Lower Bounds for External Memory Dictionaries

Gerth Stølting Brodal   Rolf Fagerberg

BRICS

University of Aarhus

Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms

Baltimore, MD, USA, January 13, 2003

Dictionary

- **Queries**
  - – membership
  - – predecessor / successor
  - – range queries . . .
- **Updates**
  - – insertions
  - – deletions

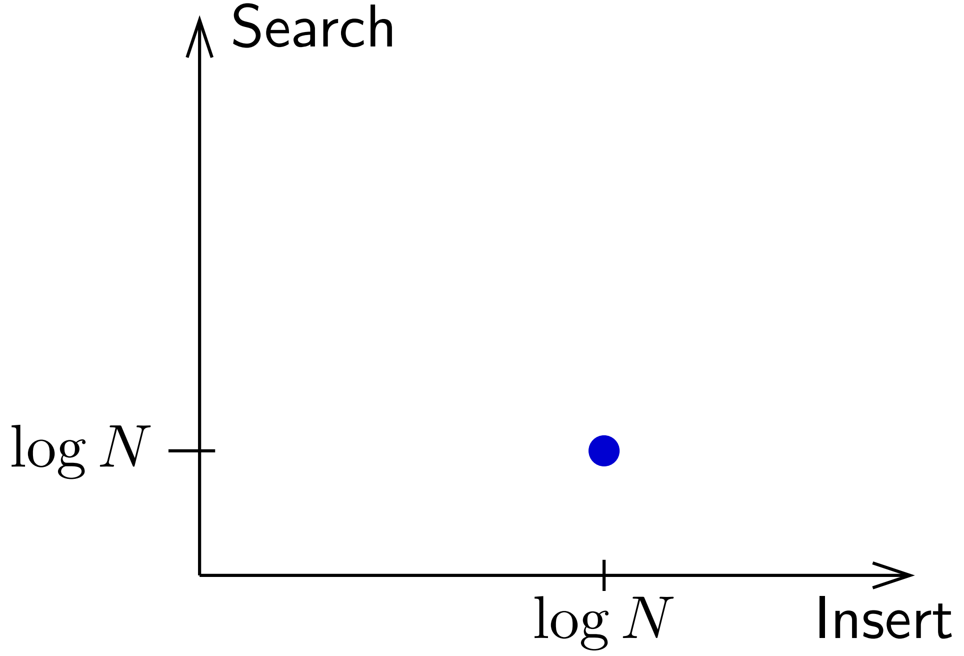
Dictionary

- **Queries**
  - – membership
  - – predecessor / successor
  - – range queries . . .
- **Updates**
  - – insertions
  - – deletions

This talk : Comparison based, membership, insertions

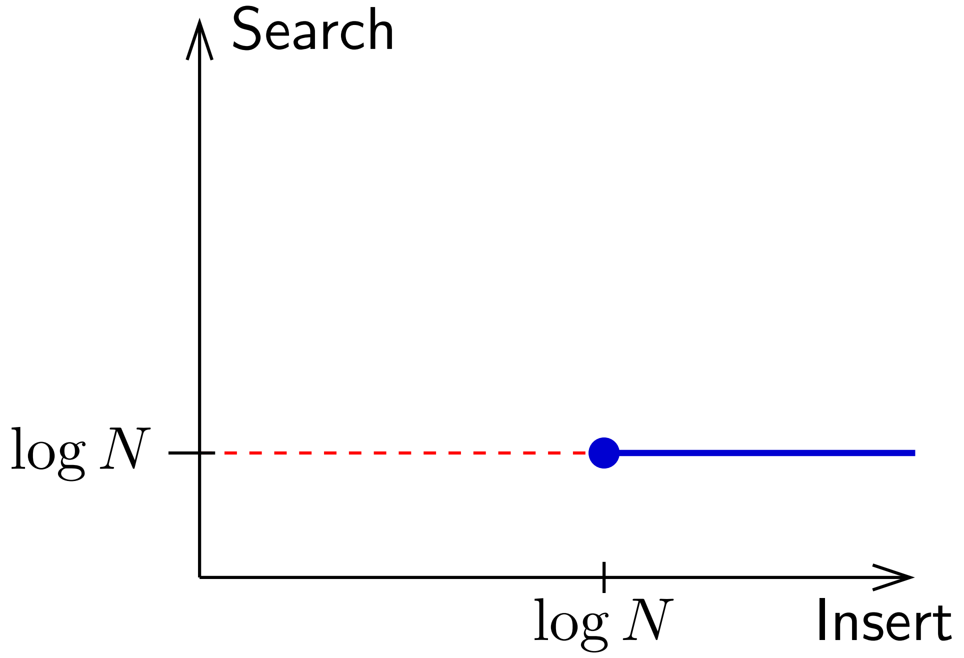
Dictionaries – Comparison Based

	Insert	**Search**
Balanced search trees	$O(\log N)$	$O(\log N)$



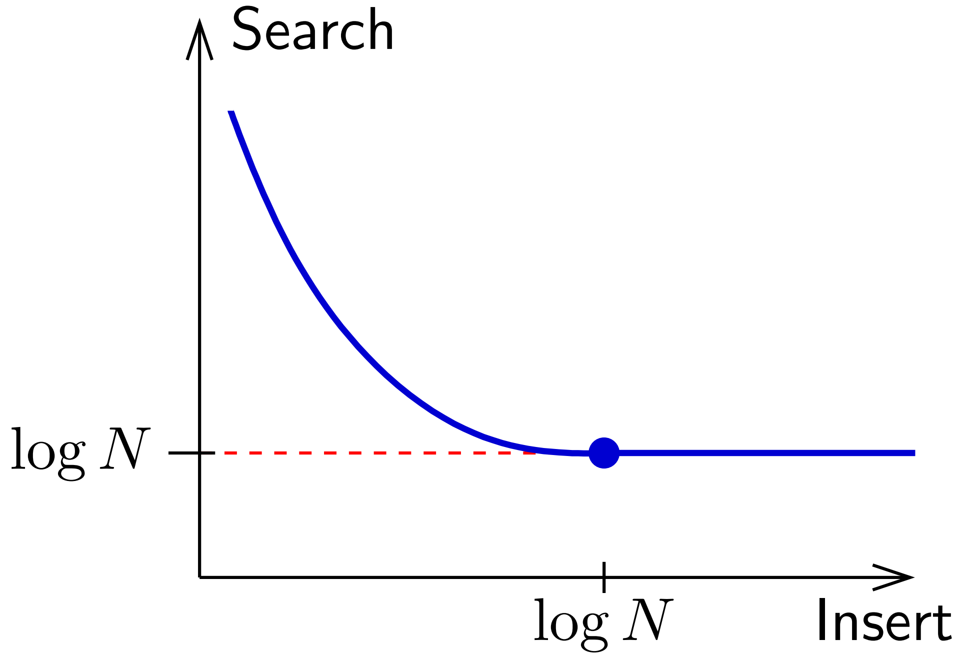
Dictionaries – Comparison Based

	Insert		Search
Balanced search trees	$O(\log N)$		$O(\log N)$
Adversary	∞	$\Rightarrow$	$\Omega(\log N)$



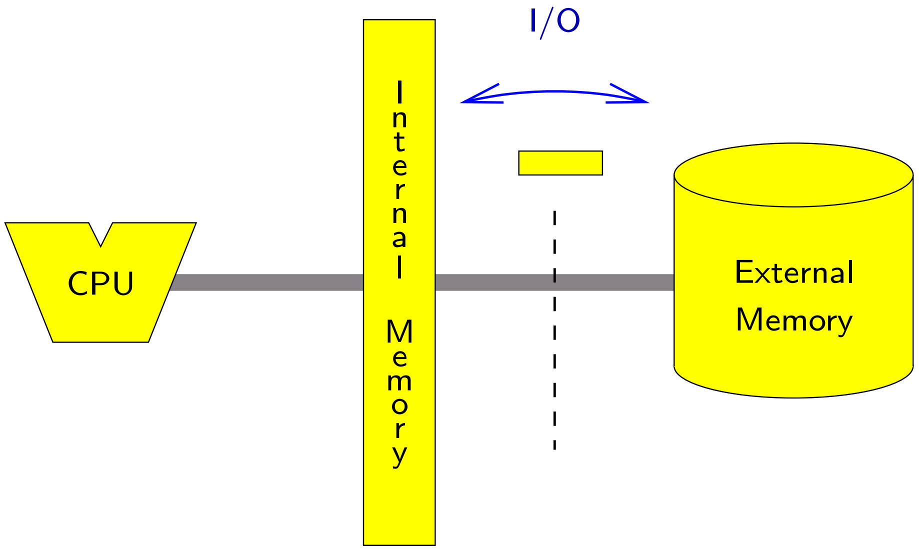
Dictionaries – Comparison Based

	Insert		Search
Balanced search trees	$O(\log N)$		$O(\log N)$
Adversary	∞	$\Rightarrow$	$\Omega(\log N)$
Borodin et al. 1982	$O(t)$	$\Rightarrow$	$N/2^{O(t)}$



