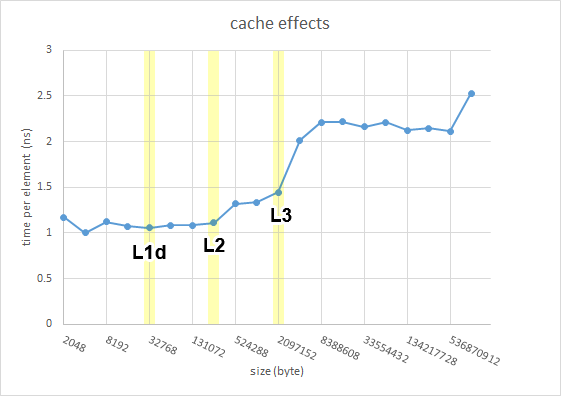
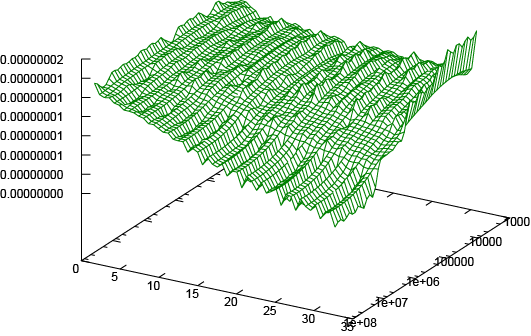
**시스템 프로그래밍 숙제1 보고서**

2013-11392 김지현

숙제 1은 일부러 메모리 캐시 미스를 유도하여, 캐시미스에 따른 계단형의 그래프를 그리고 그래프로 추정가능한 캐시의 크기와 실제 캐시의 크기를 비교하는것이 목표이다. 실험에 사용된 CPU는 **Intel i5-2556M** 이고, 결과는 아래와 같았다.



노란 선은 실험이 끝난 후, CPU의 실제 캐쉬 크기를 그래프상에 표시한것이다. L1d 캐쉬의 경우 캐시미스로 얻는 성능손실이 너무 미미해서 그래프상에 거의 드러나지 못하고, L2 캐시미스와 L3 캐시미스의경우 속도차이가 확연한것을 알 수 있다. L2 캐시미스가 일어났을경우 메모리 읽기/쓰기 시간이 **25.7%**, L3 캐시미스의경우 **57.2%** 더 들어가는것을 측정할 수 있었다. 남는 여분의 시간을 활용하여, 아래와 같은 Memory Mountain 그래프를 그릴 수 있었다.



*가로축: stride 크기, 세로축: 전체 memory 크기, 높이: 스루풋 (1 나노초당 처리할 수 있는 element 갯수)*

*부록*

아래는 첫번째 그래프를 그리는데에 사용된 코드이다. **Rust**로 짜여졌다. Rust는 llvm으로 컴파일되는 정적 언어이며 GC를 쓰지 않아, 실행에 있어 C와 아무런 차이가 없다. Memory mountain을 그리는데에 쓰인 코드 또한 GitHub(아래 링크)에서 확인할 수 있다.

https://github.com/simnalamburt/snucse.mountain

|  |
| --- |
| #![feature(std\_misc)]  extern crate time;  use time::PreciseTime;  fn main() {  println!("size,t");  for n in 10..30 {  test(n)  }  }  fn test(exp: usize) {  let stride = 16;  let size = 2 << exp;  let mut arr = vec![0u8; size];  let mask = size - 1; // x & mask == x % arr.len()  let t0 = PreciseTime::now();  for i in 0usize..64\*1024\*1024 {  arr[(i \* stride) & mask] += 1;  }  let t1 = PreciseTime::now();  let t = t0.to(t1).num\_nanoseconds().unwrap();  println!("{},{}", size, t);  } |