Grim hardware realities of functional programming

Karel Čížek
@kaja47
funkcionalne.cz

λ → Si

“We are all wired into a survival trip now. No more of the speed that fueled that 60's. That was the fatal flaw in Tim Leary's trip. He crashed around America selling "consciousness expansion" without ever giving a thought to the grim meat-hook realities that were lying in wait for all the people who took him seriously... All those pathetically eager acid freaks who thought they could buy Peace and Understanding for three bucks a hit. But their loss and failure is ours too. What Leary took down with him was the central illusion of a whole life-style that he helped create... a generation of permanent cripples, failed seekers, who never understood the essential old-mystic fallacy of the Acid Culture: the desperate assumption that somebody... or at least some force – is tending the light at the end of the tunnel.”

HST

Faustian bargain

What's possible?

<rant>
(15/15)
counting cycles
</rant>

Linear vs Binary Search

https://schani.wordpress.com/2010/04/30/linear-vs-binary-search/

Command-line tools can be 235x faster than your Hadoop cluster

http://aadrake.com/command-line-tools-can-be-235x-faster-than-your-hadoop-cluster.html

Sorting out graph processing

https://github.com/frankmcsherry/blog/blob/master/posts/2015-08-15.md

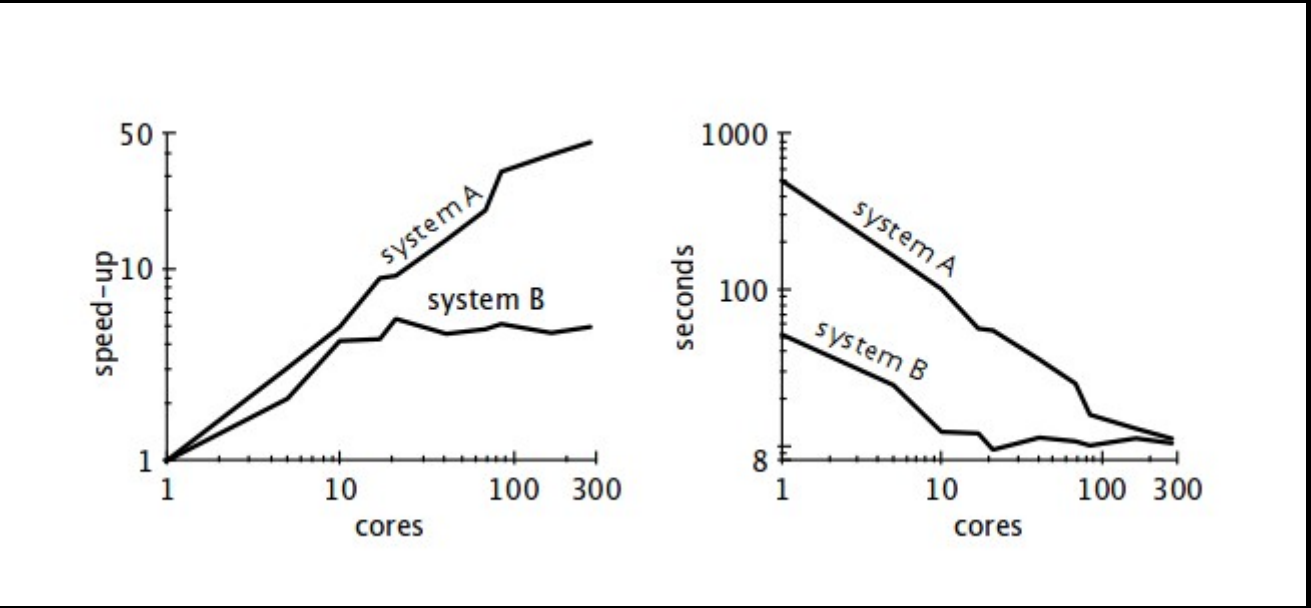
Scalability! But at what COST?

http://www.frankmcsherry.org/assets/COST.pdf
http://www.frankmcsherry.org/graph/scalability/cost/2015/01/15/COST.html

The emperor's new clothes: distributed machine learning

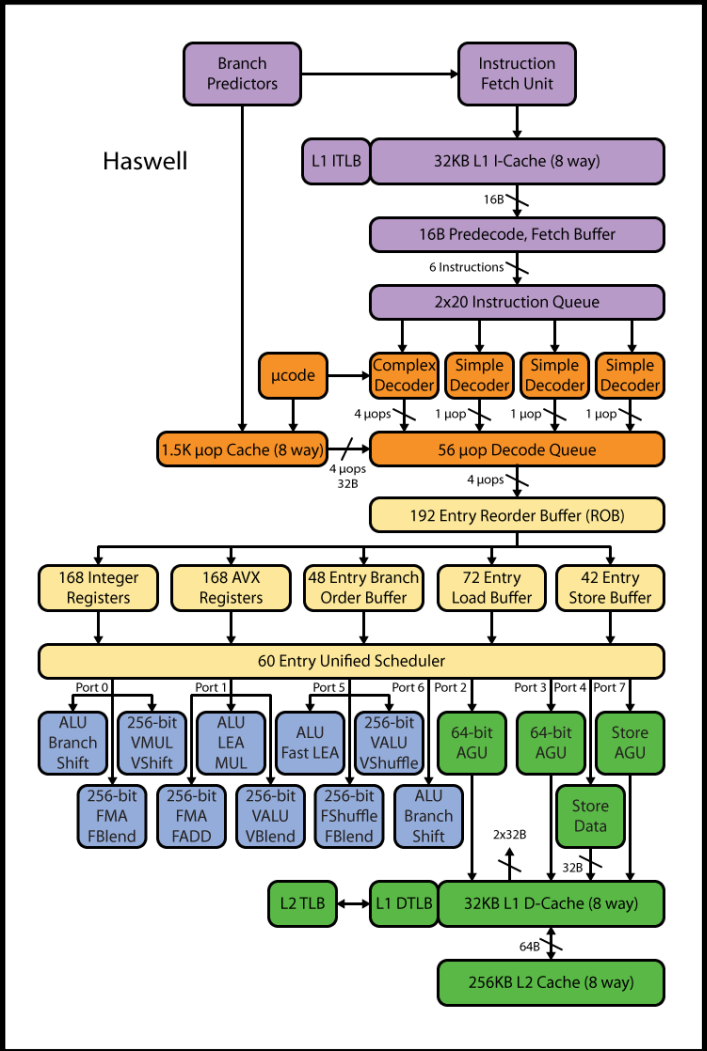
http://fastml.com/the-emperors-new-clothes-distributed-machine-learning/

The Owls are Not What They Seem



scalable system	cores	twitter	uk-2007-05
Stratosphere [6]	16	950s	-
X-Stream [17]	16	1159s	-
Spark [8]	128	1784s	$\geq$ 8000s
Giraph [8]	128	200s	$\geq$ 8000s
GraphLab [8]	128	242s	714s
GraphX [8]	128	251s	800s
Single thread (SSD)	1	153s	417s
Union-Find (SSD)	1	15s	30s

