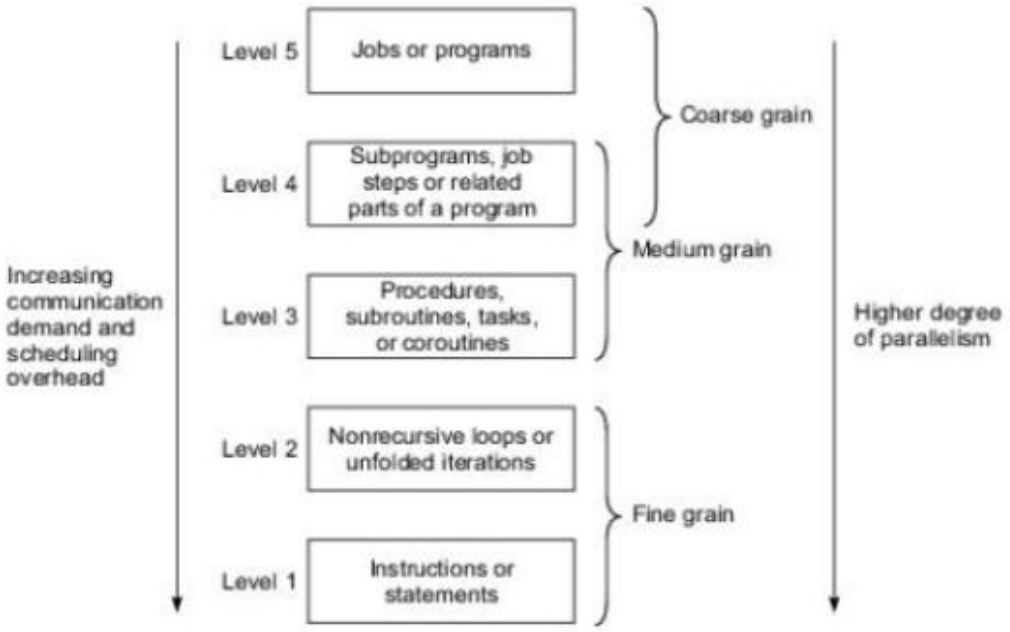
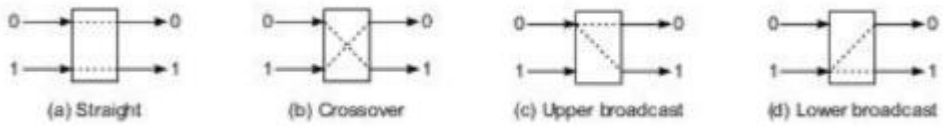
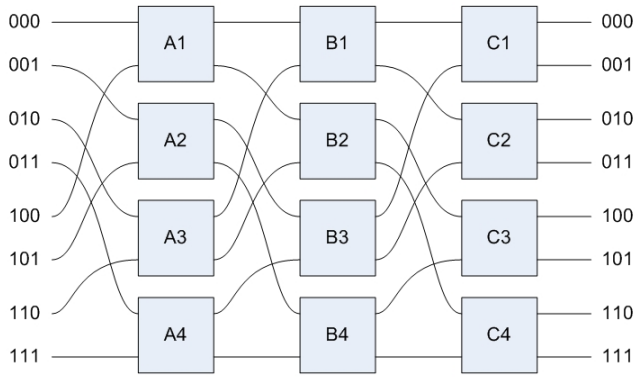
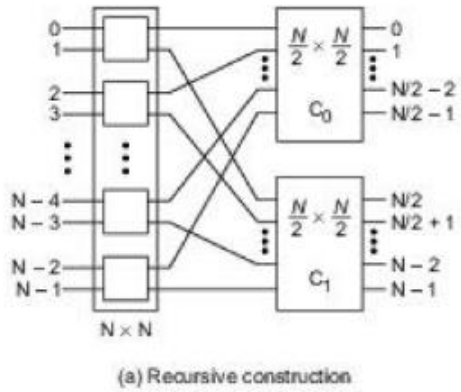
Internal Assessment Test II

