Matching Output Queueing with Combined Input and Output Queueing

Nick McKeown[*], Balaji Prabhakar[‡], and Mingyan Zhu[*]

[*]Department of Electrical Engineering, Stanford University, Stanford, CA 94305-9030, USA.

[‡]Hewlett-Packard BRIMS, Filton Road, Stoke Gifford, Bristol BS12 6QZ, England.

Abstract

At very high aggregate bandwidths, output queueing is impractical because of insufficient memory bandwidth. This problem is getting worse: memory bandwidth is improving slowly, whereas the demand for network bandwidth continues to grow exponentially. The difficulty is that output-queued switches require memories that run at a speedup of N, where N is equal to the number of switch ports. This paper addresses the following question: Is it possible for a switch to ***exactly*** *match output-queueing with a reduced speedup? We prove that if virtual output queueing is used, a combined input-output queued switch is always work-conserving if its speedup is greater than $N/2$. This result is proved using a novel scheduling algorithm - the Home Territory Algorithm (HTA).*

1: Introduction

Many commercial switches and routers today employ output queueing.[1] The advantages of output queueing are two-fold. First, it allows the throughput to be maximized: so long as no input or output is oversubscribed, the switch will support the traffic. Second, because packets are immediately placed in output queues upon arrival, it is possible to control the latency of packets through the switch. This is very important for supporting QoS in a switch or router. But output queueing is complex and expensive. Its complexity arises from a need for the switch fabric and memory to run N times as fast as the line rate, assuming an $N \times N$ switch. The ratio of a switch's internal bandwidth to its line rate is defined as the speedup of a switch. Hence, an output-queued (OQ) switch has a speedup of N.

Input queueing has lower complexity, and consequently lower cost. The switch fabric and the memory at the inputs of an input-queued (IQ) switch need only run as fast as the line rate. In other words, an input-queued switch has a speedup of 1. However, if each input maintains a single FIFO queue, input queueing suffers head of line (HOL) blocking, which limits the throughput to just 58.6%. For non-uniform traffic, the performance is even worse [1].

Output queueing is limited by the bandwidth of commercially available memories. Currently, it seems practical to implement an output-queued switch or router with an aggregate bandwidth of approximately 20Gb/s. But the continued exponential increase in demand for bandwidth is making faster and faster switches necessary. Before long, it may be impractical to build output-queued switches.

A lot of research has been done to approach the performance of output queueing using a combined input-output queued switch. A combined input-output queued (CIOQ) switch is defined as a non-blocking packet switch with N inputs and N outputs

1. When we refer to output queueing in this paper, we include designs that employ centralized shared memory.

and operates S times as fast as the line rate. With a speedup of S, a CIOQ switch can remove up to S packets from each input and deliver up to S packets to each output within a time slot. A time slot is the time between packet arrivals at input ports. Since $1 < S < N$, packets need to be buffered before switching at the inputs as well as after switching at the outputs.

Various properties of CIOQ switches have been studied, such as average packet delay, maximum throughput, packet loss probability, packet blocking probability, optimal buffer allocation, effects of arbitration policy on throughput, etc.[2][3][4] and [5]. It has been shown that, when the input traffic is independent and uniform, a CIOQ switch can achieve 99% throughput with a modest speedup of approximately four [3][4][5].

The aim of our work is different. Rather than find values of speedup that work well on average, or with relatively benign types of traffic, we aim to find the minimum speedup such that a CIOQ switch behaves *identically* to an OQ switch for *all* types of traffic. Here, 'behave identically' means that by observing only the output processes, an OQ switch is indistinguishable from a CIOQ switch. The output processes are indistinguishable if and only if they are busy at the same time, and have the same packet departure order.

An OQ switch is work-conserving since its outputs never idle when there are packets in the system for them. Therefore, a CIOQ switch that operates identically, is also work-conserving. In our work, we have searched for techniques such that a CIOQ switch can be work-conserving. Note, a work-conserving CIOQ switch may not have the same packet departure order as an OQ switch.

Our work has two results. First, we formalized the intuitive notion that a $N \times N$ switch with a single FIFO queue at each input requires speedup $S = N$ to be work conserving; i.e. it must be an output-queued switch. Second, we found that a $N \times N$ CIOQ switch that uses *virtual output queueing*[1] can be work conserving if $S > \frac{N}{2}$, *for any type of input traffic*. We proved this result using a novel scheduling algorithm: the Home Territory Algorithm (HTA).

