Recitation 4: Cilk and Cilktools

This recitation explains how to use Cilk Plus to improve the performance of programs on multicore processors, as well as how to use Cilk screen race detector, and Cilk view scalability analyzer.

1 Getting started

We recommend that you work on course machines. Since we will use gnuplot, you should enable X11 forwarding by using -X option.

```
% ssh -X username@cloudN.csail.mit.edu
```

To get a local copy of the repository for your work, you need to use git to clone it.

```
% git clone /afs/csail.mit.edu/proj/courses/6.172/student-repos/recitation4/username recitation4
```

2 Cilk Plus

2.1 cilk_spawn and cilk_sync

Compile a bad fib program in the fib directory. Then, time the execution for finding fib(45) by running

```
% time ./fib 45
```

You will get 3 different times as outputs: real, user, and sys. Real is wall time, which is what you see in a clock. User is a CPU time spent in user-mode. Sys is a CPU time spent in kernel. user + sys is the actual CPU time of the command.

Since the current fib is a serialize program, you will find that wall time is slightly higher than CPU time. Next, we want to parallelize the program to take advantage of the other 11 processors in the cloud machines.

You can do this by adding cilk_spawn in front of function calls that you want to do in parallel. You also need to add cilk_sync to wait for all spawning tasks to finish. Lastly, you should include Cilk header using

```
#include <cilk/cilk.h>
```

- **Exercise:** Parallelize fib using cilk_spawn and cilk_sync, and time the output binary.

You should find that the new version is faster than the first one (wall time). However, CPU time is higher than wall time since we use multiple processors to run the program.

2.2 cilk_for

transpose is an in-place matrix transpose program. You can replace for with cilk_for to parallelize the outer loop.

- **Exercise:** Parallelize transpose using cilk_for, and time the output binary with input size 10000.
3 Cilk screen race detector

Compile and run a simple program qsort-race in the qsort-race directory. There is a race in this program. We can use Cilk screen to detect the race by running

```
% cilkscreen ./qsort-race
```

- **Exercise:** Fix this race.

4 Cilk view scalability analyzer

qsort is a quick sort program from project 2.1. You can use Cilk screen to analyze the scalability of the program. Run the following command to get a graph of speed-up for different number of threads.

```
% cilkview -trials all 20 ./qsort.64
```