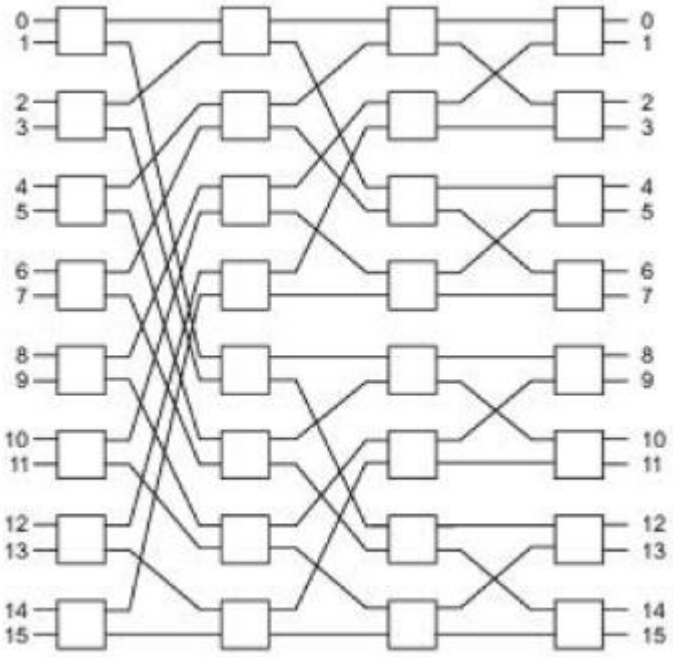
October 2019

Sub:	Advanced Computer Architecture					Sub Code	15CS72	Branch:	CSE
Date:	14/10/2019	Duration:	90 mins	Max Marks:	50	Sem / Sec:	VII A,B,C	OBE	

	Answer any FIVE FULL Questions	MARKS	CO	RBT
1 (a)	Explain the following terms for System Interconnect architecture. a) Node Degree b) Bisection Width c) Static Connection Networks d) Data Routing Functions e) Crossbar Networks a) Node Degree: The number of edges incident on node represents node degree. If the network contains directed edges then number of incoming edges represents in-degree and number of outgoing edges represents out-degree. Then node degree is the sum of the two. Hence the node degree should be kept small (constant) to reduce cost. b) Bisection Width: When the network is cut into two halves, the minimum number of edges along the cut is called as channel bisection width b. Each edge corresponds to a channel with w bit wires. Hence wire bisection width is B=bw which represents the wiring density of a network. The channel width w=B/b. c) Static Connection Networks: Static networks are formed by point to point fixed direct connections which will not change during the program execution. d) Data Routing Functions: The data routing functions are used for data exchange among multiple processors or PE in network. Commonly seen data routing functions include shifting, rotation, permutation (one-to-one), multicast (one to many), shuffle, exchange etc. These routing functions can be implemented on ring, mesh, hypercube or multistage networks. e) Crossbar Networks: A crossbar network can be visualized as a single-stage switch network. Each cross point switch can provide a dedicated connection path between a pair. The switch can be set on or off dynamically upon program demand. Example: Inter-processor Memory Crossbar network	05	CO1	L2

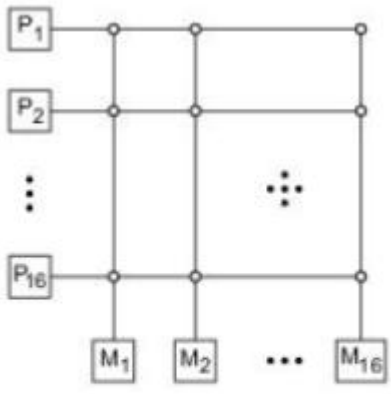
1(b)	Explain different levels of processing and how parallelism can be exploited at each level. There are different processing levels and parallelism can be exploited at each level as shown in Figure given below. Also lower the level, more finely will be the grain size.  Instruction Level Parallelism: Typical grain size would be less than 20 instructions. Also the parallelism is detected and source code will be transformed to parallel code by optimizing compilers. Grain size is Fine Loop Level Parallelism: If loop iterations are independent they can be executed in parallel by Vector Processor. But recursive loops are rather difficult to parallelize. A typical loop contains less than 500 instructions. Grain size is Fine. Procedure Level Parallelism : Here grain size is medium which typically corresponds to a procedure or subroutine with less than 2000 instructions. Detection of parallelism at this level is much more difficult than at the finer-grain levels. Subprogram Level Parallelism: The grain size may typically contain tens or hundreds of thousands of instructions. Traditionally, parallelism at this level has been exploited by algorithm designers or programmers, rather than by compilers. Job level Parallelism: This corresponds to the parallel execution of essentially independent jobs (programs) on a parallel computer. The grain size can be as high as millions of instructions in a single program. The job level parallelism is handled by OS and program loader. The grain size is coarse.	05	CO4	L2

