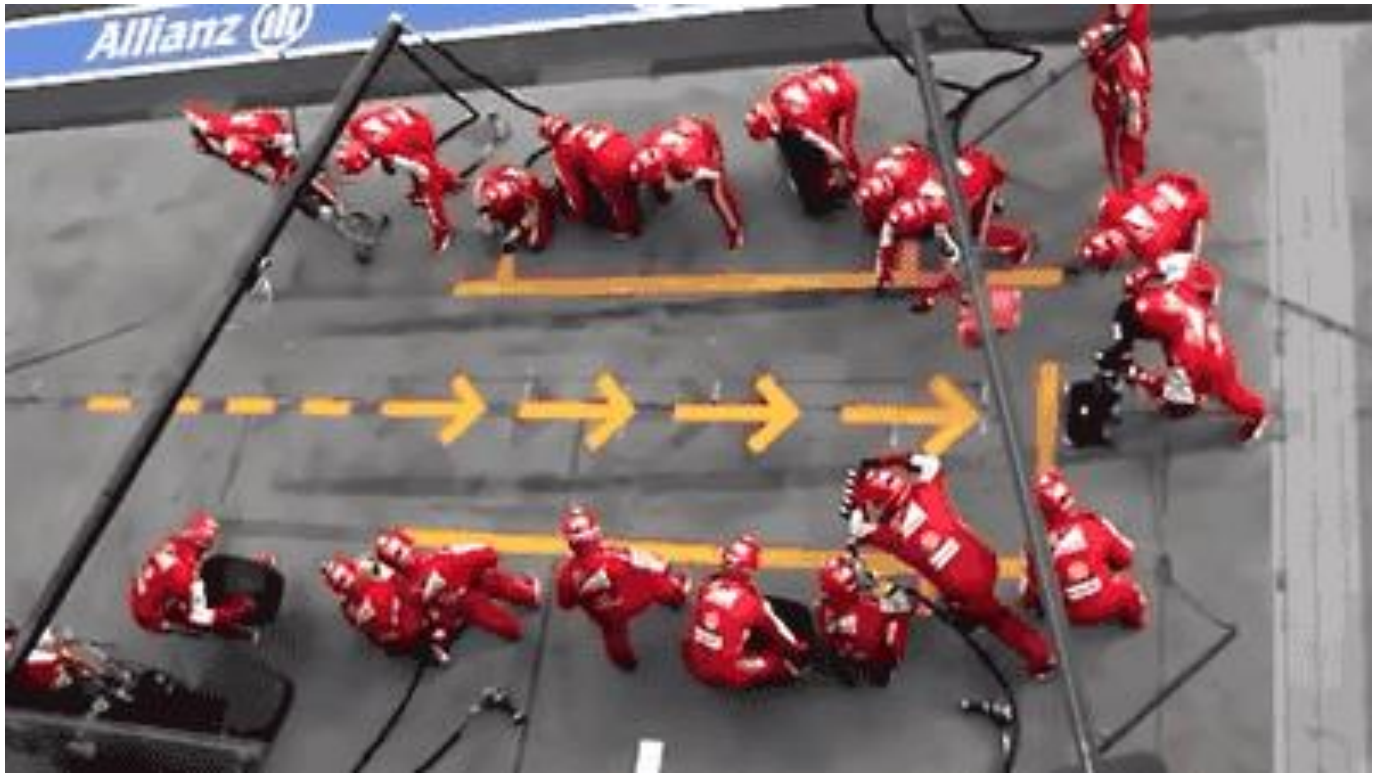
Multi-Threading I

Andrew Hu



Administrivia

- Attendance is still http://tinyurl.com/uwigem/18sp/attendance/
- Pacman projects due tonight
  - Looking for just a minimum viable product
  - If a significant amount of progress has been made by deadline, it may be extended

Project Updates

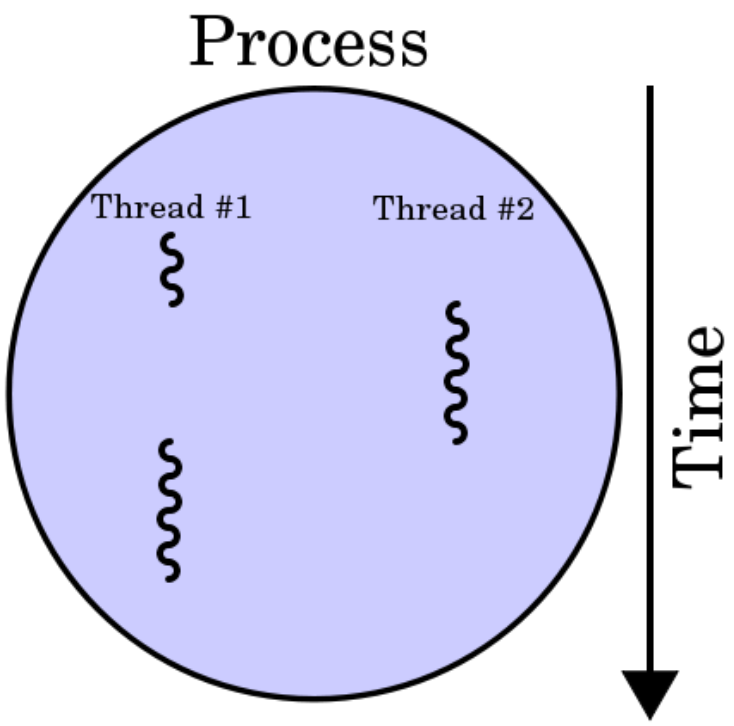
- Current order of project ideas
  1. DAWGMA Arduino Project
  2. Chromastat overhaul
  3. Video Game for outreach
- Other options
  - Microfluidics project (whatever shape this takes on)
  - CS-Research with comp bio labs

What is multi-threading?