This paper is organized as follows: Section 2 describes our combined input-output queued packet switch model, Section 3 discusses the main results of our research, and Section 4 summarizes our findings.

2: Our CIOQ Packet Switch Model

The combined input-output queued switch we consider, shown in Figure 1, is a single-stage packet switch. It has N inputs, N outputs, a non-blocking space division switch fabric, and a scheduler. Both the inputs and the outputs are labelled from 0 to $N-1$.

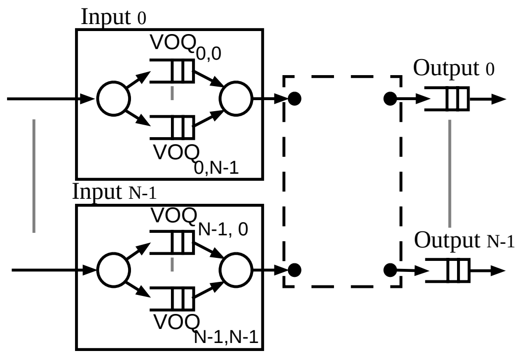

Figure 1: Our CIOQ Packet Switch

The switch we consider operates in synchronous mode where packets arrive and are transmitted synchronously. Packets are assumed to have fixed length and fixed transmission time. We choose the time between packet arrivals as our time unit, and call it a time slot.

1. This is a technique commonly used in input-queued switches to eliminate HOL blocking. With virtual output queueing, a switch maintains a separate FIFO at each input for each output.

The switch has a speedup of S, i.e. its fabric runs S times as fast as the line rate, where $1 < S < N$. A $N \times N$ switch with speedup of S can remove at most S packets from each input and can transmit at most S packets to each output in a time slot.

A CIOQ switch requires buffering at its inputs as well as at its outputs. We assume that all buffers have infinite capacity. Each output maintains a FIFO queue, and operates on a first come first serve basis. Each input maintains a separate FIFO queue for each output. Hence, there are N FIFO queues at each input. These FIFO queues are Virtual Output Queues(VOQ), where VOQ_{ij} denotes a FIFO queue at input i that queues packets destined to output j.

A scheduling algorithm selects a matching between the inputs and outputs. A matching is selected in such a way that each non-empty input is connected to at most one output and each output is connected to at most one input. For a CIOQ switch with a speedup of S, a scheduling algorithm carries out the matching S times per time slot. Our switch uses a novel scheduling algorithm called the Home Territory Algorithm (HTA). Section 3 gives a detailed description of HTA.

3: Main Results

3.1: Without Virtual Output Queues

It is intuitively clear that if a CIOQ switch maintains a single FIFO queue at each input, then a speedup of N is sufficient for it to be work-conserving. In fact, it is both necessary and sufficient:

Theorem 1: *For a CIOQ switch that maintains a single FIFO queue at each input, a speedup of N is required for it to be work-conserving.*

Proof: See Appendix A.

In an effort to match the performance of output-queueing, it is worth asking the questions: Is this limited by HOL blocking? If not, can we arrange the input queues differently and use a lower speedup to make the switch conserve work? As we shall see, the answer to the first question is “No!”, whereas the answer to the second is “Yes!”.

3.2: With Virtual Output Queueing.

In this section, we describe a novel scheduling algorithm - the Home Territory Algorithm - and prove our main result: A CIOQ switch that uses VOQ buffering and operates under HTA conserves work for all types of input traffic when its speedup is greater than $N/2$.

3.2.1: Definitions

Definition: *Phases.*

A time slot is divided into S equal intervals, where S is the speedup of the switch. Each of these intervals is called a *phase*. During phase i, denoted Φ_i, $1 \le i \le S$, a switch can remove at most one packet from each input, and can transfer at most one packet to each output.

Definition: *Output Home Territory (HT^O) and Non Home Territory (NHT^O) sets.*

Each output partitions the set of inputs into two disjoint subsets: its Home Territory (HT^O) set and its Non Home Territory (NHT^O) set. The HT^O set of output j is denoted $HT^O(j)$, and similarly the NHT^O set of output j, $NHT^O(j)$. This partitioning is based on the following static priority that each output assigns to each input.

