PERFORMANCE TUNING FOR DEVELOPERS

James Clippinger, VP, Strategic Accounts
Erin Miller, Manager, Performance Engineering

© COPYRIGHT 2016 MARKLOGIC CORPORATION. ALL RIGHTS RESERVED.

MarkLogic®

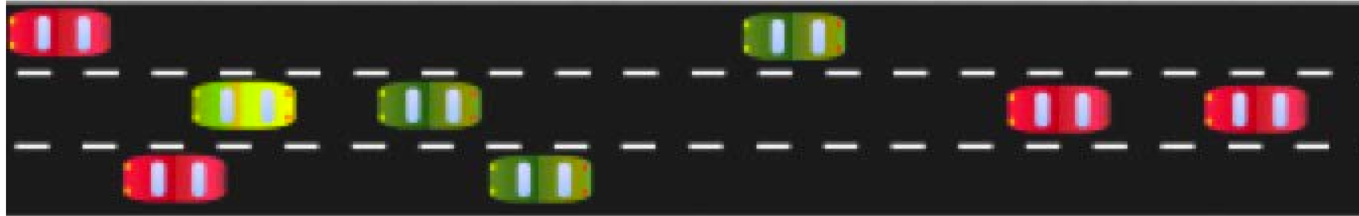
Agenda

- MarkLogic Resource Consumption for Developers
  - Understanding Systems Utilization
- Common Code Issues
  - Implicit Sorting
  - Avoiding termlist queries
- Questions

© COPYRIGHT 2016 MARKLOGIC CORPORATION. ALL RIGHTS RESERVED.

Resource Consumption: Why Developers Should Care

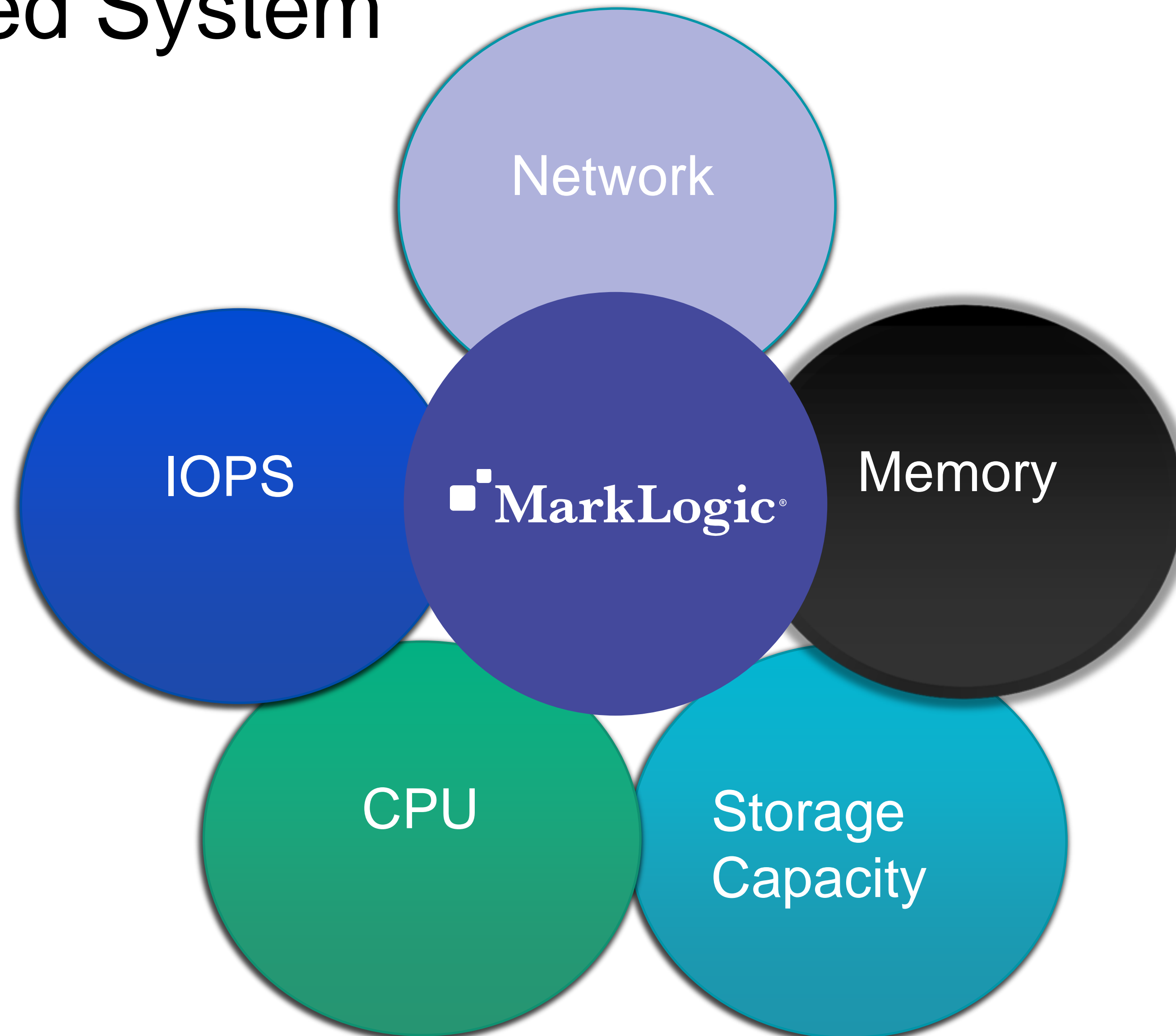
- System stuff is for the sysadmin. I just need to make sure my code is optimized.