2 (a)	Explain the following Multistage Networks given below with neat diagram a) Omega Network b) Baseline Network <u>Omega Network</u> The figure given below shows 4 possible switch modules used for constructing the Omega Network and 8*8 Omega Network. Three stages of 2 X 2 switches are needed. There are 8 inputs on the left and 8 outputs on the right. The ISC pattern is the perfect shuffle over 8 objects. The outputs from each stage are connected to the inputs of the next stage using a <u>perfect shuffle</u> connection system In general, an n-input Omega network requires $\log_2 n$ stages of 2*2 switches. Each stage requires n/2 switch modules. In total, the network uses $n\log_2 n/2$ switches. Each switch module is individually controlled.   Baseline Network The first stage contains one N*N block and second stage contains two N/2*N/2 sub blocks labeled as C0 and C1. The construction process is recursively applied to the sub blocks until the size of the sub block gets reduces to 2*2. The ultimate building blocks for sub blocks are 2*2 switches each with two legitimate connection states: straight and crossover between the two input and two outputs. A 16*16 baseline network is shown below.  	10	CO1	L2

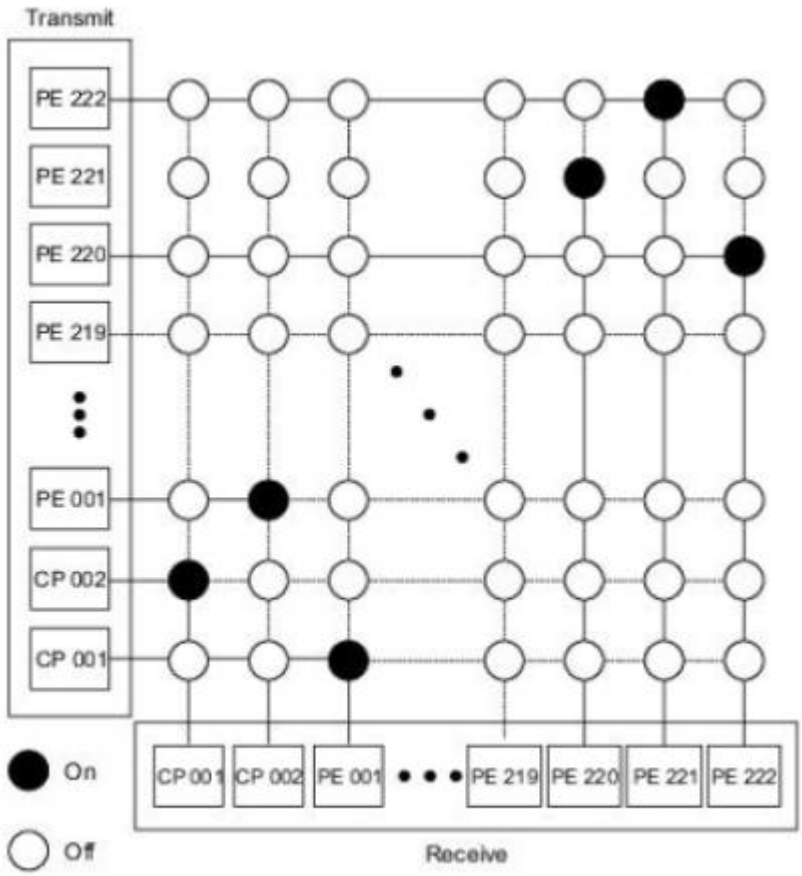
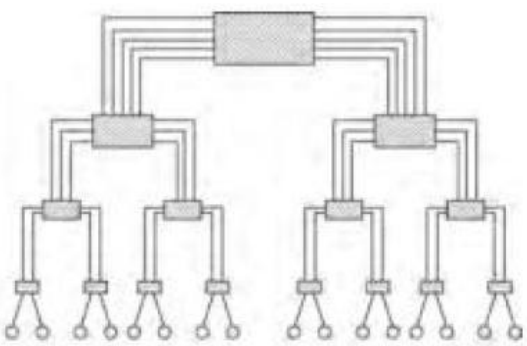


(b) A 16 × 16 Baseline network

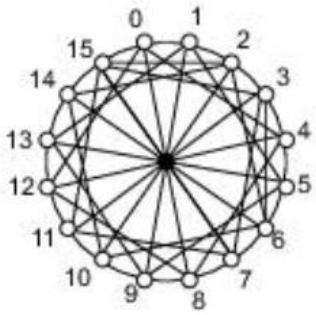
3(a)	Explain Crossbar network with a neat diagram **CrossBar Network** The highest bandwidth and interconnection capability are provided by crossbar networks. A crossbar network can be visualized as a single-stage switch network. Each cross point switch can provide a dedicated connection path between a pair. The switch can be set on or off dynamically upon program demand. Two types of crossbar networks are shown in figure given below. Inter-processor Memory Crossbar network : The pioneering C.mmp implemented a 16*16 crossbar network which connected 16 PDP11 processors to 16 memory modules, each of which had capability of 1 million words of memory cells. The 16 memory modules could be accessed by the processors in parallel. Each memory module can satisfy only one processor request at a time. Only one cross point switch can be set on in each column. However, several cross point switches can be set on simultaneously in order to support parallel memory accesses.	08	CO1	L2