HW is weird



http://www.realworldtech.com/haswell-cpu/6/

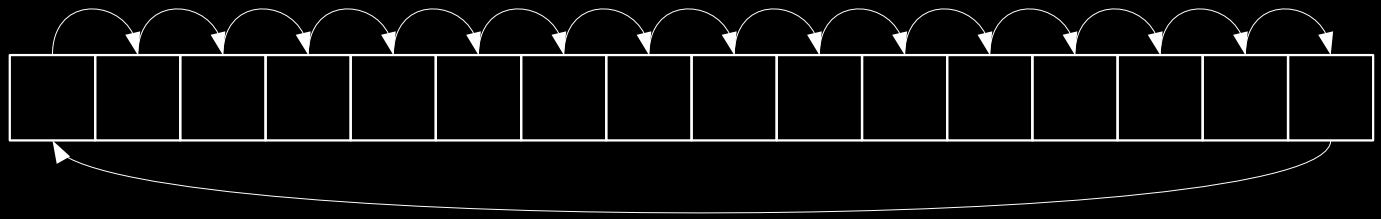
And now a little quiz

```
long sum = 0;

for (int i = 0; i < iter; i ++) {
    sum += arr[i % len];
}
```

```
long sum = 0;

for (int i = 0; i < iter; i ++) {
  sum += arr[i & (len - 1)];
}
```

```
long sum = 0;
long x = arr[0];

for (int i = 0; i < iter; i ++) {
    sum += x;
    x = arr[x];
}
```

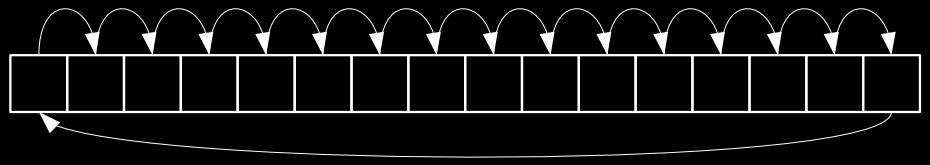
```
long sum = 0;

for (int i = 0; i < iter; i ++) {
  sum += arr[i & (len - 1)];
}
```

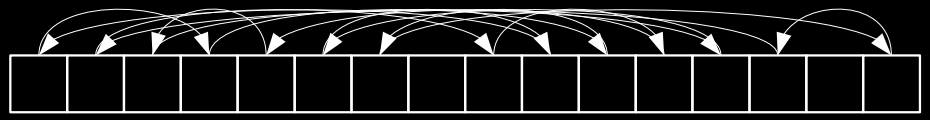


PARADIS: An Efficient Parallel Algorithm for In-place Radix Sor
http://www.vldb.org/pvldb/vol8/p1518-cho.pdf

```
long sum = 0;
int x = seqarr[0];

for (int i = 0; i < iter; i ++) {
    sum += x;
    x = seqarr[x];
}
```

```
long sum = 0;
int x = randarr[0];

for (int i = 0; i < iter; i ++) {
    sum += x;
    x = randarr[x];
}
```





(cache size/prefetch)

```
L1           0.95 s    1.8 cycles/↻   0.6 ns/↻  4.45 IPC
SEQ INDEP    1.22 s    2.4 cycles/↻   0.8 ns/↻  3.29 IPC
SEQ DEP      2.45 s    4.9 cycles/↻   1.5 ns/↻  0.97 IPC
RAND INDEP  25.55 s   51.1 cycles/↻  16.0 ns/↻  0.15 IPC
RAND DEP   130.98 s  262.0 cycles/↻  81.6 ns/↻  0.02 IPC
```

```
struct row {
  long a,b,c,d,e,f,g,h; // 64B
};

for (int i = 0; i < N; i++) {
    sums[0] += rows[i].a;
}
```

```
struct row {
  long a,b,c,d,e,f,g,h; // 64B
};

for (int i = 0; i < N; i++) {
  sums[0] += rows[i].a;
  sums[1] += rows[i].b;
  sums[2] += rows[i].c;
  sums[3] += rows[i].d;
  sums[4] += rows[i].e;
  sums[5] += rows[i].f;
  sums[6] += rows[i].g;
  sums[7] += rows[i].h;
}
```

(column DB)

```
long count = 0;
double sum = 0.0;

for (int i = 0; i < len; i++) {
  if (randarr[i] != 0) {
    sum += randarr[i] / d;
    count += 1;
  }
}
```

```
long count = 0;
double sum = 0.0;

for (int i = 0; i < len; i++) {
  if (randarr[i] > 0) {
    sum += randarr[i] / d;
    count += 1;
  }
}
```

An Experimental Study of Sorting and Branch Prediction
http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.157.9149&rep=rep1&type=pdf

AA-Sort: A New Parallel Sorting Algorithm for Multi-Core SIMD Processors
http://researcher.watson.ibm.com/researcher/files/jp-INOUEHRS/PACT2007-SIMDsort.pdf

Efficient Implementation of Sorting on Multi-Core SIMD CPU Architecture
http://www.vldb.org/pvldb/1/1454171.pdf

