

Algorithms for Low-Power Management of On-Disk Data

Michael A. Bender



Power Consumption of Disks

Power consumption of disks

- Enterprise 80 to 160 GB disk runs at 4W (idle power).
- Enterprise 1-2 TB disk runs at 8W (idle power).

Data centers/server farms use 80-160 GB disks

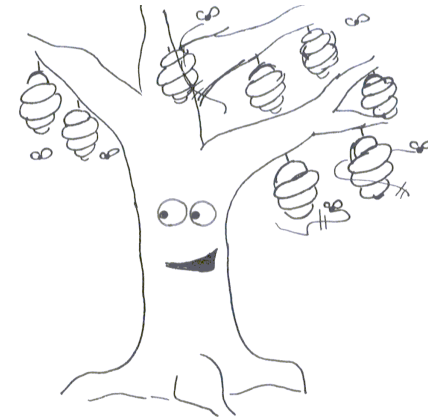
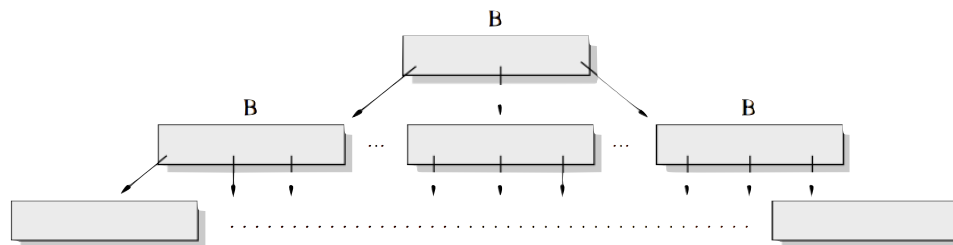
- Use many small-capacity disks, not large ones.

Using large disks may save factor of ~12

- Other considerations modify this factor
 - ▶ e.g., CPUs necessary to drive disks, scale-out infrastructure, cooling, etc.
- Why not use larger disks? One reason is performance.

Disk Performance

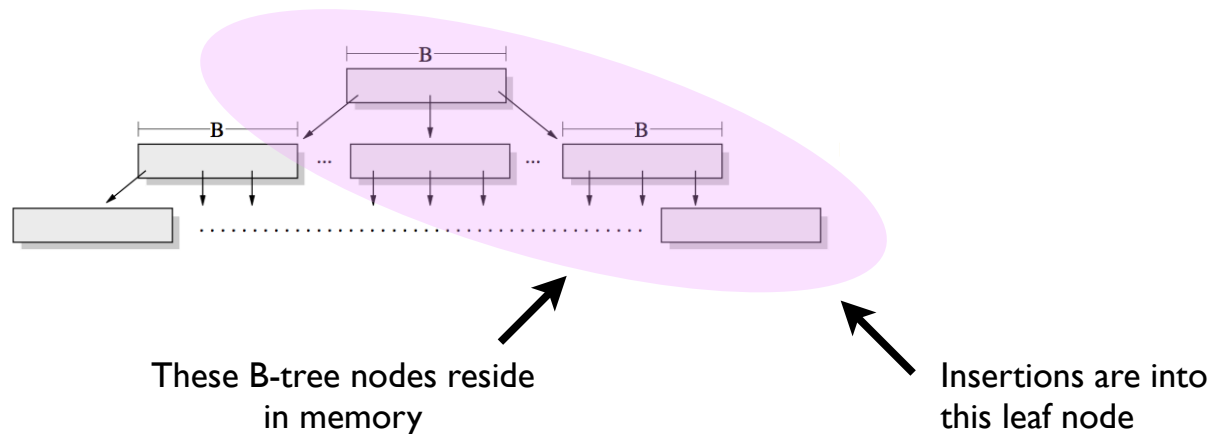
We need to examine how data is accessed.
Databases/file systems use B-tree variants.



The algorithmic characteristics of B-trees determine the system design and power consumption.

