Parallel Functional Programming Homework 4

Prof. Stephen A. Edwards, Columbia University

Due November 3rd, 2024

You will need GHC and Threadscope installed on your computer to do this assignment.

Record your answers **in a PDF file** and submit it on Courseworks.

Collect all the data for this assignment on the same computer. You will likely want to use it to develop your class project. I recommend *not* doing this in a virtual machine, which will likely skew your results.

1. Figure out which processor you are using. Report the brand, model, the number of cores, and the number of hardware threads. For example,

Brand	Intel
Model	Core i7-9700
Cores	8
Hardware Threads	8

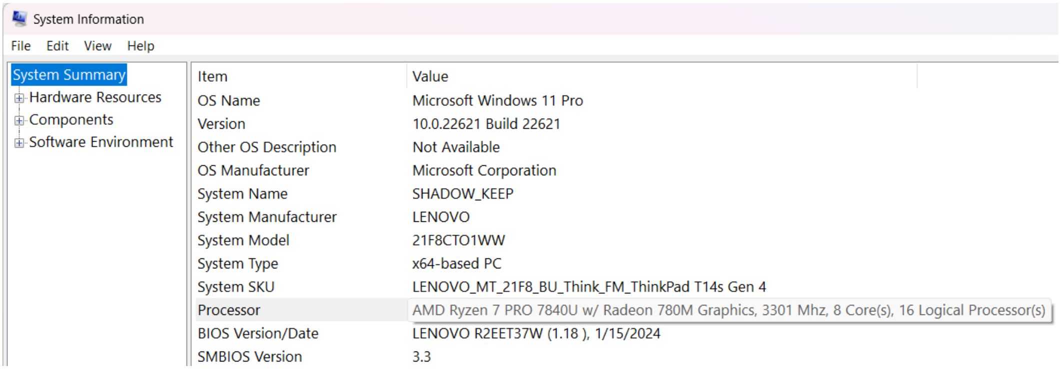
A core is a mostly-separate processor on a chip. Many modern processors use simultaneous multithreading (SMT, Intel markets it as "Hyperthreading") in which a single core interleaves the execution of two or more threads at the cycle level, which often is as efficient as running the threads on separate cores. We will consider the count of hardware threads as the available level of hardware parallelism.

Learn the brand and model of your processor and look up technical details of it online. E.g., Googling "i7-9700" leads me to Intel's website with details of the processor *https://www.intel.com/content/www/us/en/products/sku/191792/intel-core-i79700-processor-12m-cache-up-to-4-70-ghz/specifications.html*

Under Linux, `lscpu` reports this information concisely:

```
$ lscpu
Architecture:            x86_64
  CPU op-mode(s):        32-bit, 64-bit
  Address sizes:         39 bits physical, 48 bits virtual
  Byte Order:            Little Endian
CPU(s):                  8
  On-line CPU(s) list:   0-7
Vendor ID:               GenuineIntel
  Model name:            Intel(R) Core(TM) i7-9700 CPU @ 3.00GHz
    CPU family:          6
    Model:               158
    Thread(s) per core:  1
    Core(s) per socket:  8
    Socket(s):           1
```

Under Windows, run *System Information* and look at the Processor entry:



Item	Value
OS Name	Microsoft Windows 11 Pro
Version	10.0.22621 Build 22621
Other OS Description	Not Available
OS Manufacturer	Microsoft Corporation
System Name	SHADOW_KEEP
System Manufacturer	LENOVO
System Model	21F8CTO1WW
System Type	x64-based PC
System SKU	LENOVO_MT_21F8_BU_Think_FM_ThinkPad T14s Gen 4
Processor	AMD Ryzen 7 PRO 7840U w/ Radeon 780M Graphics, 3301 Mhz, 8 Core(s), 16 Logical Processor(s)
BIOS Version/Date	LENOVO R2EET37W (1.18), 1/15/2024
SMBIOS Version	3.3

This particular processor has 8 cores with two-way SMT, giving 16 hardware threads, as confirmed by the AMD website *https://www.amd.com/en/products/processors/laptop/ryzen/7000-series/amd-ryzen-7-7840u.html*

Under Mac OS X, open a terminal and run `sysctl`:

```
% sysctl -a machdep.cpu
machdep.cpu.cores_per_package: 8
machdep.cpu.core_count: 8
machdep.cpu.logical_per_package: 8
machdep.cpu.thread_count: 8
machdep.cpu.brand_string: Apple M1
```

2. Collect performance information and Threadscope graphs for nfib2 from the lecture (add a main function that runs nfib2 40):

```
import Control.Parallel(par)

nfib2 :: Integer -> Integer
nfib2 n | n < 2 = 1
nfib2 n = par nf (nf + nfib2 (n-2) + 1)
  where nf = nfib2 (n-1)
```

Report the following information:

(a) The execution time on a single thread (-N1)

(b) The execution time on all the threads you have (e.g., -N8 if your computer has 8 hardware threads)

(c) A speedup graph from 1 thread to all your hardware threads where speedup is 1 under a single thread, 3 if the program takes 1/3 of the time it took with a single threads.

(d) A Threadscope graph of the activity under all the threads you have (e.g., -N8 -ls)

(e) Statistics on the number of sparks created, converted, overflowed, etc. under all the threads you have (e.g., -N8 -s).

