Running Parallel Programming Data-Acquisition Scripts from a Windows Powershell

Mike Bailey

mjb@cs.oregonstate.edu

Oregon State University
Computer Graphics

powershell.pptx

mjb – March 9, 2023

Change the NUMT and NUMTRIES to Global int Variables

Right now, if your code is using defined constants, like this:

```
#ifndef NUMT
#define NUMT          2
#endif

#ifndef NUMS
#define NUMS         32
#endif
```

Change it to use global variables, like this:

```
int NUMT =   2;

int NUMS = 32;
```

Oregon State University
Computer Graphics

mjb – March 9, 2023

argc and argv

When you write in C or C++, your *main* program, which is really a special function call, looks like this:

```
int main( int argc, char *argv[ ] )
{
        . . .
```

These arguments describe what was entered on the command line used to run the program.

The **argc** is the number of arguments (the arg **C**ount)

The **argv** is a list of argc character strings that were typed (the arg **V**ector).
The name of the program counts as the 0th argv (i.e., argv[0])

So, for example, when you type
        ls –l
in a shell, the *ls* program sees argc and argv filled like this:
        argc = 2
        argv[0] = "ls"
        argv[1] = "-l"

---

argc and argv

So, if NUMT and NUMTRIALS are global int variables:
        int NUMT =   2;
        int NUMS = 32;

and you want to set them from the command line, like this:
./prog   1   64

Then, *inside your main program*, you would say this:
        if(argc >= 2)
                NUMT = atoi(argv[1]);

        if(argc >= 3)
                NUMS = atoi(argv[2]);

The if-statements guarantee that nothing bad happens if you forget to type values on the command line.

The *atoi* function converts a string into an integer ("ascii-to-integer").
If you ever need it, there is also an *atof* function for floating-point.

shared( ) in the *#pragma omp* Line

Also, remember, since NUMTRIALS is a variable, it needs to be declared as *shared* in the *#pragma omp* line:

#pragma omp parallel for default(none) shared(NUMS,xcs,ycs,rs,tn) reduction(+:numHits)

NUMT does not need to be declared in this way because it is not used in the for-loop that has the *#pragma omp* in front of it.

Oregon State
University
Computer Graphics
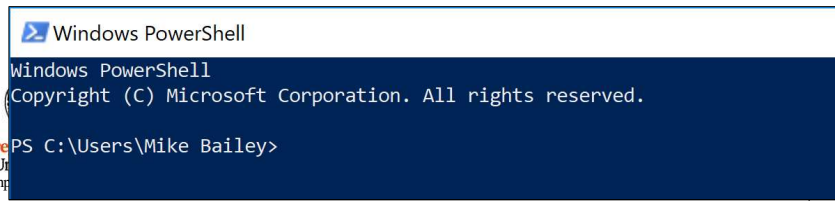
mjb – March 9, 2023

Windows Powershell

Windows comes with a shell program called *Powershell*. It might not be as familiar to most of us as some of the Linux shells are (bash, csh), but it can still be used to run multiple combinations of your program parameters in one shot.

There are a number of ways to get Powershell running. Either:

- Click on the Microsoft icon. Then scroll down to **Windows Powershell** and run **Windows Powershell.**

- *Shift right-click* in the directory you want to work in and select **Open Powershell Window**.

- Hold down the Windows key and hit the 'x' key, then select **Windows Powershell.**

The resulting window should look like this:

```
Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\Mike Bailey>
```

Oregon State
University
Computer Graphics

mjb – March 9, 2023

Change Directory to Where Your .exe File Lives

Then:

1. *cd* (change directory) to your home directory.
2. Then *cd* to the folder with your project
3. Then *cd* to the folder with your executable (*.exe)

> Windows PowerShell

```
PS C:\mjb\CS575\MonteCarlo\msvc\Debug>
```

The prompt will always tell you where you are in the file system.