- More cores?
- More Chrome thingies on Task Manager?
  - (Seriously what's up with that???)
- More windows on screen at once?

Multi-Threading w/o Multiple Cores

- Multi-threading predates multiple core CPUs
- A way to force the OS to give “equal” time to different parts of the same program
- Not actually running at the same time, just fast switching between threads



Motivating Example: Word Processing

- We need to handle keyboard input, plus the rest of the program logic
- What happens when we do a grammar check? What if we keep typing?
  - Note, most input libraries fix this problem, but in designing systems we can't assume things will just work

```
void main() {
  while (true) {
    // Look for keyboard input by waiting until
    // a key is pressed
    char input = nextKeyPress();

    // Write it to the file
    file.write(input);

    // If it's the end of a sentence, grammar
    // check the last sentence
    if (input == '.') {
      // This takes a really long time!
      grammarCheck(file);
    }
  }
}
```

Operating System

- Speed gains not *just* thanks to more cores
- Operating system allows
  - Creating different “threads” for the same program
  - Controlling threads across different cores
  - Preventing threads from writing to the same data at once
  - Scheduling which threads get time on the CPU

Software Interface

- What does a multi-threaded program even look like?

```
//Program start
//Create data to tell your new
//thread what to do
int message = 42;
//Create a new child thread
//with this message
runChild(message);

print("Hello from the parent");
```

Software Interface

- What does a multi-threaded program even look like?

```
//Program start
//Create data to tell your new
//thread what to do
int message = 42;
//Create a new child thread
//with this message
runChild(message);

print("Hello from the parent");
```

```
/*DOESN'T START UP HERE*/

//Goes from fork();

print("Hello from the child: " +
      message);
```

What gets printed?

- Let's find out!
- Code is here: http://tinyurl.com/uwigem/18sp/documents/multithread1.zip
- Compile from command line with
  - `javac Parent.java`
  - `java Parent`
- What did it print out?

```
0
100
Hello from the child: 42
200
300
400
500
Hello from the parent
```

Break it down!

Load the thread with our message, and start it

Kill time while printing out some numbers

Print out the parent's message

```
public class Parent {
  public static void main(String[] args) {
    // Create the message
    int message = 42;

    // Start the child thread
    Child childThread = new Child(message);
    childThread.start();

    // Twiddle our thumbs
    for (int i = 0; i < 5000; i++) {
      if (i % 1000 == 0) System.out.println(i);
    }

    // Print from the parent
    System.out.println("Hello from the parent");
  }

  public static class Child extends Thread { ... }
}
```

Break it down!

Make the child thread class inherit from Thread

The run() method gets run when we call Thread.start()

```
public class Parent {
  public static void main(String[] args) { ... }

  public static class Child extends Thread {

    private int message;
    public Child(int message) {
      this.message = message;
    }

    /*THIS IS THE SPECIAL METHOD*/
    public void run() {
      // Print our message
      System.out.println("Hello from the child: " +
                this.message);
    }
  }
}
```

Word Processing: Fixed

```
void main() {
  createThread(GrammarChecker);
  while (true) {
    char input = nextKeyPress();

    // Write it to the file
    file.write(input);

    if (input == '.') {
      sendMessage(grammarChecker, 1);
    }
  }
}
```

```
class GrammarChecker {
  // Running in a separate thread
  void run(int message) {
    while (true) {
      waitForMessage();
      if (message == 1) {
        checkGrammar();
      }
    }
  }

  void checkGrammar() {
    // Do the grammar check
    // ...
  }
}
```

Multi-Threading Experiment

- When is the *child thread*'s message printed, again?
- What happens when we remove the for loop that wastes time?
- What happens when we make the for loop much shorter?

```
0
100
Hello from the child: 42
200
300
400
500
Hello from the parent
```

Message Passing

```
void main() {
  createThread(GrammarChecker);
  while (true) {
    char input = nextKeyPress();

    // Write it to the file
    file.write(input);

    if (input == '.') {
      sendMessage(grammarChecker, 1);
    }
  }
}
```

```
class GrammarChecker {
  // Running in a separate thread
  void run(int message) {
    while (true) {
      waitForMessage();
      if (message == 1) {
        checkGrammar();
      }
    }
  }

  void checkGrammar() {
    // Do the grammar check
    // ...
  }
}
```

Both threads already running, but they can communicate!

Message Passing Experiment

- Code here: http://tinyurl.com/uwigem/18sp/documents/multithread1.zip
- Creates two threads, and does work in both of them
- Once the child thread is finished, it sends a message back

Parallelism vs Concurrency

- Two words for using the same thing differently
- Parallelism
  - Using multiple threads to do one operation faster
- Concurrency
  - Using multiple threads to do different things at the same time

Test

- Test

Hello, there!

```
int main() {
        // Code goes here
}
```