- I guess we needed more I/O capacity.

© COPYRIGHT 2016 MARKLOGIC CORPORATION. ALL RIGHTS RESERVED.

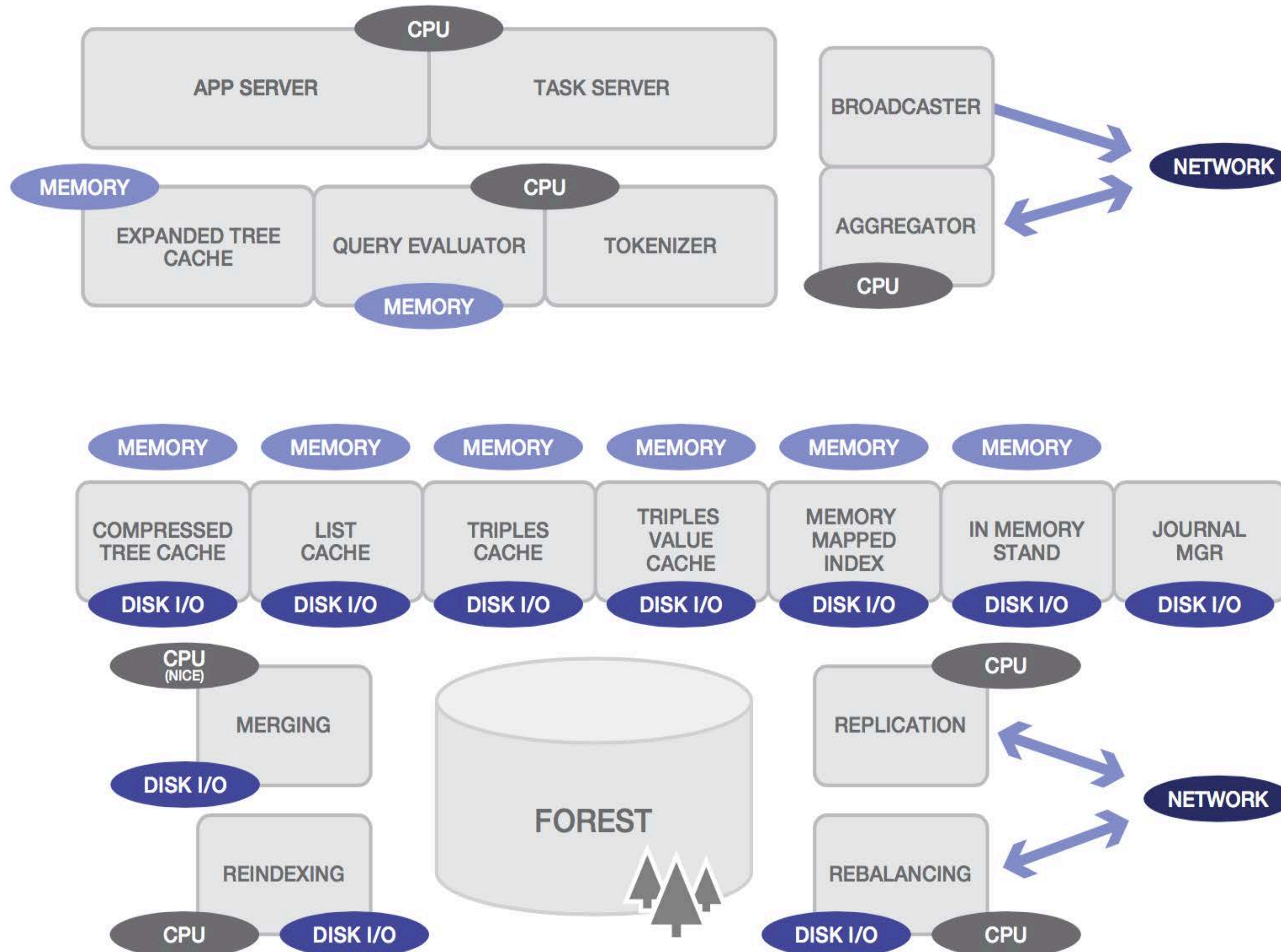
The Balanced System



© COPYRIGHT 2016 MARKLOGIC CORPORATION. ALL RIGHTS RESERVED.

