root cellar

Bleve

- Full-text Search Index
- Pluggable KV layer
  - Initially focused on cgo solution for performance
  - Now hoping to beat performance with simpler Go solution

See also https://www.cockroachlabs.com/blog/the-cost-and-complexity-of-cgo/

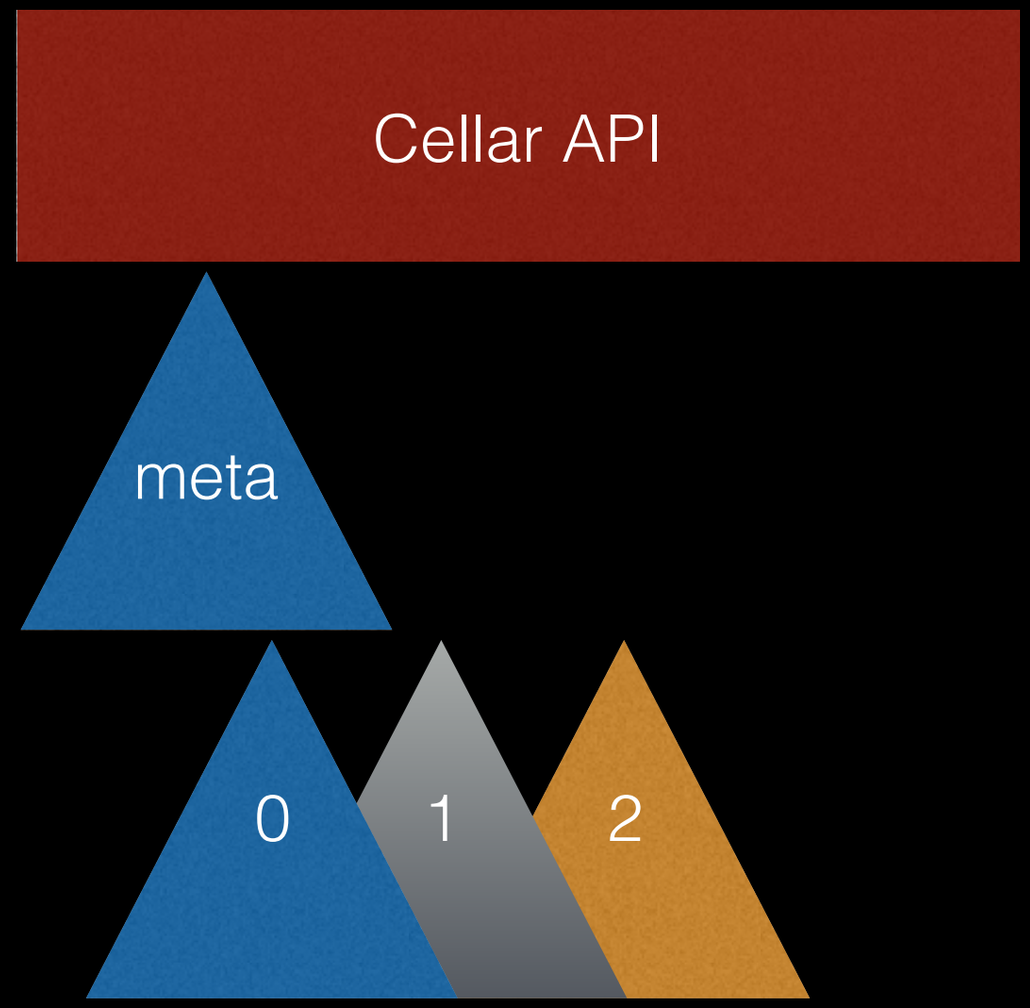
Bolt

embedded key/value database for Go

- Simple API
- Single File
- mmap B+tree

Build LSM on top of Bolt

- Instead of random updates to one large index
- Each batch becomes sequentially written stand-alone bolt instance
- Get/Seek/Next have to navigate multiple read-only bolt instances



Bolt API

- DB - file
- TX - atomic writes, snapshot reads
- Bucket - multiple (nested)
- Cursor - fast access consecutive keys

Cellar API

- DB - directory containing data bolts + meta bolt
- TX - each write TX creates new bolt db, atomically swapped into active list
- Bucket - only ONE
- Cursor - get/seek across all active bolts

