Project milestone report

Sean Zhou, Ye Lu

Project progress

- We have finished reading the related papers and make our design & implementation decisions
- We have finished the implementation of fine-grained locking BST with hand-over-hand locking based on c++ mutex
- We have finished the implementation of lock-free BST with GCC atomic builtin function `__sync_bool_compare_and_swap` as the CAS operation prototype
- We have implemented a basic test harness based on c++11 threads

Period	Goal
☑ ~~11.3 - 11.8~~	paper reading and implementation decisions discussion
☑ ~~11.9 - 11.15~~	implementation of fine-grained locking BST
☑ ~~11.16 - 11.21~~	implementation of lock-free BST

Future schedule

- During the poster session, we plan to show our benchmark results for two different BST implementations under different workloads. The demo form will mainly be with diagrams and tables showing performance results as well as some conceptual graphs explaining how the implementations work.
- The future work will be writing a test harness program to generate different kinds of workloads and apply the workload to the BST implementations. The benchmark result will be the speedup we achieved as more cores are being used.

☐ 11.22 - 11.25	Implement harness program and design workloads (Sean)
☐ 11.26 - 11.29	Preliminary correctness test with harness program (Ye)
☐ 11.30 - 12.2	Benchmark on fine grained lock BST and optimization (Sean)
☐ 12.3 - 12.5	Benchmark on lock-free BST and optimization (Ye)
☐ 12.6 - 12.9	writing final report and prepare for poster session (Ye, Sean)

Goals and Deliverables

We believe we're on track to achieve the goals in our proposal. So far we have been keeping up well with our proposed schedule in finishing both BST implementations. And additionally we've written a basic test harness and run some correctness tests. So we believe we're at least enroute to finishing all the "plan to have" features in our proposal.

The next steps would be to build up our correctness / performance test programs in order to appropriately check and benchmark our BST implementations, and to fix any correctness / performance bugs we find. If this goes well, we'll likely have time to work on implementing concurrent self-balancing, otherwise we'll focus on debugging and getting convincing results for the vanilla BST.

Here is the new list of goals we plan to hit from now before the poster section:

Plan to have

- Extend correctness test harness to ensure the correctness of our implementations and fix any bugs found
- Build up performance test harness to benchmark and compare the two implementations under different workloads, debugging any correctness oddities
- Collect results on GHC and PSC machines. Analyze the results and create graphs

Hope to have

- Implement fine-grained self-balancing BST
- Implement lock-free self-balancing BST
- Benchmark and compare the self-balancing BST implementations

Risks and Mitigation

Potentially there may be subtle concurrent bugs that are not found until our more extensive tests in the following weeks. These bugs may be hard to recreate and catch, eating up our time.

We should thus strive to test as soon as possible and be prepared for debugging.