B-trees are Fast at Sequential Inserts

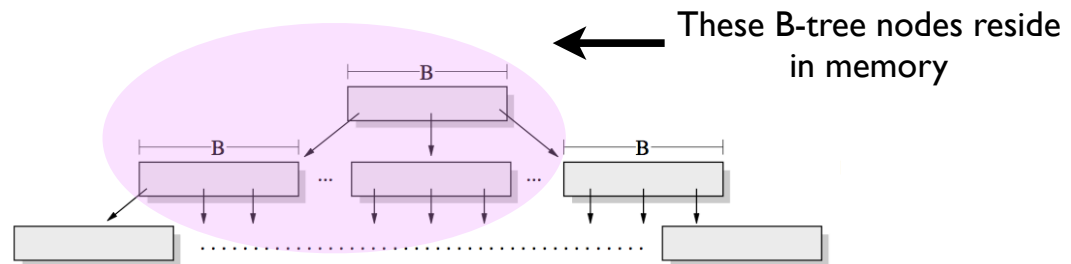
Sequential inserts in B-trees have near-optimal data locality



- One disk I/O per leaf (which contains many inserts).
- Sequential disk I/O.
- Performance is disk-bandwidth limited.

B-Trees Are Slow at Ad Hoc Inserts

High entropy inserts (e.g., random) in B-trees have poor data locality



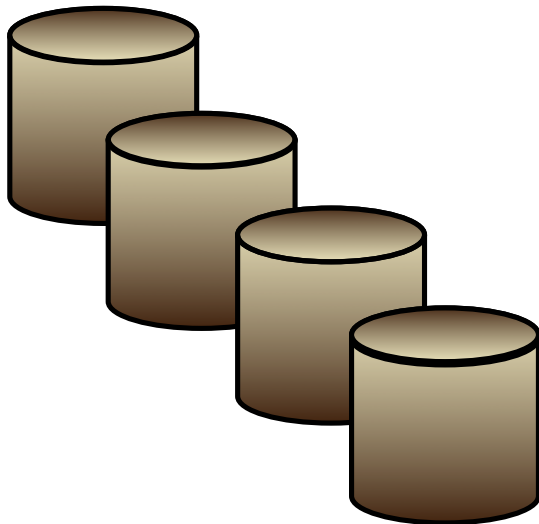
- Most nodes are not in main memory.
- Most key insertions require a random disk I/O.
- Performance is disk-seek limited.

Why Not Use Larger-Capacity Disks?

B-trees need disk systems with enough

- capacity
- bandwidth
- random I/Os ← Hard for disks

Achieve random I/Os by increasing number of disks



Fractal Tree™ Technology

Fractal Trees deliver better performance

- Drop-in replacement for B-tree
- **Bandwidth limited, not seek limited**
 - ▶ Scales as square root of capacity
- Developed at Stony Brook, Rutgers, MIT
[Bender, Farach-Colton, Fineman, Fogel, Kuszmaul, Nelson '07]
- Commercialized by Tokutek for MySQL databases