(a) Interprocessor-memory crossbar network built in the C.mmp multiprocessor at Carnegie-Mellon University (1972)

	Inter-processor Crossbar network This large crossbar 224 * 224 was actually built in a vector parallel processor VPP500 by Fujitsu Inc. (1992). The PEs are processors with attached memory. The CPs stand for control processors which are used to supervise the entire system operation, including the crossbar networks. In this crossbar, at one time only one crosspoint switch can be set on in each row and each column. The interprocessor crossbar provides permutation connections among the processors. Only one-to-one connections are provided. Therefore, the n*n crossbar connects at most n source, destination pairs at a time.  (b) The interprocessor crossbar network built in the Fujitsu VPP500 vector parallel processor (1992)			
3(b)	Compare Tree and Fat Tree topology The binary tree will have N= 2k-1nodes. Only drawback is that there will be a lot of traffic towards the root. The maximum node degree is 3 and diameter is 2(k-1). The drawback is overcome in Fat Tree where the channel width increases as we ascend from leaves to the root. The fat tree is more like a real tree in that branches get thicker toward the root. The traffic at the root node is lowered due to high channel width. The idea of a fat tree was applied in the Connection Machine CM-5.  (c) Binary fat tree	02	CO1	L3

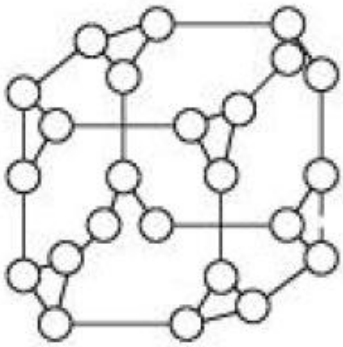
4(a)	Explain the following a) Degree of Parallelism and Parallelism Profile b) Average Parallelism a) Degree of Parallelism and Parallelism Profile The execution of a program on a parallel computer may use different numbers of processors at different time periods during the execution cycle. For each time period, the number of processors used to execute a program is defined as the degree of Parallelism (DOP). The plot of the DOP as a function of time is called the parallelism profile of a given program. The profile will fluctuate depending upon algorithmic structure, program optimization, resource utilization and run time conditions of computer system. DOP is defined under the assumptions of having unlimited resources. b) Average Parallelism Consider a parallel computer with n homogenous processor and maximum parallelism in profile is m. In ideal case n>>m. Δ is computing capacity of single processor. The total work W(computations performed) is given as follows $W = \Delta \sum_{i=1}^{m} i \cdot t_i$ $W = \Delta \int_{t_1}^{t_2} DOP(t)\, dt$ The average parallelism A is given as follows $A = \frac{1}{t_2 - t_1} \int_{t_1}^{t_2} DOP(t)\, dt$ In discrete form, we have $A = \left(\sum_{i=1}^{m} i \cdot t_i \right) \Big/ \left(\sum_{i=1}^{m} t_i \right)$	04	CO4	L2
4(b)	Explain the following topologies with neat diagram. a) Barrel Shifter b) Cube Connected Cycle a) Barrel Shifter Consider number of nodes to be N. $N=2^n$ and barrel shifter has node degree of d= 2n-1 and diameter D= n/2. This implies that node i is connected to node j if $j-i = 2^r$ for some r = [0, 1... n-1]. For N = 16 , the barrel shifter has a node degree of 7 with a diameter of 2. But. the barrel shifter complexity is still much lower than that of the completely connected network.	06	CO1	L2



(e) Barrel shifter

b) Cube Connected Cycle

This architecture is modified from the hypercube. 3-cube is modified to form 3-cube connected cycle (3-CCC) with network diameter of 6. The idea is to replace the corner vertices of the 3 cube with a ring (cycle) of 3 nodes as shown below in the figure.