External Memory Model

Aggarwal and Vitter 1988



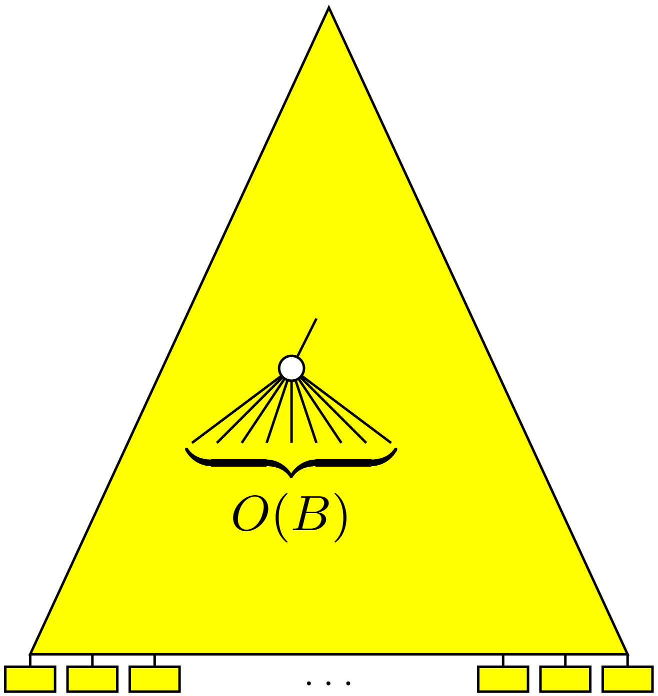
N = problem size
M = memory size
B = I/O block size

- One I/O moves B consecutive records from/to disk
- Cost : number of I/Os
- Elements can be copied and compared in internal memory

B-trees

– An External Memory Dictionary

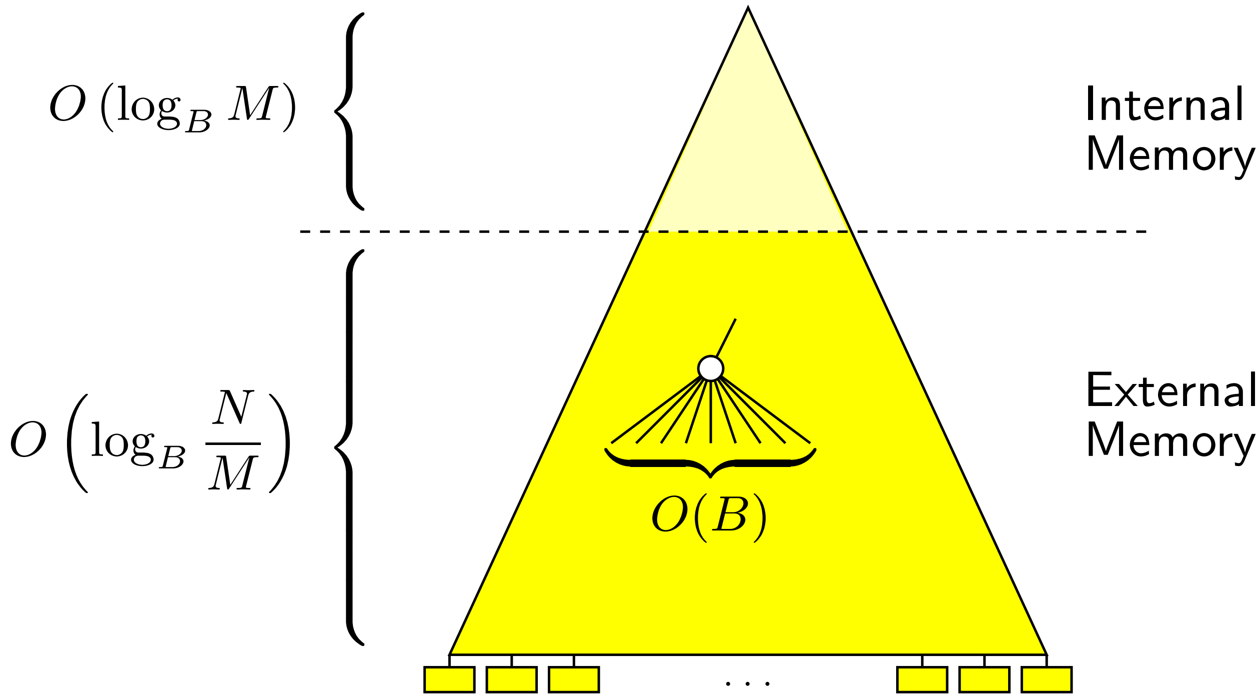
Bayer and McCreight 1972



B-trees

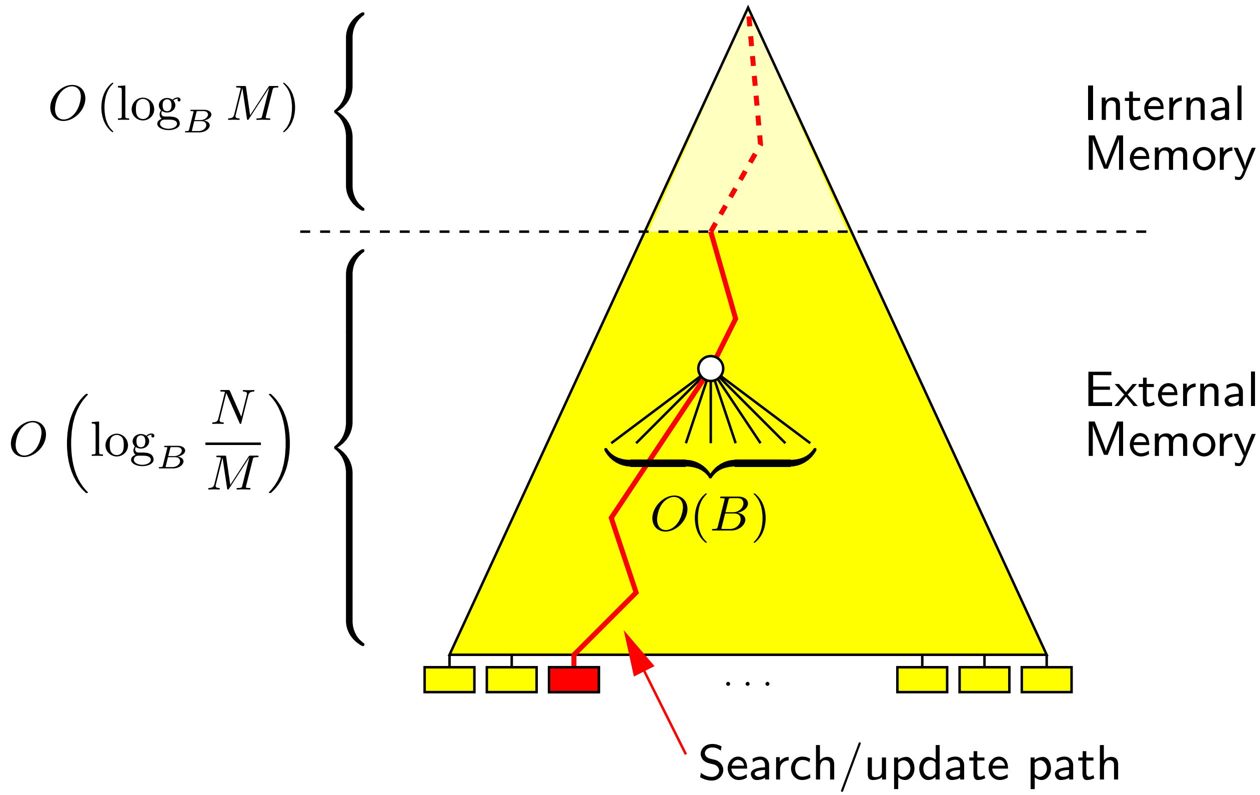
– An External Memory Dictionary

Bayer and McCreight 1972



B-trees
– An External Memory Dictionary

Bayer and McCreight 1972

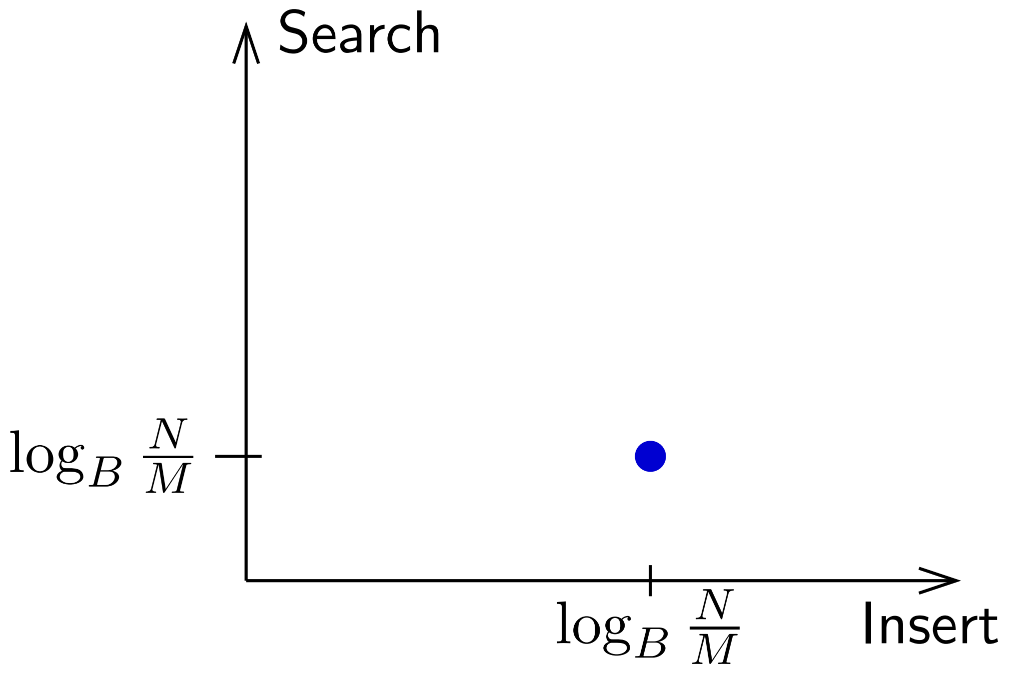


Insert
Membership

$\left.\begin{array}{l}\text{Insert}\\\text{Membership}\end{array}\right\}\ O\left(\log_B \frac{N}{M}\right)$ I/Os