Large-capacity disks can be used

- **Order of magnitude energy savings per GB**

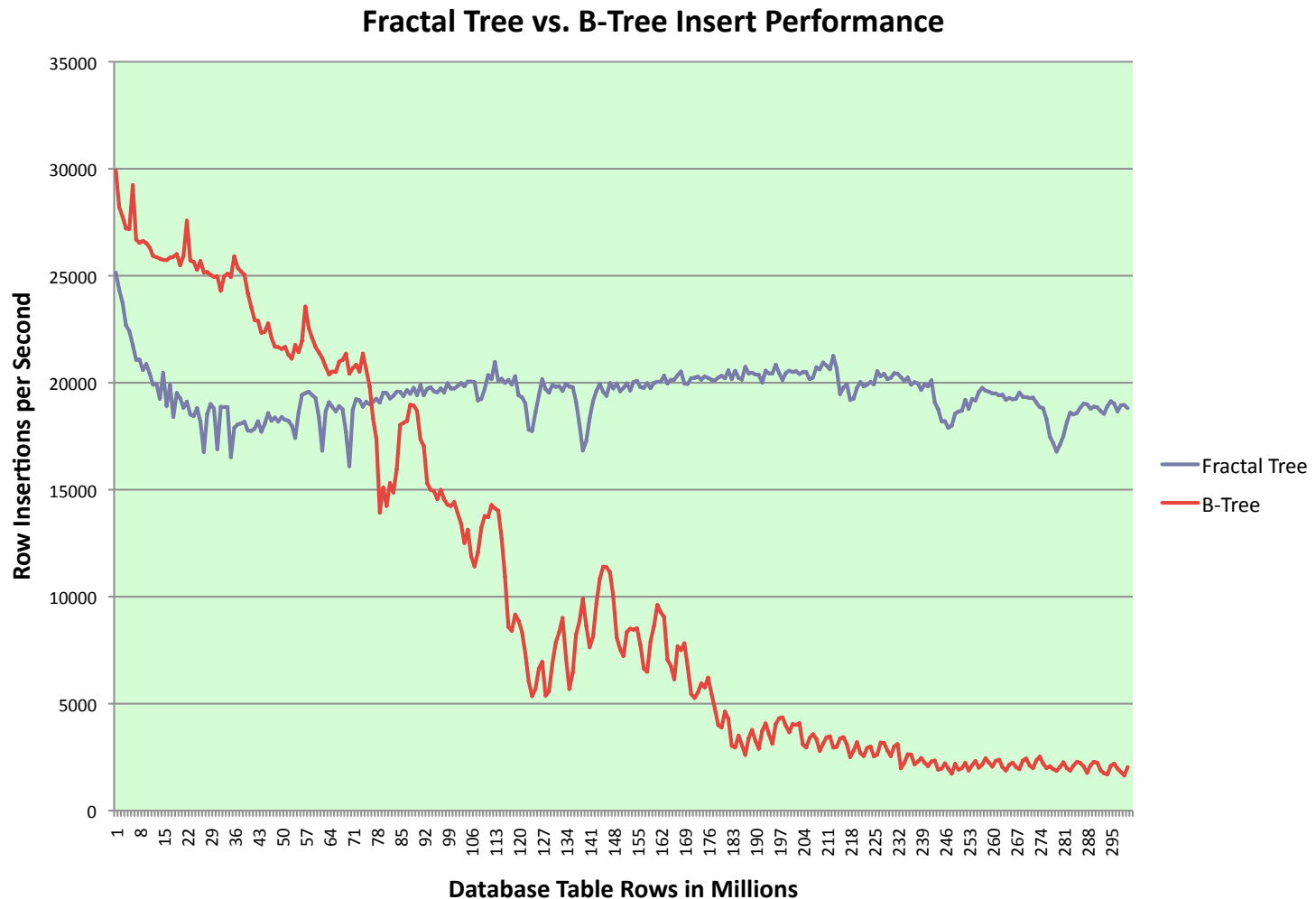
Fractal Tree Theoretical Model

	B-tree	Fractal Tree
Insert	$O(\log_B N) = O\left(\frac{\log N}{\log B}\right)$	$O\left(\frac{\log N}{B}\right)$

Example: $N=1$ billion, $B=4096$

- 1 billion 128-byte rows (128 gigabytes)
 - ▶ $\log_2(1 \text{ billion}) = 30$
- Half-megabyte blocks that hold 4096 rows each
 - ▶ $\log_2(4096) = 12$
- B-trees require $\frac{\log N}{\log B} = 30/12 = 3$ disk seeks in worst case
- Fractal Trees require $\frac{\log N}{B} = 30/4096 = 0.007$ disk seeks in worst case

Fractal Tree vs. B-tree Insertions



Inserts scale with bandwidth. No memory cliff.

Diving Deeper...

How many CPUs in a well balanced system?

