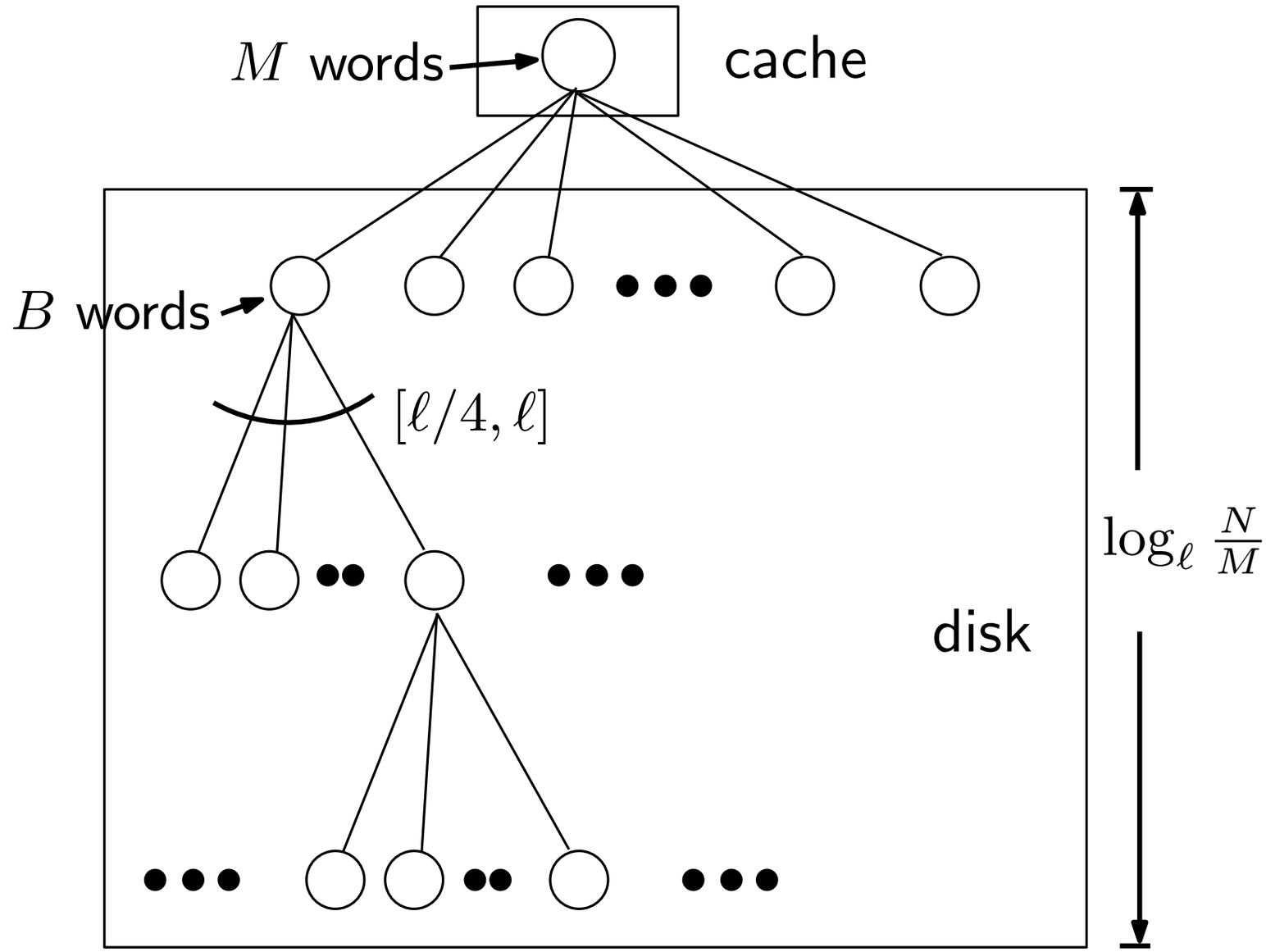
    Query: $O(\ell \log_\ell \frac{N}{M})$

    $2 < \ell < M/B$
- Range reporting: the same
- Partial-sum: the same

Upper bounds: Buffer tree for example



Upper bounds: Buffer tree for example



Very simple. But is it optimal?