MarkLogic

MarkLogic Server: E/D Node



© COPYRIGHT 2016 MARKLOGIC CORPORATION. ALL RIGHTS RESERVED.

MarkLogic®

Ingest: What Happens

- Application sends document to MarkLogic App Server
- Document loaded into in-memory stand
- Transaction written to journal
- Periodically, in-memory stand is flushed to disk
- As stands grow/change (updates, deletes) merges occur
- Merges reclaim disk space and optimize indexes

© COPYRIGHT 2016 MARKLOGIC CORPORATION. ALL RIGHTS RESERVED.

MarkLogic

Ingest: Merges and I/O Capacity

- What happens?
  - Multiple stands combined into one
  - Disk space is reclaimed
  - Indexes and lexicons are recombined and optimized
- When does it happen?
  - When you ask the server to merge (manual)
  - Automatically, based upon max merge size, merge min ratio

© COPYRIGHT 2016 MARKLOGIC CORPORATION. ALL RIGHTS RESERVED.

MarkLogic

Ingest: Resource Impact

- Storage: need total indexed forest size + 64GB/forest
  - But don't forget how DR/HA changes everything
- IOPS: Spikey I/O when in-memory stand flushed to disk
- CPU: Merges (operate as background I/O; show up as nice)
- CPU: index creation/updates (ingest + merges)
- Memory: increases as content is loaded (tree, termlist, triples, reverse)
  - Preload mapped data: all memory mapped file data pages loaded at end of merge
- Network: NAS impact; mlcp fastload avoids network hops but can only be used for new content (not updates) and when forest topology won't change

© COPYRIGHT 2016 MARKLOGIC CORPORATION. ALL RIGHTS RESERVED.

MarkLogic

Ingest: Developer Tips

- Careful of transforms
  - Try to minimize complexity
  - Think about queries that you’re writing in your ingest code and what they might do to system resources (in addition to basic ingest + merge)
  - See how you scale
- Use evals when needed
  - Lookups in the context of updates causes locking

© COPYRIGHT 2016 MARKLOGIC CORPORATION. ALL RIGHTS RESERVED.

MarkLogic

Ingest: Developer Tips

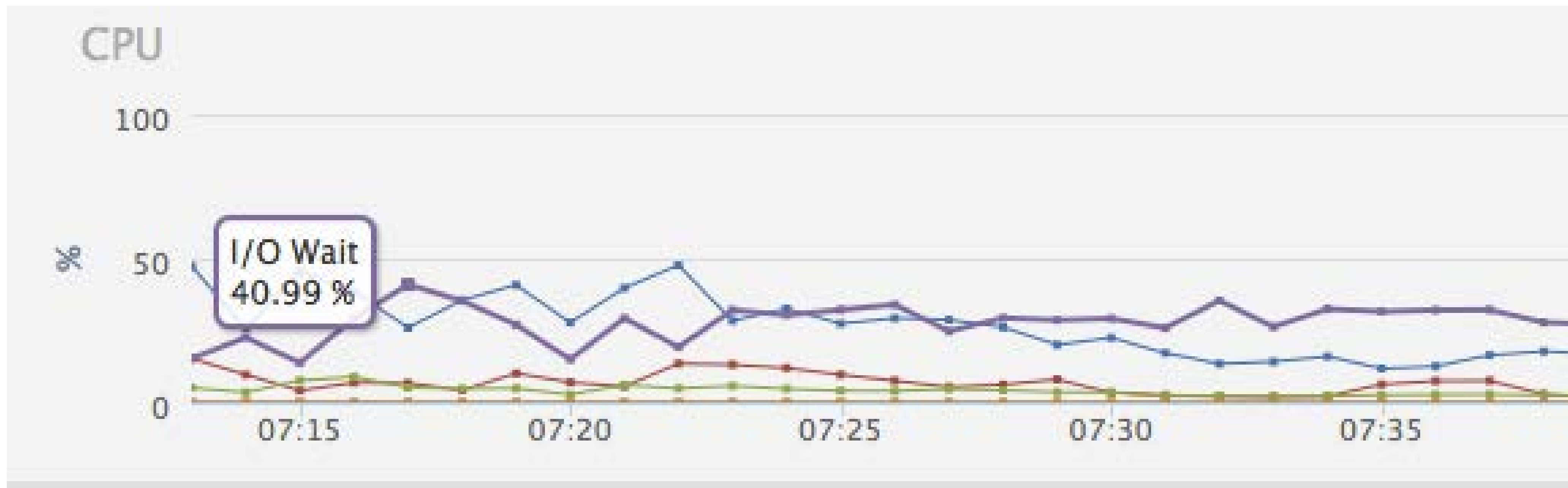
- Understand storage and I/O requirements, and propagate through your org
  - Don’t run out of merge space
- Plan for scale
  - Running your ingest code in dev isn’t enough
  - How does it scale when you have 100M docs instead of 1M?
  - Do what Clip says
  - Monitor, monitor, monitor