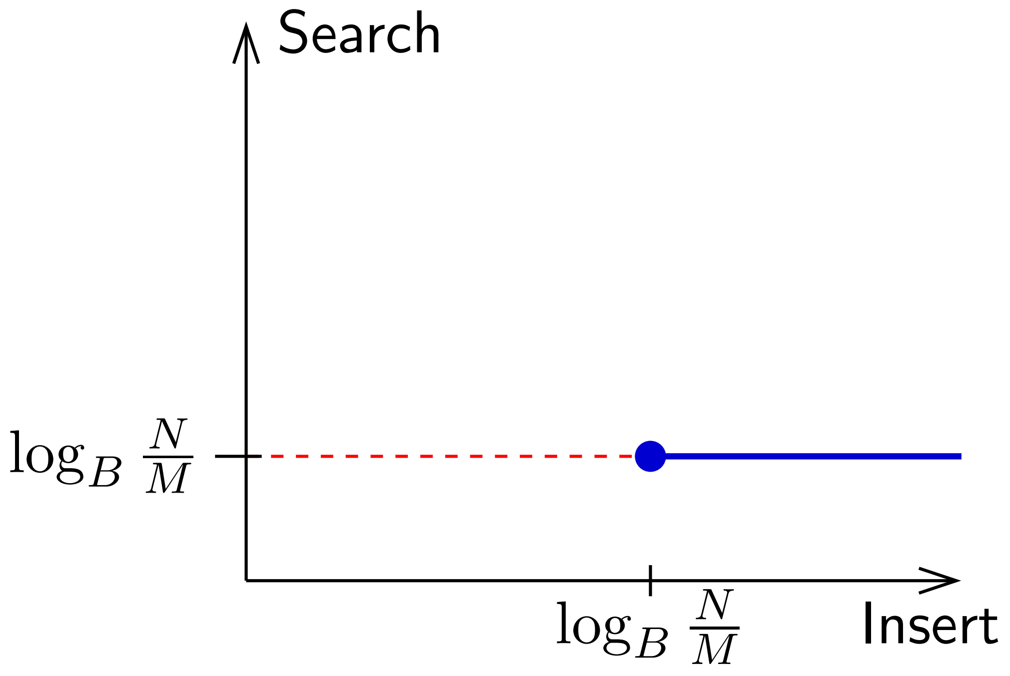
Dictionaries – External Memory

	Insert	Search
B-trees	$O(\log_B \frac{N}{M})$	$O(\log_B \frac{N}{M})$



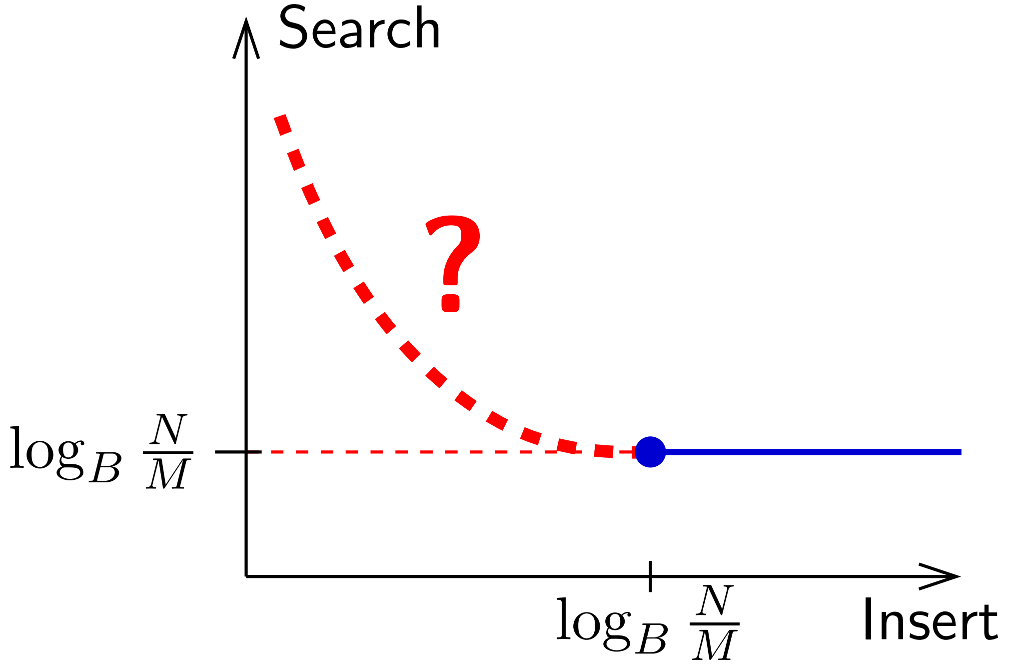
Dictionaries – External Memory

	Insert		Search
B-trees	$O(\log_B \frac{N}{M})$		$O(\log_B \frac{N}{M})$
Adversary	∞	$\Rightarrow$	$\Omega(\log_B \frac{N}{M})$