Upper bounds (cont.)

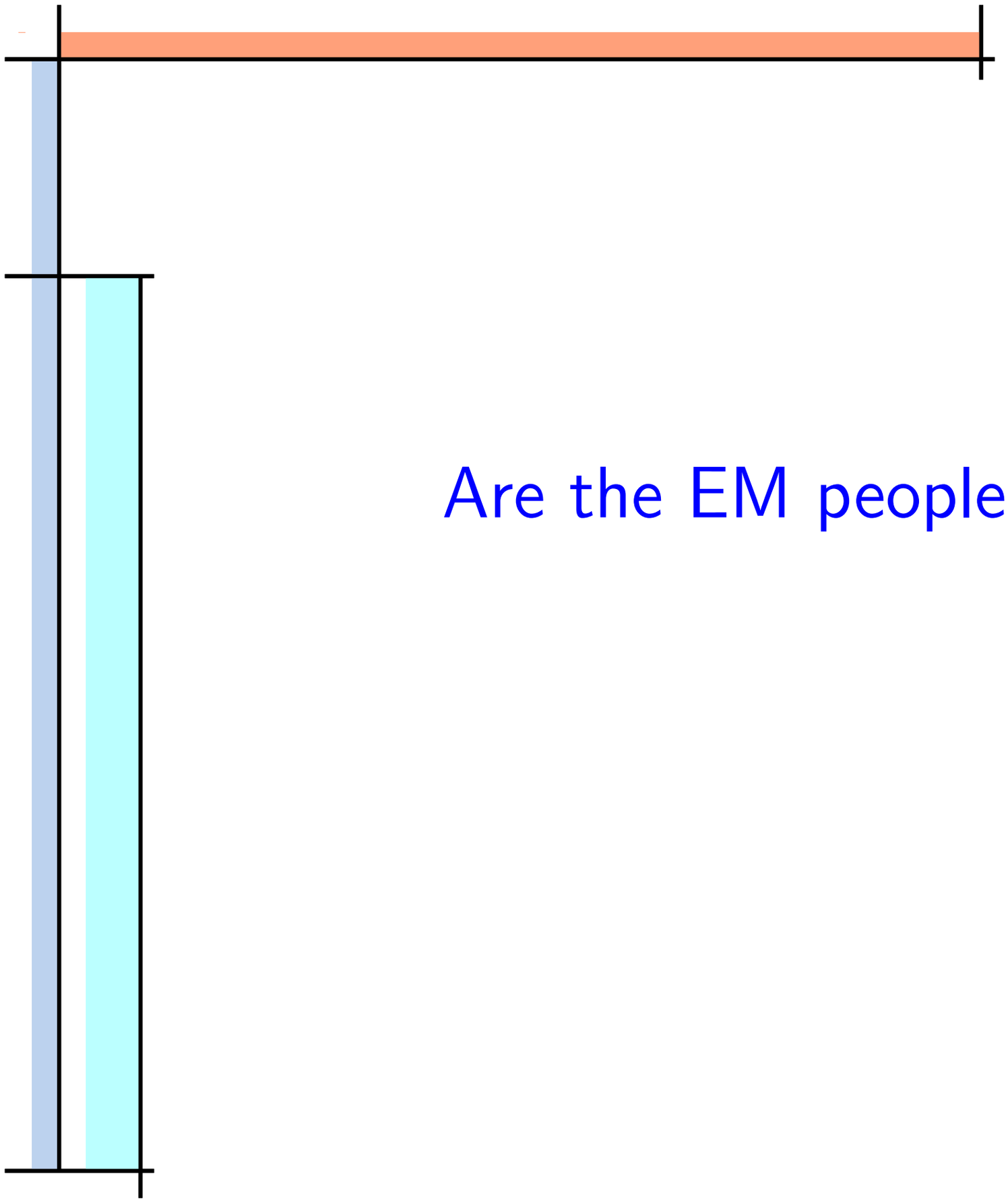
- Dictionary and membership
  - Knuth, 1973: External hashing
    Expected average cost of an operation is $1 + 1/2^{\Omega(B)}$, provided the load factor α is less than a constant smaller than 1. (truly random hash function)
  - Data structures like Arge's Buffer tree:
    Update $= O(\frac{\ell}{B} \log_\ell n)$, Query $= O(\log_\ell n)$. $(n = N/M)$

Upper bounds (cont.)