© COPYRIGHT 2016 MARKLOGIC CORPORATION. ALL RIGHTS RESERVED.

MarkLogic

Ingest: Identifying Bottlenecks

- There's only a problem if there's a problem: context is everything
- Inserts/updates and queries slow down during discrete periods
  - Let's look at the Monitoring History and Logs



© COPYRIGHT 2016 MARKLOGIC CORPORATION. ALL RIGHTS RESERVED.

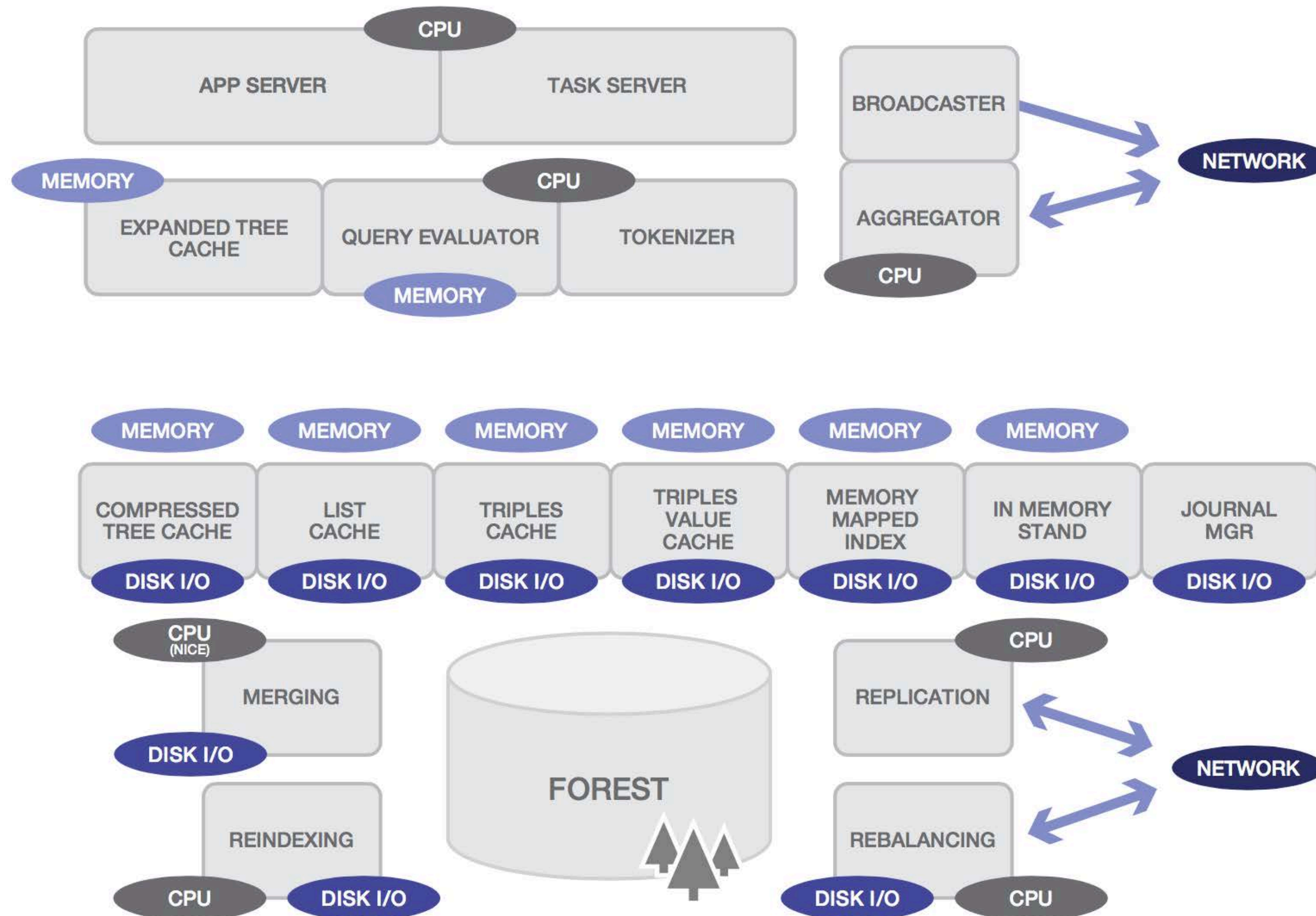
MarkLogic

Ingest: Solving the I/O Bottleneck

- Best solution: more I/O throughput
  - More nodes
  - More forests
  - More NAS
- Next best: throttle background I/O:
  - background-io-limit = 100
  - background-io-limit also applies to database backups

© COPYRIGHT 2016 MARKLOGIC CORPORATION. ALL RIGHTS RESERVED.

Understanding Queries: What Happens Where



© COPYRIGHT 2016 MARKLOGIC CORPORATION. ALL RIGHTS RESERVED.

MarkLogic®

Queries: Searches and Index Resolution

- Execute query on App Server/E-node
- For searches and XPath expressions:
  - Step 1: Index resolution: D-nodes
  - Step 2: Filtering: E-nodes
    - Load fragment from disk and verify match
- Step 2 is *optional*; use to eliminate false positives
