CSCI 5525 Project Proposal

1. Team Members: Aaron Holt and ?
Note: I'm turning this in now more for the deadline than anything. Typing this out left me wondering how feasible this idea is. I am still looking at other ideas.

2. Determinism is an important property which makes the design, implementation, and debugging of parallel programs far simpler. Many programs are intended to be deterministic but subtle problems can lead to nondeterministic outcomes. This project will explore locking mechanism to ensure deterministic parallel programming.

3. Adding parallelism to code can create unintended bugs in code. Take this C example compiled using OpenMP:

```
#include <stdio.h>
int main(void)
{
  #pragma omp parallel
   printf("Hello, world.\n");
   return 0;
}
```

Run on a 2-core machine, the desired output is likely:

```
Hello, world
Hello, world
```

However, due to the race condition the output could be something like:

```
Hello, Hello, wworld.
orld.
```

The race condition between the two threads sharing standard output makes the outcome nondeterministic. The scope of this project will include simple parallel loops such as the one above that include racy conditions. Some of these could be trivial examples such as summing up integers (where determinism isn't always necessary), but the goal will be to make user defined parallel loops deterministic.

4. There are a few methods I'm looking into. The first is a "type and effect" system in which determinism is guaranteed by compile-time checking [1] [2]. In this system, the programmer partitions the heap and specifies which methods have read/write access to different parts of the heap. (Most of the work done on this is in Java, and so unless I can find work to build on in C/C++ I will likely avoid this).
The other possibility is locking, where certain resources are withheld from threads, guaranteeing they finish in the right order.

5. The "type and effect" method has already been partially implemented in Java with good results. The deterministic code produced had similar performance to equivalent nondeterministic code. I'm still looking researching locking implementation.

6/7. Success will be demonstrated by taking unintentionally nondeterministic programs and making them deterministic. Determinism will be shown by analyzing memory accesses when using "type and effect" method. Otherwise using probability (running the program thousands of times) could give evidence for deterministic outcomes.

Bibliography

[1] U. o. I. a. Urbana-Champaign, "A Type and Effect System for Deterministic Parallel Java," 2009.

[2] E. Westbrook, R. Raman, J. Zhao, Z. Budimlic and V. Sarkar, "Dynamic Determinism Checking for Structured Parallelism," 2012.