ZCache Simulator

Keshane Gan

16 December 2015

Part I
Preliminary

There are several files included in this directory: `runsims.py`, `lruqueue.h`, `lruqueue.c`, `cache.c`, `RHcache.h`, `zcache.h`, `Makefile`. `runsims.py` runs the program for each cache configuration for each cache type for each file. `make` can be run to compile the program.

Part II
Data Structures

The structure of the cache itself is simulated simply by a two-dimensional array of ZBlocks (specified in `zcache.h`. Each ZBlock holds an address, a time stamp, and an array of (four) hashed values. Each of the hashed values point to a different ZBlock like so: `ZBlock[h][i]`, where `h` is the hashed value and `i` is the index in the array of that hashed value.

Part III
Methods and Implementation

File Reading

To improve performance and to prevent machine failure, the program will only accept gzipped files.

Hashing

I used the MD5 hash function for the hashing. The address as an integer type in C is run through the hash function and certain bits of the hash is used to index into the cache.
There were two methods that I explored.
The first method was to divide the MD5 digest value into four parts and to extract four different hash values that way. The second method was to increment the address by 1, 2, 3, or 4 and perform an MD5 hash on each of those to get the four needed hash values. Both methods have very small outcome difference.

Allocation

Once the hash values have been found, the `zallocate` function decides the location to place a block. It first checks to see if any of the hash values point to a block that has the address that it's trying to allocate. If it

does find such a block, it counts as a hit and the time_stamp of the block is updated. If not, a depth-first search is performed. For the final program, I have declared the depth at which it will search to be 4 because it was performing too poorly with a depth of 3. The path to the oldest block is saved. Then the oldest block is written over by the second-to-last block in the path and so forth until the new block is allocated.

Part IV
Results and Conclusions

Results are shown on the last page.
Overall, the zcache performed poorly, even considered against the Robin Hood cache. There are a few instances where the zcache performs slightly better, like in the libquantum trace for Configuration B. For the most part, however, z-cache has about twice as many misses as the standard and Robin Hood cache, except for the gcc trace, where there's only a 10% difference.
One problem that needs to be taken into consideration is that the MD5 hash doesn't produce a significantly uniform distribution of numbers. A sample distribution of hashes from 0 to 2047 of memory addresses from the `gcc-10M` trace is included below. A hash function that produces more evenly distributed values would probably improve hit performance. In practice however, such a function would probably take too long to realistically use in a machine. More improvements could be made.