The priority assigned by output j to input i is denoted $P_j^O(i)$. If $i > j$, then $P_j^O(i) = i - j$, else $P_j^O(i) = (i + N) - j$. In other words, output j assigns the highest priority to input $(j + 1)$ Mod N, the second highest priority to input $(j + 2)$ Mod N, and so on.

Every output ranks all inputs from priority 1 to N. Here, smaller number indicates higher priority. The top S inputs constitute its NHT^O set and the remaining inputs constitute its HT^O set, so that $|NHT^O| = S$ and $|HT^O| = N - S$.

Table 1 gives the territory sets of output j and lists the priorities it assigns to all inputs. Figure 2 shows the $NHT^O(1)$ set in a 5×5 switch with a speedup of 3. The priority of each input assigned by output 1 is shown next to the input. The arrow points to the $NHT^O(1)$.

Table 1: Output Territory Sets & Input Priorities

	Inputs	Priority(P_j^O)
$NHT^O(j)$	*(j + 1) Mod N*	*1*
	......	......
	(j + S) Mod N	*S*
$HT^O(j)$	*(j + S + 1) Mod N*	*S + 1*
	......	......
	(j + N) Mod N	*N*

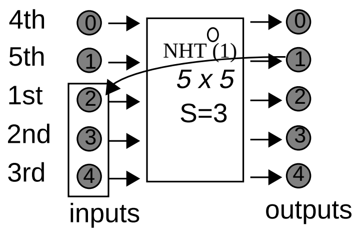

Figure 2: Non Home Territory Set of Output 1 and the Input Priorities Assigned by Output 1 in a 5x5 Switch which has a Speedup of 3.

Definition: *Input Home Territory (HT^I) and Non Home Territory (NHT^I) sets.*

Each input partitions the set of outputs into two disjoint subsets: its Home Territory (HT^I) set and its Non Home Territory (NHT^I) set. The HT^I set of input i is denoted $HT^I(i)$, and similarly the NHT^I set of input i, $NHT^I(i)$.

Again, the partitioning into Home and Non Home Territory sets is based on priorities. The priority assigned to output j by input i equals the priority assigned to input i by output j. In other words, $P_i^I(j) = P_j^O(i)$. An input then assigns the top S ranked outputs to its NHT^I set and the remaining outputs to its HT^I set.

Table 2 gives the territory sets of input i and lists the priorities it assigns to all outputs. Figure 3 shows the $NHT^I(4)$ set in a 5×5 switch with a speedup of 3. The priority of each output assigned by input 4 is shown next to the output. The arrow points to the $NHT^I(4)$.

Table 2: Input Territory Sets & Output Priorities

	Outputs	Priority(P_i^I)
$NHT^I(i)$	*(i + N - 1) Mod N*	*1*
	......	......
	(i + N - S) Mod N	*S*
$HT^I(i)$	*(i + N - S - 1) Mod N*	*S + 1*
	......	......
	i Mod N	*N*

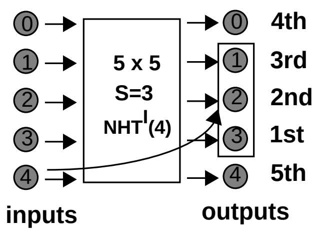

Figure 3: Non Home Territory Set of Input 4 and the Output Priorities Assigned by Input 4 in a 5x5 Switch which has a Speedup of 3.

Definition: *Home Territory VOQs and Non Home Territory VOQs.*

The VOQs at each input are divided into two disjoint subsets according to the NHT^I and HT^I sets of that input, and they are the VOQ_{NHT} set and the VOQ_{HT} set. For an arbitrary input i,

$VOQ_{NHT}(i) = \bigcup_{j \in NHT^I(i)} \{VOQ_{ij}\}$, and

$VOQ_{HT}(i) = \bigcup_{j \in HT^I(i)} \{VOQ_{ij}\}$. Each VOQ_{ij} has its own unique and fixed priority. We use $P_{VOQ_{ij}}$ to denote the priority of VOQ_{ij}, and $P_{VOQ_{ij}} = P_i^I(j)$.

3.2.2: Home Territory Algorithm

The description of Home Territory Algorithm is divided into three parts. The first part describes the service policy in a single phase. The second part describes how an input selects a granting output. The third part describes the service policy in a single time slot.

Part I. Service Policy in a Phase

The service provided by HTA in a phase has two steps, and they are:

1. conflict free matching between inputs and outputs
2. packet forwarding from inputs to outputs.

