CHAPTER 1

INTRODUCTION

In recent years, the concept of computing has shifted slowly and steadily from desktop to distributed systems. Internet is the best example for distributed systems. Media such as television, telephone and radio has considerably improved their power and range of communication. This seems to be that there are no longer any restrictions on communication with space and time requirements that have to be effectively synchronized. The Internet is a well established worldwide communications medium for the entire spectrum of communication modes such as data, voice and video in both real-time and non-real time.

In 1960 the Internet emerged as a research and university network for providing services such as e-mail, file transfer, etc. In 1980’s the annual investment for Internet is more than billions of dollars which acts as backbone of commercial markets. Today’s Internet consists millions of networks with their customers revolving around it. Internet Service Providers (ISP) serves Internet through a wired or mobile network, through dial-up modem, Digital Subscriber Line, Integrated Services Digital Network, High Speed Digital Lines, Fibre Optic, Satellite Broadband Network, Mobile Broadband Network, WiMax via a handset, SIM card or USB modem. As a result, Internet traffic is growing by a factor of 30% per year [1] [2]. In 1995 the world Internet users was 4,48,38,900/- which is less than 1% of the total population. In 2005 the world Internet users increased to 1,029,717,906 which is around 16% of the total population whereas in 2015 the number drastically increased to 2,925,249,355 which is 40% of the world total population. This shows the necessity
of Internet technology for the improvement in every human life and also for the
development of countries [3]. Fig 1.1 shows the amount of Internet users as of 2014.

Advances in fibre optic technology support ISP’s to handle the explosive
growth in number of users and traffic per user effectively. New applications such as
Internet Television, Audio and Video multicasting, Voice over Internet Protocol
(VoIP) requires higher bandwidth range of multi-terabit per second. Consequently, the
total data rate of a single fibre is increasing at a faster rate than the switching and
routing equipment that terminates and switches traffic at a carrier’s central office or
Point of Presence (PoP). This emerges the need for better switching and routing
methodologies to keep up with Internet growth.

![Fig.1.1 Internet users in the world](image)

**Fig.1.1** Internet users in the world

### 1.1 COMMUNICATION NETWORK

Communication Network provides a scalable solution to connect large amount
of systems in a view to achieve unlimited data transfer activities. Two broad
categories of communication networks are Circuit Switched Network and Packet Switched Network.

1.1.1 Circuit Switched Network

Circuit Switched Network (CSN) uses a single fixed communication path between source and destination. Telephone communication is the well known example for CSN which works with fixed range of bandwidth in which the path remains open irrespective to whether the call is actively used or not. This results in requirement for longer call period starting from connection establishment to its termination. When user makes a call, his telephone switch gets connected with local access network. Multiplexer is used to establish connection between source and destination and access networks through any high speed medium. Receiver telephone switch in-turn will connect to its local access network based on its availability. This type of network gives utmost importance to circuits rather than packets. As a result, it requires signalling to set up switch tables and also under performs in case of heavy traffic [4] [5].

1.1.2 Packet Switched Network

In Packet Switched Network (PSN), delivery of packets from source to destination involves two operations namely Routing and Switching. Routing determines the path through which the packet reaches the destination. Switching takes place at every node to switch the packet in the pre-determined route/port. PSN uses variable sized information which is segmented into fixed sized data called packets [6]. A packet consists of three fields namely Header, User Data and Trailer as shown in Fig.1.2.
The header indicates the start of new packet and contains source and destination address, packet sequence number and other routing information. User data field contains actual information to be transmitted with error control mechanisms. The trailer contains a cyclic redundancy checksum used for error detection and prevention process. Based on the destination address placed in header field of a packet, data is transmitted from source to destination system. Unlike circuit switching, each packet can be transmitted in different path ways. Based on the current traffic conditions, they can be dynamically routed through different path ways of the network. In the destination, packets are collected and reassembled to form the original information.