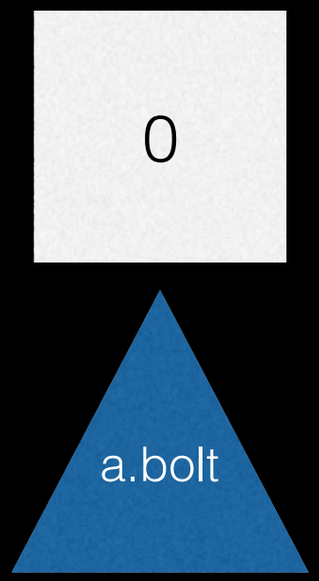
Opening a Cellar

```
c, err := Open("test", nil)

if err != nil {

   log.Fatal(err)

}

defer c.Close()
```

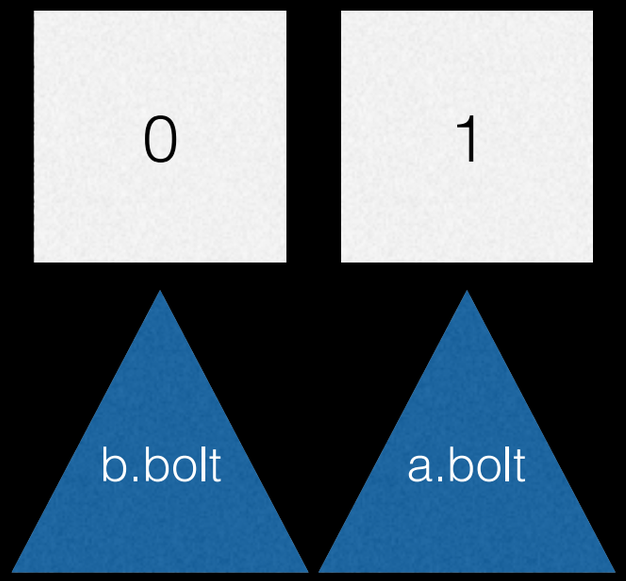
Writing Data

```
err = c.Update(func(tx *Tx) error {

   tx.Put(k,v)

   return nil

})

if err != nil {

   t.Fatal(err)

}
```

