Distributed Key-value store based on Braft (Braft-KV)

Structure

- server.cpp: server program
- client.cpp: client program for testing purpose
- **./config.sh**: configuration file for "run_server.sh", run_server.sh will `source` this script
- **./run_server.sh**: testing script for executing server clusters
- **./stop.sh**: testing script for stopping all running server process
- **/adapter**: backend database adapter
  - adapter.h: general adapter interface
  - redis_adapter.cpp/h: Redis adapter implementation
- **/redis**: util script for executing Redis instances and the corresponding configuration files
- /test: some testing scripts (not updated for multi-fields)
- /depercated: depercated classes

Usage

Build

```
mkdir build
cd build && cmake ..
```

Test on local machine

Under this example, we will start n=3 server nodes for testing. Note that the cluster will not progress if n < 2.

1. Start n Redis instances (default port: 40000, 40001, 40002), you may follow the session "Test with Redis"
2. Start n braft-kv instances

   ```
   $ cd build
   $ ./run_server.sh
   ```

3. Start testing client program

   ```
   $ ./kvstore_client
   ```

4. Perform your action

   ```
   # Perform put
   # format: put <key> <field1> <value1> ...
   ```

```
Action <get|put|del>: put stocks tsla 900 spce 30
I0222 01:56:23.228399 99041 /home/tfwong/research/braft-
kv/example/kvstore/client.cpp:187] field: tsla value: 900
I0222 01:56:23.228482 99041 /home/tfwong/research/braft-
kv/example/kvstore/client.cpp:187] field: spce value: 30

# Perform get
# format: get <key> <field1> <field2> ...
Action <get|put|del>: get stocks tsla
I0222 01:56:28.894104 99041 /home/tfwong/research/braft-
kv/example/kvstore/client.cpp:139] <GET>:
I0222 01:56:28.894131 99041 /home/tfwong/research/braft-
kv/example/kvstore/client.cpp:142] field: tsla value: 900

# Perform del
# format: del <key> (only support delete whole key)
Action <get|put|del>: del stocks
```

5. Stops all server nodes

```
$ ./stop.sh
```

Test with Redis

For n server nodes, we have to start n Redis instance as each server nodes' backend.

1. For the first time, please modify fields in each Redis configuration file in `/redis/<node id>`
   - pidfile: `<path of redis dir>/<node id>/redis-server.pid`
   - logfile: `<path of redis dir>/<node id>/redis-server.log`
   - dir: `<path of redis dir>/<node id>`
2. Start all redis instances

```
$ ./start.sh
```

3. Stop all redis instances (also clear all data in nodes)

```
$ ./stop.sh
```

Test with YCSB

To benchmark the performance of Braft-KV, we have implemented an YCSB interface for it.

Since the building of a single YCSB interface will require the whole YCSB, so we developed our interface in a separated repository.

Repository: YCSB

Testing procedure

1. Clone the repository

```
$ git clone git@bitbucket.org:ivanwong_237/ycsb.git
$ git checkout braftkv
```

2. Build YCSB core and braft-kv-binding

```
$ chmod a+x ./util.sh
$ ./util.sh build-all
```

3. Run YCSB shell with braft-kv-binding

Usage of YCSB shell: [Usage]

> of course you have to start the braft-kv according to above procedure first

```
$ ./util.sh shell
YCSB Command Line client
Type "help" for command line help
Start with "-help" for usage info
Loaded group configuration: name: KVStore conf:[/127.0.1.1:8102:0,
/127.0.1.1:8100:0, /127.0.1.1:8101:0]
Asking peer for leader: list://127.0.1.1:8102
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further
details.
Update leader: /127.0.1.1:8101:0
Connected.
# Perform insert (put)
> insert stocks MSFT=180 AAPL=315
Result: OK
6 ms
# Perform read (get)
> read stocks MSFT AAPL
Return code: OK
MSFT=180
AAPL=315
3 ms
# Perform delete (del)
> delete stocks
Return result: OK
3 ms
>
```

4. Run a workload test

```
$ ./util.sh run workloada
Command line: -t -db site.ycsb.db.BraftKVClient -p braft-
kv.group_conf=127.0.1.1:8100:0,127.0.1.1:8101:0,127.0.1.1:8102:0, -p braft-
kv.group_name=KVStore -P /home/tfwong/research/ycsb/workloads/workloada
YCSB Client 0.18.0-SNAPSHOT

Loading workload...
Starting test.
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further
details.
DBWrapper: report latency for each error is false and specific error codes
to track for latency are: []
```

```
Loaded group configuration: name: KVStore conf:[/127.0.1.1:8100:0,
/127.0.1.1:8102:0, /127.0.1.1:8101:0]
Asking peer for leader: list://127.0.1.1:8100
Update leader: /127.0.1.1:8101:0
[OVERALL], RunTime(ms), 7650
[OVERALL], Throughput(ops/sec), 130.718954248366
[TOTAL_GCS_PS_Scavenge], Count, 1
[TOTAL_GC_TIME_PS_Scavenge], Time(ms), 10
[TOTAL_GC_TIME_%_PS_Scavenge], Time(%), 0.130718954248366
[TOTAL_GCS_PS_MarkSweep], Count, 0
[TOTAL_GC_TIME_PS_MarkSweep], Time(ms), 0
[TOTAL_GC_TIME_%_PS_MarkSweep], Time(%), 0.0
[TOTAL_GCs], Count, 1
[TOTAL_GC_TIME], Time(ms), 10
[TOTAL_GC_TIME_%], Time(%), 0.130718954248366
[READ], Operations, 510
[READ], AverageLatency(us), 1457.8196078431372
[READ], MinLatency(us), 480
[READ], MaxLatency(us), 14351
[READ], 95thPercentileLatency(us), 3891
[READ], 99thPercentileLatency(us), 5827
[READ], Return=OK, 510
[CLEANUP], Operations, 1
[CLEANUP], AverageLatency(us), 2206720.0
[CLEANUP], MinLatency(us), 2205696
[CLEANUP], MaxLatency(us), 2207743
[CLEANUP], 95thPercentileLatency(us), 2207743
[CLEANUP], 99thPercentileLatency(us), 2207743
[UPDATE], Operations, 490
[UPDATE], AverageLatency(us), 3810.74693877551
[UPDATE], MinLatency(us), 1709
[UPDATE], MaxLatency(us), 36959
[UPDATE], 95thPercentileLatency(us), 7575
[UPDATE], 99thPercentileLatency(us), 17631
[UPDATE], Return=OK, 490
```