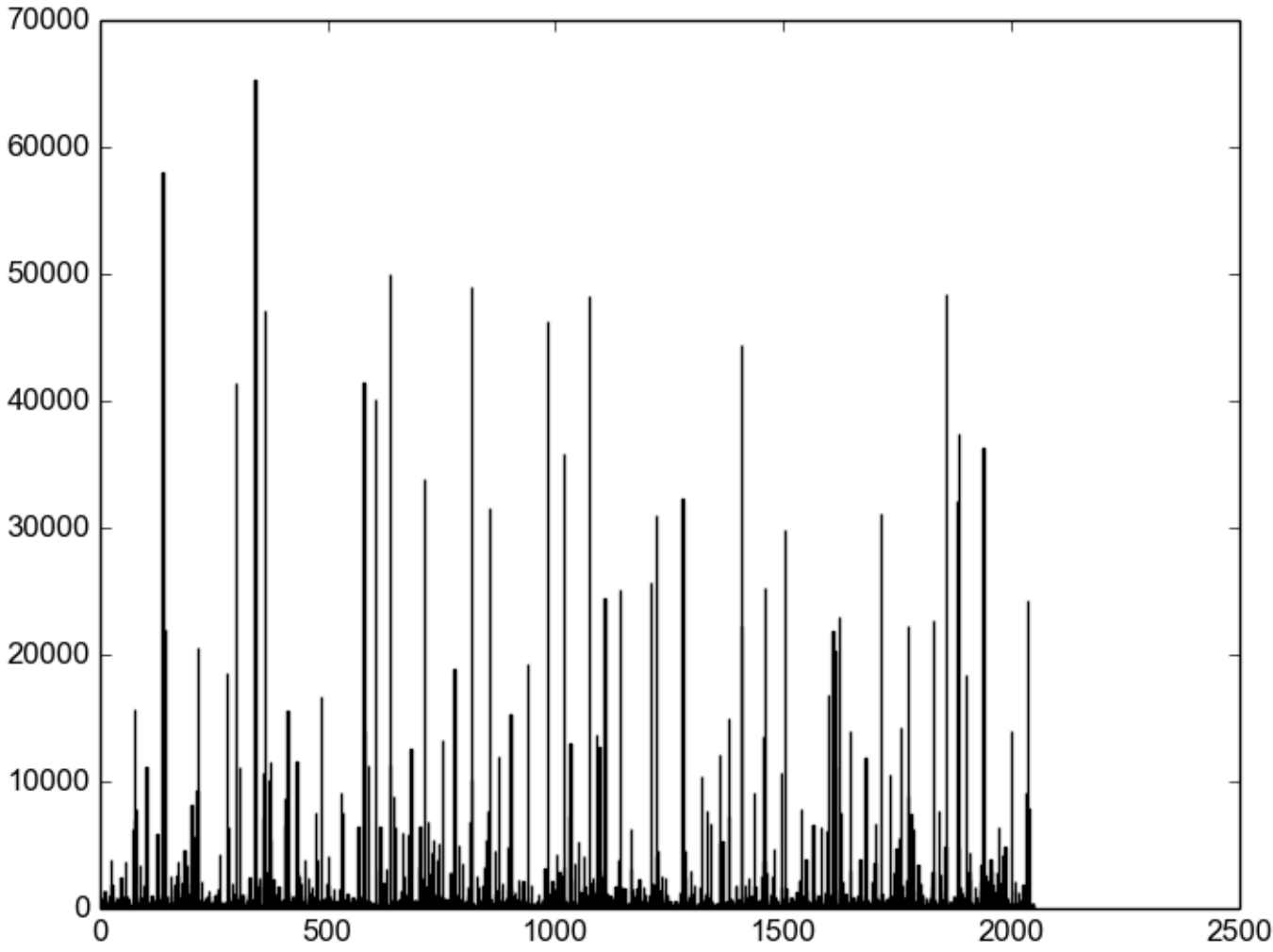
70000
60000
50000
40000
30000
20000
10000
0
0
500
1000
1500
2000
2500

Table 1: Misses in Configuration A

file	standard	robinhood	zcache
art-100M.trace-ld-st.txt.gz	18259018	18259018	18258196
gcc-10M.trace-ld-st.txt.gz	22491	21980	25871
gcc-50M.trace-ld-st.txt.gz	95974	115251	114551
go-100M.trace-ld-st.txt.gz	47245	49299	212278
hmmer-100M.trace-ld-st.txt.gz	2543164	2579800	5652263
libquantum-100M.trace-ld-st.txt.gz	8382092	8382110	8382126
mcf-100M.trace-ld-st.txt.gz	10390280	11248599	10222203
sjeng-100M.trace-ld-st.txt.gz	125469	77268	160488
sphinx3-100M.trace-ld-st.txt.gz	92866	92955	249486

Table 2: Misses in Configuration B

file	standard	robinhood	zcache
art-100M.trace-ld-st.txt.gz	16952575	16952575	18136652
gcc-10M.trace-ld-st.txt.gz	19524	19410	23886
gcc-50M.trace-ld-st.txt.gz	84113	85155	105111
go-100M.trace-ld-st.txt.gz	14626	13000	24740
hmmer-100M.trace-ld-st.txt.gz	2148104	2490786	5029502
libquantum-100M.trace-ld-st.txt.gz	8382092	8382092	8371235
mcf-100M.trace-ld-st.txt.gz	9522613	10405146	9672753
sjeng-100M.trace-ld-st.txt.gz	61825	61687	139613
sphinx3-100M.trace-ld-st.txt.gz	66631	66645	247965

Table 3: Misses in Configuration C

file	standard	robinhood	zcache
art-100M.trace-ld-st.txt.gz	11385760	11591514	9787551
gcc-10M.trace-ld-st.txt.gz	19025	19025	23860
gcc-50M.trace-ld-st.txt.gz	80363	80310	99476
go-100M.trace-ld-st.txt.gz	12255	12248	21950
hmmer-100M.trace-ld-st.txt.gz	1864839	2058534	4287880
libquantum-100M.trace-ld-st.txt.gz	49029	104607	6689463
mcf-100M.trace-ld-st.txt.gz	8957091	9664976	9279149
sjeng-100M.trace-ld-st.txt.gz	57408	57440	130953
sphinx3-100M.trace-ld-st.txt.gz	65898	65898	189460