



# Lecture 18: One-sided MPI

**CMSE 822: Parallel Computing**  
**Prof. Sean M. Couch**





# Puppy time!



**How it started**






**How it's going**





# PCA Questions

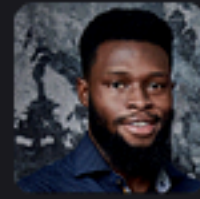
 **Matthew Zeilbeck** 10:33 PM  
PCA15: The thread that creates a group of tasks is allowed to execute them, but how long does task scheduling take? Is task creation/scheduling nearly always much faster than task execution? Is execution generally close to being done when all of the scheduling is done to the point that there is very little for the thread that created the tasks to execute? Also how much overhead is there with tasks?

 3 

- Implementation and algorithm specific!
- See [https://link-springer-com.proxy1.cl.msu.edu/chapter/10.1007/978-3-642-30961-8\\_24](https://link-springer-com.proxy1.cl.msu.edu/chapter/10.1007/978-3-642-30961-8_24)



# PCA Questions



Josué Kpodo 10:58 PM

PCA15: In section 20.2.1, it says that "if a thread is in one critical section, the other ones are all blocked". Does this mean that a mutual exclusion is basically a global lock then?



- Yes.

## Critical Sections

A critical section protects against multiple accesses to a block of code. When a thread encounters a critical section, it only enters when no other threads are in any critical section, that one or others. The following example uses an unnamed critical section.

```

1  #pragma omp critical
2
3  {
4
5  if (max < new_value)
6
7  max = new_value
8
9  }
```

- <https://software.intel.com/content/www/us/en/develop/articles/more-work-sharing-with-openmp.html>

Global, or unnamed, critical sections may unnecessarily impact performance because every thread is effectively competing for the same global critical section. For that reason, OpenMP has named critical sections. Named critical sections allow for more fine-grained synchronization so only the threads that need to block on a particular section will block. The following example improves on the previous example.

```

1  #pragma omp critical(maxvalue)
2
3  {
4
5  if (max <
6
7  new_value)
8
9  max = new_value
10
11 }
```

With named critical sections, applications can have multiple critical sections, and threads can be in more than one critical section at a time. It is important to remember that entering nested critical sections runs the risk of deadlock, which OpenMP does not detect. When using multiple critical sections, be extra careful to examine those that may be "hiding" in subroutines.



# MPI One-sided Communication

- Watch lecture by Bill Gropp: <https://www.youtube.com/watch?v=CW2EDMRyEH8>



# Final Projects

- Join appropriate Slack channel!