- Let’s think about what step 2 means
  - For a database with 100K docs
  - For a database with 100M docs

© COPYRIGHT 2016 MARKLOGIC CORPORATION. ALL RIGHTS RESERVED.

Queries: But It Worked Fine in Dev…

- First clue that you’re looking at filtering run amok: long-running queries
  - Use Monitoring Dashboard
    - http://<ml-hostname>:8000/dashboard/query
- System tools can show you I/O spikes: iostat –x 3

```
avg-cpu:  %user   %nice %system %iowait  %steal   %idle
          10.10    6.42    2.08   34.02    0.00   47.38

Device:         rrqm/s   wrqm/s     r/s     w/s   rsec/s   wsec/s avgrq-sz avgqu-sz   await  svctm  %util
sda              23.33    23.33    4.33    0.33   240.00   189.33    92.00     0.02    3.71   2.86   1.33
sdb               0.00  8238.00  450.67  130.33 67829.33 66269.33   230.81     7.83   13.45   1.13  65.77
sdc               0.00 13590.67  682.00  168.67 115616.00 110074.67   265.11     8.22    9.65   1.04  88.33
sdd               0.00  8835.67  483.00  201.33 72557.33 72168.00   211.48    90.02  148.32   1.46 100.03
```


© COPYRIGHT 2016 MARKLOGIC CORPORATION. ALL RIGHTS RESERVED.