- Dictionary and membership
  - Knuth, 1973: External hashing
    Expected average cost of an operation is $1 + 1/2^{\Omega(B)}$, provided the load factor α is less than a constant smaller than 1. (truly random hash function)
  - Data structures like Arge's Buffer tree:
    Update $= O(\frac{\ell}{B} \log_\ell n)$, Query $= O(\log_\ell n)$. $(n = N/M)$
- **Question**: Are these upper bounds tight? Nothing in between?

Compared with the rich results in RAM!

- 1D-range reporting
  - $O(\sqrt{\log N / \log\log N})$ insertion and query [Andersson, Thorup, JACM'07]
  - $O(\log N / \log\log N)$ insertion and $O(\log\log N)$ query [Mortensen, Pagh, Pătraşcu, STOC'05]
  - Other results that depend on the word size w
- Predecessor
  - $\Theta(\sqrt{\log N / \log\log N})$ insertion and query [Andersson, Thorup, JACM'07]
- Partial-sum
  - $\Theta(\log N)$ insertion query [Pătraşcu, Demaine, SODA'04]



Are the EM people just dumb?

Lower bounds: the models

- *External comparison model*. Only comparisons. Indivisibility.

Lower bounds: the models

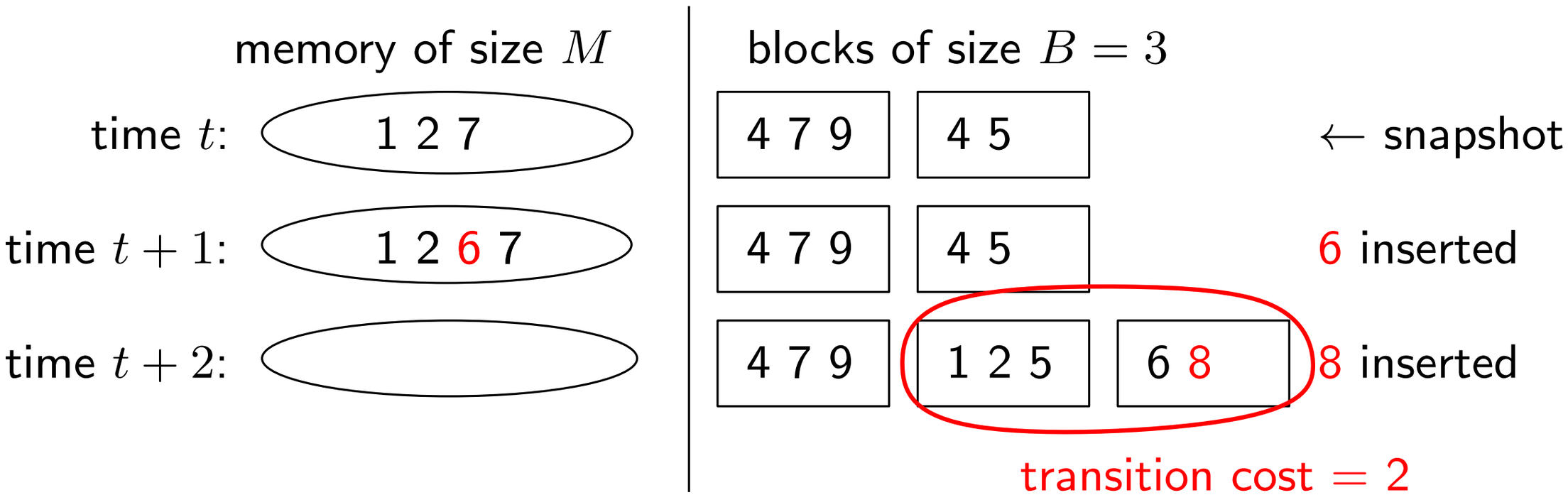
- *External comparison model*. Only comparisons. Indivisibility.
- *(General) external model*

  Any operation is allowed inside a cell. Only #cells probed is counted.
  Most general. Information theoretic.
- *Cell-probe model*. $B = \log N$.

