CS3500 Lab 7

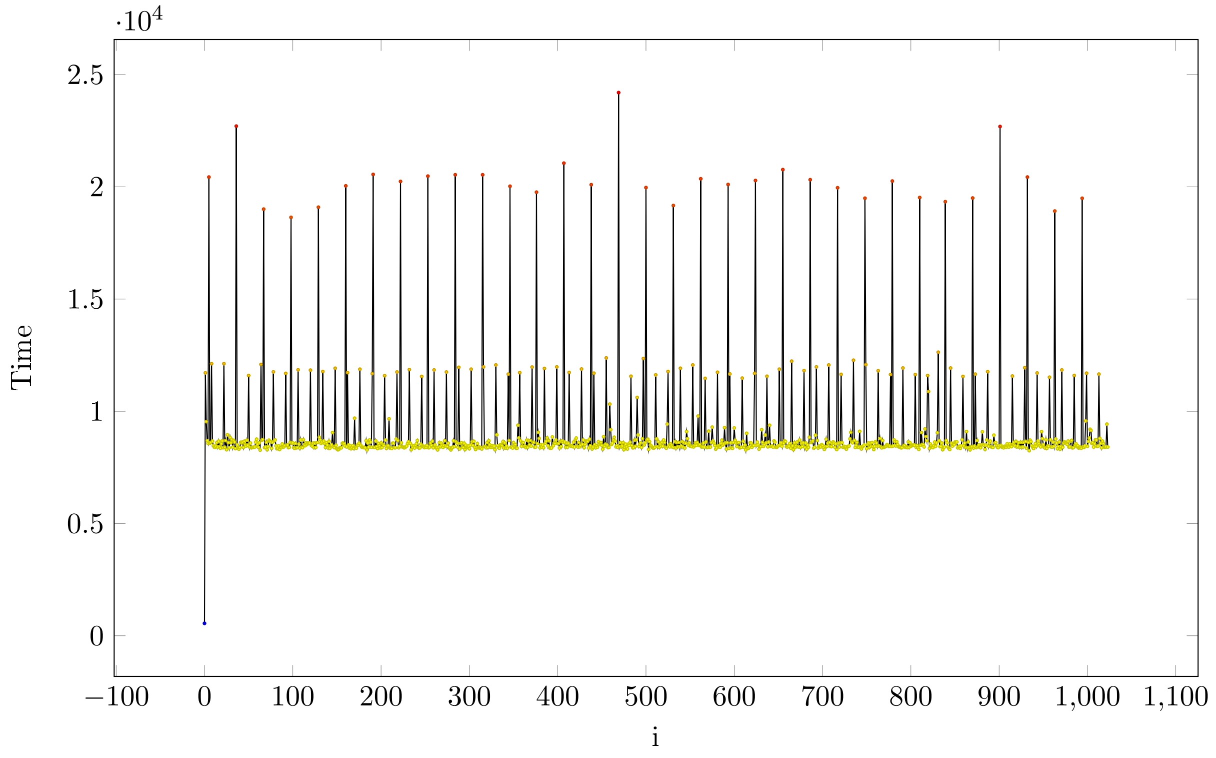
Harshit Kedia, CS17B103

Date: 20th Sep 2019

1.
   - Total time taken by first three accesses = 546 (mean) and 9955 (variance)
   - Total time taken by second three accesses = 9234 (mean) and 3634037 (variance)
   - Total time taken by third three accesses = 440 (mean) and 8452 (variance)
   - 1st is less than 2nd because stack is being used due to variable declarations, hence that page/block of data is present in cache. When we want index 1500, 3000, 4500, each of these values are present in distinct pages (1 page can contain 1024 integers only), hence all of these are brought from main memory to cache one by one.
   - 3rd is less than 2nd because when we ask for these higher indexed values again, these (block of memeory, not page though) are already present in cache and page table entry is present in TLB, hence lower cycles due to hit!

2.
   - Time taken for 10 nops: 366
   - Time taken for 100 nops: 602
   - Time taken for 1000 nops: 604
   - Time taken for 2000 nops: 456
   - Time taken for 4000 nops: 596
   - Time taken for 5000 nops: 594
   - Reasoning: The code size for higher value of "nops;" will cross 4096 bytes, hence the jump label will be on a different page than the rest of the function, thus leading to a address translation procedure for a different page. It was expected to get different values for higher value for higher number of nops than that for lower number of "nops;" due to the mentioned reason.

3.
   - Plot of variation in time
   - Reasoning on the plots: The first page is already present in cache due to usage the head pointer locally. So when we read an element from first page, it is accessed pretty fast. But afterwawrds when we try reading data from the other pages, they are brought from the main memeory, hence larger time taken. Within these Page address translations, there will be TLB L1, TLB L2 misses/ flushes. Hence the spikes on the plot. Due to priciple of locality, we bring a chunk of page entries into TLB L3, thus getting a few hits and not all misses.
   - Conclusions on size of TLB: Level 2 has 16 entries(oscillation of spikes of around 12000 cycles after every 14 values of N), but since some fixed entries are present for OS, so 16 entries in L2 sounds better(also power of 2). For Level 3 in TLB, 32 entries, as the plot oscillates/spikes periodically after 32 (around 20000 cycles). TLB L1 size cannot



be estimated because during every translation call, there will be a L1 miss (we never call a page again). Seeing L2 and L3, estimated size of L1 TLB will be around 4-8 entries.

4.
- Total time taken by first three accesses = 688580 (mean) and 843362164 (variance)
- Total time taken by second three accesses = 683 (mean) and 1492 (variance)
- Total time taken by third three accesses = 525 (mean) and 16479 (variance)
- 2nd is less than 1st because during 1st 3 reads, the entire MMAP'd huge page needs to be brought to physical memeory, hence taking huge time. But since that page contains the entire array of 8192 elements, the mean time to get these is reduced drastically.
- 3rd is less than 2nd because these and nearby data (block) will be present in cache and hence will take lesser cycles to access.
- Contrast to Question 1: Since the entire huge page is brought at once to TLB, the translation takes very less time as compared to Q1, where both the TLB and Cache will be misses. The only difference is observed in 2nd and 3rd three accesses dude to cache hit.