MarkLogic®

Queries: Filtered Searches

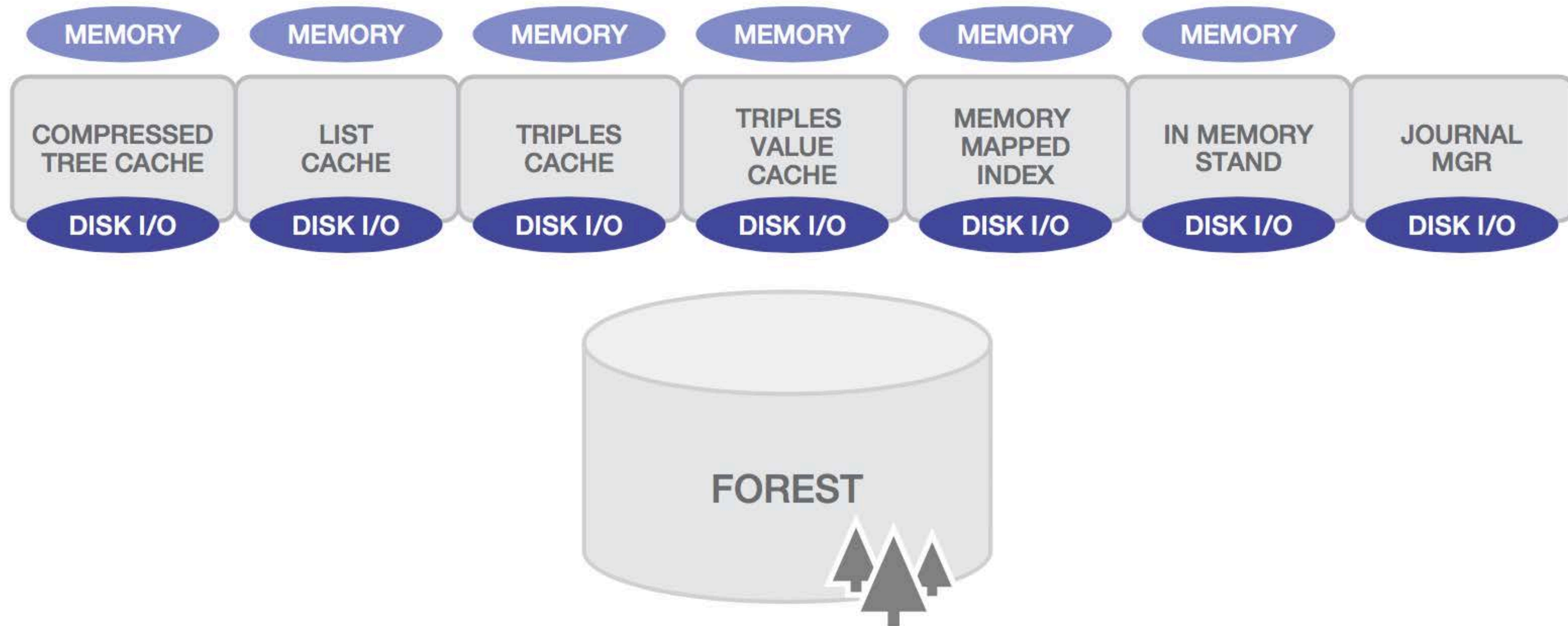
- Remember: Context is everything
- First: Tune indexes to reduce false positives
  - Goal is to resolve queries from the indexes — that will be fastest and most efficient
- Use filtering only if you expect to have false positives and your application can't tolerate false positives

© COPYRIGHT 2016 MARKLOGIC CORPORATION. ALL RIGHTS RESERVED.

MarkLogic

Queries: How About Those Caches?

- Caches (and indexes) are with the data on the D-node: Expanded Tree Cache on the E-node



© COPYRIGHT 2016 MARKLOGIC CORPORATION. ALL RIGHTS RESERVED.

MarkLogic

Queries: Caches and Their Contents

- List cache: holds termlists after they've been read off disk
- Compressed tree cache: holds the binary representation of documents after they've been read off disk
- Expanded tree cache: Each time a D-node sends an E-node a fragment over the network, it sends it in the same compressed format in which it was stored. The E-node then expands the fragment into a usable data structure.
- Triples and Triples value cache: holds triples and triples data
- Not caches, but still in memory: memory mapped files
  - Range indexes
  - Lexicons

© COPYRIGHT 2016 MARKLOGIC CORPORATION. ALL RIGHTS RESERVED.

MarkLogic

Queries: Making the Best Use of Your Caches

- Cache sizes are optimized at install-time
  - What if I add memory?
    - Increase cache sizes
  - What if I separate into D- and E-nodes?
    - Create two groups with two different cache sizes
    - E-nodes should have more ETC, less CTC and List Cache
    - D-nodes should have more CTC and List, less ETC
      - But don’t forget about reindexing

© COPYRIGHT 2016 MARKLOGIC CORPORATION. ALL RIGHTS RESERVED.

MarkLogic

Queries: Cache Partitions

- Each cache defaults to one, two, or four partitions depending upon memory size
- Increasing partitions means increased concurrency
  - But wait — it also means smaller caches
  - More threads can access the cache, but there’s less space in each cache
    - Greater chance of cache full errors

© COPYRIGHT 2016 MARKLOGIC CORPORATION. ALL RIGHTS RESERVED.

MarkLogic

Queries: Lifecycle of Cache Items

- Documents/termlists added to caches on first request
- Each cache partition is responsible for aging out stale items
- You can clear caches by restarting the server
  - Another way to say this: When you restart the server, your caches are empty
  - Make sure you've got warm-up queries for prod as needed

© COPYRIGHT 2016 MARKLOGIC CORPORATION. ALL RIGHTS RESERVED.

MarkLogic

Queries: Tips for Developers

- Don't use filtered searches unless you truly have to; tune your indexes instead
- Remember to warm caches when you start the server
- Increasing cache partitions can increase concurrency, but don't make partitions too small — watch for cache full exceptions
- Monitor I/O utilization at scale: make sure you test your queries against a prod-scale database
- Use Clip's coding tips to make code as efficient as possible
  - Efficient code uses less CPU, I/O, Memory, Network

© COPYRIGHT 2016 MARKLOGIC CORPORATION. ALL RIGHTS RESERVED.