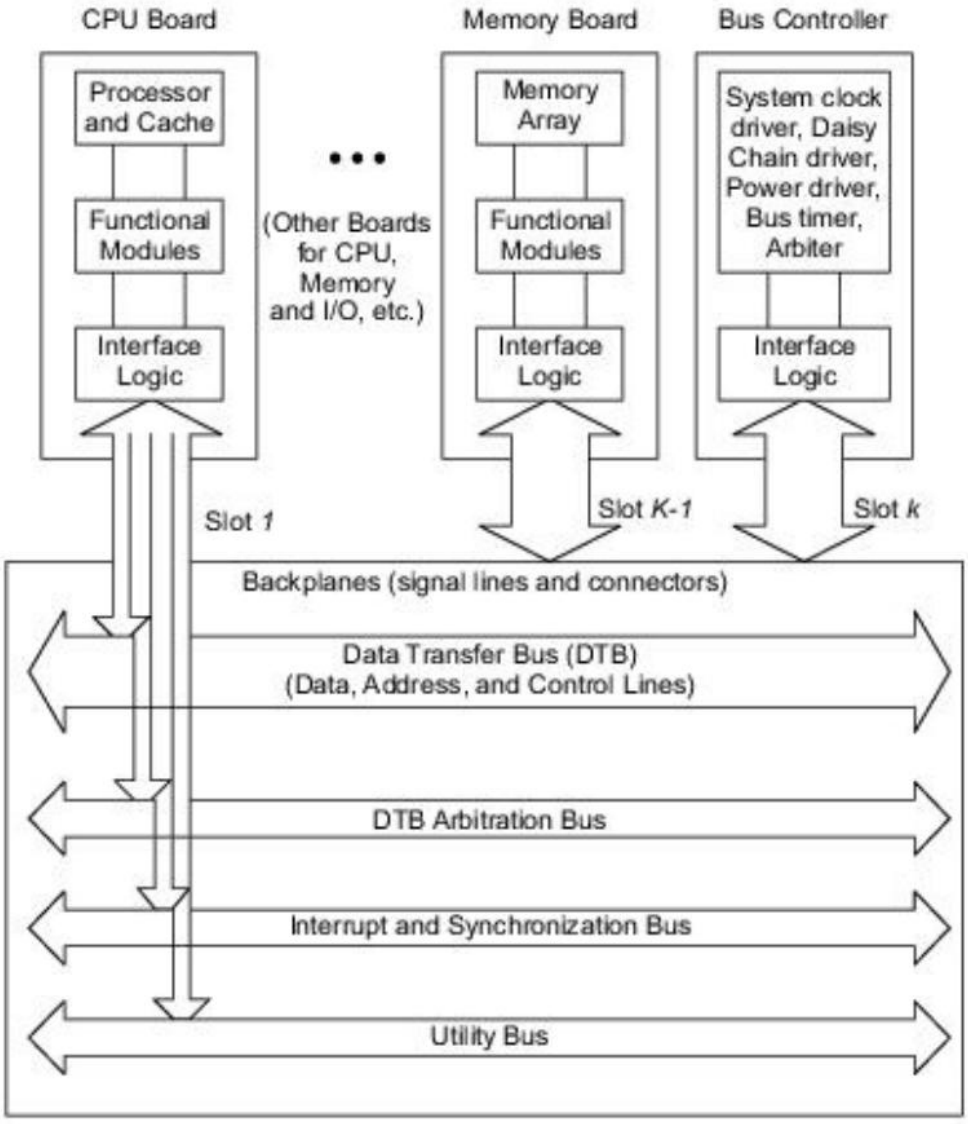


(c) 3-cube-connected cycles

In general k cube connected cycle can be formed using a k-cube with n= 2^k cycles and each cycle will have k nodes. Thus k-cube can be transformed to a k-CCC with k*2^k nodes. In general the network diameter of k-CCC is 2k. The major improvement of CCC lies in its constant node degree of 3, which is independent of the dimension of the underlying hypercube. Also CCC is better architecture for building scalable systems if latency can be tolerated in some way.

5	Explain Backplane Bus Specification with neat diagram. 1. The backplane bus interconnects processors, data storage and peripheral devices. 2. Timing protocols should be developed to arbitrate (making decision to grant bus access for particular request) among multiple requests. 3. Operational rules must be set to ensure orderly data transfers on the bus. 4. The backplane bus is shown in given figure. 5. Various functional boards are plugged into slots on the backplane which has one or more connectors for inserting the boards. <u>Data Transfer bus</u> It contains control, data and address lines. The address lines are used to broadcast the data and address. The number of address lines is proportional to the logarithm of the size of the address space. Address modifier lines can be used to define special addressing modes. The data lines are often proportional to the memory word length. The DTB	10	CO3	L3

Control lines are used to indicate read/write, timing control, and bus error-conditions.



Bus Arbitration and Control

The process of assigning the DTB bus to the requesting processor is called arbitration. The requester is called as master and processor receiving request is called slave. Special dedicated lines for managing the arbitration process. Utility lines include signals for managing the power up and power down sequences of the system. A special bus controller board has backplane control logic, such as the system clock driver, arbiter, bus timer, and power driver

Functional Modules: Various functional modules are given below

Arbiter: It accepts bus requests from the requester module and grants control of the DTB to one requester at a time.