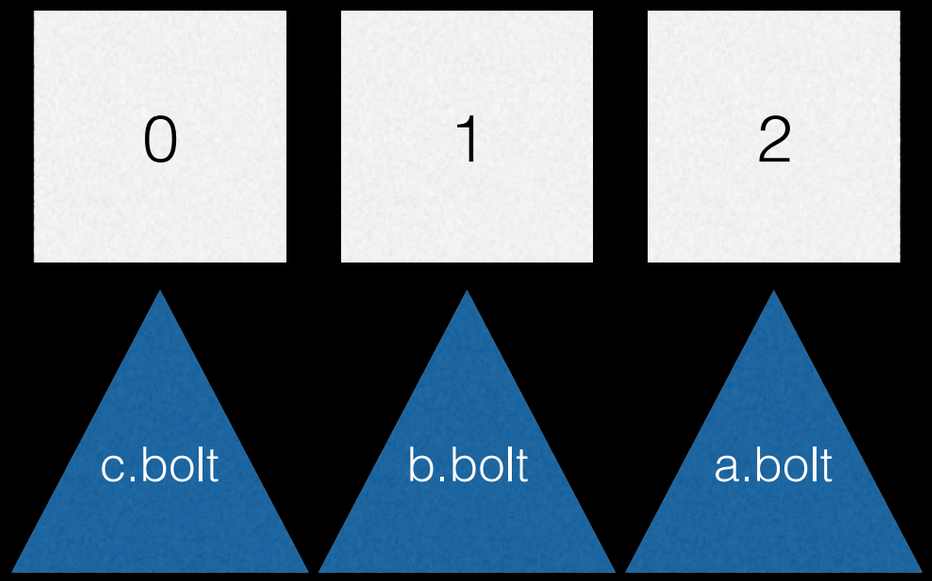
cellar bolt list



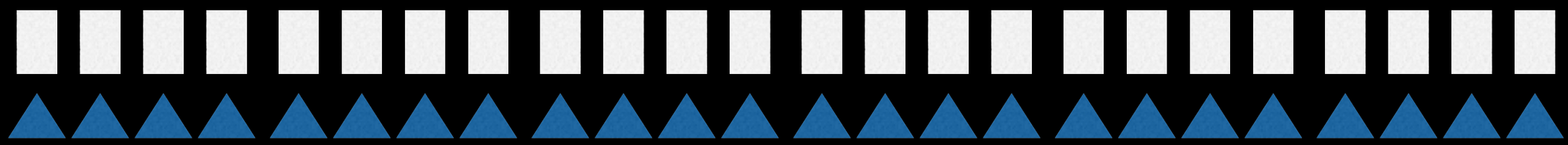
cellar bolt list



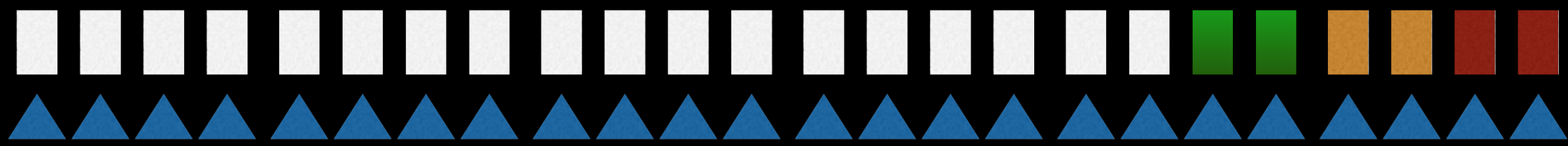
cellar bolt list



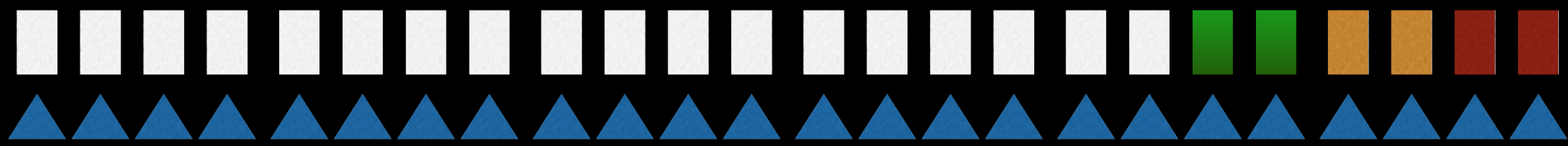
cellar bolt list