COMMON CODE ISSUES

MarkLogic

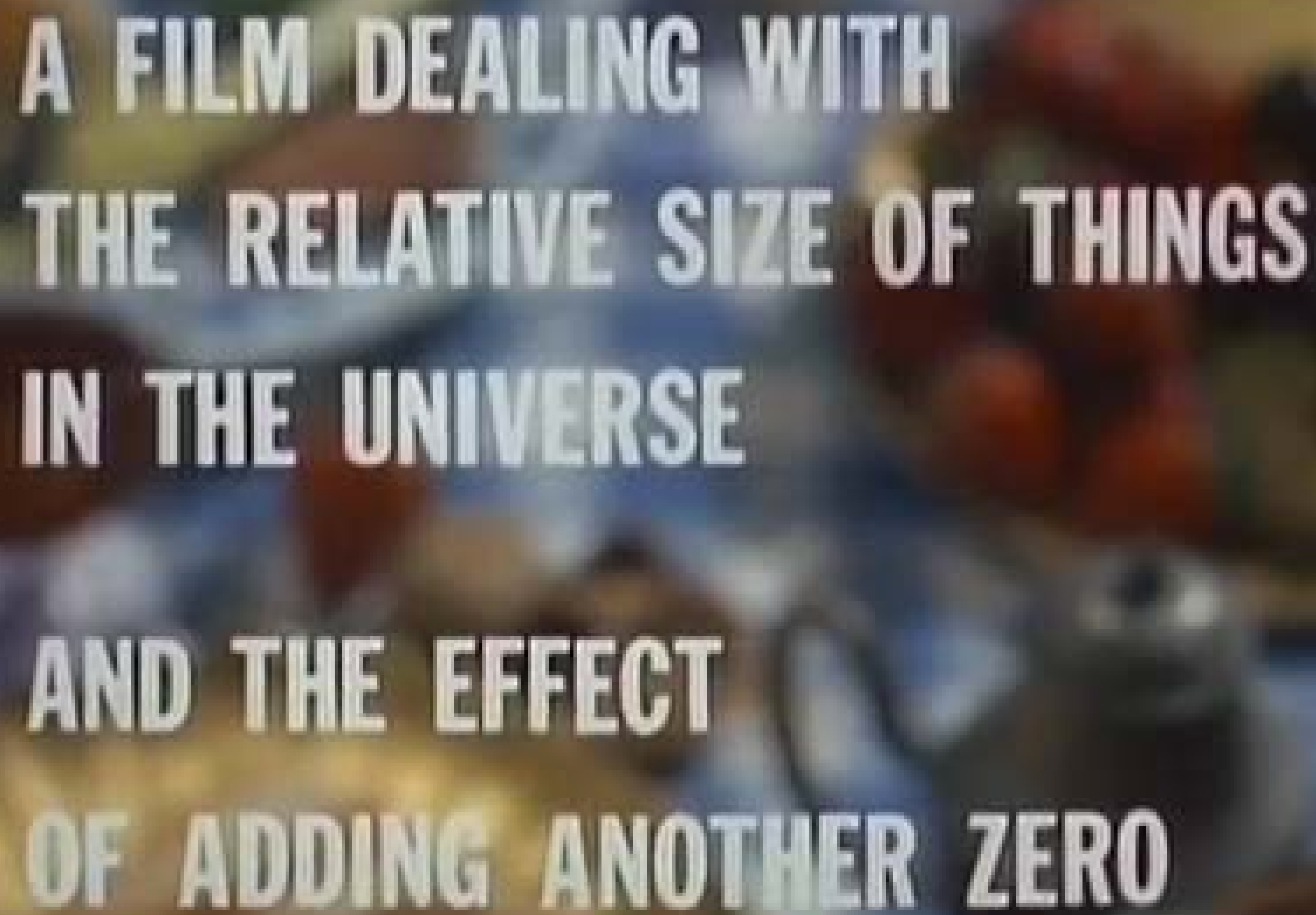
Refresher: What Reading 1MB Takes

Action	Time (µs)	# per second
Reading 1MB from 2133 MT/s RAM	7	~143,000
Reading 1MB over a 10GigE network	1,000	1,000
Reading 1MB from disk	4,000	250

Adapted from http://norvig.com/21-days.html#answers

© COPYRIGHT 2016 MARKLOGIC CORPORATION. ALL RIGHTS RESERVED.

POWERS
OF TEN


A FILM DEALING WITH
THE RELATIVE SIZE OF THINGS
IN THE UNIVERSE
AND THE EFFECT
OF ADDING ANOTHER ZERO

Priority Items: Disk

- Unnecessary fragment reads
- Creating inefficient cache behavior
- Using termlists when in-memory options available

© COPYRIGHT 2016 MARKLOGIC CORPORATION. ALL RIGHTS RESERVED.