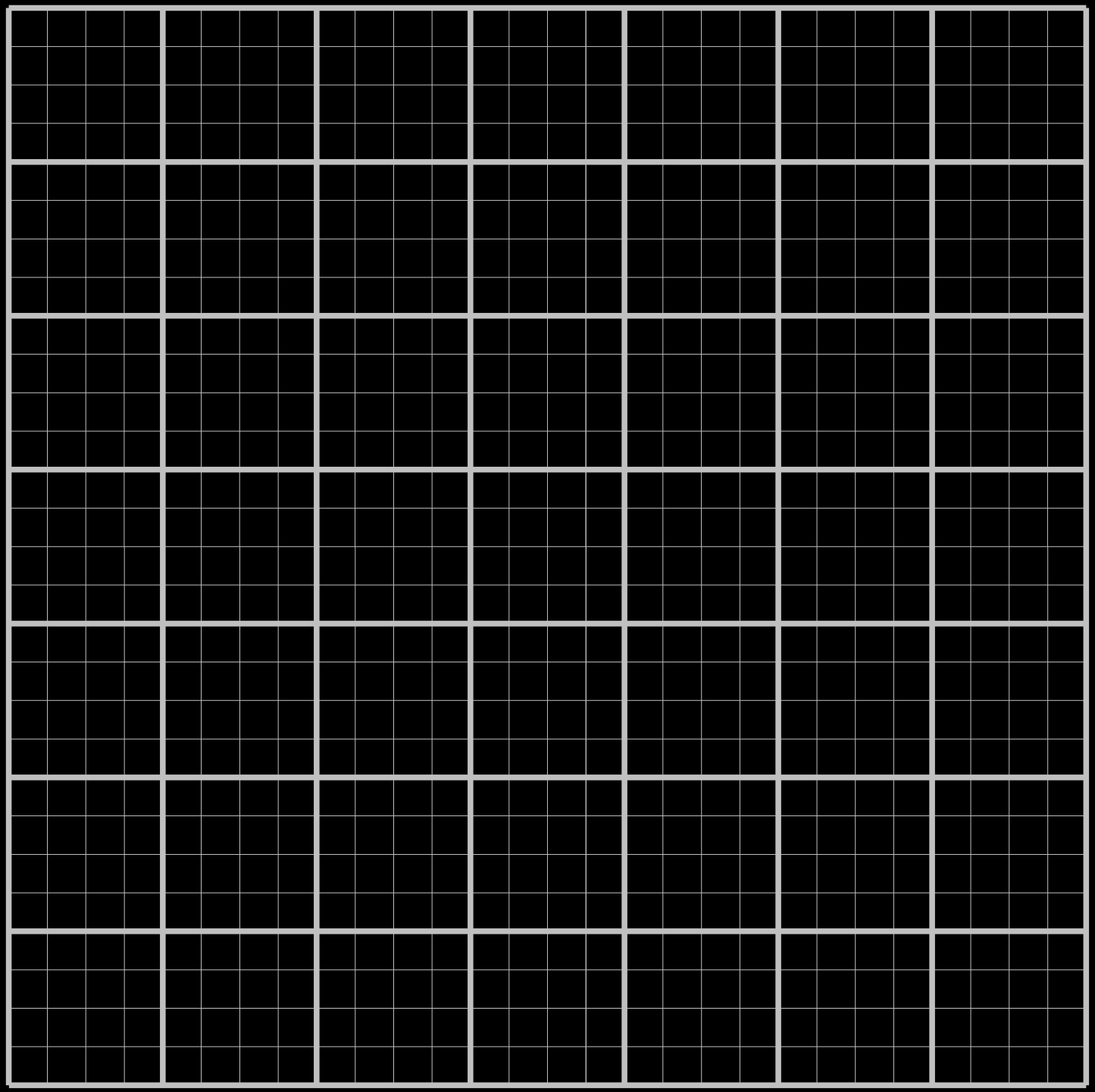
(CMOV/ARM/Itanium/GPU)

Small
Local
Sequential
Predictable

http://funkcionalne.cz/2014/10/procesory-a-jejich-architektura-sebrane-spisy/

Asymptotes v. Constants

‹anecdotes›
-O0/-O3
4x ALS/threadstorm
tiling/naïve cover trees
DJBX33A
sort vectors
DBSCAN
Radix sort grouping
Kmett's log vs $\log^2$
6.5x Spark & github.com/non/debox
‹/anecdotes›



Know Your ~~Enemy~~ Hardware

„without expertise, without a habitual knowledge of the familiar, the new that is taking shape can hardly be understood“ – Adorno

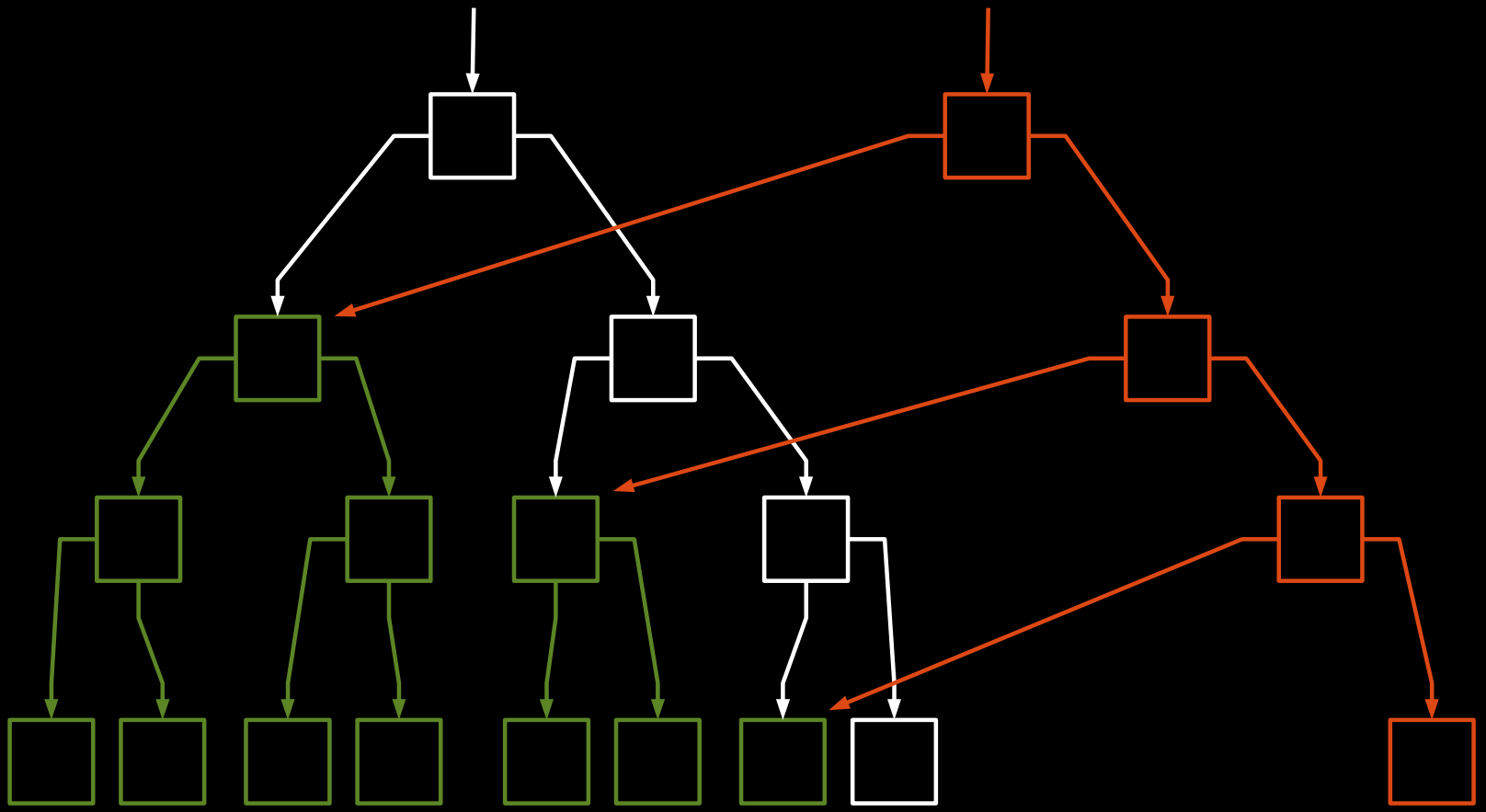
„The greatest trick the Amazon ever pulled was convincing the world the hardware doesn't exist“

FP?