Unlike circuit switching, packet switching may use any resources made available by the various inactive sources. Therefore it is well adapted to applications where the data transfer is not stable [7]. Another major advantage is its capability to use the network bandwidth effectively through multiplexing. As circuit switching uses dedicated line for data transfer, queuing mechanisms are not needed whereas for packet switching multiple sources are involved therefore they use queuing mechanisms. Recent study [2] confirms the emergence of optical switch transmission will reach a saturation point and therefore packet switch will remain economically cheaper.
1.2 PACKET SWITCHING ARCHITECTURES

A packet switch, also known as backplane or switch fabric is a controller to various networked devices involved in data transfer process. Fig.1.3 shows a network switch connected with workstations and servers. The task of the backplane is to transfer the packets between workstations and servers. Each resource in the network is connected with a switch port through which the communication takes place.

Fig.1.3 Network Switch connected with different workstations and servers

1.2.1 Fabric Stages and Losslessness

A switch fabric can be either in single stage or multistage fashion. A single stage switch is a non-blocking fabric and uses single path to connect the input with output port. They are easy to build, comprehend and analyze. They exhibit strong performance characteristics over multistage fabrics [8] [9]. However, single-stage switches are not scalable compared to multi-stage switches as their cost grows with respect to increase in number of ports. A multistage fabric is a set of single-stage
fabrics which works in parallel. Its advantage is the flexibility it offers to extend to large number of ports. But due to its implementation cost single stage switch fabric is usually preferred which is then integrated to multistage fabric, if required.

Most of the fabrics are lossless because they tend to drop packets in different situations [10]. For instance, simultaneous arrival of packets to different inputs, destined to the same output gives rise to a phenomenon called output contention. Switch fabric which suffer from output contention will lead to packet loss. Therefore to avoid packet loss, a packet-switch must contain some sort of Queueing. That is, when a packet loses contention, must be queued. Therefore a switch fabric must include queuing along with switching process. Placement of queue in the most appropriate location of the switch fabric is most important as it has significant impact on its performance, hardware cost and implementation feasibility.

1.2.2 Typical Switch Fabrics

Three types of switch fabrics are used today. They are Bus architecture, Shared Memory architecture and Crossbar architecture.

*Bus Architecture*

Bus architecture is simple to implement which uses single medium where only one packet is transmitted at a timeslot. Also it needs to take care of input-output contention for the transfer to takes place. Data is transmitted through the bus using Time Division Multiplexing (TDM) as shown in Fig.1.4. Here both the input and output ports contact the memory through the bus. Shared memory and its access are controlled by an arbitration mechanism. The bus should be capable to serve all the ports in the given period of time. Major challenge in Bus architecture is its ability to
work with MultiGigabit speed process and the drawback is its capacity to hold packets. ATOM switch [11] is an example for Bus architecture.

**Fig.1.4** Shared Medium Bus

**Shared Memory Architecture**

In this fabric, a buffer is shared as shared memory by the incoming packet which subsequently reduces the amount of packet loss. All the input and output port share a common memory as shown in Fig.1.5. Memory allocation is handled by a centralized manager called Memory Manager and each output port manages its output queueing process. Major drawback with this fabric is the need for high speed memory. If the fabric port number is N and the link speed is S, then a single port shared memory must run at the speed of 2NS. Therefore these fabrics require speedup to achieve maximum performance which results in high implementation cost. Prelude switch [12] is an example for shared memory architecture.
Crossbar Architecture

A crossbar fabric consists of a two-dimensional array of crosspoint switches, one for each input-output pair as shown in Fig.1.6. Thus an NxN switch contains N^2 crosspoints through which a packet can move from any input port to any output port, subject to its availability. In any interconnection networks, crossbar is the most preferred fabric because of its low cost, enhanced scalability and non-blocking properties.

