# SHANGHAITECH UNIVERSITY
# 2022-2023 SEMESTER 1 FINAL EXAMINATION
# CS121 PARALLEL COMPUTING

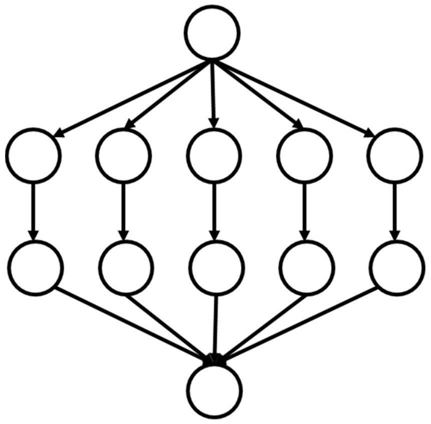
December 2022 Time Allowed: 100 minutes

*Write your answer for each problem on a separate piece of paper, with your name and student ID at the top.*

*In all problems in which you are asked to design algorithms, you should clearly describe how your algorithm works and argue why your algorithm is correct, in addition to providing code / pseudocode as instructed.*

*All answers must be written neatly and legibly in English.*

1a. Consider the task graph shown below. Each node represents a subtask which takes unit time to perform. An arrow from one subtask to another means the first subtask must finish before the second subtask can start. What is the maximum speedup for performing this task using two identical processors? What is the maximum possible speedup using any number of identical processors? What is the least number of identical processors needed to achieve the maximum possible speedup?



(5 points)

EXAM CONTINUES ON NEXT PAGE

1b. Consider the code shown below. State why the code cannot be parallelized, and modify it so that it can be parallelized. Make the minimum modifications possible. The modified code must compute the same results as the original code. Write your parallel code in OpenMP.

```
for (i=0; i<n; i++) {
   x[i] = y[i];
   y[i+1] = z[i];
}
```

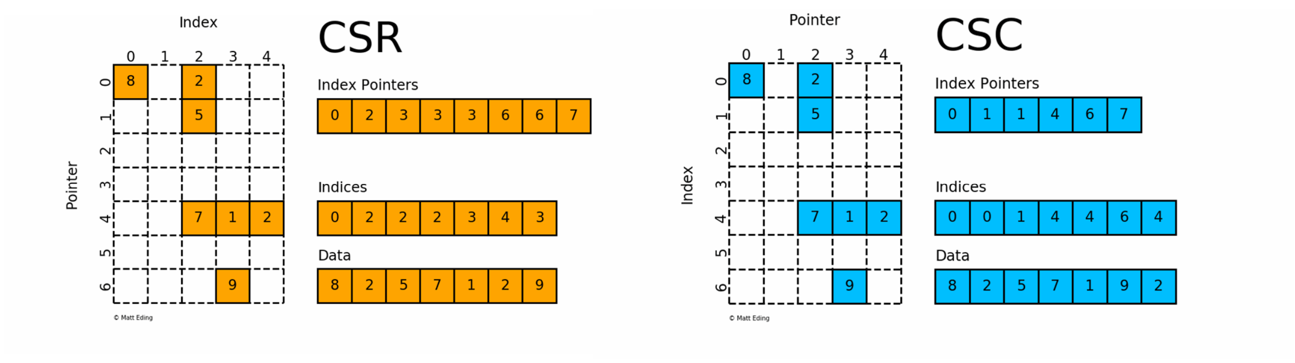
(10 points)

1c. Consider the code shown below. State why the code cannot be parallelized, and modify it so that it can be parallelized. Make the minimum modifications possible. The modified code must compute the same results as the original code. Write your parallel code in OpenMP.

```
for (i=0; i<n; i++)
   for (j=1; j<n; j++)
      A[i][j] = A[i+1][j-1]
```

(10 points)

2. Recall that the compressed sparse row (CSR) and compressed sparse column (CSC) formats can be used to efficiently store sparse matrices. An example of the CSR and CSC representations of a matrix is shown below. Given a matrix stored in CSR, write an efficient OpenMP program to convert it to CSC format. Try to make your program as efficient as possible. Describe how you address performance issues such as load balancing, synchronization, etc.



(25 points)

EXAM CONTINUES ON NEXT PAGE

3. Let $A$ be an array of size $n$ and let $k \le n$, for positive integers $n, k$. Define $S_i = \sum_{j=i}^{i+k-1} A[j]$, for $0 \le i \le n-k$. Design a PRAM algorithm which computes all the values $S_0, \dots, S_{n-k}$ using $n$ processors in $O(\log n)$ parallel time and $O(n)$ total work. Note that the running time and work should be independent of $k$.

(25 points)

4a. Consider the CUDA kernel `foo` shown below. The variable `A` points to an array that has been allocated in GPU global memory, `n` is the size of `A`, and `m` points to a float allocated in GPU global memory. Describe in words what `foo` computes.

```
1  __global__ void foo(float *A, int n, float *m) {
2      int tid = threadIdx.x;
3      int i = blockIdx.x*blockDim.x + threadIdx.x;
4
5      __shared__ float A_sh[blockDim.x];
6
7      if (i<n)
8         A_sh[tid] = A[i];
9      else
10        A_sh[tid] = 0;
11     __syncthreads();
12
13     for (int s=blockDim.x/2; s>0; s=s/2) {
14        if (tid < s)
15           A_sh[tid] = max(A_sh[tid], A_sh[tid+s]);
16        __syncthreads();
17     }
18
19     if (tid == 0 )
20        atomicMax(m, A_sh[tid]);
21 }
```

(10 points)

4b. What is the purpose of the `__syncthreads()` operation on line 11?

(5 points)

4c. What is the purpose of the `__syncthreads()` on line 16? Describe what can go wrong if it was removed.

(5 points)

4d. Suppose you have a GPU supporting up to 1024 threads per thread block, and 1536 threads per SM. Show how you would call kernel `foo` from the CPU host to achieve the highest performance.

(5 points)

END OF EXAM