referential transparency
equational reasoning
immutability
Lazyness

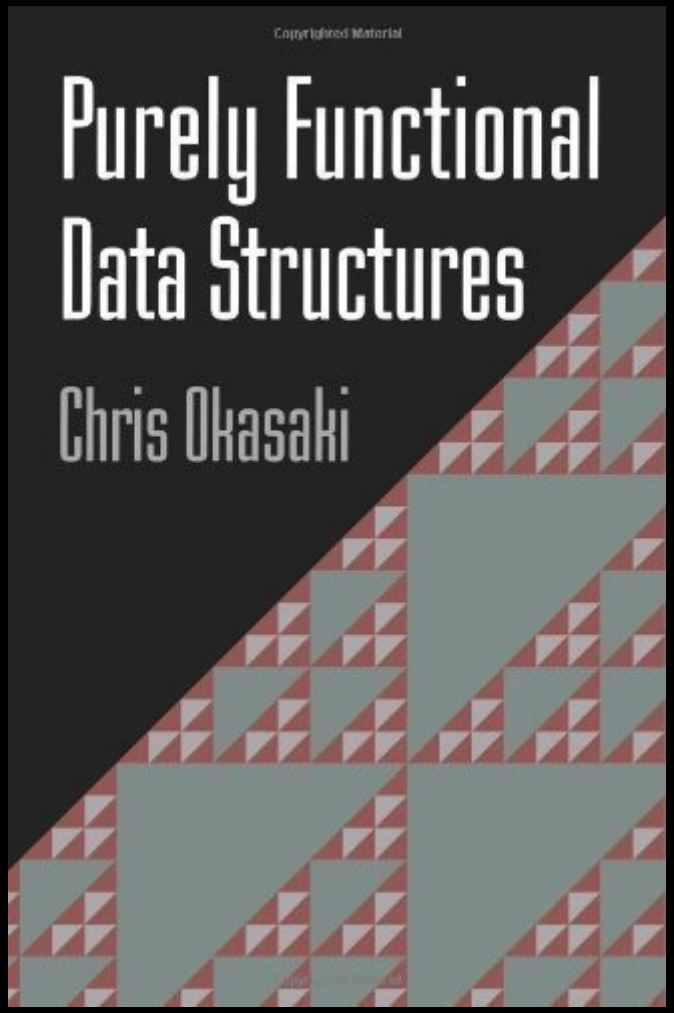
Why Functional Programming Matters
http://www.cse.chalmers.se/~rjmh/Papers/whyfp.pdf

Persistence & structural sharing



Advanced Data Structures Course
http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-851-advanced-data-structures-spring-2012/lecture-videos/

Making Data Structures Persistent
http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.133.4630


Copyrighted Material
Purely Functional
Data Structures
Chris Okasaki

Allocation

```
foldr (+) 0 [1..10]
  map f (map g xs)
```

deforestation
stream fusion
escape analysis
(Azul)

Trace-based Just-in-time Compilation for Lazy Functional Programming Languages
https://dl.dropboxusercontent.com/u/3265448/schilling.thesis.final.2014-01-20.pdf

Compression in Apache Kafka is now 34% faster
http://www.confluent.io/blog/compression-in-apache-kafka-is-now-34-percent-faster