Most of the high performance switches such as Ethernet switch, ATM switch, etc. prefers crossbar architectures because it is

- Simple to implement
- Allows high speed operations (in Gbps) with point to point connectivity
- Increases the aggregate bandwidth through multiple I/O transactions
- Avoids packet loss through queueing
- Enhanced scalability due to limitation in bus transfer and memory access bandwidth
Cisco switches are example for crossbar architecture which is commercially successful.

1.3 TYPES OF CROSSBAR SWITCHES

Different types of crossbar switches are Input Queued Switch, Output Queued Switch, Combined Input Output Queued Switch and Buffered Crossbar Switch.

1.3.1 Input Queued (IQ) Switch

A crossbar switch is said to be Input Queued Switch if queueing takes place before switching. On arrival of a packet, switch examines its destination address from the header field and determines the output port through which it is forwarded. All the packets received at the input port cannot be switched in a timeslot and therefore a queue is required to avoid packet loss. A crossbar switch with queue in the input side is called as Input Queued Switch [13] and is shown in Fig.1.7.
An input queued switch has the crossbar running at the same speed of the Line Rate R. At each timeslot, queue at the input side will not receive or depart more than one cell. As a result memory needs to operate on twice the speed of line rate i.e 2R speed to receive or depart a cell. This makes the IQ switch to build high bandwidth at low cost and with scalability features. But an input-queued (IQ) switch with a single FIFO queue at each input has a poor performance due to Head-of-Line (HOL) blocking.

**Head-of-Line Blocking**

All the cells need to be switched are placed in a FIFO queue. At each queue the first cell that is ready for switching is termed as Head of Line cell. At each timeslot, Input Queued Switch with FIFO queue will switch only one cell from an input port to a destined output port. Suppose, HOL cell of more than one queue destined to same output port, then HOL blocking occurs [14]. This is because only one cell can be transferred to the output port at a time. HOL blocking considerably
reduces the throughput performance of the IQ switch. Virtual Output Queuing [15] [16] [17] is a simple queue structure which eliminates the HOL blocking problem and is discussed in the next section.

**Virtual Output Queuing**

A Virtual Output Queue (VOQ) is deployed for each input port instead of FIFO queue. The size of VOQ depends on the size of the output port and therefore there will be $N^2$ input queues where each queue follows FIFO procedure. For example, a queue of size 4x4 has 4 input and 4 output ports. All the 4 input queues have 4 VOQ's each as the size of the output port is 4. In VOQ, each input maintains a separate queue for output. At each timeslot, a cell is switched from a VOQ to each output and therefore HOL blocking is eliminated [18].

![Virtual Output Queuing architecture](image)

Fig.1.8 Virtual Output Queuing architecture

Fig.1.8 shows the Virtual Output Queue for an NxN switch. State information of all the VOQ’s are maintained by the scheduler. At each timeslot, the scheduler
collects the status information from all the input and chooses the cells needed to be transferred to the output. This process takes place based on the employed scheduling algorithm. It uses request-grant command to complete the entire transfer activities in the given timeslot. With the inclusion of VOQ in IQ switches, the throughput performance has been improved by more than 25% [17] [19].

1.3.2 Output Queued (OQ) Switch

A crossbar switch is said to be Output Queued Switch if switching takes place before queueing. In this architecture, a queue is placed on the output side of the switch to receive the packets whose destinations fail to collect it. The Output Queued Switch has no queue on the input side and therefore all the received packets must be switched immediately to the output port. Failure to do so leads to packet loss [20]. At each timeslot, if the input port is able to switch all the packets to its output port then the switch can attain maximum throughput but is practically not feasible. To switch the packets simultaneously whenever it arrives, too much of interconnection and memory bandwidth (for speedup) is required. This increases the implementation cost of the fabric which is a major disadvantage.