Bus Timer: It measures the time each data transfer takes on the DTB and terminates the DTB cycle if a transfer takes too long.

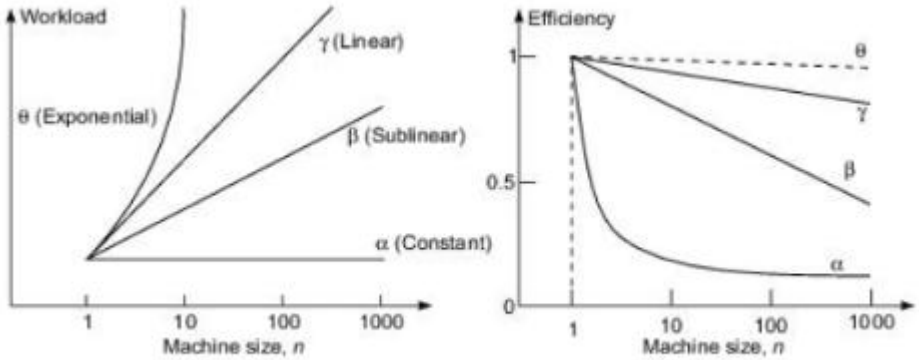
Interrupter Module: It generates an interrupt request along with its status /ID information.

Location monitor: Monitors the data transfer over the DTB bus

Power Monitor: It watches the status of the power source and signals when power becomes unstable.

	System clock driver: It is a module that provides a clock timing signal on the utility bus. Physical Limitations 1. Due to electrical, mechanical, and packaging limitations, only a limited number of boards can be plugged into a single backplane bus. 2. The bus system is difficult to scale and mainly limited by contention. 3. Multiple backplane buses can be mounted on the same backplane chassis.			
6 (a)	Explain characteristics of the parallel algorithm The characteristics of parallel algorithms are listed below. 1. Deterministic algorithm: For a given particular input, the computer will always produce the same output but in the case of non-deterministic algorithm, for the same input, the compiler may produce different outputs in different runs. Hence Deterministic algorithms are implementable on real machines and have polynomial time complexity. 2. Computational granularity: Granularity decides the size of data items and program modules used in computation. In this sense, we also classily algorithms as fine-grain, medium -grain, or coarse-grain. In fine-grained parallelism, a program is broken down to a large number of small tasks. In coarse-grained parallelism, a program is split into large tasks. Medium-grained parallelism is a compromise between fine-grained and coarse-grained parallelism, where we have task size and communication time greater than fine-grained parallelism and lower than coarse-grained parallelism. Most general-purpose parallel computers fall in this category. 3. Parallelism Profile: The effectiveness of parallel algorithms is determined by the distribution of degree of parallelism. 4. Communication patterns and synchronization requirements: Communication patterns address both memory access and interprocessor communications. The patterns can be static or dynamic. Static algorithms are more suitable for SIMD or pipelined machines, while dynamic algorithms are for MIMD machines. The synchronization frequency also affects the efficiency of algorithm. 5. Uniformity of Operations: It indicates that same or uniform operations are performed on the dataset and data is partitioned across various parallel nodes for data level parallelism. Suitable for SIMD processing or pipelining. 6. Memory requirement and data structures: Memory efficiency gets affected by the usage of different data structures and data movement patterns in algorithm. The efficiency of parallel algorithm can be computed by analyzing the space and time complexity	06	CO1	L2

6 (b)	Write note on following benchmarks. 1. Dhrystone 2. Whetstone Dhrystone: Developed by Reinhold Weicker in 1984, Dhrystone is a synthetic benchmark software program used to test a computer's processor's integer performance. The unit KDhrystones/s is often used in reporting the results. The Dhrystone benchmark version 1.1 was applied to a number of processors. DEC VAX11/780 scored 1.7 KDhrystones/s performance. Whetstone: This is a Fortran-based synthetic benchmark assessing the floating-point performance, measured in the number of KWhetstones/s that a system can perform. The benchmark includes both integer and floating-point operations involving array indexing, subroutine calls, parameter passing, conditional branching, and trigonometric and transcendental functions.Both the Dhrystone and Whetstone are synthetic benchmarks whose performance results depend heavily on the compilers used. Both benchmarks were criticized for being unable to predict the performance of user programs. The sensitivity to compilers is a major drawback of these benchmarks.	04	CO1	L2
7(a)	Explain all the application models and efficiency curve with neat diagram. Efficiency Curves • If the workload is unchanged or constant, the efficiency will decrease rapidly as shown in the figure. Hence it is necessary to increase the machine size and problem size proportionally. Such a system is known as a scalable computer for solving scalable problems. The efficiency curve is denoted as α which corresponds to Amdahl's law • If the workload is linear it is ideal case and efficiency curve denoted by γ is almost flat. • If the workload is not linear, the second choice is to have sub linear scalability as close to linearity as possible as illustrated by curve $\boldsymbol{\beta}$ in figure. The sub linear efficiency curve $\boldsymbol{\beta}$ lies somewhere between curves α and γ. • If the workload has exponential growth pattern the system is poorly scalable. The reason is that to keep a constant efficiency or a good speedup, the increase in workload with problem size becomes explosive and exceeds the memory or IO limits. Thus the efficiency curve $\boldsymbol{\theta}$ is achievable only with exponentially increased memory capacity.	08	CO1	L2