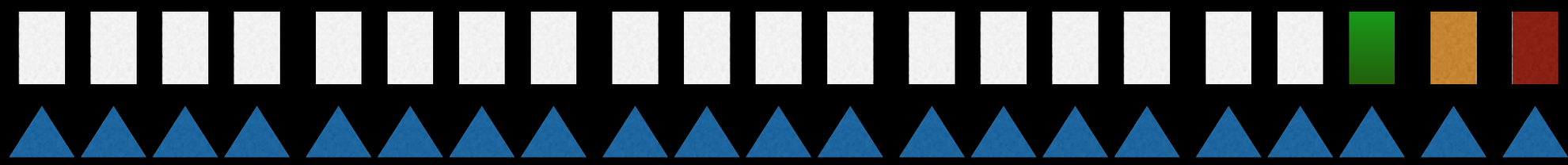
cellar merge



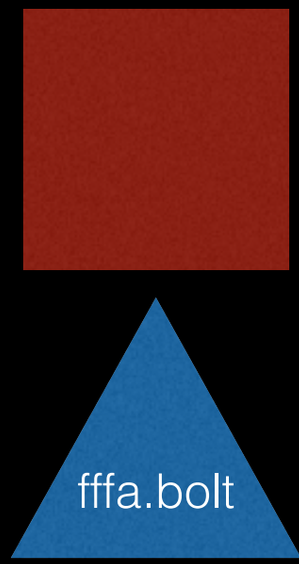
adjacent segments merged concurrently



adjacent segments merged concurrently



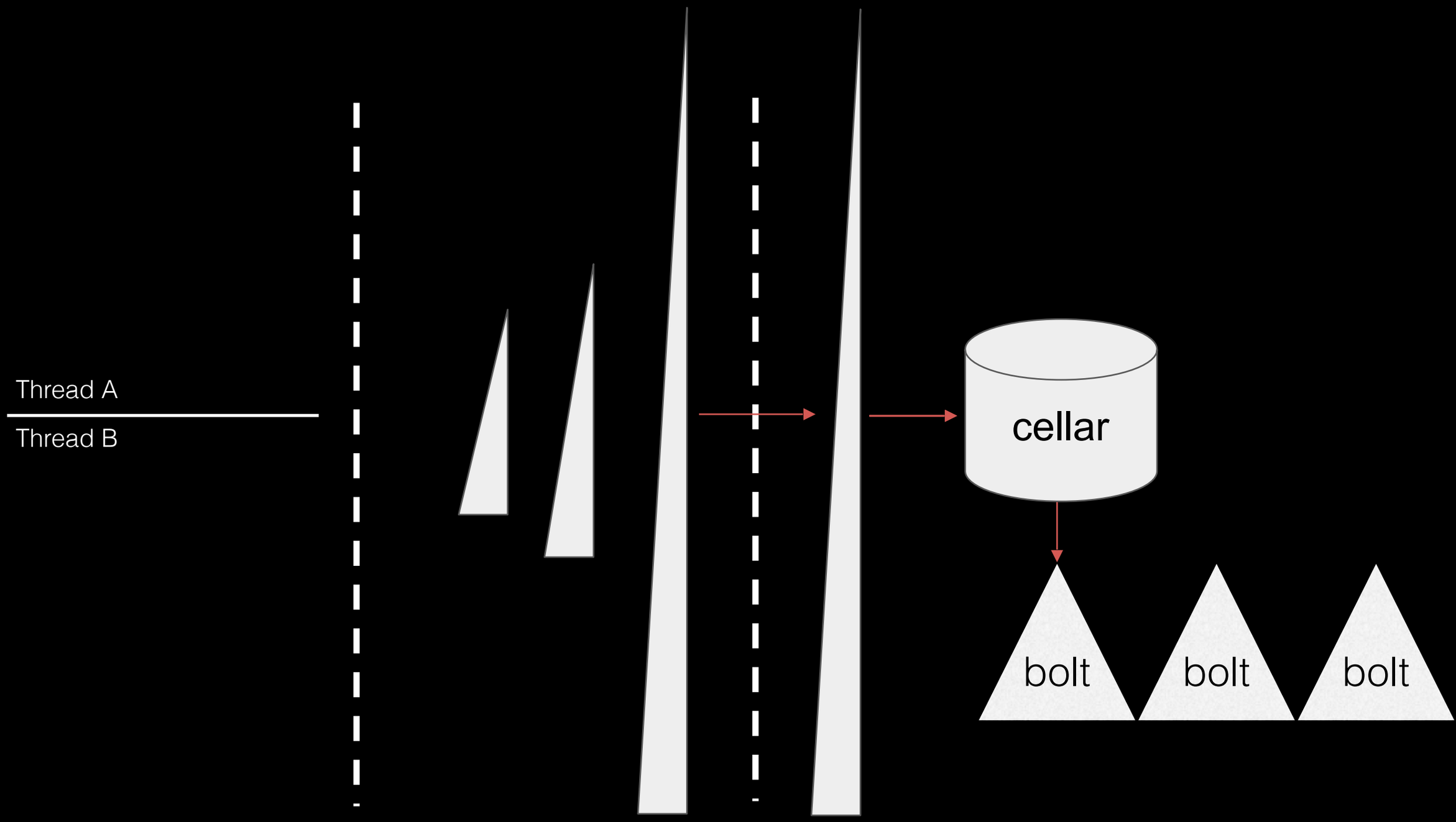
eventually everything is merged back to single bolt