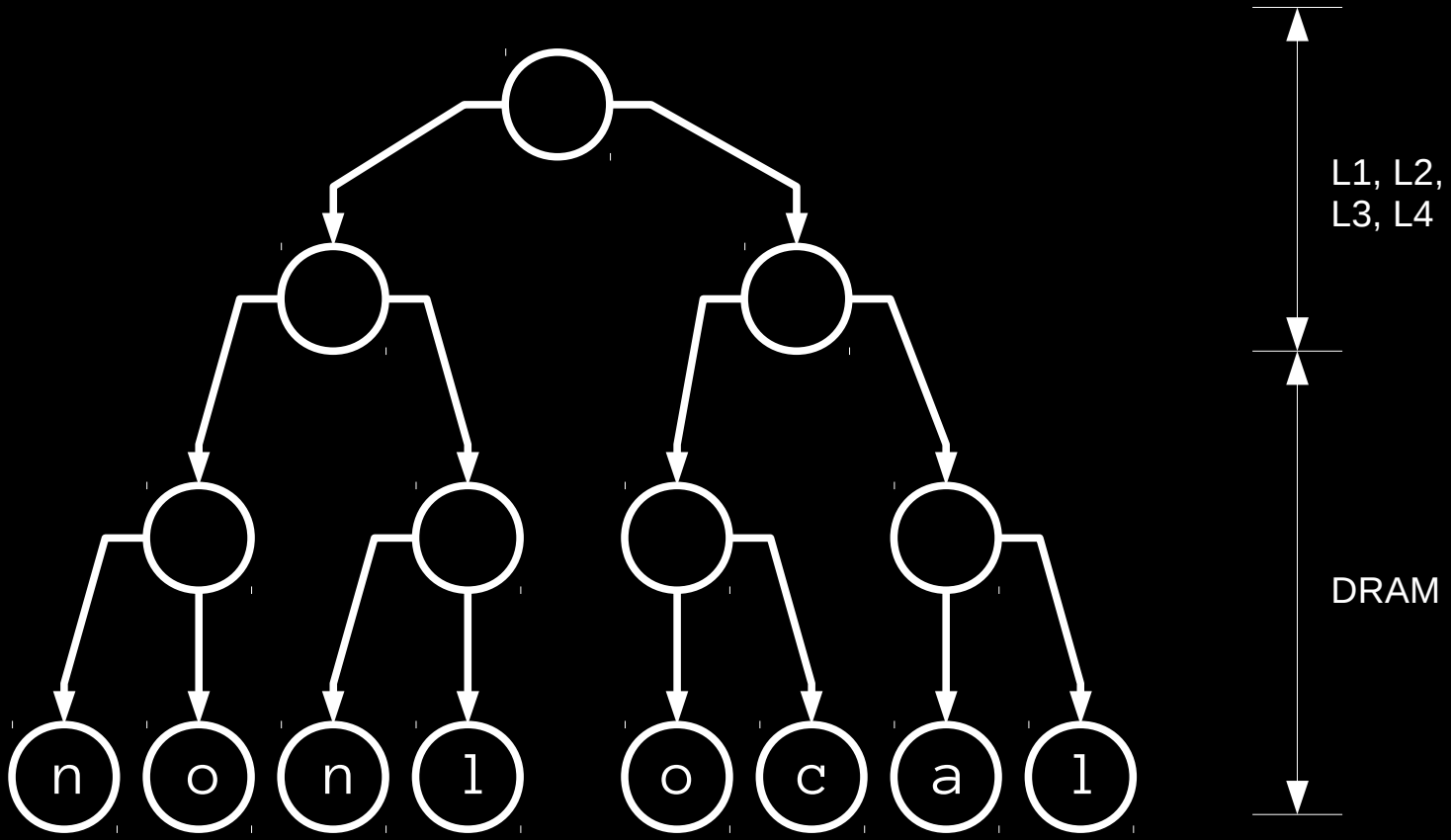
Bandwidth

allocation/eviction
double the bandwidt, double the fun

<joke>
thread local maps
perf
Algebird monoids
</joke>

Trees

O(right)
~30
GC/G1

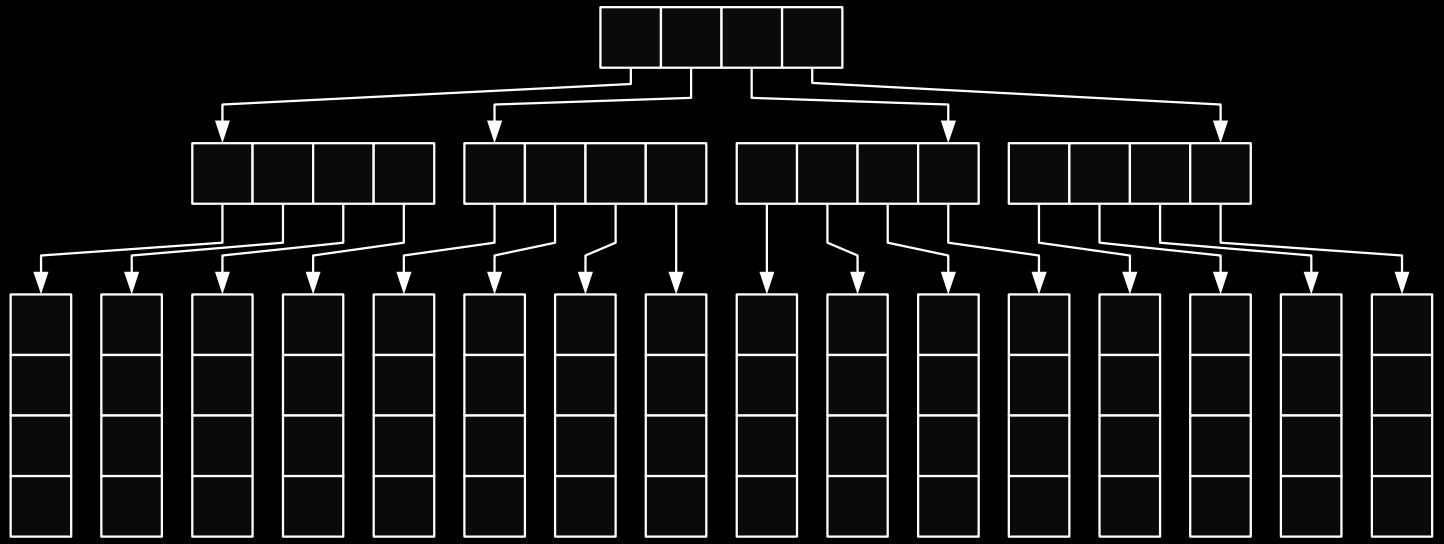


(cache oblivious DS/van Emde Boas)
(gather binary search)

