Tertiary Incentive For Data Transmission Using Switches

Abstract
Load-Balancing switch is a simple architecture and have high forwarding bandwidth. The original Load-balancing switch has mis-sequencing problem which hinders the performance of underlying TCP applications. This problem has been solved at the cost of either algorithmic complexity or special hardware requirements. In this paper we introduce three stage load balancing switch architecture to solve this mis-sequencing problem. It achieves a high forwarding capacity while preserving the order of packets without the need of costly online scheduling algorithms. Theoretical analysis and experimental results show that this three stage load balancing switch provides a transmission delay which can be solved by output queued switch plus a constant, indicating the same forwarding capacity as an output-queued switch. Therefore 100 % throughput has been obtained while using this three stage load balancing switch.

Keywords
Networking, Relational Database Management System, Circular Queue

I. Introduction
Most high performance packet switches today use input queuing, and a non-blocking (usually crossbar) switch fabric. To overcome head-of-line blocking and enable high throughput, the input buffers are arranged as virtual output queues (virtual output queues). To simplify the tasks of memory management and scheduling, a time slot which has a fixed size is used and hence arriving variable length packets are segmented into fixed size packets, or "cells". Each time slot a centralized scheduler examines the contents of the virtual output queues to determine the configuration of the switch fabric for the next time slot. Numerous papers have studied this approach, and have proposed new scheduling algorithms that are simple to implement provide throughput guarantees or provide delay guarantees. The scheduler often limits the scalability of the system because of the frequency and complexity of its decisions. Introduces a remarkable two-stage switch, within which each stage uses a trivial settled sequence of configurations. The switch is straightforward to implement at high speed and has been proven to supply 100 percent turnover for a broad category of traffic. What is more, there’s a sure between the common delay of the two-stage switch which of a perfect output-queued switch. However, in its simplest type, the switch mis-sequences packets by Associate in Nursing discretionary quantity.

In Networking there has been an excellent deal of interest recently in load equalization switches because of their easy design and high forwarding capability. A typical load-balancing switch consists of a two-stage switch cloth and a one-stage buffer sandwiched between the two switch fabrics where each fabric executes a periodic connection pattern. The idea behind this kind of design is that the input traffic with an arbitrary distribution becomes uniformly distributed through the first-stage switch fabric and then is fully forwarded by the second-stage switch fabric. Thus, high output is achieved whereas the design remains ascendible. To address the mis-sequencing problem, several new load-balancing switch designs have been proposed in the literature. The mis-sequencing problem is resolved by either resequencing out-of-order packets at the outputs or preventing it from occurring. The two schemes with the first-in–first-out (FIFO) and the earliest deadline first (EDF) scheduling policies are proposed to resequence mis-sequenced packets at the output ports, respectively. In the FIFO-based load-balancing switch, a jitter controller is placed in front of the second-stage virtual output queue (virtual output queue) to impose proper delays for packets so that they arrive to the resequencing buffer in sequence. In the EDF-based load-balancing switch, packets are scheduled out of the resequencing buffer according to their deadlines.

II. Modules
- Load – Balancing Scheme
- Multipath Switching System
- Multistage Multiplane Close Switches
- Performance Analysis

A. Load Balancing Scheme
In this paper, we introduce a new load-balancing switch, called the three-stage load-balancing switch, which effectively solves the mis-sequencing problem without the need of costly online scheduling algorithms or hardware speedup. The architecture which we studied in this and has been briefly described. The architecture described in this paper significantly extends these initial efforts, providing a theoretical analysis and evaluating its performance through extensive simulations.

B. Multipath Switching System
The forwarding process in the first two stages is straightforward. For the first crossbar, when Output i.e., Internal Input is connected to input i.e., External Input the packet at the head of S1-Virtual Output Queue is transmitted to Internal Input. Upon arriving at Internal Input, the packet is written to the Virtual Output Queue corresponding to its destined Internal Output. For instance, packet is written to S2-Virtual Output Queue. For the second crossbar, when output i.e., Internal Output is connected to Internal Input the packet at the head of S2-Virtual Output Queue is transmitted from S2-Virtual Output Queue to Internal Output. Upon arriving at Internal Output, the packet is written into the memory block which is reserved.

C. Multistage Multiplane Close Switches
A Circular Queue uses a head pointer and a tail pointer to keep track of the beginning and the end of the circular queue inside the Circular Queue, respectively. The queue length of a Circular Queue is defined as the distance between the tail and head pointers, i.e., the number of the memory blocks that are being re-served by incoming packets. A memory block is released when the packet occupying it is forwarded out of the Circular Queue. At the very beginning, the tail and head pointers coincide with each other, indicating a queue length of zero. The length of a Circular Queue increases as memory blocks are assigned to newly arrived packets. As the queue grows, the tail pointer moves forward inside the Circular Queue. The tail pointer is updated ac- cording to (1), as described before. When the tail pointer catches up with the head pointer, the Circular Queue becomes full and no more memory blocks are available. In this...
case, newly arrived packets are dropped until a packet is forwarded out of the Circular Queue.

The length of a Circular Queue decreases as packets are transmitted from the Circular Queue to the corresponding External Output port. To preserve the packet order, Circular Queues forward out packets from their heads. After the packet at the head is delivered to the External Output port, its memory block is released, and the head pointer points to the next memory block, in accordance with (1). Since packets with the same destined External Output ports travel through different paths to the Circular Queues, it is likely that some packets will arrive in Circular Queues earlier than their preceding ones, forming “holes” inside the Circular Queues. A hole refers to a segment of one or more consecutive empty memory blocks. If a head pointer reaches such a hole, it cannot move forward until the hole is filled up. The existence of the holes indicates that the arrival order of the packets to stage 3 is different from their arrival order to stage 1.

