Cache simulator c++

Jacob Sundqvist (jacsu246)

User guide:

My cache simulator runs on my raspberry pi (raspbian). Should work for other unix systems such as ubuntu or debian but that is untested.
Statistics will be written into the third argument of the program (lru_stats.txt or ttl_stats.txt in the examples below).

To compile use the provided makefile:
> make

To run the simulator use:
LRU:
> cat input.txt | ./msc option=0 x=50000 y=2 lru_stats.txt > output.txt
TTL:
> cat input.txt | ./msc option=1 y=2 t=5 ttl_stats.txt > output.txt

Input data:

The input data in gen_input.txt and gen_input_big.txt has been generated using the provided matlab script gen_requests.m.

gen_requests.m

This scripts generates all the characteristics using different distributions:

- Time_stamps: Exponential distribution
- Servers: Uniform distribution
- Clients: Uniform distribution
- Files: Uniform distribution (Should have been ZIPF but I did not get that working in time)

gen_input.txt

- Number of servers: 10
- Number of clients: 1000
- Number of files: 100
- Number of requests: 10000

gen_input_big.txt

- Number of servers: 50
- Number of clients: 1000
- Number of files: 1000
- Number of requests: 100000

Output data:

The output data will specify what happens to each requests. Is it a cache hit or cache miss? This file is mainly for debugging.

Statistical data:

This file shows cache statistics such as hit rate, miss rate, number of requests and uncached requests.

Tests:

Test1:

Test1 uses the gen_input.txt as input. This test is intended as a base test and test2 is intended to improve these base results.

LRU:

This test uses the following parameters for LRU: X=50000, Y=3 and yields the following result:

```
################################
Cache stats using LRU scheme
Cache_size: 50000
Priority 3
Timelimit 0
Statsfile: ./test1/lru_stats.txt
################################
Total requests: 10000
Total cache requests: 6380
Total cache hits: 1030 (16.1442%)
Total cache misses: 5350 (83.8558%)
Total unstored: 3620
################################
```

TTL:

This test uses the following parameters for TTL: Y=3, T=30 and yields the following result:

```
################################
Cache stats using TTL scheme
Cache_size: 2147483647
Priority 3
Timelimit 30
Statsfile: ./test1/ttl_stats.txt
################################
Total requests: 10000
```

Total cache requests: 6380
Total cache hits: 18 (0.282132%)
Total cache misses: 6362 (99.7179%)
Total unstored: 3620
################################

Test2:

Test2 uses the gen_input.txt as input. Has been formed to improve the results from test1. For LRU this meant to increase the cache size. For TTL this meant drastically increasing the allowed time for a cache line before it expires.

LRU:

This test uses the following parameters for LRU: X=50000, Y=3 and yields the following result:
################################
Cache stats using LRU scheme
Cache_size: 500000
Priority 3
Timelimit 0
Statsfile: ./test2/lru_stats.txt
################################
Total requests: 10000
Total cache requests: 6380
Total cache hits: 5740 (89.9687%)
Total cache misses: 640 (10.0313%)
Total unstored: 3620
################################

TTL:

This test uses the following parameters for TTL: Y=3, T=10000 and yields the following result:
################################
Cache stats using TTL scheme
Cache_size: 2147483647
Priority 3
Timelimit 10000
Statsfile: ./test2/ttl_stats.txt
################################
Total requests: 10000
Total cache requests: 6380
Total cache hits: 2397 (37.5705%)
Total cache misses: 3983 (62.4295%)
Total unstored: 3620
################################

Test3:

Test 3 uses a very large input file and I was unable to complete the statistics in time.

Results:

From the test1 and test2 we can clearly see that a bigger cache will increase the cache hit ratio for LRU and a larger time interval before cache eviction for TTL will increase the cache hit ratio.

Discussion:

From my experiments I can clearly see that my assumptions about file distributions and time_stamps are naive. Both schemes suffer from this by a reduced hit ratio. The TTL scheme is based upon non-even distributions which makes my assumptions for the input result in worst-case scenarios.