Vectors

Effectively constant*

$\log_{32}(\text{Int.MaxValue}) \approx 7$



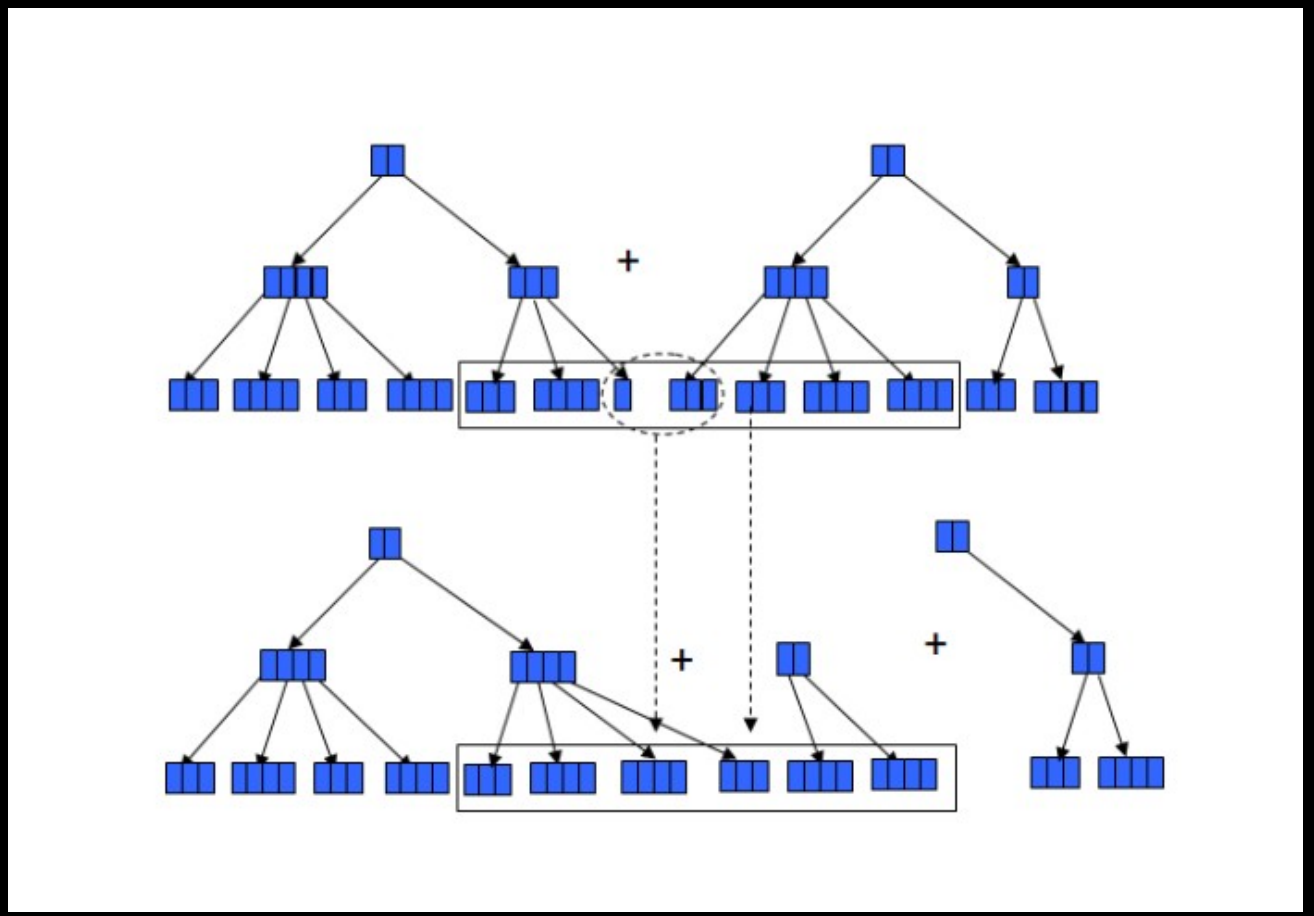
256B nodes

append buffer/Node

RRB trees

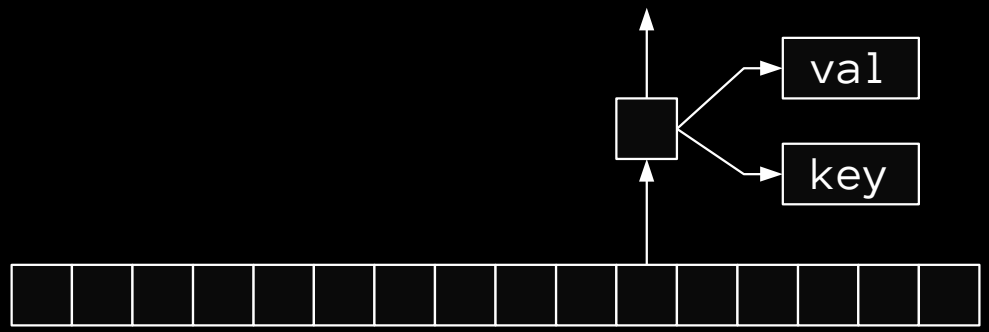
log merge
modulo

RRB-Trees: Efficient Immutable Vectors
http://infoscience.epfl.ch/record/169879/files/RMTrees.pdf



Hashmaps

dreams about O(1) from 1953
PRNG



Large HashMap overview: JDK, FastUtil, Goldman Sachs, HPPC, Koloboke, Trove
http://java-performance.info/large-hashmap-overview-jdk-fastutil-goldman-sachs-hppc-koloboke-trove/

(StringIntDictionary)
(BF/MurMur3)

HAMT

hash array mapped tries
~~resize~~
M1/M2/M3/M4

Ideal Hash Trees
http://lampwww.epfl.ch/papers/idealhashtrees.pdf

Fast And Space Efficient Trie Searches
http://infoscience.epfl.ch/record/64394/files/triesearches.pdf

The FP is already here — it's just not very evenly distributed.

SnapQueue

Lock-Free Queue with Constant Time Snapshots

https://axel22.github.io/resources/docs/snapqueue.pdf

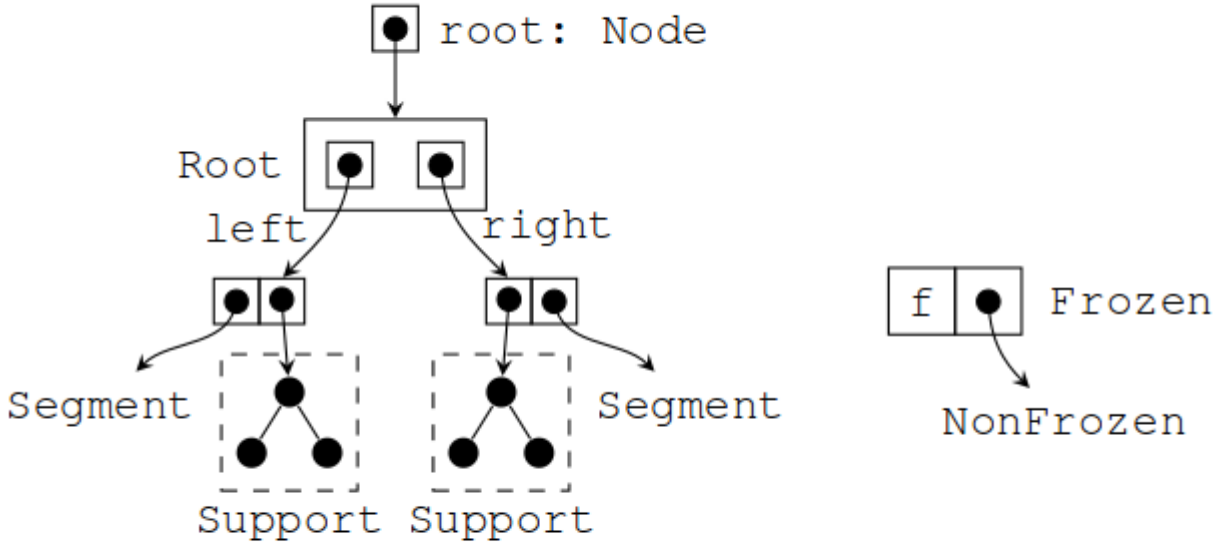
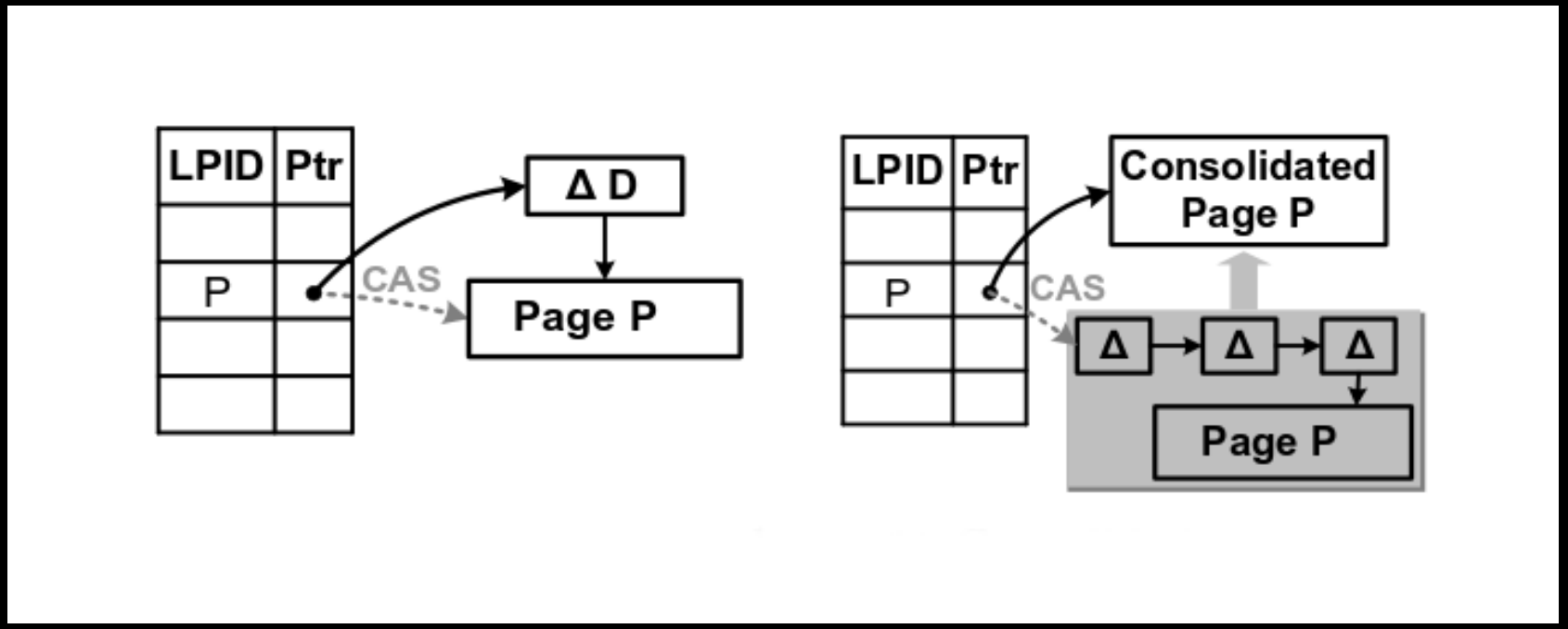