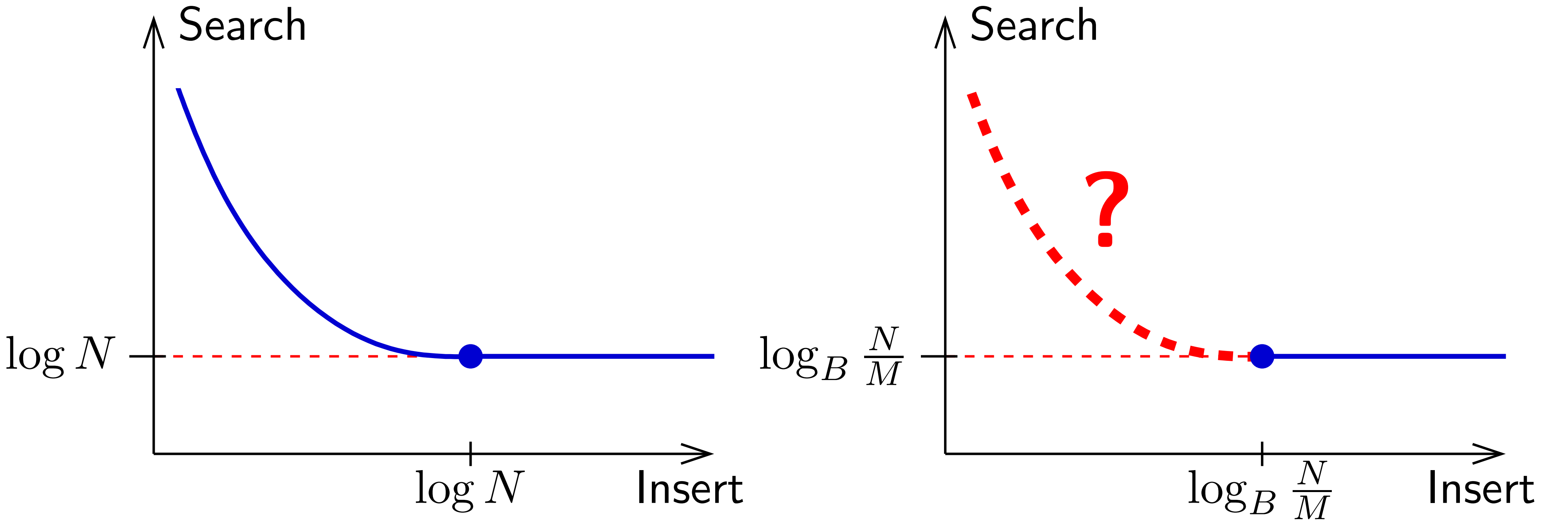


Dictionaries – External Memory

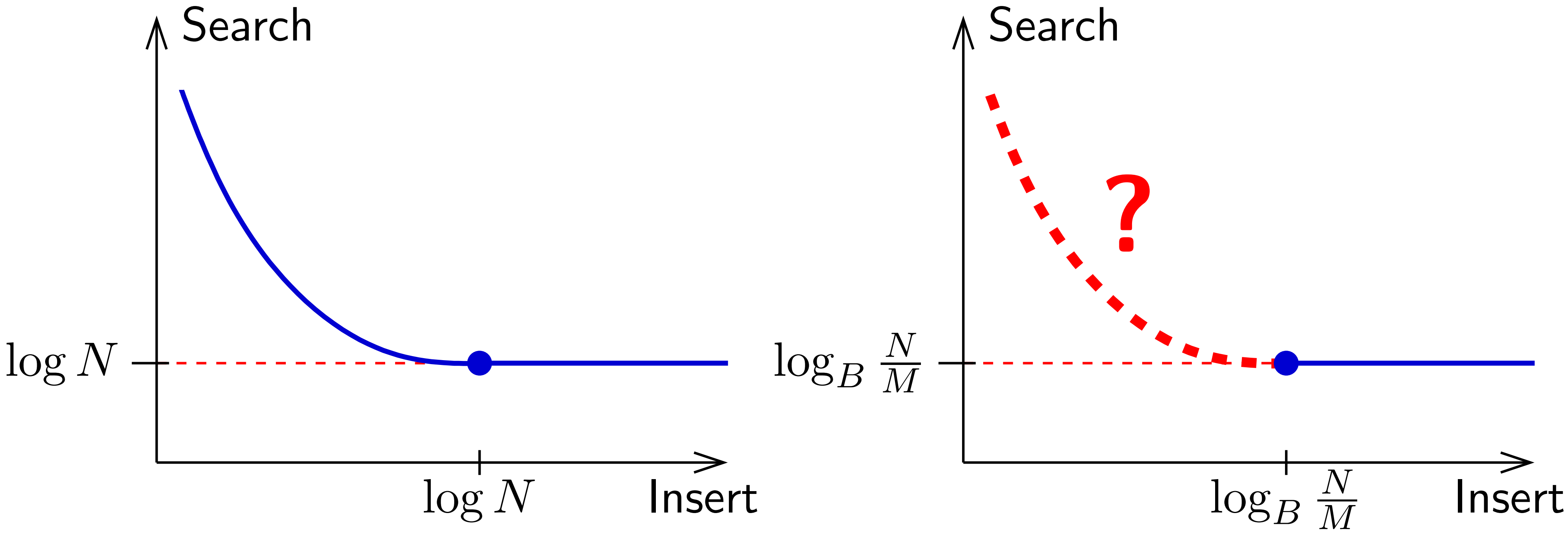
	Insert		**Search**
B-trees	$O(\log_B \frac{N}{M})$		$O(\log_B \frac{N}{M})$
Adversary	∞	$\Rightarrow$	$\Omega(\log_B \frac{N}{M})$



