COSC 301: Operating Systems

Lab 7: The memory hierarchy

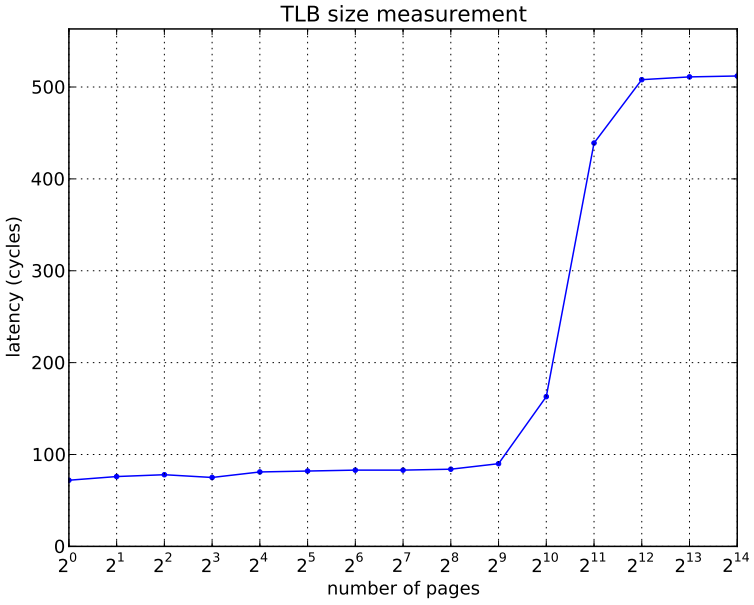
For this lab, we will investigate performance characteristics of the memory hierarchy in a computer system[#ostep]_.

There are two graphs to consider in this lab (shown below, posted on Moodle, and distributed in color in lab). Our goal for this lab is to *infer* characteristics about the memory hierarchy of a computer system based on measurements shown in the graphs.

In the following, remember that a CPU typically has multiple levels of caches, each of differing sizes, along with some number of TLB entries, which may also be in some hierarchy. These caches all exist in an effort to reduce the number of times we need to access main memory, which can be slow. Consider that the CPU has to access main memory for:

1. A data cache miss. Assuming that there is a TLB hit and we obtain the correct virtual->physical address translation, the data may or may not be available in one of the caches (L1, L2, or possibly L3), so we may need to access main memory for the data.
2. A TLB miss, but a cache hit. In this case, if there is no translation cached in the TLB for the virtual->physical mapping, we have to walk the page tables, which requires access to main memory. Once the correct mapping is located, the data requested may actually be in cache. (Note also that the cache line containing a page table entry may also be one of the CPU caches.)
3. A TLB miss, and a cache miss. If there is no virtual->physical mapping cached in the TLB, we have to walk the page tables which results in main memory access. If the requested data are also not in a cache, we have to retrieve the data from main memory as well.

The TLB



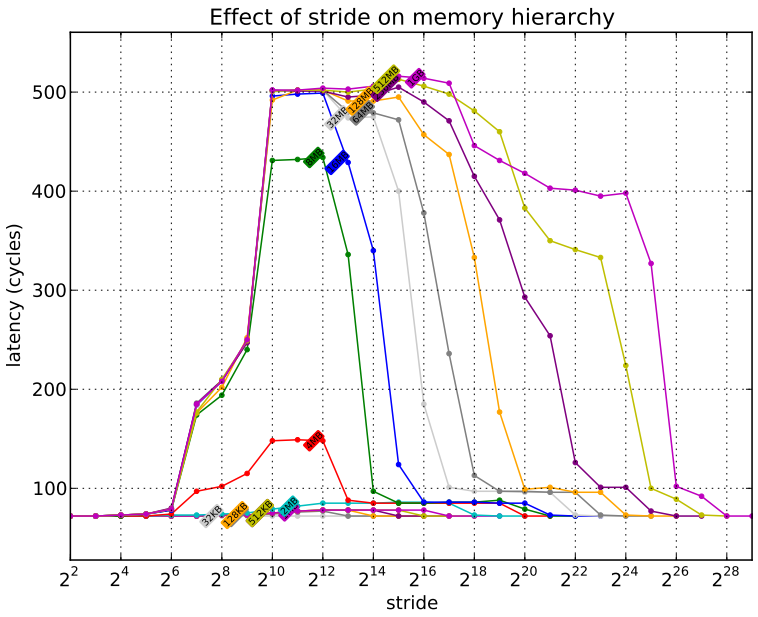
This graph was produced by a program (`tlb.c`, in the lab github repo) that creates a large array (n times the CPU page size, for some large n), and benchmarks the number of CPU cycles required to access the first four bytes of some number of consecutive pages. We vary the number of consecutive pages from 1 (2^0) to 16384 (2^14).

Note that the y axis is the number of CPU cycles required to read the four bytes of each page, on average. That is, it is not measured directly in *time*, but rather in CPU cycles.

Discuss the following questions:

1. How many TLB entries are there on the computer system from which this graph was produced?
2. Are there multiple levels of TLBs? How do you know one way or the other?
3. Assuming that the CPU runs at 3 GHz, about how many nanoseconds is the main memory access time?

Full memory hierarchy



This graph was produced by a program (`stride.c`, in the lab github repo) that does the following (in pseudocode):

```
for an array size in the range 32KB (2^15) to 1GB (2^30), by power of 2:

    for stride length from 1 to array size:

        measure the average memory access time (in CPU cycles) when
        "striding" across the array at the given stride length
```

Note that this graph was produced on the *same* computer system as the first graph. Any insights gained from analyzing the first plot are applicable to this graph. The page size for this CPU is 4KB (4096 bytes, or 2^12 bytes). There are separate curves plotted for each array size. Each point on each curve corresponds to a different stride length for a given array.

If you think about it, as we vary the array size, we will be varying the number of pages required to hold the array, as well as varying the fraction of the array that can be held in a given cache level (recall from COSC 201 that data are retrieved from memory at the granularity of a cache line, which typically varies from 8 bytes to 128 bytes, or more).

Discuss the following questions:

- What is the cache line size? How can you tell?

- How large are data caches? Can you infer that there are multiple levels of data caches? How can you tell?
- How many TLB entries do there appear to be? Are there multiple levels of TLBs? How can you tell? Does it match the inferred number from the first graph?
- Assuming that the CPU runs at 3 GHz, about how many nanoseconds is the main memory access time?
- Can you infer anything about the associativity of TLBs or data caches?

[1] This lab is based on the homework assignment in <`http://pages.cs.wisc.edu/~remzi/OSTEP/vm-tlbs.pdf`>.