- Depends on usage model.
- For I/O bound systems, one CPU can drive many disks.
- For bandwidth bound systems, a small number of drives can soak a CPU.

A Fractal Tree can use more CPUs per disk

- So maybe the previous comparison isn't fair to B-trees...
- or maybe it isn't fair to Fractal Trees.

Other Power Metrics

What about insertions/joule?

- (200 seeks per sec)/(10 watts=joules/sec)
= 20 seeks per joule
- We already calculated seeks per insert
 - ▶ E.g., 3 versus 0.007

	B-tree	Fractal Tree
Insert	$O(\log_B N) = O\left(\frac{\log N}{\log B}\right)$	$O\left(\frac{\log N}{B}\right)$

Fractal Trees provide two orders of magnitude improvement in insertions per joule

- Versus one order of magnitude in Watts/GB

Algorithms that Ride the Technology Curve

Where the power goes in disks

- electronics -- used to be larger, now small
- drag -- superlinear in RPM
- seek cost -- superlinear in access time

Power cost

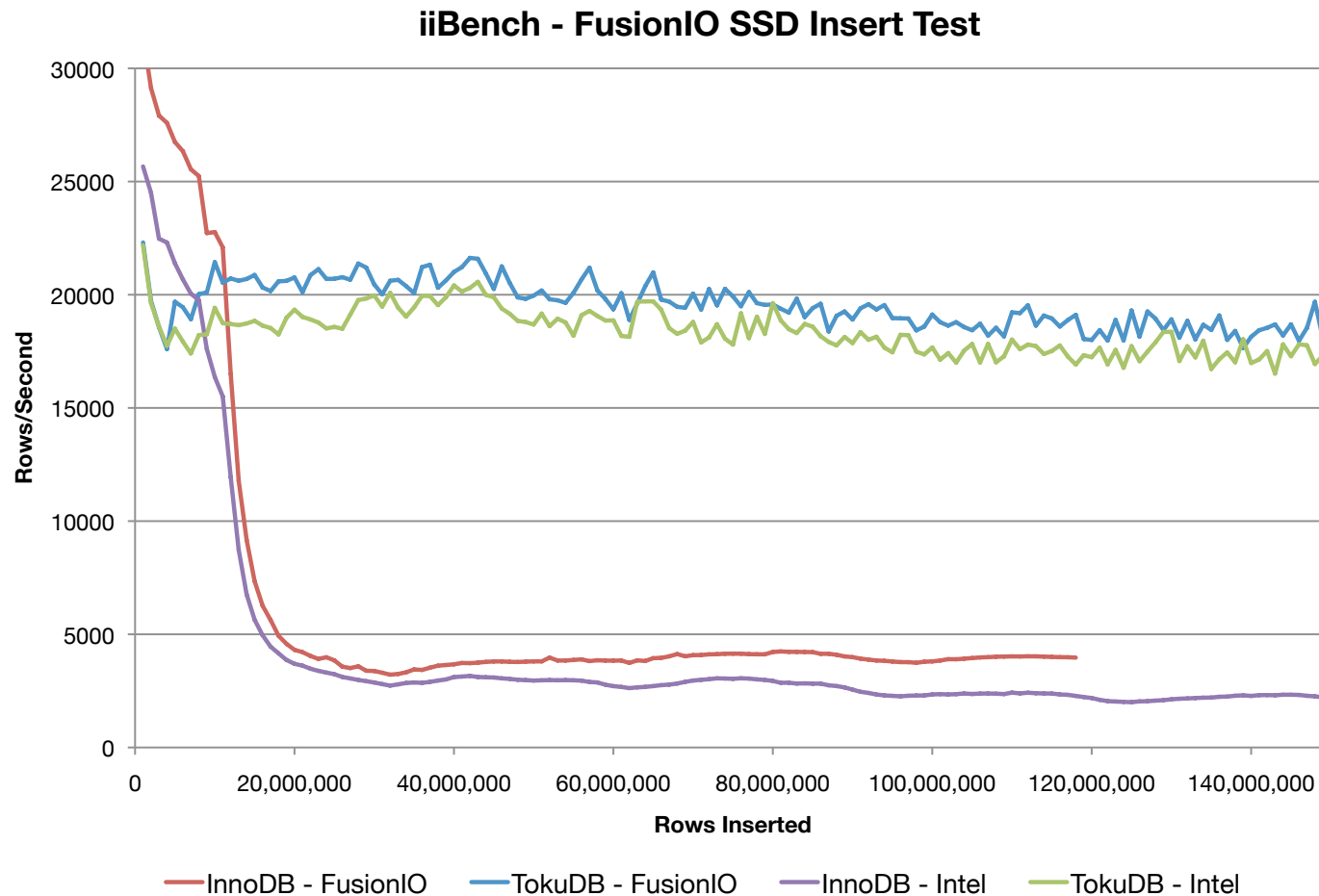
- Random I/O is expensive. Hardly improving over time.
- Bandwidth and capacity are cheap. They grow over time.
 - ▶ Bandwidth grows as square root of capacity.