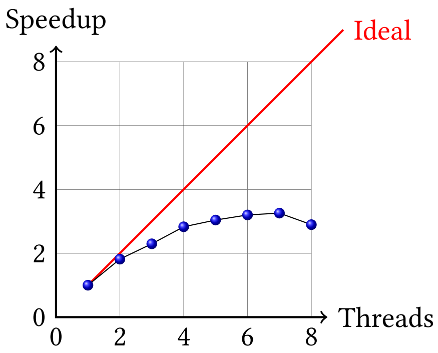
(f) A Threadscope graph of spark creation, conversion, and pool size for a single thread when running with all hardware threads

Here is an exemplary answer, taken from the lecture slides:

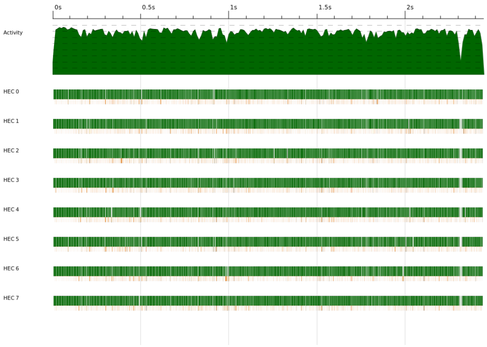
(a) 7.312s

(b) 2.524s

(c)

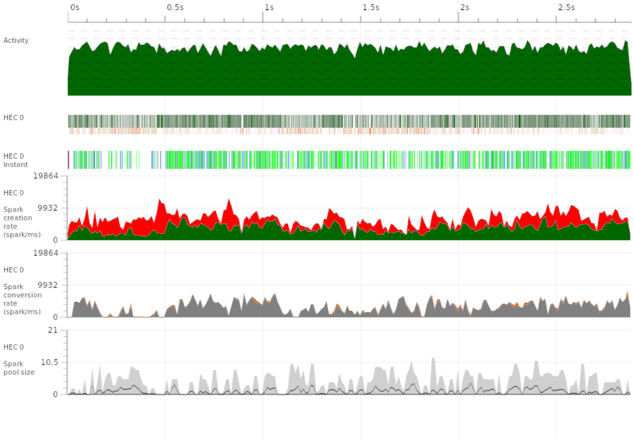


(d)



(e) `SPARKS: 194364898 (10671 converted, 121033805 overflowed, 0 dud, 2594777 GC'd, 46510093 fizzled)`

(f)



3. Collect performance information and Threadscope graphs for `nfib4` from the lecture. Add a `main` function that runs `nfib4 depth 40` for a depth provided as a command-line argument.

   ```
   nfib4 :: Int -> Int -> Integer
   nfib4 0 n          = nfib n
   nfib4 _ n | n < 2 = 1
   nfib4 d n = nf1 ‘par‘ nf2 ‘pseq‘ nf1 + nf2 + 1
     where nf1 = nfib4 (d-1) (n-1)
           nf2 = nfib4 (d-1) (n-2)

   nfib :: Int -> Integer
   nfib n | n < 2 = 1
   nfib n = nfib (n-1) + nfib (n-2) + 1
   ```

   Report the following information, in each case using all the hardware threads your processor provides:

   (a) The number of sparks total, converted, GC’ed, and fizzled for `nfib4 12 40`

   (b) A speedup graph over depths 1–13 where the speedup is the total time (sum over all threads) divided by the elapsed time (time to completion) as reported by `-s` in “Total time” line.

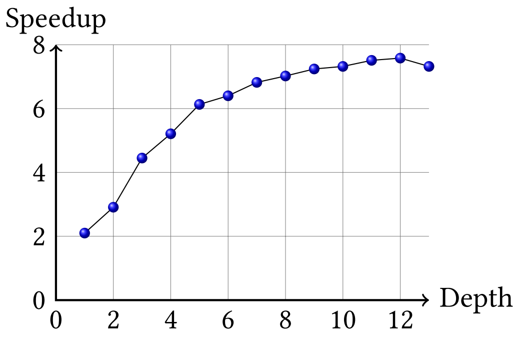
   (c) A graph from Threadscope for depth 4 showing 16-way parallelism with some load balancing issues.

Here is an exemplary answer, taken from the lecture slides from last year:

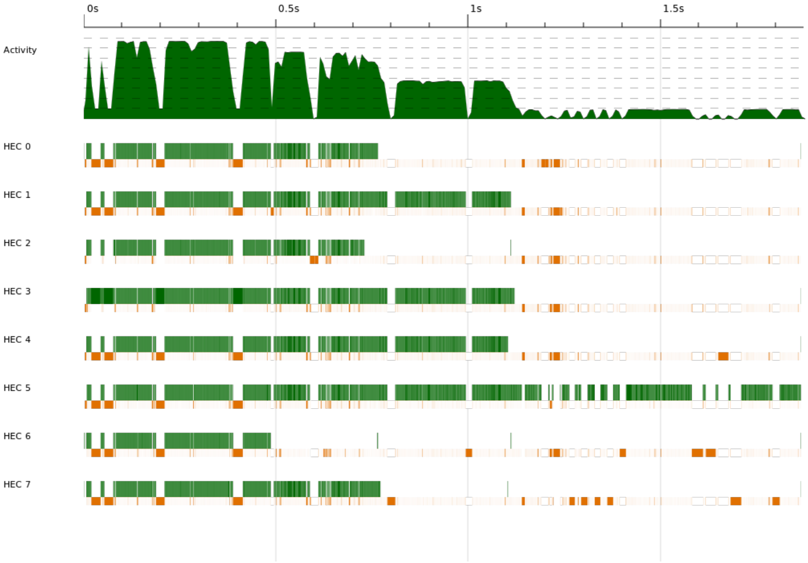
(a)

```
SPARKS: 4106 (160 converted, 0 overflowed, 0 dud,
436 GC'd, 3509 fizzled)
```

(b)



(c)


0s
0.5s
1s
1.5s
Activity
HEC 0
HEC 1
HEC 2
HEC 3
HEC 4
HEC 5
HEC 6
HEC 7