(a) Four workload growth patterns (b) Corresponding efficiency curves

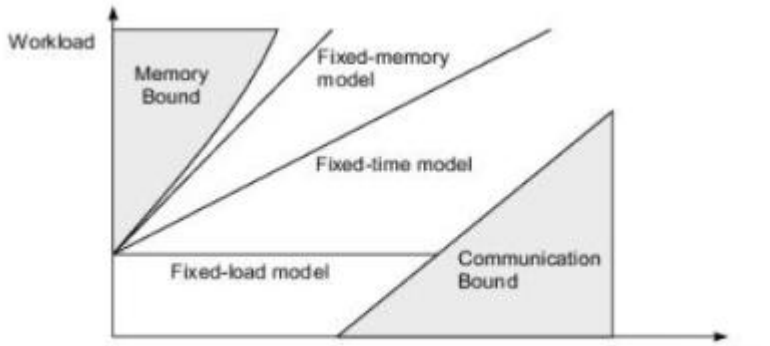
Application Models

There are three speedup performance models defined below which are bounded by limited memory, limited tolerance for latency of IPC and limited IO bandwidth.

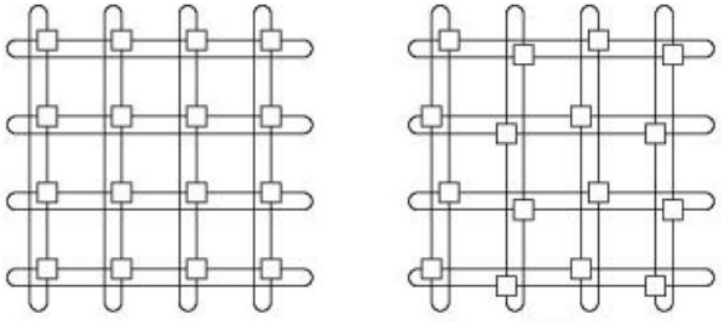
Fixed Load Model: It corresponds to a constant workload with efficiency curve α . It is limited by communication bound shown in shaded area in figure.

Fixed Time Model: It corresponds to linear workload with efficiency curve γ.

Fixed Memory Model: It corresponds to a workload between the curve γ and $\boldsymbol{\theta}$. It is limited by memory bound shown in shaded area in figure.



(c) Application models for parallel computers

7(b)	What is Torus Topology? What are advantages of Folded Torus over Torus? The torus has ring connections along each row and along each column of the array. In general, an n*n binary torus has a node degree of 4 and a diameter of 2(n/2). The torus is a symmetric topology. Folded Torus has uniform wire length as shown in the figure given below  (a) Traditional torus (a 4-ary 2-cube) (b) A torus with folded connections	02	CO1	L3

8	Explain Amdahl's law with neat diagram	10	CO4	L2

Amdahl's law for fixed workload

The time critical applications where the goal is to minimize the turnaround time provided the major motivation behind the development of fixed load speedup model and Amdahl's law.

Here we consider a fixed workload and hence as the number of processors increases in parallel computer the workload is distributed to more processors for parallel execution. Consider Degree of Parallelism(DOP) as DOP= i and i> =n where n is the number of processor/machine size. Assume the workload as W_i and it is executed by all n processors and m is maximum parallelism observed and Δ is computing capacity of each processor.