An Output Queued Switch is depicted in the Fig.1.9 with NxN input-output ports. At each timeslot, $N$ cells must switch through $N$ input port simultaneously. If each external link runs at a rate $R$, then the memory must run at a speed of $(N+1)R$. This requirement is known as the internal speedup of a switch [21]. As the users in the network increases, there will be demand in memory bandwidth which is difficult to handle with OQ switches.
1.3.3 Combined Input-Output Queued Switch

Speedup is one of the solutions to overcome HOL blocking problem. It is defined as the ratio at which the internal fabric must operate in comparison to the external links. During switching process, when a speedup of more than 1 is used, then CIOQ architecture is required. Combined Input-Output Queued Switch (CIOQ) as shown in Fig.1.10 uses queue on both the sides of the switch [22] [23]. Thereby packet loss at both the sides of the switch can be avoided. Building a high speed switch is not a constructive process when its implementation cost increases in accordance with the required speedup.
1.3.4 Buffered vs Bufferless Crossbar switches

Bufferless scheduling algorithms have reached their practical limitations due to higher port numbers and data rates. Internal BCS have started to attract researchers, because of the great potential they have in solving the complexity and scalability issues faced by their bufferless predecessors [24]. The increasing demand for terabit switches and routers mean that future commercial packet switches must be implemented with reduced scheduling complexity. The buffered crossbar architecture can inherently implement distributed scheduling schemes and has been considered a viable alternative to bufferless crossbar switches to improve performance. The presence of internal buffers drastically improves the overall performance of the switch as it offers two distinct advantages. First, the adoption of internal buffers makes the scheduling totally distributed, dramatically reducing the arbitration complexity. Second, and most importantly, these internal buffers reduce (or avoid) output
contention as they allow the inputs to make cell transfers concurrently to a single output.

Among various architectures, buffered crossbars are preferred because they have simpler scheduling algorithms than a bufferless crossbar. In a bufferless crossbar, the scheduler must find a matching between inputs and outputs that leads to complex scheduling algorithms. Also the buffered crossbar does not need a centralised scheduler since scheduling is distributed in BCS. It can be pipelined to run at high speed, making buffered crossbars appealing for high performance switches and routers. It also considerably reduces the hardware cost. Most researchers have proved that buffered crossbars provide good throughput for admissible uniform traffic with simple algorithms.

1.4 BUFFERED CROSSBAR SWITCHES

In Buffered Crossbar Switches (BCS), a buffer is placed on all the crosspoints of the switch. For an \( N \times N \) switch, a total of \( N^2 \) buffers are used. The presence of these buffers avoids the input-output contention hence making the scheduling process much simpler and distributed. Therefore a cell can be switched on the input side without knowing the status of the output port. Consequently the complexity involved in the scheduling process has been reduced.

Normally BCS uses a FIFO queue on the input side which is called Combined Input Crosspoint Queued (CICQ) switch [25]. CICQ switches offers good performance under both uniform and non-uniform traffic patterns [26] but suffers from HOL blocking. Therefore FIFO queue is replaced with VOQ which is called as Buffered Crossbar Switch with Virtual Output Queue fabric (BCS/VOQ) proposed by [26]. BCS/VOQ avoids HOL blocking thereby switch performance is further
improved. Moreover these fabrics are said to be stable because they are free from packet loss.

Fig.1.11 4x4 BCS/VOQ Switch

A 4x4 BCS/VOQ switch fabric is shown in the Fig.1.11 with 4 input and 4 output ports. A buffer is placed on each crosspoints and number of VOQ on the input side is 4.

1.4.1 BCS/VOQ Scheduling

The performance of the Buffered Crossbar Switch is based on the employed scheduling algorithm. Scheduling algorithm decides when and where to switch the packets and therefore designing an appropriate scheduling algorithm is must for any BCS [27] [28]. The main goal is to find the appropriate match between input and output ports. Since BCS is distributed in nature, the fabric uses two schedulers: Input Scheduler and Output Scheduler. Input Scheduler selects a HOL cell from each VOQ and transfers it to the crosspoint buffer. Thus a one-to-one matching is established between input and crosspoint queue during every timeslot. Also in parallel, the Output Scheduler establishes a match between a crosspoint buffer and the output port and the cell is transmitted. Both input and output scheduling takes place at each timeslot by
switching their respective cells. Upon completion of output scheduling, state information of crosspoint buffers are supplied to input through Flow Control Mechanisms. Researchers around the world have proposed various scheduling algorithms for BCS/VOQ switches and are discussed under literature survey section.

