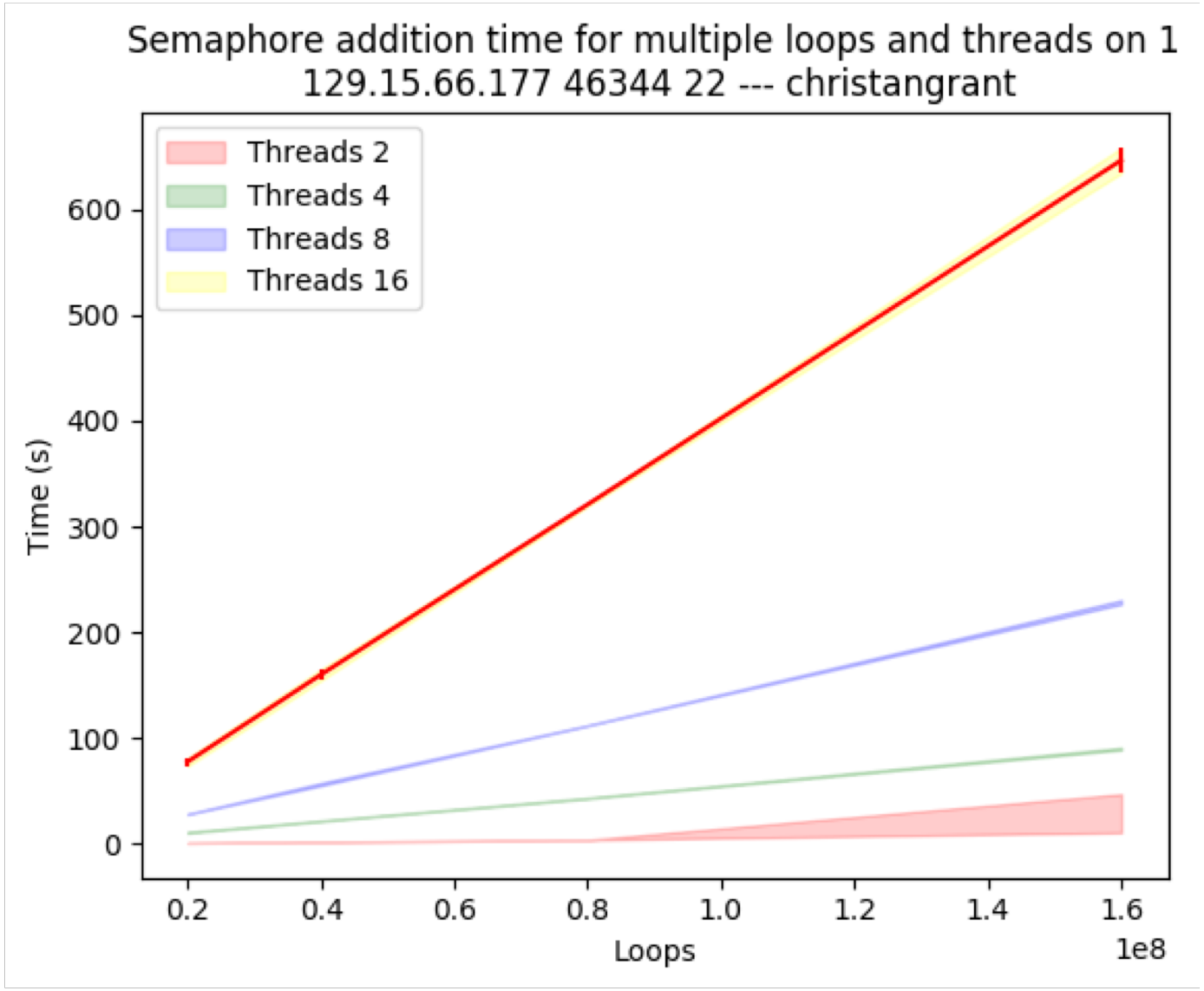
Semaphores and Threads
HW4

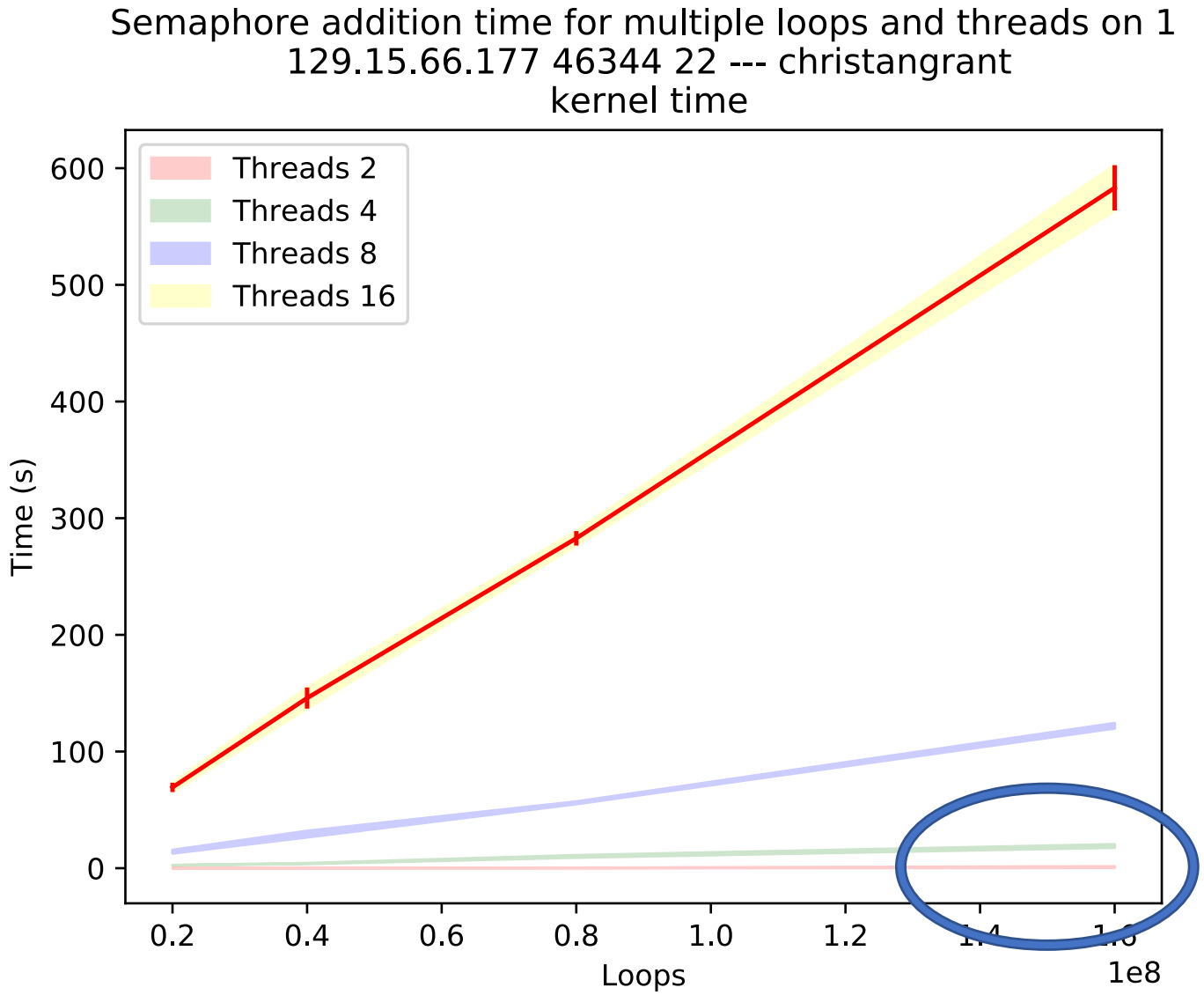


More threads are not always better

There may be an increase in management for new threads.

(Same graph but) Sys/Kernel time

Time spent in the kernel is more deterministic



Time

- There are three main ways of recording time in *nix systems.
  - Real time → This is the amount of time as measured by a real life clock.
  - User Time → The amount of time spent in user mode.
  - Sys Time → The amount of time spent in kernel mode.
- Time is recorded on a per-process basis.
- Timing is stored within the process itself.
- When a child thread is terminated, its timing statistics are saved in the process structure.
- User and Sys times are recorded across CPUs so User + Sys times may be > than the real time.

Maintaining Mutual Exclusion

- Used a (binary) semaphore
- Other options:
  - Mutex Lock http://man7.org/linux/man-pages/man3/pthread_mutex_lock.3p.html
  - Spin Lock http://man7.org/linux/man-pages/man3/pthread_spin_lock.3.html

Integer overflow

- 160000000 * 16 is 2,560,000,000

```
christangrant@myosinstance:/$ getconf INT_MAX
2147483647
```

- Better?

```
christangrant@myosinstance:/$ getconf ULONG_MAX
18446744073709551615
```

- Adding long types may be slower.

Other ways to increase program speed

- Use more than one resource to reduce competition.
- Tune parameters program size.
- If adding beyond billions reconsider atomic actions.
- Others?