UNIVERSITY OF SOUTHERN CALIFORNIA

CS402 Assignment 01 – Coarse Grained Locking

Michael Gerow

February 22, 2013

1 Coarse Grained Locking

The coarase grained locking for the database is quite simple. We only have one global mutex for the entire tree and lock and unlock it from within the query, add, and xremove functions.

1.1 Testing

I have a few methods for testing coarse grained. First I tried simply calling 250 instances of the build in test2 and unpausing them all at the same time just to see if it ran though. After that I constructed a test that adds a series of numbers to the database, queries them, and then deletes them all. I then ran this with 250 instances. Because of the way the adds and removes are set up we should end up with an empty database at the end no matter what order the clients executed in, and sure enough we do.
This specific test can be run using this command (from within the mt_db directory):

```
./server_coarse < test/250_add_remove | tail -n 900 | less
```

The tail and less command are only there to grab the final dump of the database so that we can verify that the database is in fact empty.
Of course, manually scanning all these results can be a little tiring and can take a long time, so I wrote a little python script (called test.py) that will take care of this for you. It runs all three of the locking methods five times each and verifies that the database is in the correct state afterwards and prints "PASSED" if it is good. Otherwise it prints "FAILED".