Lower bounds: the models

- *External comparison model*. Only comparisons. Indivisibility.
- *(General) external model*

  Any operation is allowed inside a cell. Only #cells probed is counted.
  Most general. Information theoretic.
- *Cell-probe model*. $B = \log N$.
- *Dynamic indexability* (for reporting problems) #blocks changed



Lower bounds: Predecessor

[Brodal, Fagerberg, SODA'03]: (External comparison model)

$$q \log\left(uB \log^2 n\right) = \Omega\left(\log n\right) \qquad n = \frac{N}{M}$$

$$q \geq n \Big/ \left(\frac{M}{B}\right)^{O(uB)}$$

Lower bounds: Predecessor

[Brodal, Fagerberg, SODA'03]: (External comparison model)

$$q \log \left(uB \log^2 n \right) = \Omega \left(\log n \right) \qquad n = \frac{N}{M}$$

$$q \geq n \Big/ \left(\frac{M}{B} \right)^{O(uB)}$$

Current upper bounds:

q	u
$\log n$	$\frac{1}{B} \log n$
$\log_B n$	$\frac{B^\epsilon}{B} \log_B n$
$(\frac{M}{B})^\epsilon \log_{M/B} n$	$\frac{1}{B} \log_{M/B} n$

Lower bounds: Predecessor

[Brodal, Fagerberg, SODA'03]: (External comparison model)

$$q \log\left(uB \log^2 n\right) = \Omega\left(\log n\right) \qquad n = \frac{N}{M}$$

$$q \geq n \Big/ \left(\frac{M}{B}\right)^{O(uB)}$$

Current upper bounds:

q	u
$\log n$	$\frac{1}{B} \log n$
$\log_B n$	$\frac{B^\epsilon}{B} \log_B n$
$(\frac{M}{B})^\epsilon \log_{M/B} n$	$\frac{1}{B} \log_{M/B} n$

$q \geq \frac{\log n}{\log \log n}$

$u \geq \frac{1}{B} \cdot \frac{1}{\log^2 n}$

$u \geq \frac{B^\epsilon}{B} \cdot \frac{1}{\log^2 n}$

$q \geq n^\epsilon$

holds only for $u < 0.5 \frac{1}{B} \log_{M/B} n$

Lower bounds: Range reporting

[Yi, PODS'09]: (Dynamic indexability model)
(holds also for predecessor)

$$\begin{cases} q \cdot \log(uB/q) = \Omega(\log B), & \text{for } q < \alpha \log B, \alpha \text{ is any constant;} \\ uB \cdot \log q = \Omega(\log B), & \text{for all } q. \end{cases}$$

Lower bounds: Range reporting

[Yi, PODS'09]: (Dynamic indexability model)
(holds also for predecessor)

$$\begin{cases} q \cdot \log(uB/q) = \Omega(\log B), & \text{for } q < \alpha \log B, \alpha \text{ is any constant;} \\ uB \cdot \log q = \Omega(\log B), & \text{for all } q. \end{cases}$$

Current upper bounds:

q	u
$\log n$	$\frac{1}{B} \log n$
$\log_B n$	$\frac{B^\epsilon}{B} \log_B n$
$(\frac{M}{B})^\epsilon \log_{M/B} n$	$\frac{1}{B} \log_{M/B} n$

Lower bounds: Range reporting

[Yi, PODS'09]: (Dynamic indexability model)
(holds also for predecessor)

$$\begin{cases} q \cdot \log(uB/q) = \Omega(\log B), & \text{for } q < \alpha \log B, \alpha \text{ is any constant;} \\ uB \cdot \log q = \Omega(\log B), & \text{for all } q. \end{cases}$$