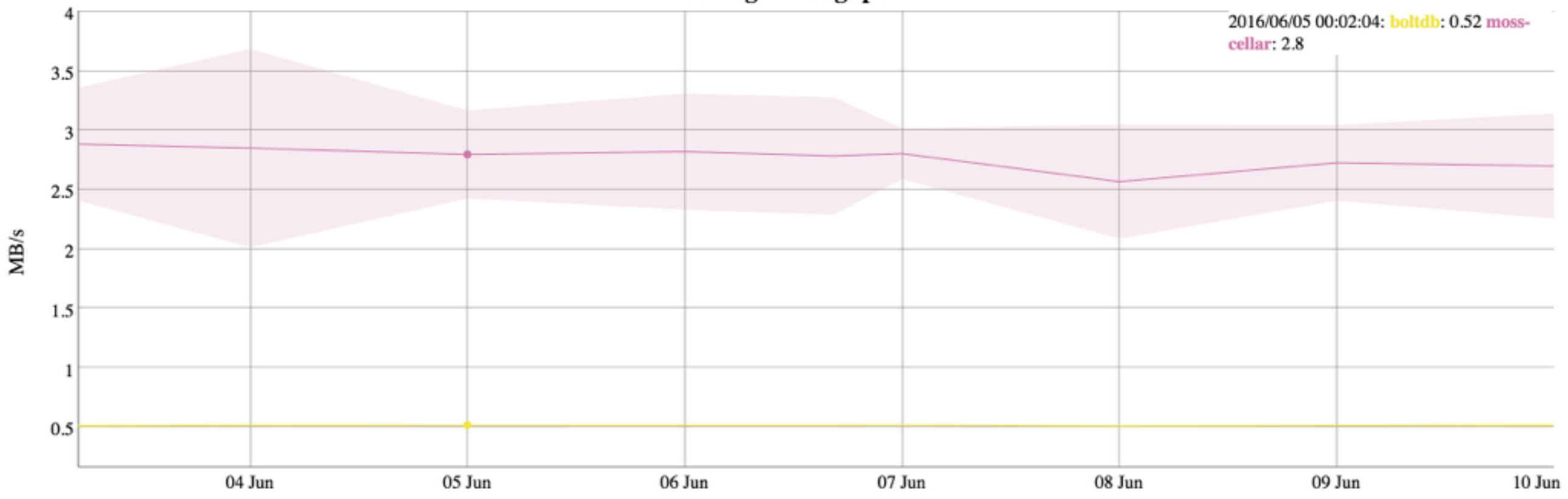
moss

memory
oriented
sorted
segments

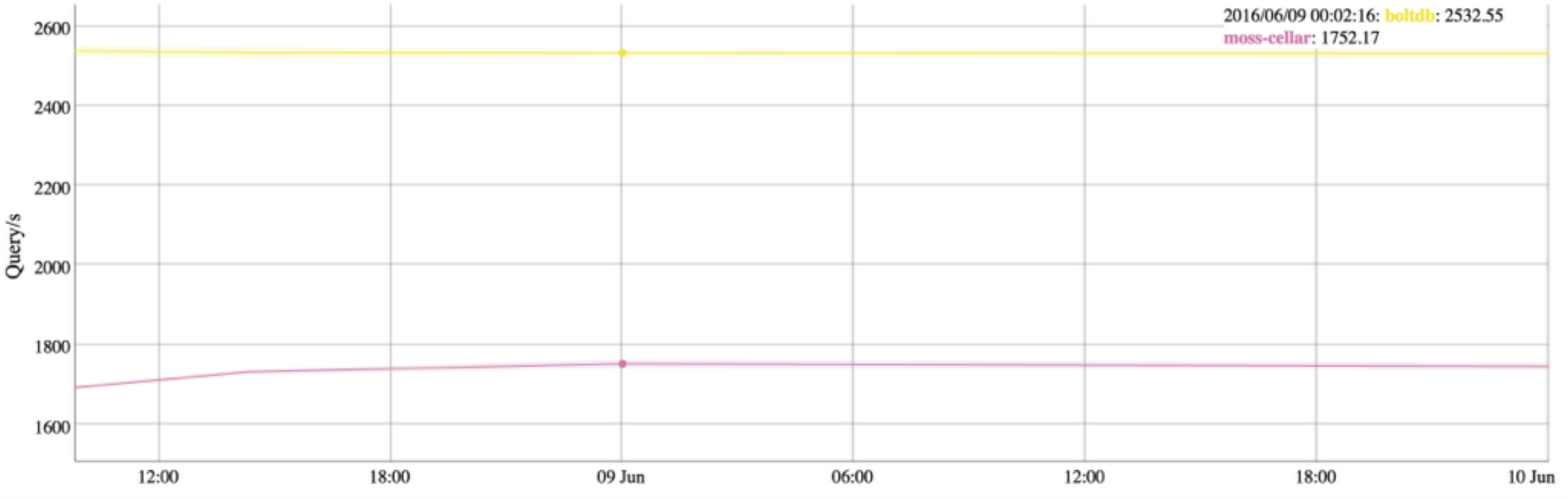
the moss/cellar fit



Indexing Throughput



Term Query (low)



moss + cellar
root cellar

Root Cellar

- Moss - https://github.com/couchbase/moss
- Bolt - https://github.com/boltdb/bolt
- Cellar - https://github.com/couchbaselabs/cellar

- @mschoch - @blevesearch - blevesearch.com