Consequence

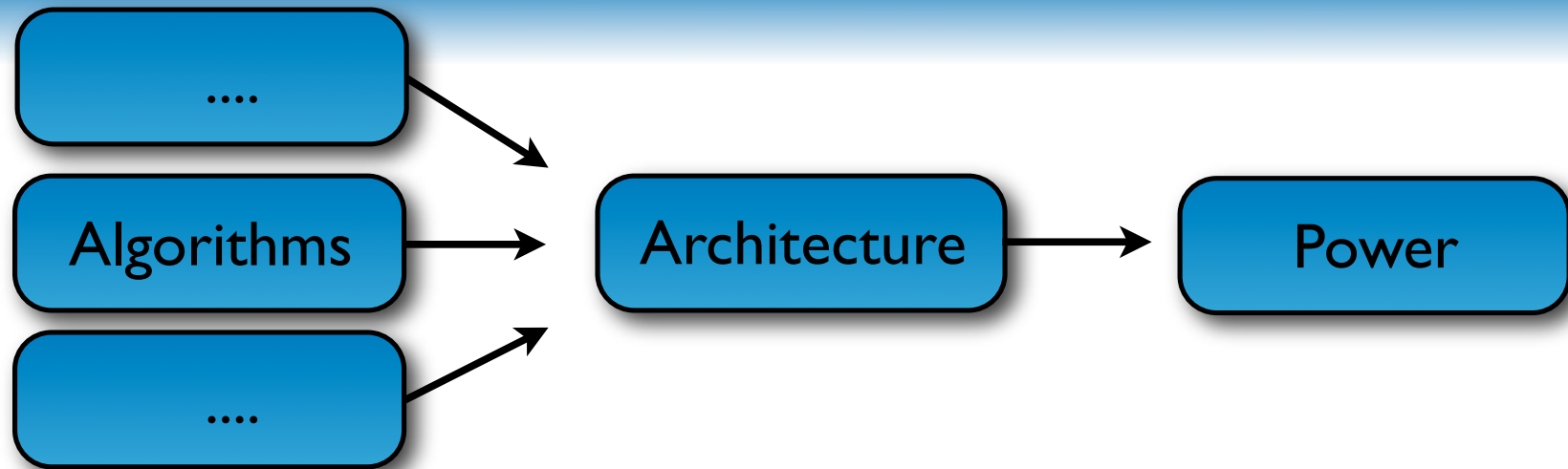
- B-trees remain slow because random I/O is expensive.
- We want data structures (such as Fractal Trees) that ride the technology curve.

SSDs Introduce Other Energy Issues

(But SSDs don't save B-trees)



Power-Efficient Data Centers



Energy-efficient data centers/server farms could have many slow cores and large-capacity disks

- Most have the reverse (server CPUs, small disks)

Building these requires foundational research in

- I/O-efficient/on-disk/concurrent data structures
- Parallel computing
- Online thread scheduling