Xi Liu, xl3504, Midterm Exam

You may use the textbook, slides, and any notes you have.
But you may not use the internet.
You may NOT use communication tools
to collaborate with other humans. This includes
but is not limited to G-Chat, Messenger, E-mail, etc.
Do not try to search for answers
on the internet it will show in your answer and you will
earn an immediate grade of 0.
Anyone found sharing answers
or communicating with another student
during the exam period will earn an immediate grade of 0.
"I understand the ground rules and
agree to abide by them. I will not share answers
or assist another student during
this exam, nor will I seek assistance from another
student or attempt to view their answers."

Problem 1

a.

yes, if the prediction is correct, then time is saved by not waiting for the conditional jump; also, running shorter jobs first may reduce turnaround time

b.

If each one of the 8 cores is	The maximum number of processes that can execute on the whole processor is:	The maximum number of threads that can execute on the whole processor is:
Only pipeline	8	8
Superscalar with no hyperthreading and each core has four execution units	8	8
Four-way hyperthreading but without branch prediction	32	32
Four-way hyperthreading with branch prediction	32	32

c.

false, different processes have different virtual address spaces. for each distinct process, the operating system maintains a different page table for the process, so different processes' virtual pages map to different physical page frames; conventionally, a page size ($2^{12} = 4096$ bytes) is a lot bigger than a cache line size ($2^6 = 64$ bytes), so 2 different physical pages cannot be contained in a same cache line, so there is no false sharing among processes that do not share memory

Problem 2

a.

yes

$span_1 =$ time of $\{A \to B \to E \to F \to G\} = T_\infty = 40$ nanoseconds

$span_2 =$ time of $\{A \to C \to E \to F \to G\} = T_\infty = 40$ nanoseconds

$span_3 =$ time of $\{A \to D \to F \to G\} = T_\infty = 40$ nanoseconds

b.

minimum number of cores for maximum speedup = 3

$$T_1 = \sum_{i \in \{A,\ldots,G\}} (\text{time of task}_i) = (10{+}5{+}5{+}10{+}5{+}10{+}10)ns = 55 \text{ nanoseconds}$$

$$speedup = \frac{T_{serial}}{T_{parallel}} = \frac{T_1}{T_3} = \frac{55 \text{ nanoseconds}}{40 \text{ nanoseconds}} = 1.375$$

time interval (ns)	process 1	process 2	process 3
$[0, 10]$	A		
$[10, 15]$	B	C	D
$[15, 20]$	E	D	
$[20, 30]$	F		
$[30, 40]$	G		

c.

yes, we are facing load imbalance since process 2 is idle for time interval $[0, 10] \cup [20, 30] \cup [30, 40]$; process 3 is idle for time interval $[0, 10] \cup [15, 20] \cup [20, 30] \cup [30, 40]$

d.

let p be the number of cores available to use

$$efficiency = \frac{speedup}{p} = \frac{1.375}{3} = 0.458\overline{3}$$

e.
execution time = (instruction count) · (cycles per instruction) · (cycle time)

$$ET = IC \cdot CPI \cdot CT$$

$$CPI = \frac{ET}{IC \cdot CT}$$

$$ET = T_1 = \sum_{i \in \{A,\ldots,G\}} (\text{time of task}_i) = 55 \text{ nanoseconds} = 55 \cdot 10^{-9} seconds$$

$$IC = 7 \cdot 100 \text{ instructions} = 700 \text{ instructions}$$

$$frequency = 4GHz = 4 \cdot 10^9 Hertz = 4 \cdot 10^9 \frac{cycle}{second}$$

$$CT = \frac{1}{frequency} = \frac{1}{4 \cdot 10^9} = 2.5 \cdot 10^{-10} \frac{second}{cycle}$$

$$CPI = \frac{55 \cdot 10^{-9} seconds}{(700 \text{ instructions})(2.5 \cdot 10^{-10} \frac{second}{cycle})} = 0.314286 \frac{cycle}{instruction}$$

Problem 3

a.

	process 0	process 1	process 2
x	4	6	8
y	8	8	2
z	8	4	8

b.

without the break statement, when my_rank = 1, the code within the original case 2 block will be executed

before process 1 finishes the original code block of case 1, process 0, 1, and 2 would initially behave like the version of code where "break;" is not removed, but then for process 1: after the execution of the original case 1 code block, then x, y, z will be reassigned with new values such that $x = 3;\ y = 2,\ z = 1;$ then the last 2 additional MPI_Bcast() functions calls can cause the program to hang since all of the other processes in the communicator have finished their MPI_Bcast()

c.

yes

with hyperthreading, 1 physical core can be used as 2 virtual or logical cores by the operating system, allowing process 0 and process 1 to both execute on the same core and at the same time

time interval (ns)	process 1	process 2
$[0, 10]$	A	
$[10, 15]$	B	D
$[15, 20]$	C	D
$[20, 25]$		E
$[25, 35]$		F
$[35, 45]$		G