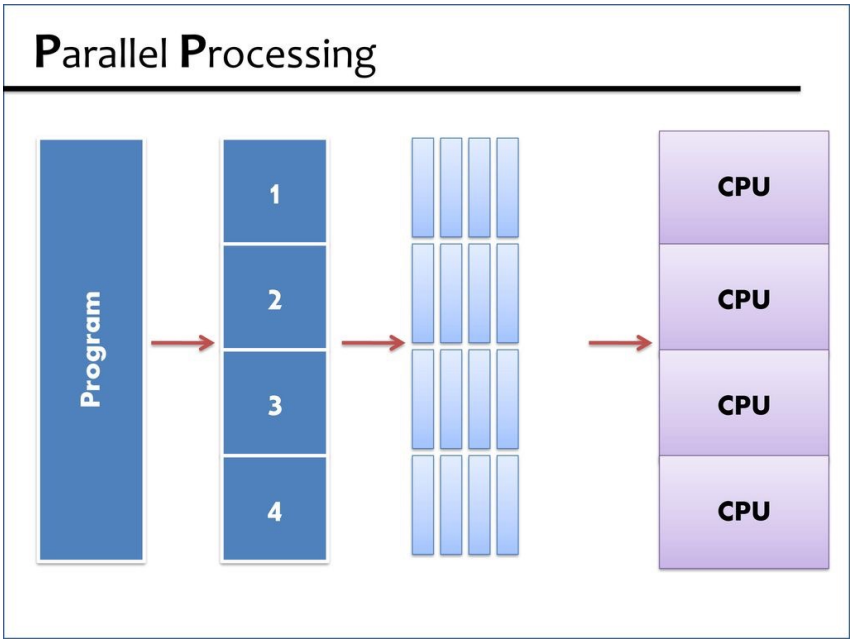
Harold Molina-Bulla DTSC-UC3M

PARALLEL PROGRAMING IN PYTHON



2022

Harold Molina-Bulla DTSC-UC3M

PARALLEL PROGRAMING IN PYTHON

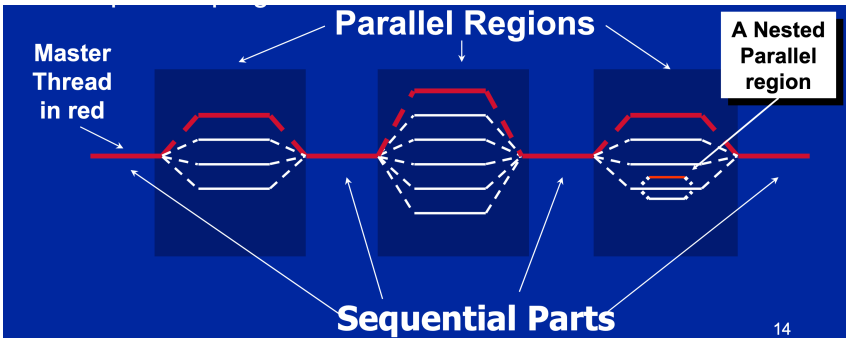
- Problems:
  - Python interpreter is single thread, the it does not support parallel programing
- Solution:
  - Use parallel execution threads
  - In a multicore machine will be multiprocess

STEPS TO PARALLEL PROGRAMING

- Create threads
- Synchronize parallel tasks
- Parallel loops
- Synchronize single masters
- Memory models

PARALLEL PROGRAMED PROGRAM

- Our program will have sequential parts and parallel regions, even nested parallel regions.



FIRST STEP: CREATE EXECUTION THREADS

- To implement parallel programs in Python we will use multiprocessing module
  - Threads
  - Pool of threads
  - Synchronize threads
  - Sharing memory
  - Queues of execution
- Very powerfull tool, if it is well programmed

CREATE PARALLEL THREADS

- We need create a pool of threads, to execute our code in parallel
- With multiprocess module, we will use the class “Pool”

```
from multiprocessing import Pool

def f(x):
    return x*x

if __name__ == '__main__':
    with Pool(5) as p:
        print(p.map(f, [1, 2, 3]))
```

will print to standard output

```
[1, 4, 9]
```

CREATE PARALLEL THREADS

- Parameters of Pool class constructor:
  - Num of paralellel processes.
    - BIG QUESTION: How many processors we will use? How many processors we have?
    - **Recommendation**: use the same number of physical cores in your computer.
    - What happens if we asks for more processors than we have?
  - Second question: How will we design our code in order to improve or, at least, do not degradate the execution?

STEPS TO DESIGN OUR PARALLEL PROGRAM

1. Analyze which parts of our will be parallelized
   1. We need to analyze the granularity level we will need or which is the best option in our environment.
2. Analyze how we will handle the memory: shared memory, by parameter, etc…
3. Analyze how we will return the results:
   1. Collecting results
      1. Advantage: easy to control
      2. Disadvantage: handle the return order
   2. Shared memory
      1. Advantages: Faster, do not take care about the execution order
      2. Disadvantages: memory hazzards and overhead to avoid

PRACTICAL WORK

- Download the Jupyter Notebooks
  - benchmark.ipynb
    - An script which executes several times the same operation, varying number of cpus, and estimate the improvement.
  - FirstParallel.ipynb
    - This Notebook shows how we can implement a Pool of processes , and program a simple dot product.
    - Shows how we can analyze the execution using profiling
    - We will need to analyze the profiling and improve the program

FIRST PARALLEL PROGRAM

- In local computer:
  - Copy the FirstParallel.ipynb and benchmark.ipynb in your local drive, in a well know folder
  - Open anaconda command line session
  - Change to the folder where you save the ipynb
  - Execute the command:
    - jupyter-notebook or jupyter-lab
  - It will execute the local server, and open in a browser the current folder
  - Select the notebook and follow the instructions