Current upper bounds:

q	u
$\log n$	$\frac{1}{B} \log n$
$\log_B n$	$\frac{B^\epsilon}{B} \log_B n$
$(\frac{M}{B})^\epsilon \log_{M/B} n$	$\frac{1}{B} \log_{M/B} n$

Assuming $\log_B n = O(1)$, or $B \geq n^\epsilon$, all bounds are tight

Lower bounds: Range reporting

[Yi, PODS'09]: (Dynamic indexability model)
(holds also for predecessor)

$$\begin{cases} q \cdot \log(uB/q) = \Omega(\log B), & \text{for } q < \alpha \log B, \alpha \text{ is any constant;} \\ uB \cdot \log q = \Omega(\log B), & \text{for all } q. \end{cases}$$

Current upper bounds:

q	u
$\log n$	$\frac{1}{B} \log n$
$\log_B n$	$\frac{B^\epsilon}{B} \log_B n$
$(\frac{M}{B})^\epsilon \log_{M/B} n$	$\frac{1}{B} \log_{M/B} n$

Assuming $\log_B n = O(1)$, or $B \geq n^\epsilon$, all bounds are tight

Can't be true for $B = o(\sqrt{\log n \log\log n})$, since the *exponential tree* achieves $u = q = O(\sqrt{\log n / \log\log n})$ [Andersson, Thorup, JACM'07].

Lower bounds: Partial sum

[Pătraşcu, Tarniţă, ICALP'05] (Cell probe model)

$$u \log \left(\frac{q}{\log n} + 2 \right) = \Omega \left(\frac{1}{B} \log n \right)$$

Lower bounds: Partial sum

[Pătraşcu, Tarniţă, ICALP'05] (Cell probe model)

$$u \log\left(\frac{q}{\log n} + 2\right) = \Omega\left(\frac{1}{B}\log n\right)$$

Current upper bounds:

q	u	
$\log n$	$\frac{1}{B}\log n$	tight
$\log_B n$	$\frac{B^\epsilon}{B}\log_B n$	
$(\frac{M}{B})^\epsilon \log_{M/B} n$	$\frac{1}{B}\log_{M/B} n$	tight

Lower bounds: Partial sum

[Pătraşcu, Tarniţă, ICALP'05] (Cell probe model)

$$u \log\left(\frac{q}{\log n} + 2\right) = \Omega\left(\frac{1}{B}\log n\right)$$

Current upper bounds:

q	u	
$\log n$	$\frac{1}{B}\log n$	tight
$\log_B n$	$\frac{B^\epsilon}{B}\log_B n$	
$(\frac{M}{B})^\epsilon \log_{M/B} n$	$\frac{1}{B}\log_{M/B} n$	tight

