Computer Science 605.411

Module 12 Example Set 3

1. A certain program consists of a purely sequential part (Part 1) that must be executed before the remaining Part 2. Part 1 accounts for 24% of the instructions in the program. Part 2 can be split into four sections that correspond to 18%, 32%, 14% and 12% of the instructions in the program. The sections in part 2 are independent and can be executed in parallel. If each instruction in the program takes 1 cycle, what speedup would be provided by executing the program on a quad-core system versus a single core system? Round your answer to two decimal places.

Part 1 would take 0.24T time units. The longest section in part 2 takes 0.32T.
So the speedup = 1/(0.24+0.32) = 1/0.56 = 1.79

2. A program is to perform two sums: one is a sequential sum of 100 scalar variables and the other is the matrix sum of two 1000 by 1000 element arrays. Performing a single addition (the sum of two individual numbers) takes 1 cycle. The scalar sum must be computer before the matrix sum. What speedup is provided for the same program by a system that has 1000 processors compared to using a single processor?

Using one processor takes 99 + 1000000 = 1000099 cycles.
The sequential sum is computed in 99 cycles by a single processor. Each scalar variable has a different address that can not be computed based on a processor number.

The matrix sum would generate a matrix result such that CU,k] = AU,k] + BU,k].
Each matrix contains 1000000 elements. Each of the 1000 processors could compute the sum of the corresponding rows in the two matrices in parallel. This would take 1000 cycles to generate the matrix sum. So the total time would be 99 + 1000 = 1099.
The speedup = 1000099/1099 = 910.01

3.A Quad-core system is used the compute the cummulative sum of the elements in an 8- element vector. Assume that the time required on any core to perform an addition is 2 nano- seconds. What speedup factor would this system provide compared to a single processor system if only the time required to perform the required additions is taken into account and no other overhead?
The cummulative sum = A[O] + A[l] + A[2] +A[3] + A[4] + A[S] + A[6] + A[7]
Partitioning the 8 elements into two groups of 4 elements each, a separate core could add a pair of elements in 2 ns to generate 4 partial sums:
A[O] + A[l], A[2] +A[3], A[4] + A[S], A[6] + A[7]

Then two cores could each add together 2 of these partial sums in 2 ns to generate 2 partial sums:
A[O] + A[l] + A[2] +A[3], A[4] + A[S] + A[6] + A[7]

Then one core would add the final 2 partial sums in 2 ns to generate the result. Thus a total 6 ns would be required.

A single core would have to perform 7 additions,which takes 14 ns. So the speedup = 14/6 = 2.33

4.A program that runs on one of the 3.3GHz processors within a NUMA multi-processor has an average CPI of 2. If an instruction references the remote memory, it takes an additional 300 ns. The program completes in 860 million cycles if none of the instructions reference the remote memory. How many additional cycles would the program take if 5% of the instructions reference the remote memory?

Since the average CPI=2, the instruction count= 860 million cycles / 2 cycles per instruction = 430 million instructions.
The 3.3 GHz clock rate corresponds to a 0.3 ns cycle time. Hence, 300ns corresponds to 300/0.3 = 1000 clock cycles. Therefore the number of extra cycles required if 5% of the instructions

reference the remote memory is 0.05* 430 million*lOOO = 21.5 billion cycles

5.Consider a multi-core processor with heterogenous cores: A, B C and D where core B runs twice as fast as A, core C runs three times as fast as A and core D runs at the same speed as core A.An application needs to execute a function that takes a single argument. Hence the function will be called once for each of the elements in an array of 256 elements. Core A requires T time units to execute the function for each array element.

a) Ifthe work is distributed among the cores as follows:

Core A	32 elements
Core B	128 elements
Core C	64 elements
Core D	32 elements

How many time units will be required to process the entire array using the function? Assume that no stalls of any type occur and expressyour answer as N*T. That is specify the integer value for N. Total execution time = max(32/1, 128/2, 64/3, 32/1) = 64 (time units)

b) Ifthe work is distributed among the cores as follows:

Core A	48 elements
Core B	128 elements
Core C	80 elements
Core D	Unused

How many time units will be required to process the entire array using the function? Assume that no stalls of any type occur and expressyour answer as N*T. That is specify the integer value for N. Total execution time = max(48/1, 128/2, 80/3, 0/1)= 64 (timeunits)

6.A program runs on a multi-core system in which each core is rated at 500 MIPS.The program consists of five threads: four independent threads (A,B,C and D) each of which generates partial results. The four sets of partial results are then taken as input by a single thread (E) to produce the final result. Threads A, B, and C each executes 20 million instructions and thread D executes 38 million instructions. Thread E executes 12million instructions in generating the final result. What speedup factor would be provided by increasing the number of identical cores in the system from 2 to 4?