HTA finds a conflict free matching in multiple iterations. All inputs and outputs are initially unmatched at the beginning of the first iteration. Only those inputs and outputs that are unmatched at the end of the previous iteration are eligible for matching in subsequent iterations. Iterations are carried out until no further matching is available.

Each iteration has three steps, which operate in parallel on unmatched inputs and unmatched outputs. These three steps are:

1. Each unmatched input sends a request to every output for which it has a queued packet.
2. If an unmatched output receives any requests, it grants to the requesting input with the highest priority.
3. If an unmatched input receives any grants, it accepts an output based on the policy outlined in Part II.

Part II. How an Input Selects a Granting Output

This selection process is carried out in each iteration of every phase. And it has two steps:

1. The input divides its granting outputs into two disjoint sets: the Unserved Set, consisting of outputs that haven't been served by the input in a previous phase of the current time slot; and the Served Set, consisting of outputs that have been served at least once in a previous phase of the current time slot.
2. The input selects a granting output from one of these sets in the following way: If the Unserved Set is non-empty, then the output with the highest priority in that set is selected, otherwise the output with the highest priority in the Served Set is selected.

In other words, an output cannot receive consecutive services at an input, unless all other requesting outputs have been served at least once at this input.

Part III. Service Policy in a Time Slot

HTA carries out the single phase service policy S times per time slot, where S is the speedup of the switch.

3.2.3: Mimicking output-queueing with the HTA

The main result of this section is Theorem 2, which establishes that a CIOQ switch operating under HTA and with speedup bigger than *N/2* conserves work. We will assume that the switch is completely empty at time *t=0* and that packets arrive after this time.

Lemma 1: *The VOQ_{NHT}s of all inputs are empty at the end of every time slot. In other words, for any input i, if*

$$\sum_{j \in NHT^I(i)} |VOQ_{ij}(t)| = 0 \text{, then}$$

$$\sum_{j \in NHT^I(i)} |VOQ_{ij}(t+1)| = 0 .$$

Proof:

We prove this lemma by induction. Suppose that the VOQ_{NHT} set at each input is empty at the end of time *t*. We will show that if VOQ_{ij} is non-empty at the beginning of time $t+1$, where $P_j^O(i) = k$ and $k \le S$, then it will be empty at the end of time $t+1$.

Suppose VOQ_{ij} isn't served at any phase before Φ_k. Then at Φ_k, input i has the highest priority among all requesting inputs at output j. By the induction hypothesis, all NHT VOQs were empty at the end of time *t*. Given that there is at most one packet arrival per time slot per input, there can be at most 1 packet in each of the NHT VOQs for output *j*. Hence there can be at most $k-1$ competing inputs that have higher priority than input *i* at output j. But, these inputs must have been served during phases Φ_l, $l < k$. Therefore HTA ensures that output j grants input *i*'s request at or before Φ_k. And input i accepts output *j*'s grant, because output j is guaranteed to be in input *i*'s Unserved Set. Hence, if $P_j^O(i) = k$ and $k \le S$, VOQ_{ij} is guaranteed to be cleared/emptied no later than Φ_k. Thus, there will be no packets in VOQ_{NHT}s at the end of time $t+1$. ❐

Lemma 2: *At any time slot, if $|VOQ_{ij}| \ne 0$ and $j \in HT^I(i)$, then at input i, there can be at most one granting output j', such that $P_i^I(j') > P_i^I(j)$ and $j' \in NHT^I(i)$.*

Proof:

All VOQ_{NHT}s are empty at the end of every time slot (Lemma1) and at most one packet can arrive at each input per time slot. If the newly arrived packet at input i is destined to output j' and $j' \in NHT^I(i)$, then $P_i^I(j') > P_i^I(j)$. Before input i can accept any granting outputs from its Home Territory set, it must use one phase to clear/empty $VOQ_{ij'}$. Otherwise, $VOQ_{NHT}(i)$ is empty, and input i can start immediately to accept granting outputs from its Home Territory set. ❐

Theorem 2: *A $N \times N$ combined input-output queued packet switch using the Home Territory Algorithm is work conserving for all types of input traffic iff its speedup is greater than $N/2$.*

Proof:

Consider any output j for which there is a cell at some input. We show that output j will be serviced and hence that HTA conserves work. There are essentially two cases to consider.