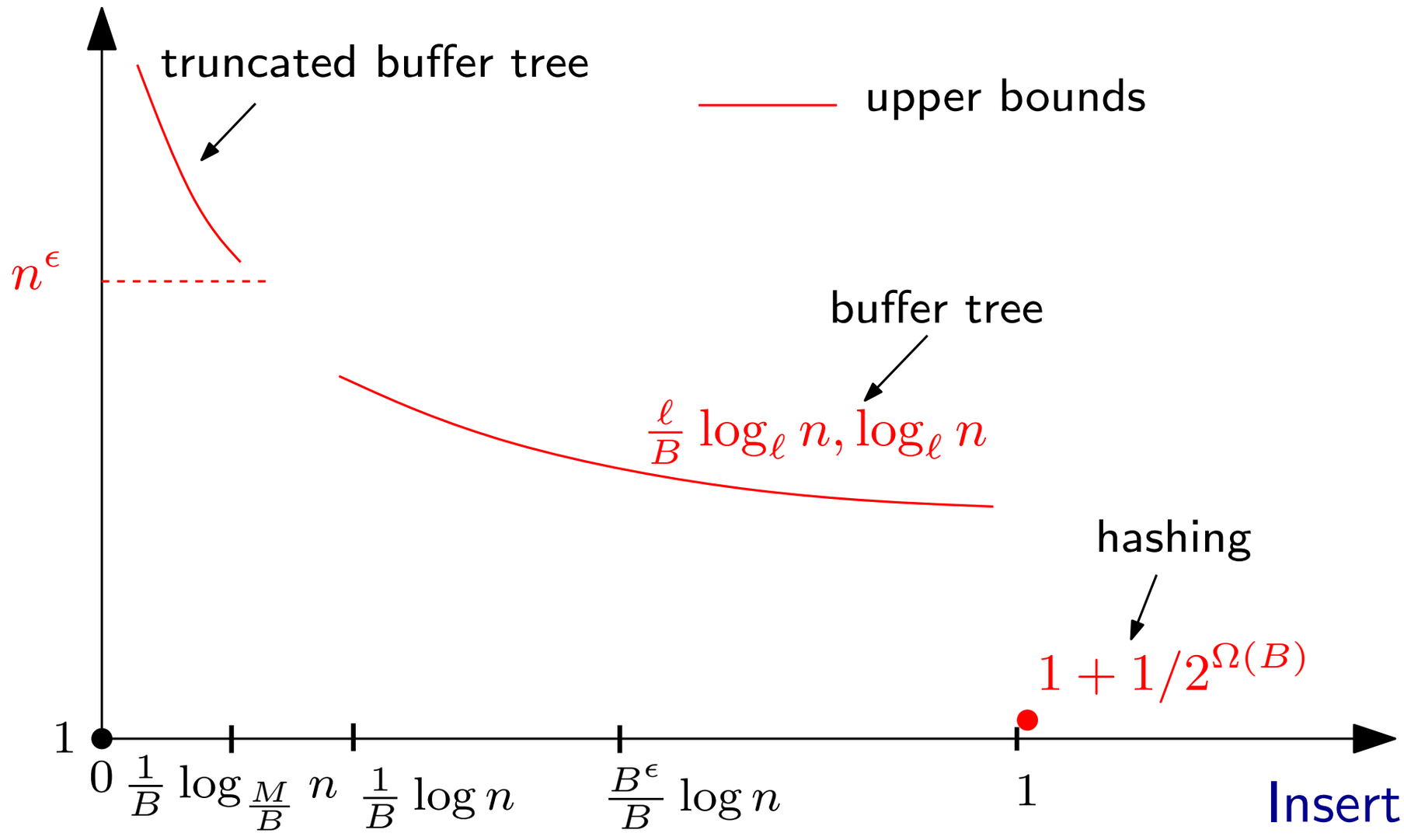
Conclusions:

- Partial-sum is a "nice" problem in internal & external memory.
- Predecessor and range reporting are "nice" in external memory only for sufficiently large B.

