Introduction to the Jarvis Cluster

1. Request an Account

To request an account please visit the following URL:
http://bluesky.cs.iit.edu/jarvis

user: iit
pass: iit2014

2. Connecting to Jarvis

To connect to Jarvis please use an ssh client.
By default <user> is the first portion of your <user>@hawk.iit.edu email address.
Example skrieder@hawk.iit.edu
<user>=skrieder

ssh <user>@jarvis.cs.iit.edu

Windows:
Putty/Cygwin
skrieder@jarvis.cs.iit.edu

Linux/Unix/Mac:
ssh skrieder@jarvis.cs.iit.edu

3. Writing our first GPU program.

Let's take a look at our first GPU program, Vector Add. For this program we have two vectors that we want to add. Rather than adding them on the CPU we will transfer the vectors into GPU memory and do the addition on the GPU and transfer the result back to the host CPU.

3.1 VectorAdd.cu

Let's create a new file called "vecadd.cu" with the following code:

```
#include <stdio.h>
	__global__ void add(int *a, int *b, int *c) {
	*c = *a + *b;
}

int main(void) {
	int a, b, c; // host copies of a, b, c
	int *d_a, *d_b, *d_c; // device copies of a, b, c
	int size = sizeof(int);
	// Allocate space for device copies of a, b, c
	cudaMalloc((void **)&d_a, size);
	cudaMalloc((void **)&d_b, size);
	cudaMalloc((void **)&d_c, size);
	// Setup input values
	a = 2;
	b = 7;
	// Copy inputs to device
	cudaMemcpy(d_a, &a, size, cudaMemcpyHostToDevice);
	cudaMemcpy(d_b, &b, size, cudaMemcpyHostToDevice);
	// Launch add() kernel on GPU
	add<<<1,1>>>(d_a, d_b, d_c);
	// Copy result back to host
	cudaMemcpy(&c, d_c, size, cudaMemcpyDeviceToHost);
	//
	printf("Our result is %d\n", c);
	// Cleanup
	cudaFree(d_a); cudaFree(d_b); cudaFree(d_c);
	return 0;
}
```

3.2 Compiling

Create a new file called “compile.sh” and add the following lines:

```
#! /bin/bash
nvcc vecadd.cu
```

Now simply run:

```
$./compile.sh
```

This generates an a.out file.

3.3 Submitting our job.

Finally, create a new file called “submit.sh” with the following code:

```
#!/bin/bash
#
#$ -cwd
#$ -j y
#$ -m n
#$ -pe mpich 1
#$ -S /bin/bash
#
./a.out
```

$qsub submit.sh

3.4 Checking on our job

After you have submitted your job to the queue you can check that status of your job by running:
$ qstat

3.5 Viewing the output of the job

The job id will be a unique integer you can view the output by running the cat command on the auto-generated file.
$ cat

Appendix. Additional Questions

Please carefully ensure that your questions has not already been answered in the documentation. If you still have a question please send an email to: jarvis-admin [at] googlegroups [dot] com