Parallel Computing
Homework Assignment 3 Solutions
[Total 30 points]

1.
[2] No, higher speedup does not imply higher efficiency.
[3] Efficiency is an indication of how well you do with extra resources. For example, by doubling the number of cores, we see an increase of 2x increase in speedup but making use of 70% only of these cores.

2.
a.
MIPS = #instructions in millions / total time in seconds
total time in seconds = total # of cycles * cycle time = total number of cycles / frequency

[2] Implementation 1:
#instructions in million = 1
total time in sec = $(5*10^5*10 + 5*10^5*1)/(2*10^9) = 2.75*10^{-3}$
MIPS = $1/(2.75*10^{-3})$= 363.63

[2] Implementation 2:
#instructions in million = 2
total time in sec = $2*10^6/2*10^9=10^{-3}$
MIPS = $2/(10^{-3})$ = 2000 MIPS

b.
CPI = total number of cycles / total number of instructions

[2] Implementation 1: CPI = $(5*10^5*10 + 5*10^5*1)/1000000 = 5.5$
[2] Implementation 2: CPI = $(2*10^6)/2000000 = 1$

c.
ET = IC*CPI*CT = #cycles/frequency

[2] Implementation 1: ET = $(5*10^5*10 + 5*10^5*1)/(2*10^9) = 2.75*10^{-3}$ seconds
[2] Implementation 2: ET = $(2*10^6)/ (2*10^9) = 10^{-3}$ seconds

d.

[1 each row: -0.5 for each mistake]

Measurement	Impl. 1	Impl. 2	Better Implementation
IC	10^6	$2*10^6$	1
MIPS	363.3	2000	2
CPI	5.5	1	2
ET	$2.75*10^{-3}$ sec	10^{-3} sec	2

3.

[1] Simply ensure that no two threads on different cores will be updating elements of the same cache block.

[1] Since each cache block is 32 bytes = 8 integers, then any two threads must be working on elements that are 8 elements away in the array from each other.

[1] This can be easily done by giving each thread 256/4 = 64 consecutive elements to work on.

It is the same answer for a, b, and c. Because the answer has nothing to do with the cache size or associativity.