IMPLICIT SORTING

```
cts:search(fn:doc(), …)[1 to 10]
```

```
(cts:search(fn:doc(), …)
    /element())[1 to 10]
```

MarkLogic

Implicit Sort and Filter

- Fetch all documents matching search in relevance-ranked order and load into cache
- Sort cached documents into document order
- Return first ten root elements

```
(cts:search(fn:doc(), …)
      /element())[1 to 10]
```

MarkLogic

xdmp:query-meters()

- Check expanded tree cache statistics for query
- Higher than expected? Hunt it down.

```
<qm:query-meters>
  <qm:elapsed-time>PT0.047488S</qm:elapsed-time>
  …
  <qm:expanded-tree-cache-hits>1000</qm:expanded-tree-cache-hits>
  <qm:expanded-tree-cache-misses>10</qm:expanded-tree-cache-misses>
```

```
(: Sorts however many fragments match cts:search() :)
(cts:search(...)/element())[1 to 10]

(: Brings at most 10 fragments into cache :)
cts:search(...)[1 to 10]/element()

(: Later XPath results in sort of all matching fragments :)
let $results := cts:search(fn:doc(), cts:word-query("foo"))
(: Many lines redacted :)
let ($results//x)[1 to 10]
```

AVOIDING TERMLIST QUERIES

```
cts:search(fn:doc(),
  cts:and-query((
    cts:word-query("unlikely phrase"),
    cts:collection-query("large-collection")
  ))
)
```

```
xdmp:plan(cts:search(fn:doc(), …))
=>
<qry:query-plan>
…
  <qry:term-query weight="0">
    <qry:key>15910352410585266235</qry:key>
    <qry:annotation>collection(large-collection)</qry:annotation>
  </qry:term-query>
```

```
cts:search(fn:doc(),
  cts:and-query((
    cts:word-query("unlikely phrase"),
    cts:element-range-query(xs:QName("xdmp:collection"),
      "=", "large-collection",

("collation=http://marklogic.com/collation/codepoint"))
  ))
)
```

```
xdmp:plan(cts:search(fn:doc(), …))

=>

<qry:query-plan>

…

   <qry:range-query weight="0" min-occurs="1" max-occurs="4294967295"
xmlns:xs="http://www.w3.org/2001/XMLSchema">

      <qry:key>8082969983947382622</qry:key>

      <qry:annotation>element(xdmp:collection)</qry:annotation>

      <qry:lower-bound xsi:type="xs:string"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">large-
collection</qry:lower-bound>

      <qry:upper-bound xsi:type="xs:string"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">large-
collection</qry:upper-bound>

   </qry:range-query>
```

MarkLogic

Priority Items: Network

- Avoiding unnecessary round trips
- Reduce payload sizes

© COPYRIGHT 2016 MARKLOGIC CORPORATION. ALL RIGHTS RESERVED.

```
(: get primary keys; so fast! :)
cts:uris((), (),
    cts:word-query("foo"))[1 to 10]
```

```
(: don’t know why this is slow :)
let $uris := cts:uris((), (),
    cts:word-query("foo"))[1 to 10]
for $uri in $uris
return fn:doc($uri)
```

```
(: faster now :)
fn:doc(cts:uris((), (),
    cts:word-query("foo"))[1 to 10])
```

```
(: fastest :)

cts:search(fn:doc(),
     cts:word-query("foo"),
     (cts:index-order(cts:uri-
reference()))
     )[1 to 10]
```

MarkLogic

Priority Items: Memory

- Lexicon calls?
- Consider memory use holistically

© COPYRIGHT 2016 MARKLOGIC CORPORATION. ALL RIGHTS RESERVED.

MarkLogic®

Additional Resources

- MarkLogic Performance: Understanding System Resources
  - http://developer.marklogic.com/learn/understanding-system-resources
- Inside MarkLogic Server
  - http://developer.marklogic.com/inside-marklogic
- MarkLogic Documentation
  - http://docs.marklogic.com/guide/performance
  - http://docs.marklogic.com/guide/cluster

© COPYRIGHT 2016 MARKLOGIC CORPORATION. ALL RIGHTS RESERVED.

MarkLogic®

Q&A

© COPYRIGHT 2016 MARKLOGIC CORPORATION. ALL RIGHTS RESERVED.

MarkLogic®

Thank you!

- erin.miller@marklogic.com
- clip@marklogic.com

© COPYRIGHT 2016 MARKLOGIC CORPORATION. ALL RIGHTS RESERVED.