CS 150: Sharing Between Processes

Cynthia Taylor
Oberlin College
April 28th 2014

Last class: Spawning Processes

- Useful when you have tasks individual processes can do

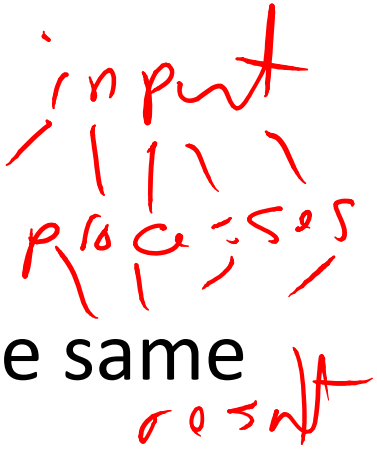
- Your web browser spawns a new process whenever you download a website in a new tab
  - Can render one site while other sites download

This Class: Processes Working on the Same Problem

- Recall from last class: Your computer can run several processes at once, as well as switching between them

- Want processes to be able to work together on the same problem to solve it faster

Process Pools



- Set of n processes that all work on the same problem
- Define a function that does what you want
- Split up input to function among n processes
- Get results from each process, combine

Syntax

```
def func(arg):
    …
    return x

p = Pool(processes=n)
Results = p.map(func,[arg1,arg2,..,argn])
```

named arg

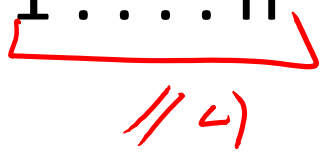
num of processes

size n

Pool.map takes a list of arguments

- If using n processes, list is of size n
- Function must take one argument
  - BUT that argument can be a list or tuple
- Result is a list of the return values from each process

- Write code to create a pool of 4 processes, and use them to compute the sum of the numbers 1 . . . . n



Will we always get faster results with more processes in the pool?

A. Yes

B. No

C. I don't know

Why doesn’t it get faster forever?

- Creating process and switching between them takes time and resources

- Need to balance speedup of adding more processes with resources of managing them

Shared Memory

- Note that our processes still aren't really cooperating
- Variables in one process cannot be modified by another process
- To change that, we use *shared memory*

RawValue

- New type of object

- Lets us share variables between processes

```
r = RawValue("i",0)
```

Takes in a code for the variable type (i is for integer), and a value

RawValue Demo

Why do we end up with the wrong values?

- Recall we switch between processes

Everything is okay

Proc1: x = r.value

Proc1: y = x+1

Proc1: r.value = y

Proc2: x = r.value

Proc2: y = x+1

Proc2: r.value = y

Everything is NOT okay

Proc1: x = r.value

Proc2: x = r.value

Proc1: y = x+1
Proc1: r.value = y

Proc2: y = x+1
Proc2: r.value = y

Would it fix this to say

```
r.value = r.value + 1
```

?

A. Yes

B. No

C. I don't know

get r.value
add 1 to r.value
set r.value

Just making it one line is not enough

- Python gets translated down to a bunch of smaller instructions by your computer
- Each python instruction because a bunch of machine instructions
- Processes can switch between machine instructions

Next Time

- Multiple Processes Fix Shared
- Lab 10 – Tuesday at 10 pm
- Prelab 11 – Wednesday in class