D. Performance Analysis

Compared to the CR switch and the FB switch, the three stage load balance switch yields significantly larger average delays under light traffic. The large average delays are contributed by a combination of the additional third stage and the purely deterministic packet-forwarding algorithm in the third stage. Due to the periodic connection patterns running on the three crossbars, if a packet cannot get forwarded at the current time-slot, it has to wait for one connection cycle (i.e., time-slots) before the next connection comes. What is more, as packets are forwarded in increasing order of their positions in the circular queues, a packet has to wait for all its prior packets to be served before it gets forwarded. Thus, a packet is very likely to be blocked in the circular queue by its prior packets that experience different delays in the first two stages due to the randomness of the input traffic. The combined effect of the two factors leads to the large packet delays of the three stage load balance switch even under light traffic.

III. Circular Queue

What is the need for logical Circular Queues in the third stage? The Circular Queues provide an architectural support for output load-balancing, which further leads to finite buffers in the second stage. Consider a slightly “modified” three stage load balance switch where we do not use such Circular Queues, namely, each memory bank alone is used as a re-sequencing buffer. With the memory block-reserving mechanism, every MB still maintains a circular queue and forwards packets from the head. As each MB is associated to an External Output, the third switch fabric is also removed. Note that without the third stage, any particular Virtual Output Queue holds packets for one External Output, where as when using the third stage, it holds packets for multiple External Outputs. During a time-slot, a Circular Queue processes up to requests/grants, receives up to packets, and sends at most one packet to its corresponding External Output port. The request/grant signals are conveyed by a specific control path. This signaling approach is similar to what is used in an Input Queue switch with a centralized Isilp algorithm. The packet arrivals in a Circular Queue are processed by the MBs, each performing one memory write at a time-slot. As packets are always forwarded from the heads of the Circular Queues and the MBs are connected to the different External Output ports, each MB only needs to perform one memory read at a time-slot. Thus, the Circular Queues remain as scalable as other components in the three stage load balance switch.

IV. Architecture Diagram

A. Architecture Overview

We now introduce the switch architecture shows the architecture of an three stage load balance switch consisting of three stages of buffering and switching. The input and output ports of the whole switch are called external inputs and external out-ports, respectively. In contrast, the input and output ports of the middle stage (i.e., the second stage) are called internal inputs and internal outputs, respectively. Note that the Internal Inputs coincide with the output ports of the first stage, and so do the Internal Outputs with the input ports of the third stage. The External Input, Internal Input, Internal Output, External Output ports are indexed with, respectively. In the three stage load balance switch, each of the three switch fabric is an crossbar running with a periodic sequence of connection patterns such that each input is connected to every output once during consecutive time-slots, and vice versa. Concretely, we choose for each crossbar a periodic sequence such that every output is connected to every input in a round-robin (RR) fashion in order of increasing input port index.

Stage 1:

There are virtual output queues at each External Input port. Each virtual output queue corresponds to one of the Internal Input ports. A combination of a flow splitter and a load balancer is placed at each External Input port to evenly distribute packet flows into the virtual output queues, where a flow is defined as an External Input–External Output pair (e.g., see [2] and [4]). Denote by Switch-1-virtual output queue the th virtual output queue at External Input port. All packets buffered in Switch-1-virtual output queue are designed to be transmitted to Internal Input. Packets are stored in Switch-1-virtual output queue in an FIFO manner.
Stage 2:
There are virtual output queues at each Internal Input port. Each virtual output queue corresponds to one of the Internal Output ports. Denote by Switch-2-virtual output queue the th virtual output queue at Internal Input port. All packets buffered in Switch-2-virtual output queue are designed to be transmitted to Internal Output port. Packets are stored in Switch-2-virtual output queue in an FIFO manner.

Stage 3:
Between the second and third crossbars are memory banks (MBs), each lying in front of one of the inputs of the third crossbar. An MB is nothing but an array of contiguous memory blocks, where each memory block can hold one packet. Each MB is composed of memory blocks and is evenly divided into segments.

V. Conclusion
In this paper, we introduced a three-stage load-balancing switch and studied its performance via theoretical analysis and experimental simulation. In particular, we demonstrated a method to resolve the mis-sequencing problem by using a nested load-balancing scheme (i.e., a combination of the input and output load-balancing mechanisms). Moreover, we rigorously proved that the transmission delay of the three stage load balance switch is bounded by that of an Output Queue switch plus a constant that depends only on the number of the input/output ports, and, as a result, it achieves the same forwarding capacity as the Output Queue switch. By simulations, we showed that the constant difference between the average packet delays of the three stage load balance and Output Queue switches holds for several typical input traffic models. Finally, we extended our analysis onto the three stage load balance switch as well as their impacts. The output load-balancing mechanism is critical to the performance of the three stage load balance switch in that in combination with input load-balancing, it produces finite queue lengths at the first two stages, regardless of the input traffic pattern. This property implies that packets are buffered only in the third stage, i.e., in circular queues, in the event of congestion. In this sense, the circular queues in the three stage load balance switch correspond to the output queues in the Output Queue switch: A circular queue is always busy whenever its length exceeds a certain threshold. Such a property leads to the relationship between the three stage load balance and Output Queue switches, that is, the length of a circular queue is upper-bounded by that of an output queue, plus a constant, which in turn implies the identical forwarding capacities of them, as shown in Theorem 4. Like prior load-balancing switches, the three stage load balance switch exemplifies the possibility of achieving high performance using common hardware resources. We believe that the deterministic operations of the three stage load balance switch help facilitate more hardware-based packet-forwarding functionalities.

References