ulm university universität
uulm

Institut of Numerical Mathematics
Dr. Andreas F. Borchert and Dr. Michael C. Lehn
Mladjan Radic

16 November 2016
Quiz 6

High Performance Computing I (WS 2016/2017)

Deadline: 23 December 2016, 2pm

Notice:

Please type your responses in a simple text file named "quiz06.txt" and submit it on our server *thales* using the following command:

```
thales$ submit hpc quiz06 quiz06.txt
```

Question 1

What is the purpose of the classes *std*::*lock_guard* and *std*::*unique_lock*? In which cases is the class *std*::*lock_guard* sufficient, where is *std*::*unique_lock* required?

Question 2

In our *ThreadPool* class, the method *submit* calls *cv.notify_one*() while the destructor uses the method *cv.notify_all*(). When to use which? Would the *ThreadPool* class work if submit would use *cv.notify_all*() instead? Would the *ThreadPool* class work as expected if the destructor would use *cv.notify_one*() instead? If not, why not?

Question 3

Our last version of the *ThreadPool* class uses the type *std*::*packaged_task*<*T*()> for its jobs instead of *std*::*function*<**void**()> as in our previous implementations. This is the implementation of the *submit* method of the same class:

```
template<typename Task>
std::future<T> submit(Task task) {
    Job job(task);
    auto result = job.get_future();
    std::unique_lock<std::mutex> lock(mutex);
    jobs.push_back(std::move(job));
    cv.notify_one();
    return result;
}
```

Is *std*::*move* required here, i.e. would *jobs.push_back*(*job*) work as well? And if not, why not?