**Properties of Scheduler**

Some of the properties of scheduling algorithm are [29]

- Each scheduler is designed to produce high throughput
- All the VOQ’s are equally served to avoid starvation
- Scheduler should be fast to find a match between input and output port
- Simple to implement
- Less implementation cost
- Execute the schedulers without speedup i.e. use of speedup in the switch fabric should be avoided

1.4.2 Buffering

The process of placing a buffer in the crosspoint of BCS/VOQ is termed as Buffering. It avoids input-output contention problem by distributing the scheduling schemes. Considering the hardware limitations and switch performance, it is difficult to finalize the buffer size. Buffers with ideal sizes can prevent the switch from packet overflowing [30]. Normally buffer size of 1 cell size is used by the researchers for stable performance. In certain cases, researchers increased the buffer size to further increase the switch performance which resulted in high implementation cost [31] which refer both space and time requirement. If the buffer size is unlimited, then there is no need for arrival schedule (i.e.) as the input can be directly sent to the buffer. Because of the implementation cost, unlimited buffer size is practically not possible in the BCS/VOQ switches.
1.5 SIMULATION ENVIRONMENT

In this dissertation, JNetworkSim [55] is used to perform all the simulation activities. JNetworkSim is an object oriented network switch simulator written in Java and is developed by Nick McKeown Group, Stanford University. It is considered as a suitable replacement and upgrade for C based simulator SIM. It is not a topology simulator but permits simulation with other algorithms and its data structures. It is modular and works very fast comparing to other switch simulators. Simulation in JNetworkSim is based on timeslots and is done in three steps: arrival of new cells in the input, transfer of cells from input to output of the switch and departure of cells from the output. JNetworkSim include many numbers of plugins to support the structure of buffered crossbar switch, various traffic patterns and performance metrics. For any additional features, any number of required plugins can be developed and included in the kit.

1.6 TRAFFIC MODELS

In a BCS/VOQ switch, traffic decides the way in which the input packets are supplied to the different VOQ’s. A switch performance is assessed through various traffic models. Two commonly used traffic models are Uniform Traffic Model (UTM) and Non-uniform Traffic Model (NTM).

1.6.1 Uniform Traffic Model

Traffic is said to be uniform if all the arrival processes have the same arrival rate, and destinations are uniformly distributed over all the outputs. Two types of Traffic Models are Uniform Traffic and Non-uniform Traffic [17]. Researchers commonly use the uniform traffic for evaluating their proposed schedules.
Under uniform, Cells are generated independently with a probability of p, and the arrival rate is spread uniformly on all the output ports N. Bursty Uniform Traffic resembles the real Internet traffic. Incoming packets are segmented into fixed sized cells to form a bursty traffic. The bursty uniform traffic is modelled by a two-state markov chain which is denoted as "BUSY" and "IDLE". In the BUSY state, packets are generated only to a selected output port with a mean length known as burst size and no packets are generated during the IDLE state.

1.6.2 Non-uniform Traffic Model

Traffic is said to be non-uniform if all the arrival processes have different arrival rate, and generates a variation in distributing the cells over all the outputs. It is normally used to simulate the Internet traffic as they are normally non-uniform and asymmetric. Two commonly used models are Non-uniform i.i.d. traffic and Non-uniform Bursty Traffic.

Traffic is called independent and identically distributed (i.i.d.) if and only if:

- Every arrival is independent of all other arrivals, both at the same input and at different inputs.
- All arrivals at each input are identically distributed.

1.7 PERFORMANCE ASSESSMENT METRICS

Metrics to assess the performance of a switch are throughput, average cell latency/delay, input queue occupancies and average waiting time.