Comparisons vs. I/Os



Comparisons vs. I/Os

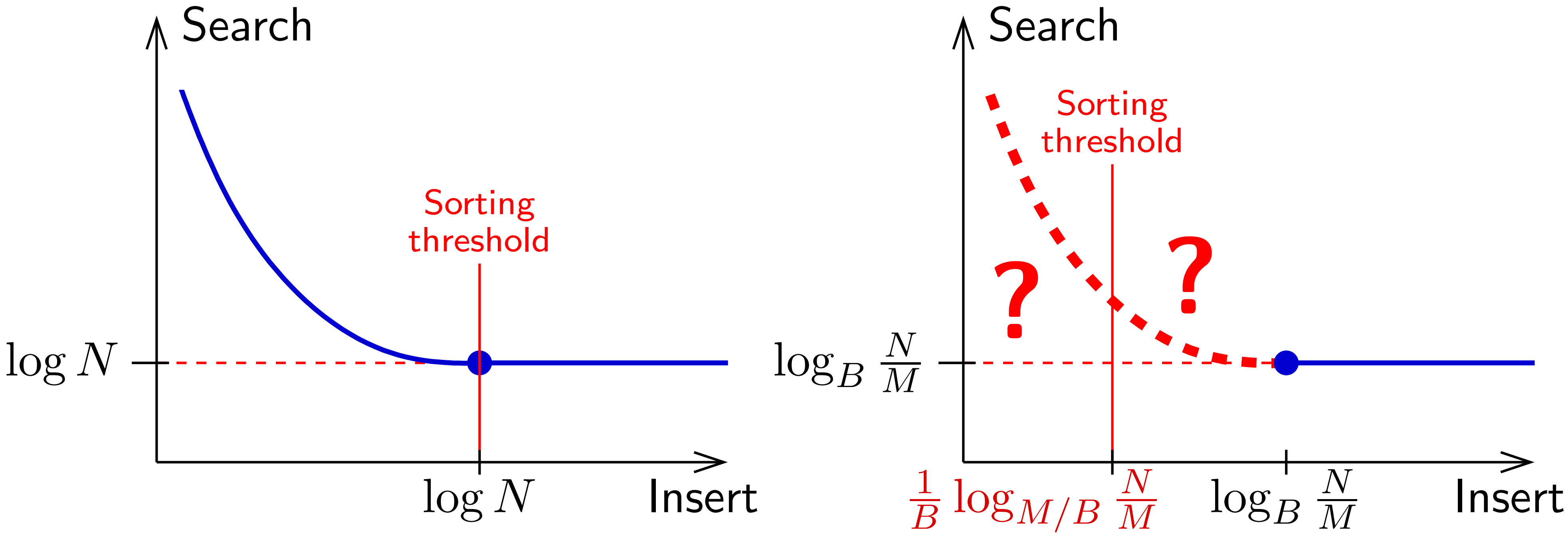


Sorting

Comparisons	$\Theta(N \log N)$
I/Os	$\Theta(\frac{N}{B} \log_{M/B} \frac{N}{M})$

Aggarwal and Vitter 1988

Comparisons vs. I/Os

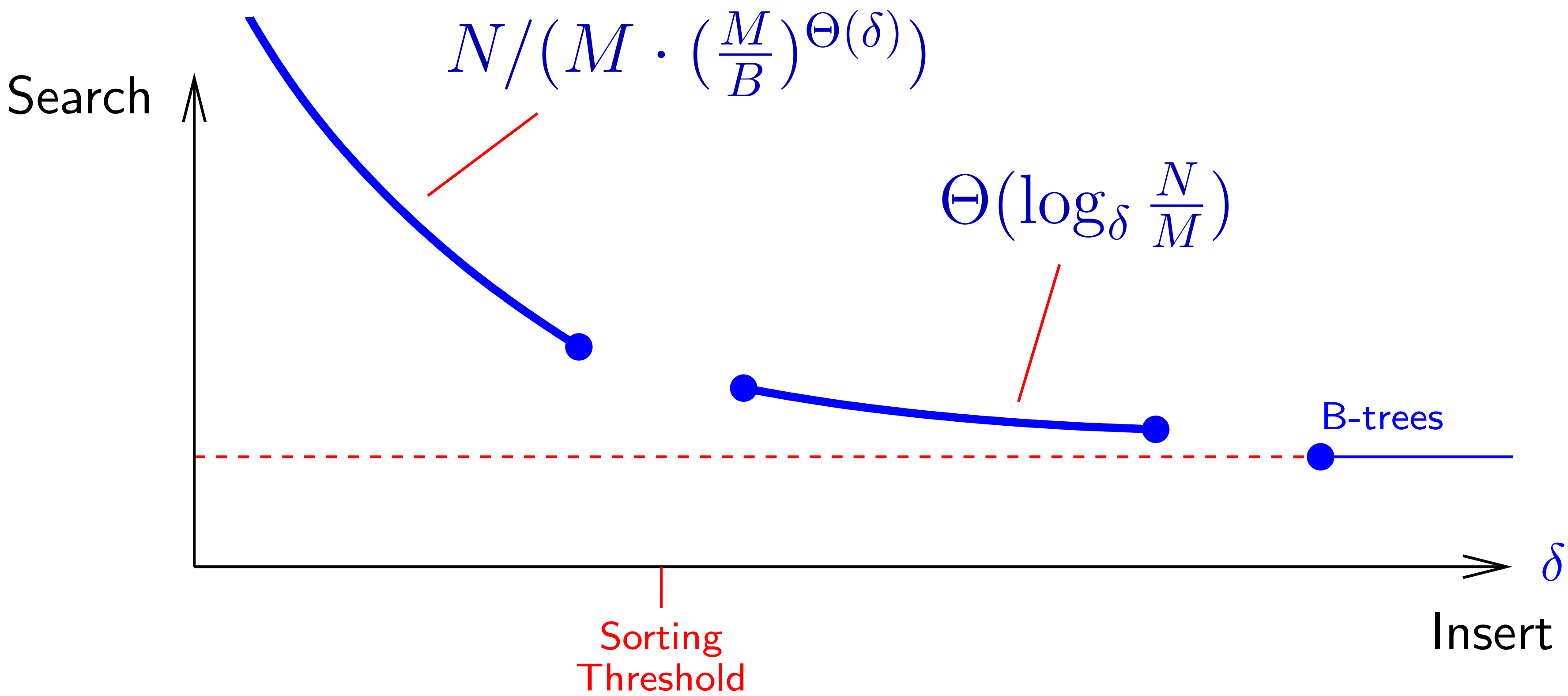


Sorting

Comparisons	$\Theta(N \log N)$
I/Os	$\Theta(\frac{N}{B} \log_{M/B} \frac{N}{M})$

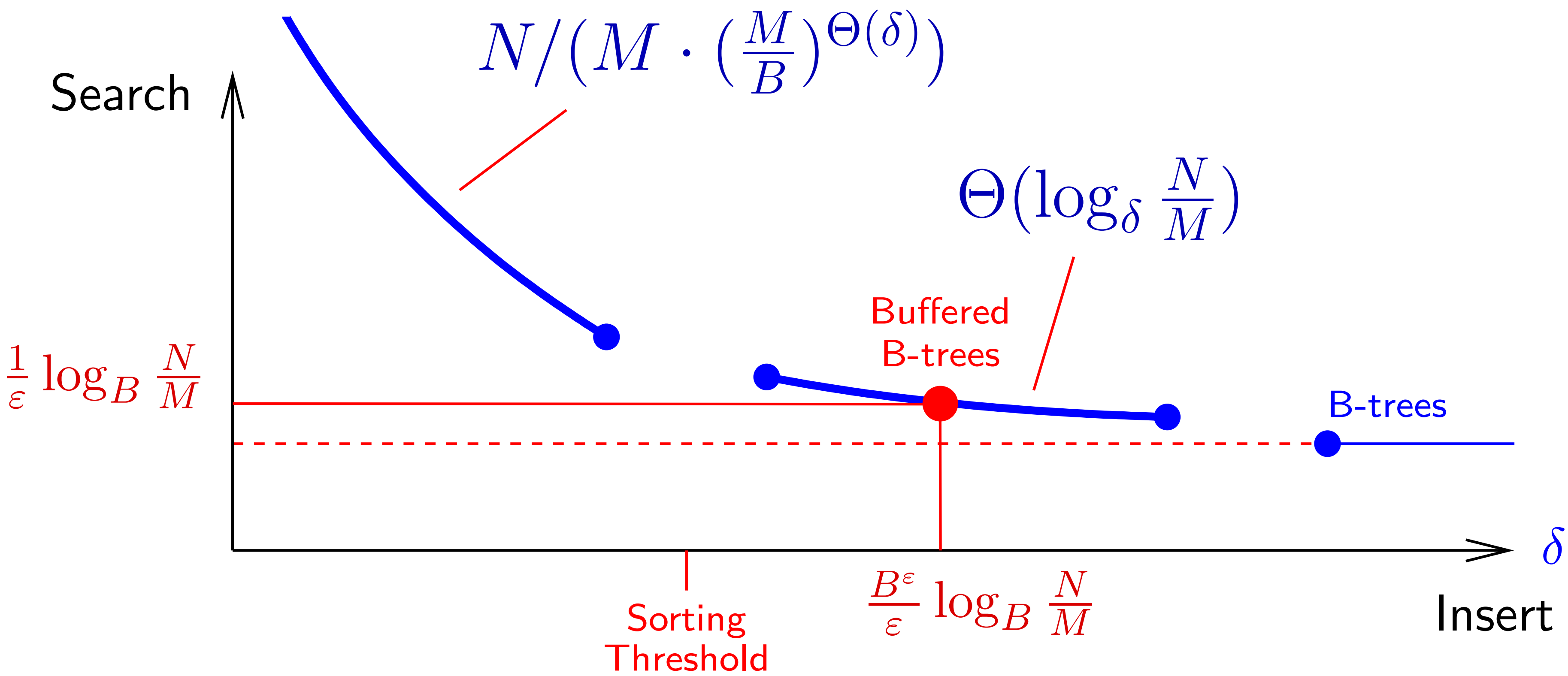
Aggarwal and Vitter 1988

Results



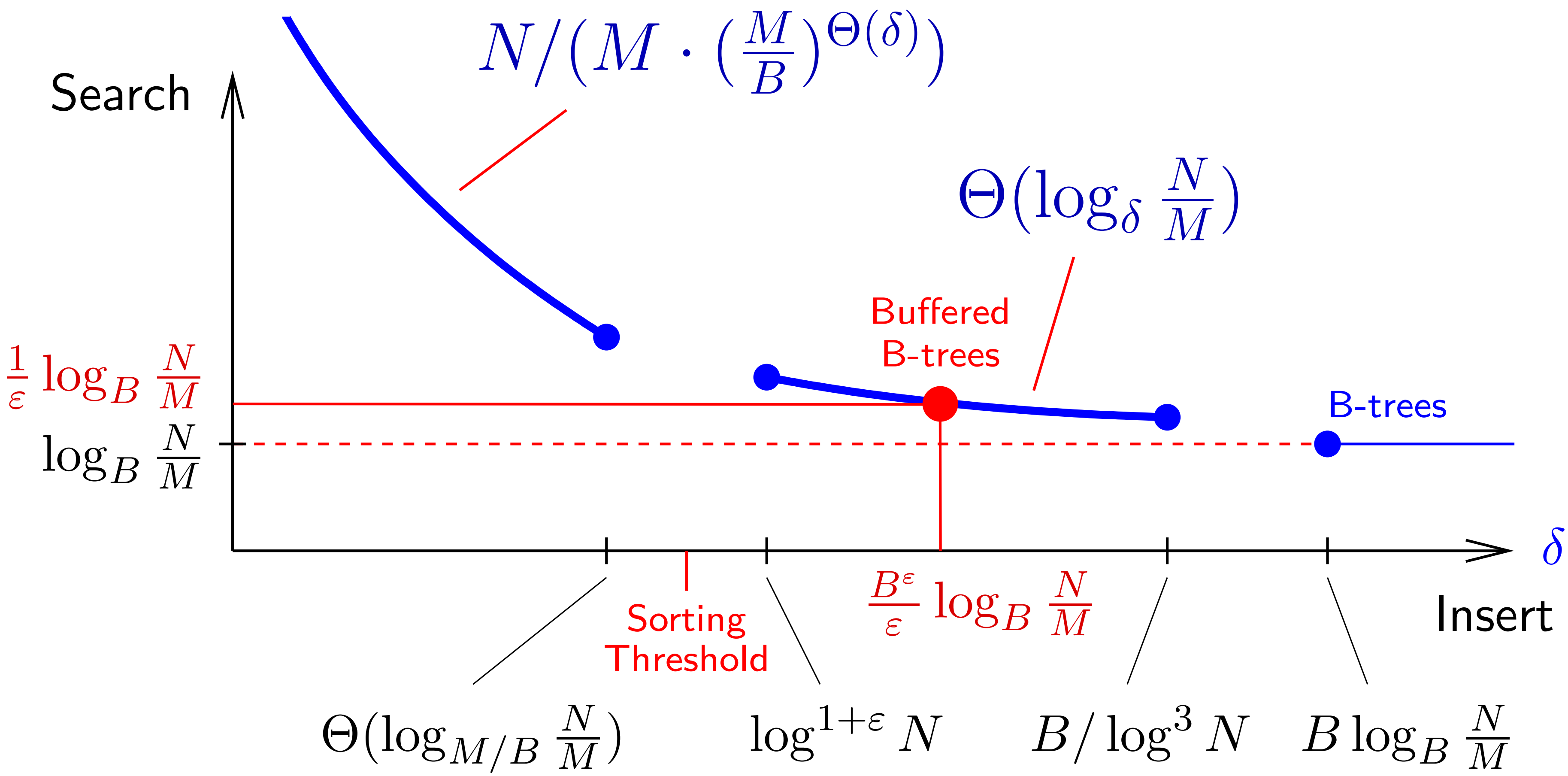
δ = number of I/Os for B insertions

Results



δ = number of I/Os for B insertions

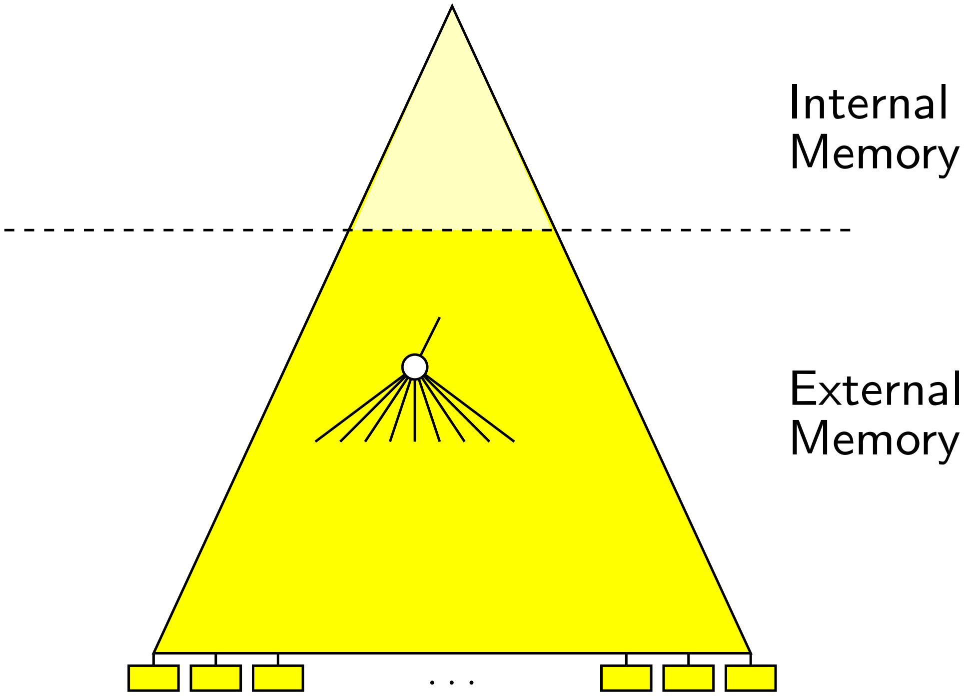
Results



δ = number of I/Os for B insertions

Buffered B-trees

– how to speedup B-tree updates by a factor $B^{1-\varepsilon}$



Searches

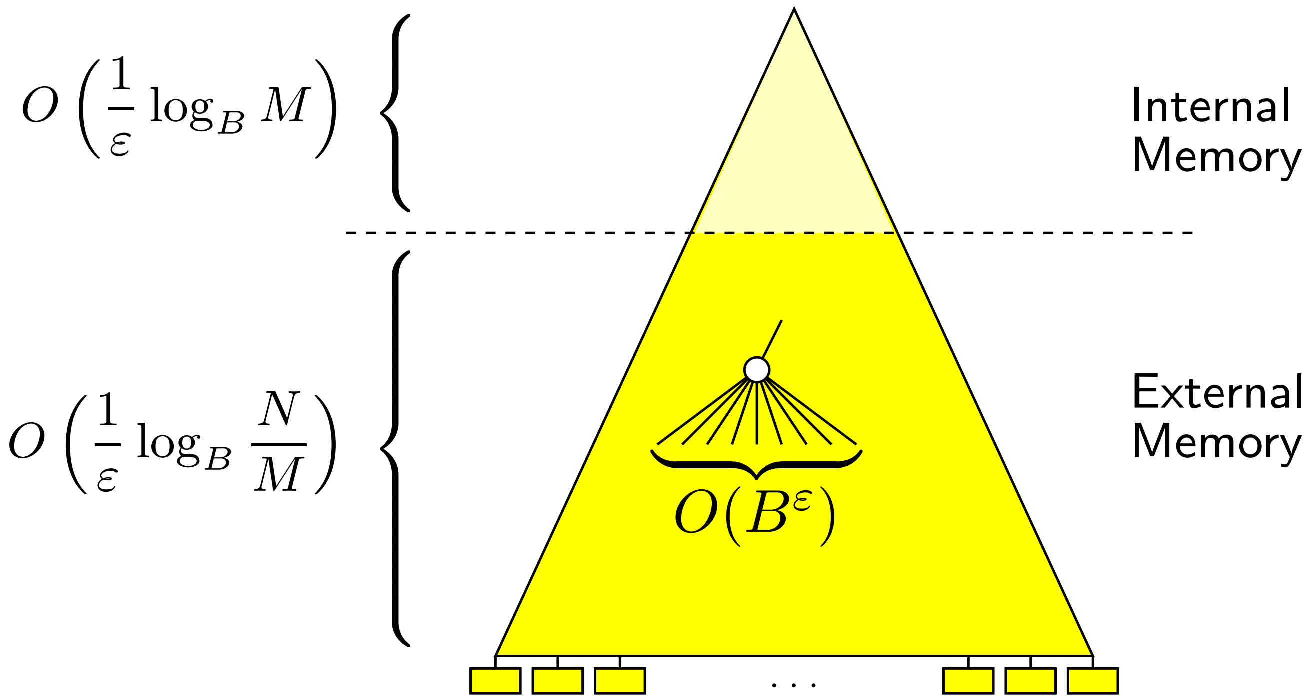
$$O(\frac{1}{\varepsilon}\log_B\frac{N}{M})$$

B insertions