Oregon State
University
Computer Graphics

mjb – March 9, 2023

Running an Executable

So, if you have *cd*'ed to where your executable (.exe) file lives, you can run it from the command line like this:

> Windows PowerShell   —   □   ×

```
PS C:\mjb\CS575\MonteCarlo\msvc\Debug> ls        Ask to see the files in this folder


    Directory: C:\mjb\CS575\MonteCarlo\msvc\Debug


Mode              LastWriteTime       Length Name
----              -------------       ------ ----
-a----        4/13/2020   8:28 PM       47104 MonteCarlo.exe      Yup, here it is
-a----        4/13/2020   8:28 PM      403468 MonteCarlo.ilk
-a----        4/13/2020   8:28 PM      626688 MonteCarlo.pdb


PS C:\mjb\CS575\MonteCarlo\msvc\Debug> ./MonteCarlo.exe      Execute it
 2 threads ;    50000 trials ; megatrials/sec =   57.42
PS C:\mjb\CS575\MonteCarlo\msvc\Debug>
```

Oregon State
University
Computer Graphics

mjb – March 9, 2023
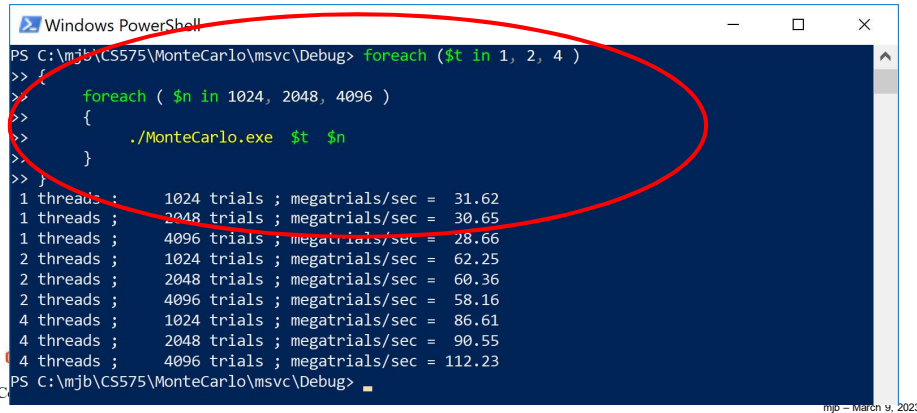
Running a Loop

But, here's the cool part. Type:

```
foreach ( $t in 1, 2, 4  )
{
        foreach ( $n in 1024, 2048, 4096)
        {
                ./MonteCarlo.exe   $t   $n
        }
}
```
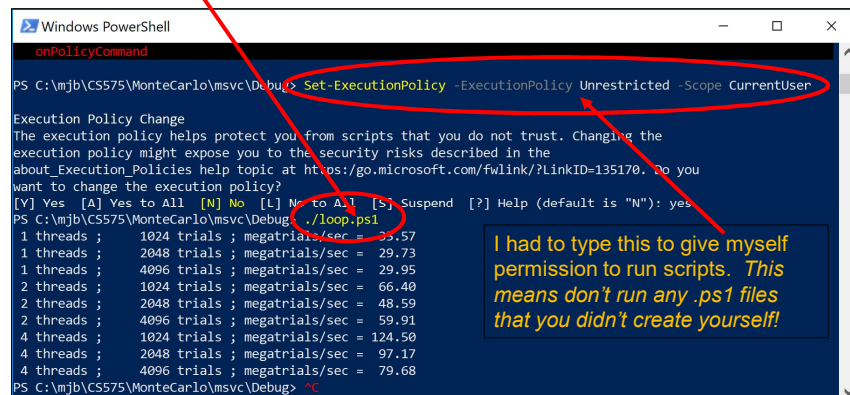
followed by Enter:

Running a Loop from a File

You can also use a text editor like *notepad* or *notepad++* and put these lines into a file called, say, l**oop.ps1** (ps1 is the Powershell file extension).

Then, you can run this script from Powershell just by typing it:

I had to type this to give myself permission to run scripts. *This means don't run any .ps1 files that you didn't create yourself!*

Instead of printing these lines to the screen, you probably want to print them to a text file that can then by imported by Excel.

University
Computer Graphics

mjb – March 9, 2023

Diverting Output to a File from Powershell

To divert the printf's to a file, do this:

./loop.ps1 > out.csv

or this:

./loop.ps1 1> out.csv

To divert *both* printf's and fprintf(stderr,…)'s together, do this:

./loop.ps1 2>&1> out.csv

Oregon State
University
Computer Graphics

mjb – March 9, 2023