A, B, C and D must complete before E executes.With 2 cores any pair of threads selected from A, B and C could run in parallel and would take time T = 20/500 = 0.04 seconds to complete. The thread not included in the pair would execute in parallel with D which would take 38/500 seconds. Thread E would then take 12/500 seconds for a total of (20+38+12)/500 =70/500 seconds.

Using 4 cores A,B,C and D could each execute on a separate cores,butit wouldtake 38/500 seconds to complete the four threads. Thread E would then take an additional 12/500 seconds for a total of 50/500 seconds. Hence the speedup would be (70/500) / (50/500) =70/50 =1.4.

7.A dual processor SMP system includes an LI data cache for each processor and employs the MESI protocol to maintain cache consistency. Each cache is a direct-mapped copy-back cache that contains 8192 cache lines each of which is 256 bytes in size. A write-allocate policy is used for each cache. One process, P1, runs on the first processor at the same time that another process, P2, runs on the other processor. P 1accesses a variable X with an initial value of 80 that resides in memory at address Ox400800CO. P2 accesses a variable Y with an initial value of 200 that resides in memory at address Ox400800F8.

a) Into which line within Pl 's cache will the memory block containing the variable X be loaded?
The block containing X would map to line Ox400800 MOD 8192 =2048 in Pl's cache

b) Into which line within P2's cache will the memory block containing the variable Y be loaded?
The block containing Y would map to line Ox400800 MOD 8192 =2048 in P2's cache.

c) All of the lines in each processor's data cache are initially invalid. For the following accesses in the order listed, show the MESI state of the affected cache line before and after the reference and explain why any change in state occurs:

Reference	Pl's cache line state before	Pl's cache line state after	P2's cache line state before	P2's cache line state after
Pl increments X by 1	I	M	I	I
P2 multiplies Y by 2	M	I	I	M
Pl reads X	I	S	M	S
P2 reads X	S	S	S	S
P2 increments Y by 3	S	I	S	M
Pl decrements Y by 4	I	M	M	I
P2 multiplies X by 2	M	I	I	M
Pl multiplies Y by 2	I	M	M	I

To increment X, P1must read the block containing X into its cache and modify the line.

To multiply Y by 2, P2 must read the block containing Y into its cache and modify the line.But Pl must first write its copy of the line back to memory. So Pl's copy changes from M to I. P2's copy changes from I to M.

When Pl reads X again, the block containing X must be loaded into Pl's cache.But P2's copy must first be written back to memory and its state changes from M to S. Pl's line changes from I to S.

When P2 reads X, the line in P2's cache containing S remains in the shared state and Pl's copy is unaffected.

When P2 increments Y by 3, its line changes state from Sto M and Pl's line containing the copy changes state from Sto l.

For Pl to decrement Y by the modified line in P2's cache must first be written back to memory and its state changes from M to I. Pl must load the block into its cache and modify the line. So the state of its line will change from I to M.

When P2 multiplies X by 2, its copy of the line changes from I to M after Pl writes its copy back to memory and changes its state from M to I.

Finally, Pl multiplies Y by 2 causing P2 to write its copy back to memory and invalidate its line. The block is loaded into Pl's cache and the state of the line in Pl's cache changes from I to M.

8.A message-passing program runs on two processors within a computer cluster.The time required to send a message is 1000 cycles. It takes 500 cycles to complete the RECEIVE operation once a message is available. The two tasks within the program perform the following actions:

Processor 0
Computes for 1000 cycles
Receives message 1 from processor 1
Computes for 2000 cycles
Sends message 2 to Processor 1
Computes for 500 cycles

Processor 1
Computes for 500 cycles
Sends message 1 to Processor 0
Computes for 500 cycles
Receives message 2 from processor 0
Computes for 7000 cycles

When a processor sends a message, it waits untilthe transmission completes before continuing . Whatis the total number of cycles that would elapse by the time the program compl|etes?

Both processors start at the same time. After 500 cycles processorl sends a message to processorO. ProcessorsO attempts to receive the message which arrives after a total of 1500 cycles have elapsed. The RECEIVE completes for processorO after a total of 2000 cycles. At this time Processorl attempts to receive a message. But processorO computes for 2000 cycles before sendingthe message. So the message arrives at processorl after a total of 5000 cycles. After a total of 5500 cycles, processorO completes its task. Processorl completes its RECEIVE after a total of 5500 cycles and computes for an additional 7000 cycles. So processorl completes its task after a total of 12500 cycles.