$$O(\frac{B^\varepsilon}{\varepsilon}\log_B\frac{N}{M})$$

Buffered B-trees

– how to speedup B-tree updates by a factor $B^{1-\varepsilon}$



- B-tree with degree $\Theta(B^\varepsilon)$

Searches

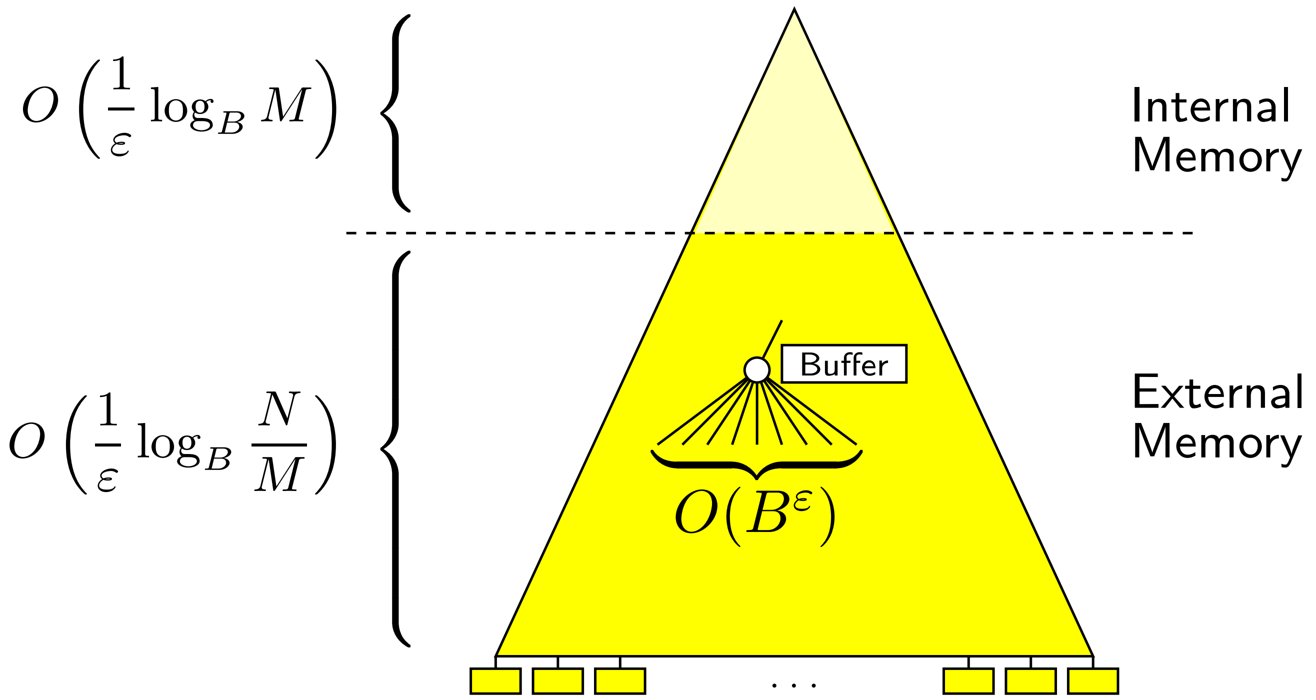
$O(\frac{1}{\varepsilon}\log_B \frac{N}{M})$

B insertions

$O(\frac{B^\varepsilon}{\varepsilon}\log_B \frac{N}{M})$

Buffered B-trees

– how to speedup B-tree updates by a factor $B^{1-\varepsilon}$



- B-tree with degree $\Theta(B^\varepsilon)$
- Buffers of $O(B)$ delayed insertions

Searches

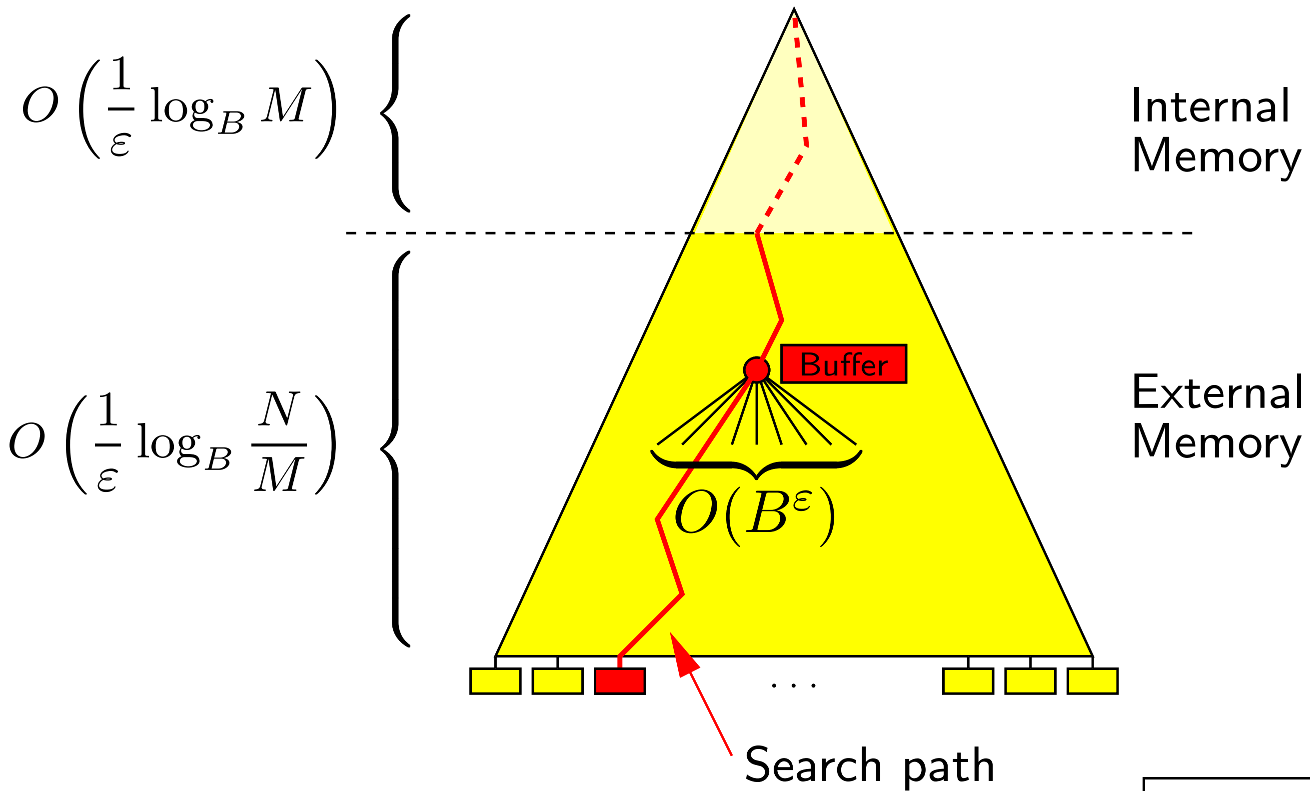
$O(\frac{1}{\varepsilon}\log_B \frac{N}{M})$

B insertions

$O(\frac{B^\varepsilon}{\varepsilon}\log_B \frac{N}{M})$

Buffered B-trees

– how to speedup B-tree updates by a factor $B^{1-\varepsilon}$



- B-tree with degree $\Theta(B^\varepsilon)$
- Buffers of $O(B)$ delayed insertions