Lowe bounds: Membership

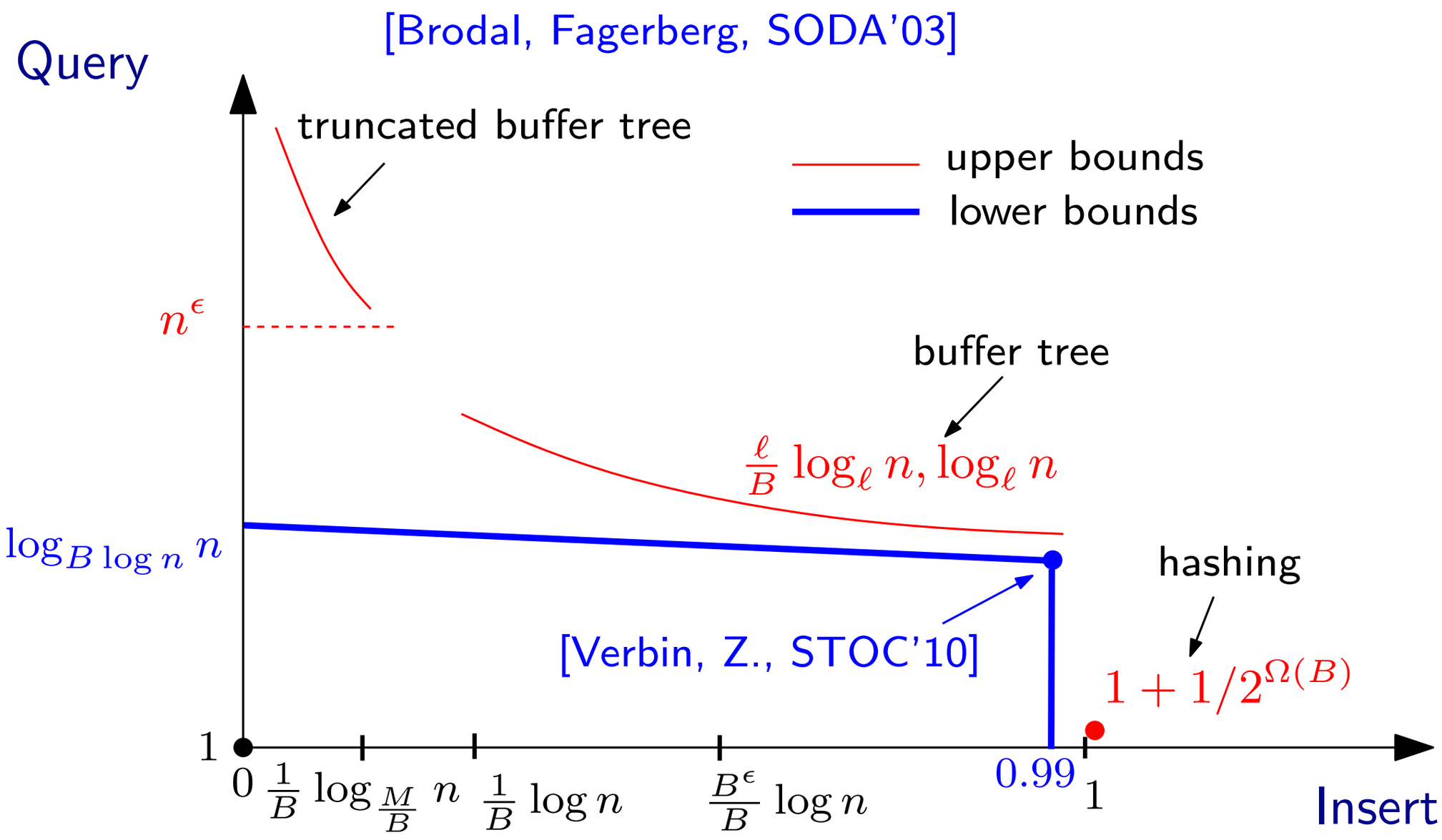
(General external memory model)

[Brodal, Fagerberg, SODA'03]