1.7.1 Throughput

Switch Utilization or Throughput is the number of cells received per output port per clock cycle. It is measured as the ratio of input load to output load in a given period of time. In a 4x4 BCS/VOQ switch, high throughput is said to be transferring 4
cells by both input and output scheduler in a given timeslot. A scheduling algorithm is considered stable if it provides 100% throughput and it keeps the input buffer size bound in number of cells. Throughput is one of the very important metric in deciding the performance of the switch.

1.7.2 Average Cell Latency

Cell latency is the total time taken by a cell to switch through input port to output port. It is measured as the difference between the cell departure time at the output port and cell arrival time in the input port or VOQ.

\[
\text{Cell latency} = \text{cell departure time at the output port} - \text{cell arrival time in the input port or VOQ}
\]

In other words, the sum of input queuing delay, transfer time through the switch fabric and output queuing delay provides the average cell delay which is represented in timeslots. Average Cell Latency is computed through the averages of all the cell delays measured during the respective simulation period or for the amount of cells used for switching. It is one of the very important metric which shows the efficiency of the switch.

1.7.3 Input Queue Occupancy

It is one of the important metric to control the cell loss in the input queue/VOQ. Based on the computed value, the size of the FIFO queue or VOQ in the input port can be redefined to avoid cell loss. Stability of a switch with respect to employed schedulers and switch fabric can be determined by the respective metric.

1.7.4 Average Waiting Time

Waiting time is measured as the number of timeslots that a cell is waiting in the virtual output queue / input queue (depends on architecture).
Average Waiting Time is computed through the averages of the entire cell waiting time. This metric is important when analysing the impact of starvation during a schedule process.

1.8 ORGANIZATION OF THE THESIS

The contributions of the dissertation are organized in chapters. Chapter 2 provides the literature survey on the necessary background of the work. It surveys on various switch scheduling algorithms deployed on input queued switches, output queued switches and buffered crossbar switches. The concept of all the algorithms is studied and understood its advantages and shortcomings. Performance of various buffered crossbar scheduling algorithms are analyzed and compared with each other for different load structures. Problems identified with the existing schedulers are listed. Finally the objectives of the dissertation are also listed.

Chapter 3 presents our first contribution, the design and implementation of Prioritized Queue with Round Robin Scheduler (PQRS) for Buffered Crossbar Switches. To avoid HOL blocking, Buffered Crossbar Switch is equipped with Virtual Output Queue on the input side. Proposed scheduler reduces the starvation effect by stabilizing the average waiting time of the HOL cell in the queue. Simulation result shows that the proposed scheduler achieves high throughput with less delay compared to Longest Queue First-Round Robin Scheduler.

In Chapter 4, PQRS is updated as Delay based Prioritized Queue with Round Robin Scheduler (D-PQRS). An incredible effort is made to further reduce the starvation effect. As a result, throughput performance is further increased with

\[
\text{Waiting time} = \text{cell departure time at the VOQ} - \text{cell arrival time in the VOQ}
\]
minimum delay. In Chapter 5, D-PQRS is implemented in Buffered Crossbar Switch with different buffer sizes. Analysis is made to understand the impact of buffer sizes on the proposed scheduler. Simulation result confirms larger sized buffers have a significant impact in case of average cell latency rather than throughput performance.

In Chapter 6, a study is made on the multicast traffic flow scheduling. Our proposed scheduler D-PQRS is further modified to support the multicast traffic and is called as D-PQRS-M. As BCS is considered as an effective architecture for multicast support, D-PQRS-M is implemented on it. It uses both the fanout splitting and no fanout splitting strategy. Simulation shows that the proposed scheduler performance is better than MXRR scheduler in terms of throughput and average cell latency. Also D-PQRS-M performance under fanout splitting is better than no fanout splitting.

Finally, Chapter 7 provides concluding remarks on the presented work. Chapter summarizes the contributions on the dissertation and gives the future research directions.