Searches

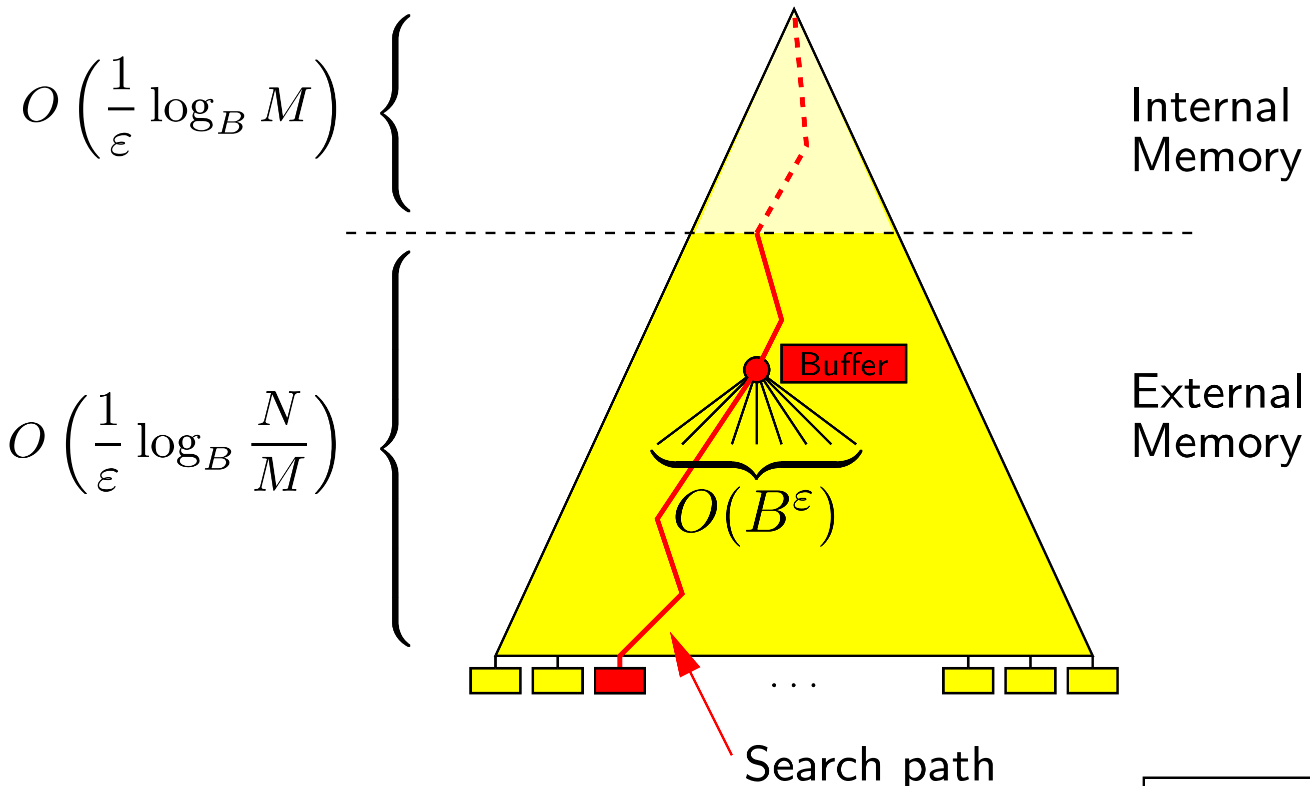
$O(\frac{1}{\varepsilon}\log_B \frac{N}{M})$

B insertions

$O(\frac{B^\varepsilon}{\varepsilon}\log_B \frac{N}{M})$

Buffered B-trees

– how to speedup B-tree updates by a factor $B^{1-\varepsilon}$



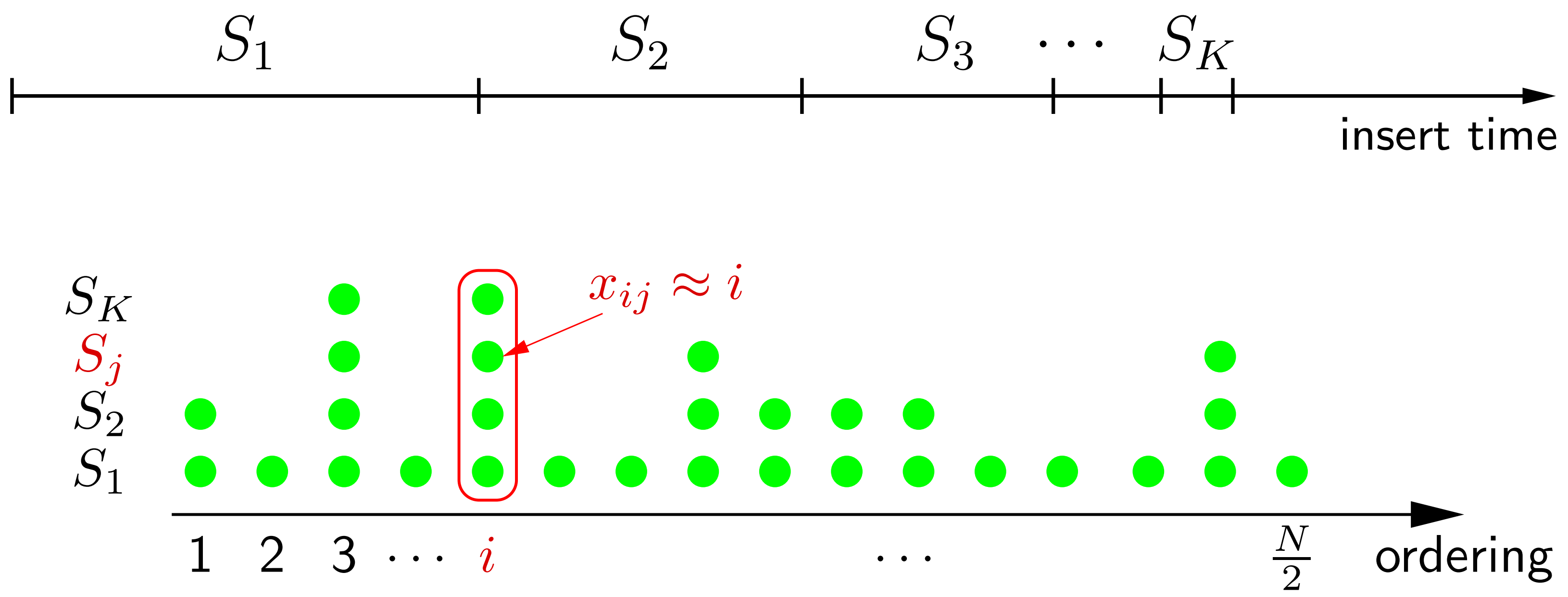
- B-tree with degree $\Theta(B^{\varepsilon})$
- Buffers of $O(B)$ delayed insertions
- On buffer overflow move $O(B^{1-\varepsilon})$ elements to a child with one I/O

Searches
$O(\frac{1}{\varepsilon}\log_B \frac{N}{M})$

B insertions
$O(\frac{B^{\varepsilon}}{\varepsilon}\log_B \frac{N}{M})$

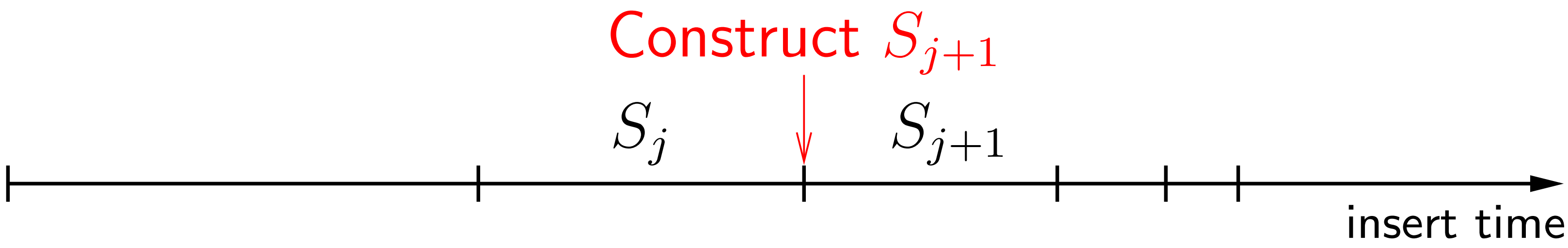
Lower Bound

– optimality of buffered B-trees



- Adversary online constructs $S_1, \ldots, S_K$
- Constructs i such that x_{ij} has not been in internal memory since S_j was inserted, for all $j = 1, \ldots, K$
- $x_{i1}, \ldots, x_{iK}$ form an antichain, i.e. search requires $\geq K$ I/Os

Lower Bound (Cont.)