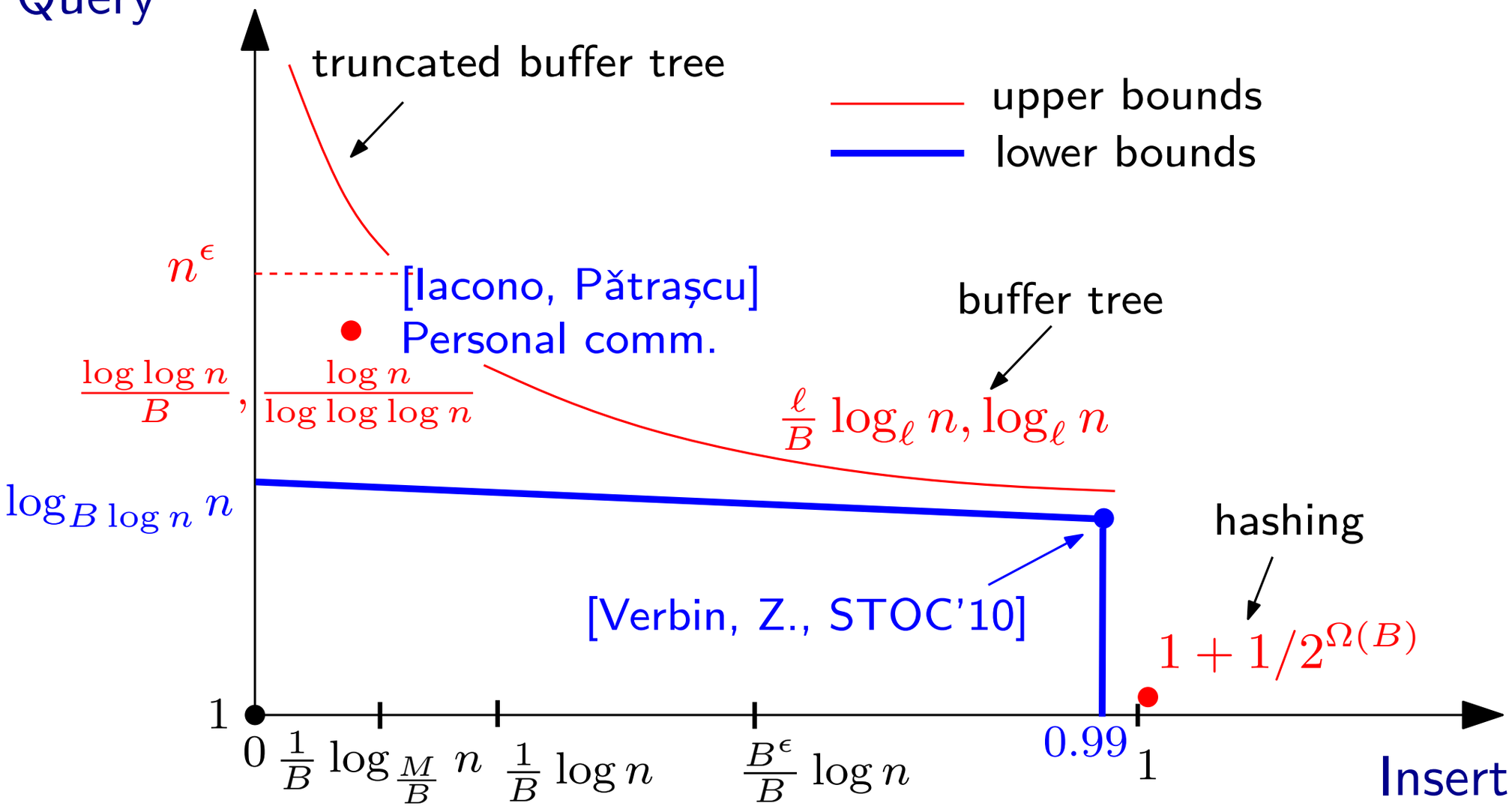
Lowe bounds: Membership

(General external memory model)



Lowe bounds: Membership

(General external memory model)



Implications of the LB for the Membership

- A strong dichotomy result:

  when designing an external memory data structure for dynamic membership,

  - either use external hash $(t_u = t_q = 1 + o(1))$
  - or use buffer tree $(t_u = o(1), t_q = O(\log_B n))$

Implications of the LB for the Membership

- A strong dichotomy result:

  when designing an external memory data structure for dynamic membership,

  - either use external hash $(t_u = t_q = 1 + o(1))$
  - or use buffer tree $(t_u = o(1), t_q = O(\log_B n))$

- Striking implications:

  the query complexities of many problems such as

  1D-range reporting, predecessor, partial-sum, etc.,

  are all the same in the regime where $1/B^{0.99} < t_u < 1$!

Two conceptual messages

1. **Buffering is impossible to achieve in the EM model with sublogarithmic query time.**

2. **EM model is a "cleaner" model than RAM in certain perspectives**.

Some future works

- We still cannot handle fast updates.

  e.g. if $t_u = O(1/B)$, $t_q = \Omega(n^\epsilon)$?

Some future works

- We still cannot handle fast updates.

  e.g. if $t_u = O(1/B)$, $t_q = \Omega(n^\epsilon)$?

- Lower bounds of other dynamic problems in the external memory.

1. union-find, need super-log query time

   if we want to batch up the updates?

2. priority queue:

   $\max\{\texttt{insert}, \texttt{delete}, \texttt{deletemin}\} \geq \frac{1}{B} \log_2 n$?

3. 1D range reporting: $\max\{\texttt{B} \cdot \texttt{updates}, \texttt{query}\} \geq \log n$?

The End

THANK YOU

Q and A