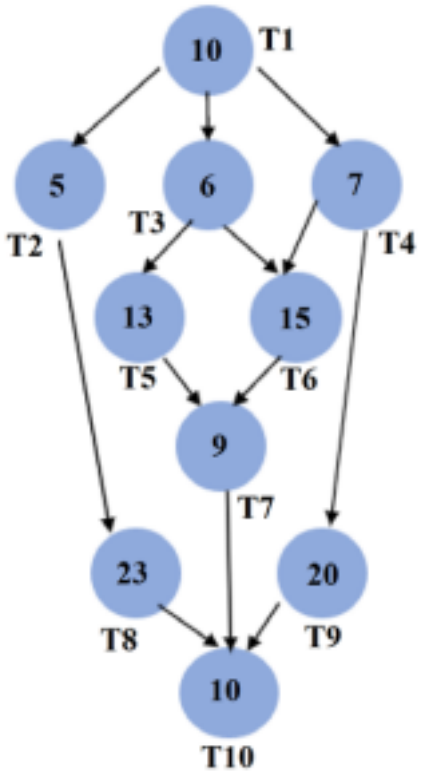
National University of Computer and Emerging Sciences, Lahore Campus

Student : Name:_____ Roll No._____ Section:_____

Question 1



For the task graph given above, determine:

a. Maximum degree of concurrency

b. Critical path

c. Critical path length

d. Maximum possible speedup assuming large number of process are available e.

Minimum number of processes needed to obtain the maximum possible speedup f.

Maximum speed-up if number of processes are limited to 2

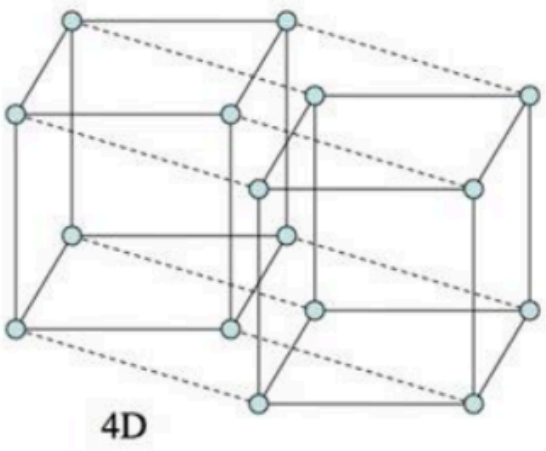
Question 2:

Calculate (a) cost (b) diameter (c) bisection width and (d) arc connectivity for:

i. 4x4 two-dimensional mesh with wraparound links

ii. 4x4 two -dimensional mesh with no wraparound links

iii. Four-dimensional hypercube (size=*16 Nodes*)



Question 3

Write a multithreaded program using 'OpenMP' to perform matrix operations as instructed below. You will perform Data Decomposition as well as task decomposition. Your program should provide the following functionality. You can paste your code in this document. Teaching assistant can ask any student to run the code or he can conduct a viva to check the authenticity.

Part A:

1. Take a matrix of size (m x n) where 'm' and 'n' values are taken as input from user. Initialize the matrix by some random values or user input.

2. Then

- **if the last digit of your student ID is (0 to 4) then create number of threads equal to number of rows'm' and each thread will sort a row using Quick Sort in Ascending order.**
- **if the last digit of your student ID is (5 to 9) then create number of threads equal to number of rows 'm'and each thread will sort a row using Quick Sort in Descending order.**

3. QuickSort should be implemented using Recursive decomposition. Each subtask will be done by a new thread.

Part B:

All values of the matrix should be multiplied to get the product of the complete matrix. You have to perform different experiments in this part to do a comparative analysis.

i. **Change the number of threads**
ii. **Change mapping scheme (Static, Dynamic, Guided) and chunk size.**

You can call the function omp_get_wtime() at the beginning and end of the program to find the elapsed time. The function omp_get_wtime() returns a double value. You can create a table to report these results.

	Chunk Size				
	Default	**n/nthreads**	**1**	**Random**	**random**
Static					
Dynamic					
Guided					