- Let $\hat{I}$ be the indexes i where
  - $x_{ij} \in S_j$ but is not in internal memory after inserting S_j
  - $x_{i1}, \ldots, x_{i(j-1)}$ have not been read into internal memory by the $\delta \frac{|S_j|}{B}$ I/Os during the insertion of S_j
- Construct $I \subset \hat{I}$ such that all blocks in external memory contain $O(\frac{B}{\delta})$ elements x_{ij} where $i \in I$
  - Existence by randomized sampling with probability $O(1/\delta)$ and Chernoff bounds, provided $B/\delta = \Omega(\log N)$
- Let $x_{i(j+1)} \in S_j$ iff $i \in I$

$$K = \Theta(\log_\delta \frac{N}{M})$$

Lower Bound — Below Sorting

Insert		Search
$\frac{\delta}{B}$	$\Rightarrow$	$\frac{N}{M \cdot \left(\frac{M}{B}\right)^{\Theta(\delta)}}$

Lower Bound — Below Sorting

Insert		Search
$\dfrac{\delta}{B}$	$\Rightarrow$	$\dfrac{N}{M \cdot \left(\frac{M}{B}\right)^{\Theta(\delta)}}$

- W.l.o.g. memory and each block totally ordered after each I/O

$$N \log M + \delta \frac{N}{B} \left(B \log \frac{M}{B} \right) \quad \text{comparisons}$$

Insert in internal memory

Merging a block with internal memory

Lower Bound — Below Sorting

Insert		**Search**
$\frac{\delta}{B}$	$\Rightarrow$	$\frac{N}{M \cdot \left(\frac{M}{B}\right)^{\Theta(\delta)}}$

- W.l.o.g. memory and each block totally ordered after each I/O

$$N \log M + \delta \frac{N}{B} \left(B \log \frac{M}{B} \right) \quad \text{comparisons}$$

Insert in internal memory

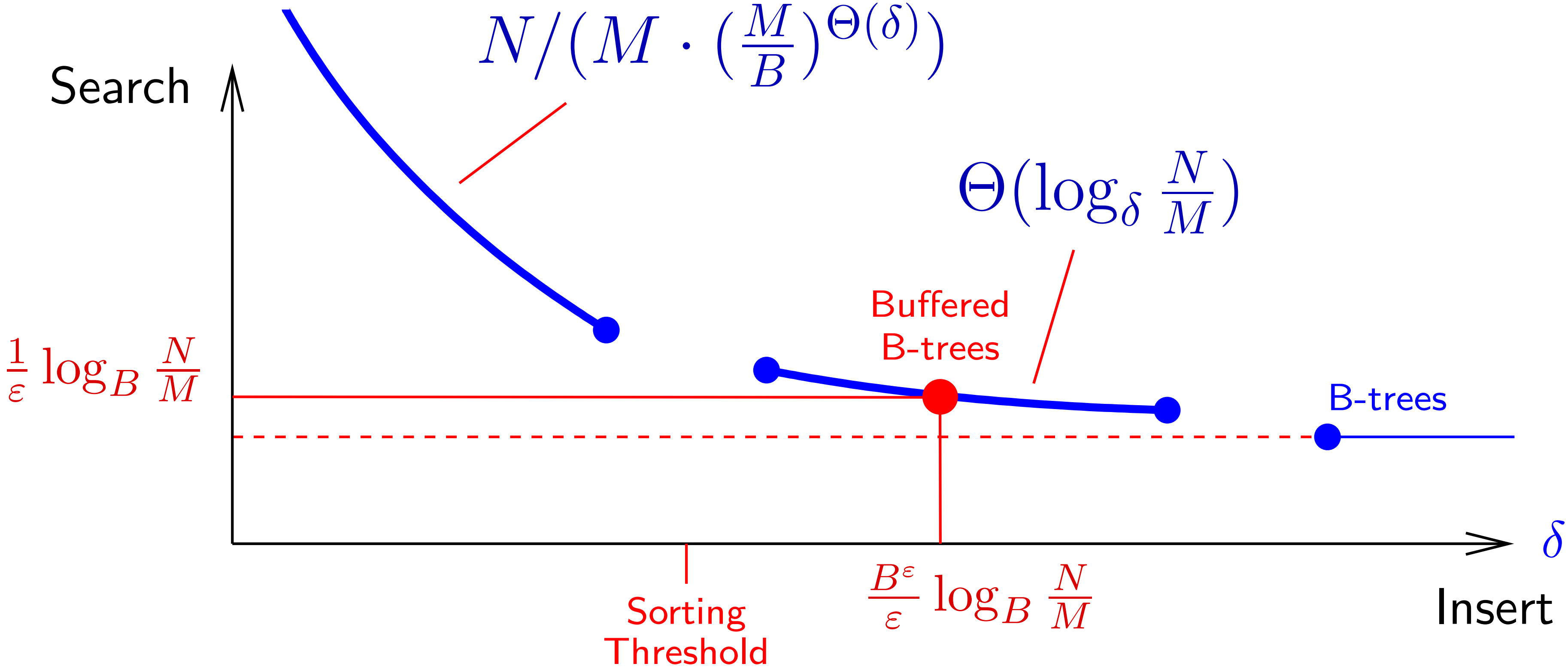
Merging a block with internal memory

- Antichain of size (Borodin et al. 1982 / Dillworth's lemma)

$$\frac{N}{2^{\log M + \delta \log \frac{M}{B}}} = \frac{N}{M \cdot \left(\frac{M}{B}\right)^{\delta}}$$

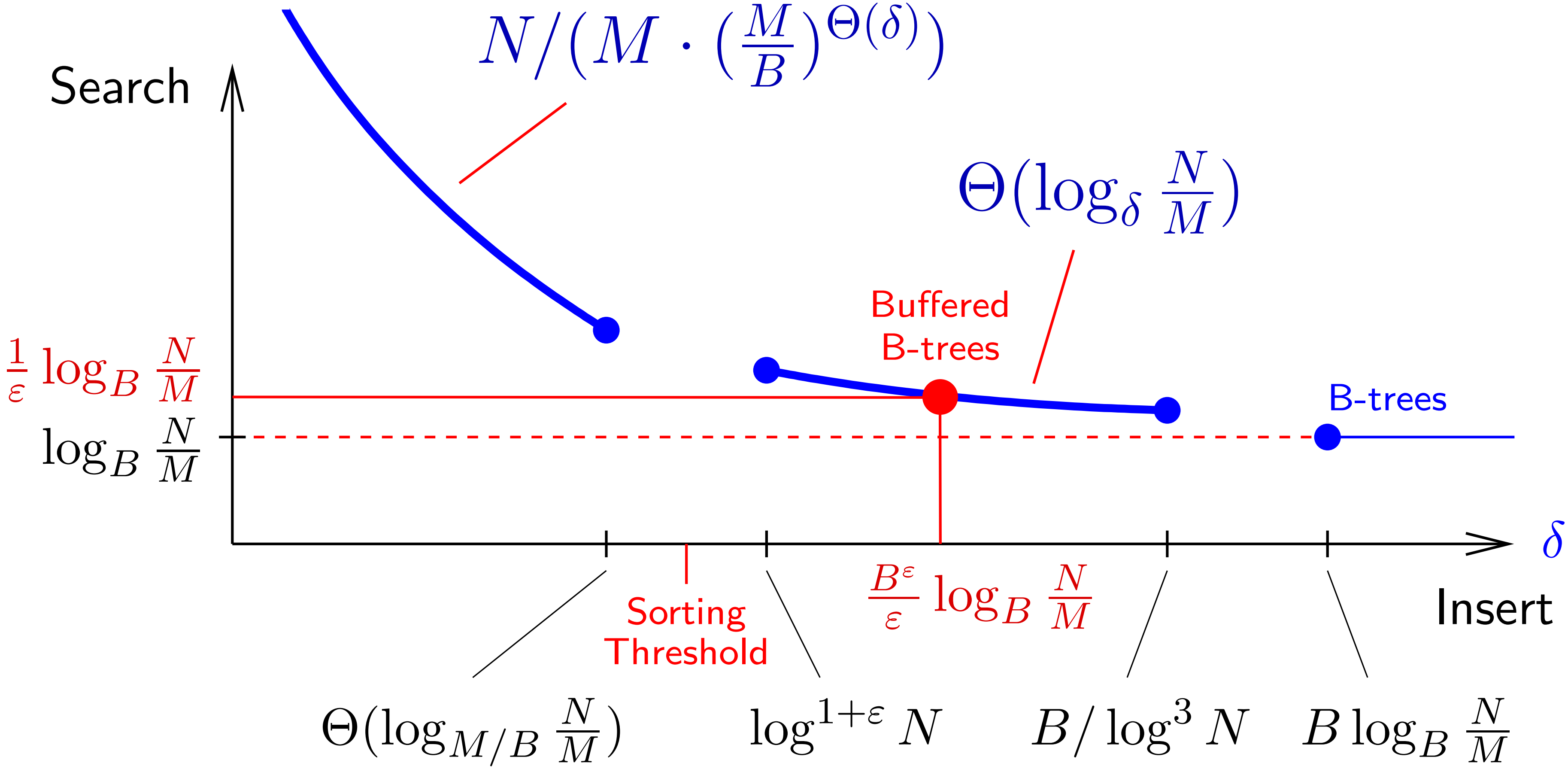
of which all elements except one are in distinct blocks

Conclusion



$N/(M \cdot (\frac{M}{B})^{\Theta(\delta)})$
$\Theta(\log_\delta \frac{N}{M})$
Search
Buffered
B-trees
$\frac{1}{\varepsilon} \log_B \frac{N}{M}$
B-trees
δ
Insert
Sorting
Threshold
$\frac{B^\varepsilon}{\varepsilon} \log_B \frac{N}{M}$

Conclusion



$N/(M \cdot (\frac{M}{B})^{\Theta(\delta)})$
Search
$\Theta(\log_\delta \frac{N}{M})$
Buffered
B-trees
$\frac{1}{\varepsilon} \log_B \frac{N}{M}$
$\log_B \frac{N}{M}$
B-trees
δ
Insert
Sorting
Threshold
$\frac{B^\varepsilon}{\varepsilon} \log_B \frac{N}{M}$
$\Theta(\log_{M/B} \frac{N}{M})$
$\log^{1+\varepsilon} N$
$B/\log^3 N$
$B \log_B \frac{N}{M}$