

HW09 - Load Balancing

CS5500

Chelsi Gupta

November 6, 2018

1 Steps for Implementation

- I have made 2 functions in my program:
 - **fill_queue:**
This function fills the initial queue with 5 tasks.
 - **determine_random_destination:**
This function determines a random destination where the process will send its excess tasks.
- The basic unit of work, a task, is represented by an integer i in the range $[1-1024]$, is to sleep the processor for $i/1000$ seconds.
- First I created an empty task queue for each process and filled it using the fill_queue function.
- The recv_task and recv_token variables represent the tasks and tokens that the process will receive from other processes.
- The req_task and req_token are the requests generated from Irecv of tasks and tokens.
- The process does the following actions before it receives a task from other processor:
 - If queue size becomes greater than 7 then it sends 1 task to a random destination using Isend. If this random destination is smaller than the current process rank, then the current process becomes black and the token as well(i.e. token=1)
 - After that if the queue is not empty it takes out 1 task and perform it.
 - It then generates a random number between 0 to 10. If this number is less than 5, then 2 new tasks are added to the task queue.

- If at any point the task queue becomes empty and no task has been sent backwards then the process will forward a white token(token=0) forward.
- At the end if process that started execution first receives a white token then we will say that White/Black ring termination has occurred and system has exited safely.
- After receiving a task from other process, the current process will append it to its task queue.
- I compiled and ran the program using 'mpirun -np 3 python load_balancing.py'.

2 Code

```

import time
start_time = time.time()
import random
import queue
import numpy as np
from mpi4py import MPI
comm = MPI.COMM_WORLD    # Defines the default communicator
ANY = MPI.ANY_SOURCE

size = comm.Get_size() # Stores the number of processes in size.
rank = comm.Get_rank() # Stores the rank (pid) of the current process

def fill_queue(task_queue):
    count = 0
    while(count < 5):
        i = random.randint(1,1024)
        count += 1
        task_queue.put(i)

    return task_queue

def determine_random_destination(rank, size):
    r1 = range(0,size)
    r2 = [rank]
    r = list(set(r1) - set(r2))
    dst = random.choice(r)
    return dst

task_queue = queue.Queue(maxsize=16)
task_queue = fill_queue(task_queue)

recv_task = np.zeros(1)
token = 0
recv_token = np.zeros(1)

while True:

```

```

req_token = comm.Irecv(recv_token, source=ANY, tag=0)
req_task = comm.Irecv(recv_task, source=ANY, tag=1)

while req_task.Get_status()==False:

    if task_queue.qsize() > 7:
        random_dest = determine_random_destination(rank, size)
        if random_dest < rank:
            token = 1

        sTask = task_queue.get()
        comm.Isend(np.array(sTask), dest=random_dest, tag=1)

    if task_queue.qsize() > 0:
        do_task = task_queue.get()
        time.sleep(do_task/1000)
        print('Process_',rank,'_performed_',do_task,'_units_of_work.')

    if random.randint(0, 10) < 5:
        for _ in range(2):
            x = random.randint(1,1024)
            task_queue.put(x)

    if task_queue.empty():
        comm.Isend(np.array([token]), dest=(rank+1)%size, tag=0)

        req_token.wait()

        if recv_token[0] == 0:
            print('Process_',rank,'_all_done.')
            print("_____%s____seconds____" % (time.time() - start_time))
            quit()

    print("Process_",rank,"_received_work.")
    req_task.wait()
    task_queue.put(recv_task[0])

```

3 Timing Information

I tried running the program with 3 processors and the table below shows the time taken by each process to complete its execution:

#Process No.	Time-taken(in seconds)
P0	33.587
P1	13.26
P3	33.583

4 Output

```
chelsi@chelsi-HP-Z220-CMT-Workstation: ~/Parallel_Computing/hw09
Process 0 performed 544 units of work.
Process 2 performed 779 units of work.
Process 0 performed 519 units of work.
Process 2 performed 90 units of work.
Process 0 performed 424 units of work.
Process 2 performed 814 units of work.
Process 0 performed 622 units of work.
Process 2 performed 501 units of work.
Process 0 performed 336 units of work.
Process 1 all done.
--- 13.267043113708496 seconds ---
Process 2 performed 863 units of work.
Process 2 performed 409 units of work.
Process 2 performed 32 units of work.
Process 2 performed 161 units of work.
Process 2 performed 169 units of work.
Process 2 performed 795 units of work.
Process 2 performed 679 units of work.
Process 2 performed 831 units of work.
Process 2 performed 706 units of work.
Process 2 performed 153 units of work.
Process 2 performed 545 units of work.
Process 2 performed 283 units of work.
Process 2 performed 420 units of work.
Process 2 performed 175 units of work.
Process 2 performed 751 units of work.
Process 2 performed 704 units of work.
Process 2 performed 323 units of work.
Process 2 performed 738 units of work.
Process 2 performed 988 units of work.
Process 2 performed 21 units of work.
Process 2 performed 880 units of work.
Process 2 performed 703 units of work.
Process 2 performed 865 units of work.
Process 2 performed 184 units of work.
Process 2 performed 521 units of work.
Process 2 performed 81 units of work.
Process 2 performed 1003 units of work.
Process 2 performed 522 units of work.
Process 2 performed 981 units of work.
Process 2 performed 845 units of work.
Process 2 performed 441 units of work.
Process 2 performed 794 units of work.
Process 2 performed 541 units of work.
Process 2 performed 124 units of work.
Process 2 performed 65 units of work.
Process 2 performed 418 units of work.
Process 2 performed 265 units of work.
Process 2 performed 425 units of work.
Process 2 performed 808 units of work.
Process 2 performed 94 units of work.
Process 0 all done.
Process 2 all done.
--- 33.583059310913086 seconds ---
--- 33.58794593811035 seconds ---
chelsi@chelsi-HP-Z220-CMT-Workstation:~/Parallel_Computing/hw09$
```

Figure 1: My Output