Figure 7. SnapQueue Illustration

The Bw-Tree

A B-tree for New Hardware Platforms

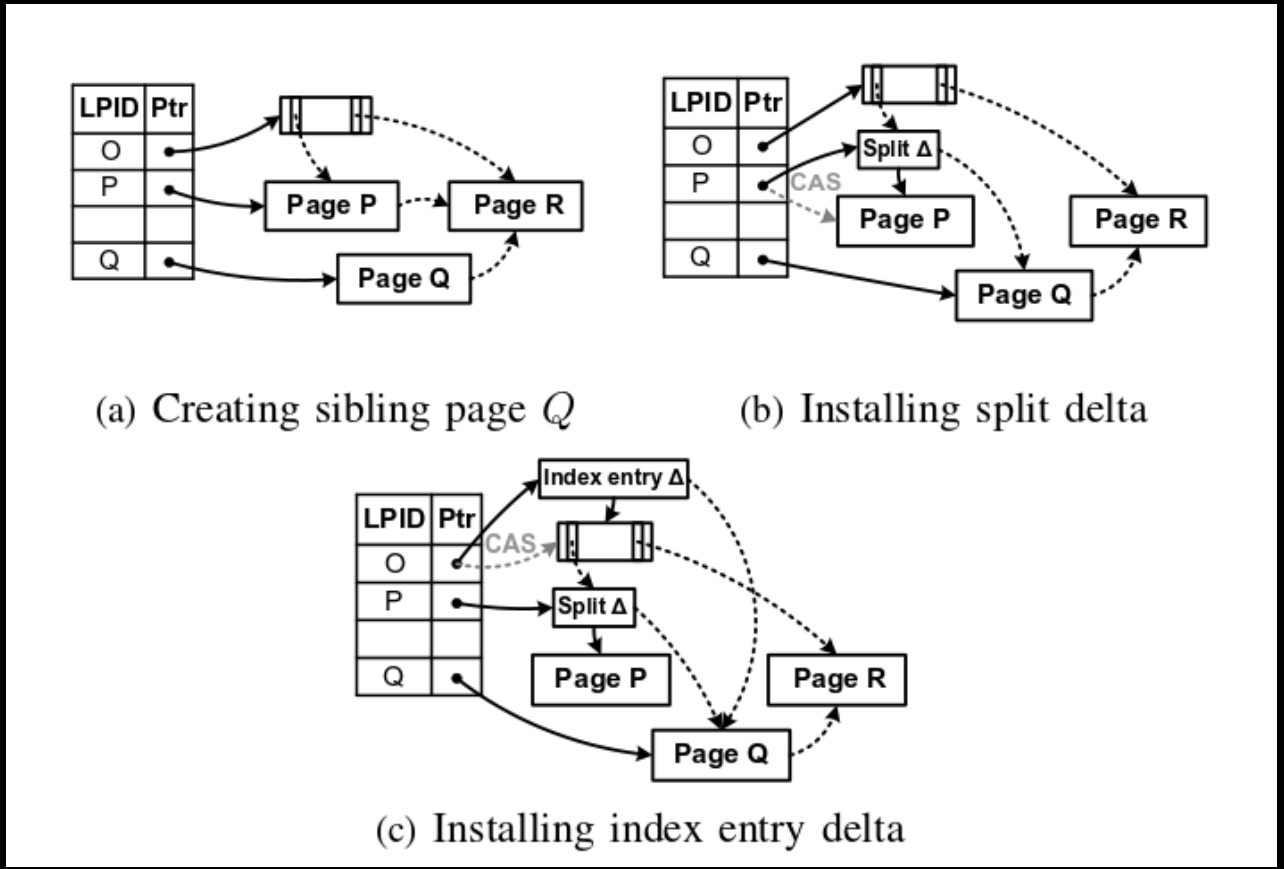
http://research.microsoft.com/pubs/178758/bw-tree-icde2013-final.pdf