The execution time of W_i is

$$t_i(n) = \frac{W_i}{i\Delta}\left\lceil \frac{i}{n} \right\rceil$$

The response time is

$$T(n) = \sum_{i=1}^{m} \frac{W_i}{i\Delta}\left\lceil \frac{i}{n} \right\rceil$$

Now we define the fixed load speedup factor S_n as ratio of T(1) to T(n). T(1) indicates response time for the uni-processor system and hence

$$T(1) = \sum_{i=1}^{m} Wi/\Delta$$

$$S_n = \frac{T(1)}{T(n)} = \frac{\sum_{i=1}^{m} W_i}{\sum_{i=1}^{m} \frac{W_i}{i}\left\lceil \frac{i}{n} \right\rceil}$$

Let Q(n) be overhead contributed by interprocessor communication, latencies of memory access etc.Hence we can rewrite S_n as

$$S_n = \frac{T(1)}{T(n)+Q(n)} = \frac{\sum_{i=1}^{m} W_i}{\sum_{i=1}^{m} \frac{W_i}{i}\left\lceil \frac{i}{n} \right\rceil + Q(n)}$$

Also Q(n) is dependent on application as well as machine.

Consider a situation where the system can operate in only two modes i.e. sequential mode with DOP=1 and perfectly parallel mode with DOP=n Then S_n is given as follows

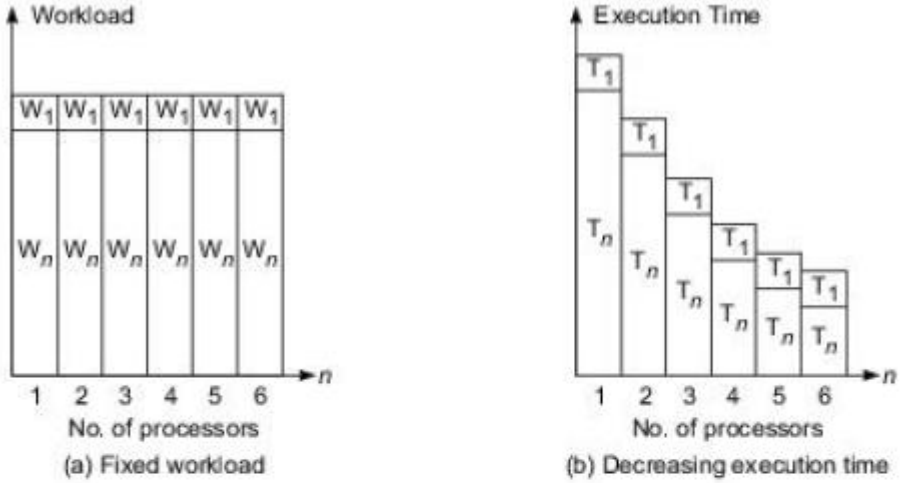
$$S_n = \frac{W_1 + W_n}{W_1 + W_n/n}$$

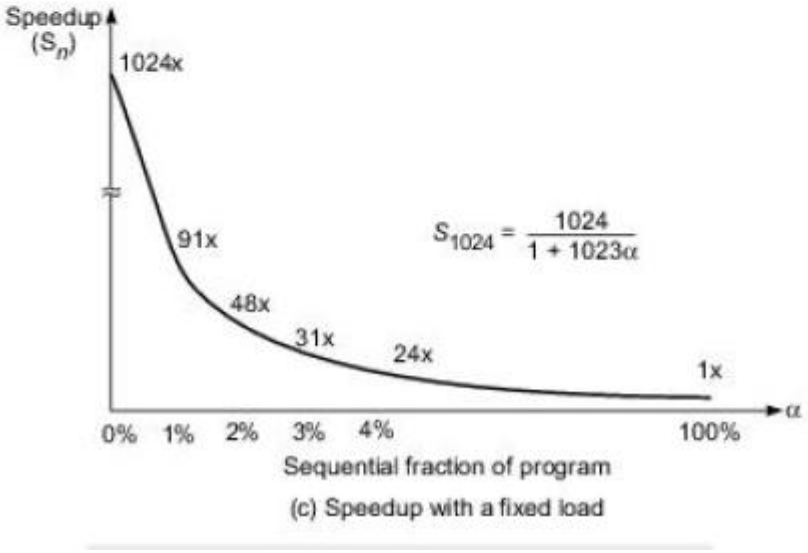
Hence execution time reduces when the parallel portion of the program is executed by n

processors, but the sequential portion of the program does does not change and requires constant amount of time as shown in figure below. Thus

$W_1+W_n=\alpha + (1- \alpha) = 1$

Here α **represents** the percentage of the program that must be executed sequentially and 1- α corresponds to portion of the code that can be executed in parallel. The total amount of workload W_1+W_n is kept constant.





The speedup curve decreases very rapidly as the α increases. This means that with a small percentage of the sequential code the entire performance cannot go higher than 1/ α. Hence this is the sequential bottleneck in the program which cannot be solved by increasing the number of processors.