The Bw-Tree

A B-tree for New Hardware Platforms

http://research.microsoft.com/pubs/178758/bw-tree-icde2013-final.pdf

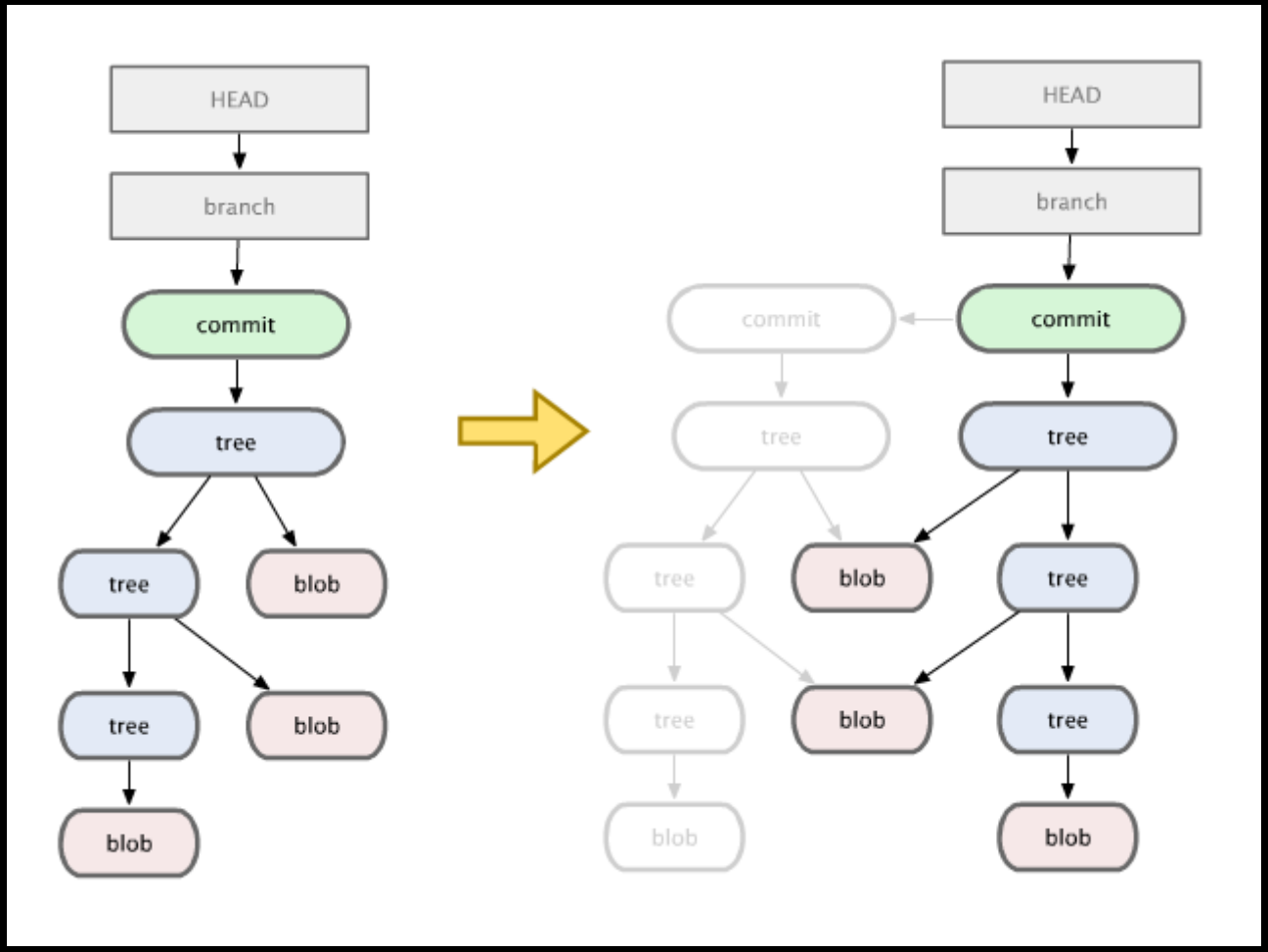


(a) Creating sibling page Q

(b) Installing split delta

(c) Installing index entry delta

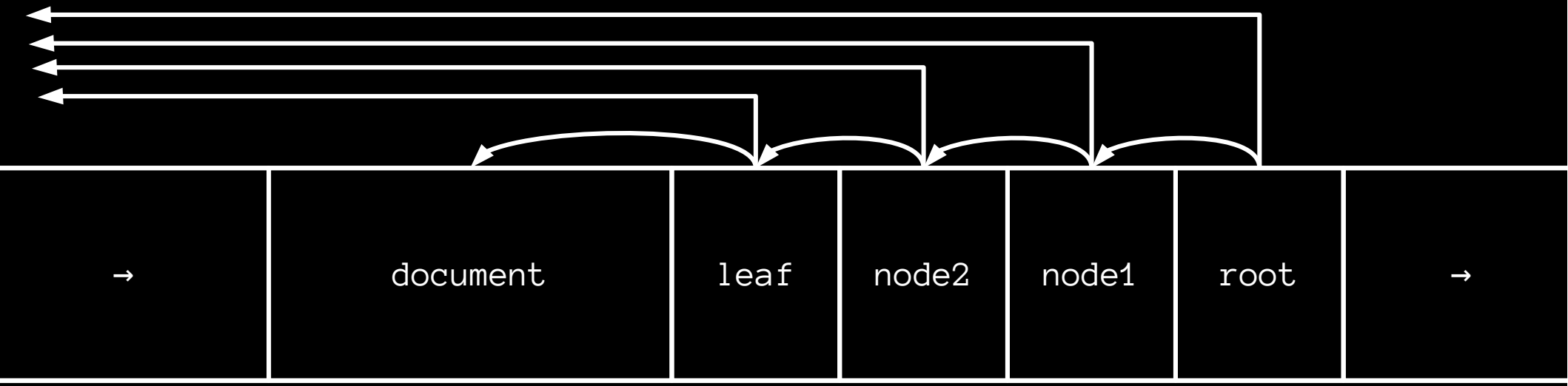
Git



http://eclipsesource.com/blogs/2009/12/13/persistent-trees-in-git-clojure-and-couchdb-data-structure-convergence/

https://speakerdeck.com/schacon/introduction-to-git

CouchDB



CouchDB's File Format
https://plus.google.com/+rkalla/posts/CyvwRcvh4vv

LevelDB
Lucene
Druid.IO
MonetDB

Log Structured Merge Trees
http://www.benstopford.com/2015/02/14/log-structured-merge-trees/

Log-structured file systems: There's one in every SSD
https://lwn.net/Articles/353411/

A Comparison of Fractal Trees to Log-Structured Merge (LSM) Trees
http://forms.tokutek.com/acton/attachment/6118/f-0039/1/-/-/-/-/lsm-vs-fractal.pdf

Stratified B-trees and versioning dictionaries
http://arxiv.org/pdf/1103.4282v2.pdf

Diff-Index: Differentiated Index in Distributed Log-Structured Data Stores
http://researcher.ibm.com/researcher/files/us-wtan/DiffIndex-EDBT14-CR.pdf

Out-of-order CPU

register renaming

FP is not the key to concurrency lack of communication is

contention/cache coherence

An Overview of Kernel Lock Improvements
http://events.linuxfoundation.org/sites/events/files/slides/linuxcon-2014-locking-final.pdf

Asynchronized Concurrency:
The Secret to Scaling Concurrent Search Data Structures
http://infoscience.epfl.ch/record/207109/files/ascy_asplos15.pdf

Adaptive lock-free maps: purely-functional to scalable
https://dl.acm.org/citation.cfm?id=2784734

<namedrop>
Erlang accidentally has the right properties to exploit multi-core architectures - not by design but by accident. - Joe Armstrong
</namedrop>

Make it fast, make it GPU

map/filter/reduce/monoids
#pragma omp parallel for
-fopenmp
~~alocate & recur~~
Von Neumann/dataflow/OOO

Programming on Parallel Machines - GPU, Multicore, Clusters and More
http://heather.cs.ucdavis.edu/~matloff/158/PLN/ParProcBook.pdf

A Primer on Memory Consistency and Cache Coherence
https://lagunita.stanford.edu/c4x/Engineering/CS316/asset/A_Primer_on_Memory_Consistency_and_Coherence.pdf

<qa>
MM?
</qa>

CAS hell

atoms/agents/refs
contention
retries
starving
eventuality
confluent persistency/CDRT/CALM/commutative group
agents queues
MVCC
commute

Staring into the Abyss: An Evaluation of
Concurrency Control with One Thousand Cores
http://vldb.org/pvldb/vol8/p209-yu.pdf

Hardware transactional memory

RTM/HLE
fragile as f***
no guarantee of forward progress

What's the deal with Hardware Transactional Memory?
https://www.youtube.com/watch?v=eBAJ1eNrksM

Early Experience with a Commercial Hardware Transactional Memory Implementation
http://vglab.cse.iitd.ac.in/~sbansal/csl862-os/readings/htm_experiences.pdf

Accelerating Native Calls using Transactional Memory
https://blogs.oracle.com/dave/resource/wttm15-dice.pdf

Hardware Transactional Memory in Java, or why synchronized will be cool again.
http://vanillajava.blogspot.cz/2014/02/hardware-transactional-memory-in-java.html

</presentation>

Karel Čížek
@kaja47
funkcionalne.cz

NVRAM