1. Output *j* has a cell at a NHT VOQ. In this case, Lemma 1 asserts that this cell will be forwarded and j will not idle.

2. The only cell for j is in the HT at some input i. If this is the case, j can loose contention to at most one NHT VOQ at input i (as a consequence of Lemma 2) and to at most $N-S-1$ other HT VOQs at input i. This means it can loose contention in at most $N-S$ phases. But, the speedup $S > N/2$ implies that $S > N-S$. Therefore, there is a phase at which output j will be served at input i, even after all these higher

priority outputs are served.

This proves sufficiency. ❐

To prove the necessity of Theorem 2, we present the following traffic pattern which defeats the Home Territory Algorithm when the speedup is less than or equal to $N/2$.

Given $S = n$, if a $2n \times 2n$ switch fails to conserve work, then a $(2n+1) \times (2n+1)$ switch also fails. Therefore, we choose to use a $2n \times 2n$ switch to show the traffic pattern.

The following steps enunciate the proof:

- Consider a $2n \times 2n$ switch operating under HTA.
- Suppose its speedup $S = n$.
- Pick an arbitrary input i.
- Choose VOQ_{iJ} to be the target VOQ, where $P_{VOQ_{iJ}} = 2n$. Observe that we have picked a VOQ which has the lowest priority at input *i*. The traffic pattern has been designed to stress the target VOQ such that at some time one of its packets is required at output *J* and yet can't be forwarded because of HTA, thus causing the switch to fail to conserve work.
- The VOQs at input i are shown in Figure 4. The shaded area represents VOQ_{NHT}, while the non-shaded area represents VOQ_{HT}. The priority of each VOQ is also indicated in the figure.

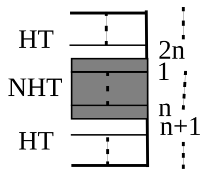

Figure 4: VOQs at input *i*

The following traffic pattern defeats HTA:

1. Identify output *j*, $j \in HT^I(i)$ and $P_{VOQ_{ij}} = n+1$.
2. Let all inputs receive packets destined to output *j*.
3. Repeat Step 2 until VOQ_{ij} has enough packets to stay non-empty till such time a packet from the target VOQ fails to be delivered to its output. (Note: It is a priori possible to determine how long Step 2 should be repeated to achieve said failure.)
4. Repeat Steps 1 & 2 for all output j's, $j' \in HT^I(i)$ and $n + 2 \leq P_{VOQ_{ij'}} \leq 2n - 1$.
5. Repeat Step 1 for target VOQ, i.e. VOQ_{iJ}.
6. Now choose an output j'', such that $j'' \in NHT^I(i)$. Let $VOQ_{ij''}$ receive a packet per time slot from now on.
7. Repeat Step 6 until the FIFO queue at output *J* becomes empty.
8. Now output *J* is waiting for the packet at VOQ_{iJ}. However, this packet can't be forwarded according to HTA, since there are n non-empty VOQs at input *i* that have a higher priorities than the target VOQ. Because $S = n$, the switch fails to conserve work.

This traffic pattern proves necessity. (Note: It may be instructive to construct the traffic pattern taking n=4 and S=2.) ❐

Intuitive Explanation of the Theorem.

For a CIOQ switch that uses HTA to be work conserving, its minimum required speedup is determined by the expression $max(|HT| + 1, |NHT|)$. According to the definition of NHT, $|NHT| = S$. Also during any time slot, to guarantee work conservation, $|HT| + 1$ phases are needed to serve all non-empty VOQs at least once. Hence, it becomes obvious that a speedup of roughly $N/2$ is able to ensure work conservation.

4: Conclusion and Future Work

In this paper, we have proved that a

combined input-output queued switch that uses VOQ buffering can be work-conserving for any type of input traffic if its speedup is greater than $N/2$. This result is proved using a novel scheduling algorithm, the Home Territory Algorithm.

This result only meets part of our original goal, which is to find techniques such that a CIOQ switch can behave identically as an OQ switch. Here, 'behave identically' means their output processes are indistinguishable, given that the input traffic is the same. HTA only guarantees these processes to be busy at the same time. To make the processes indistinguishable, we also need techniques which can guarantee identical packet departure order. This has been our current research focus.

5: References

[1] Karol, M.; Hluchyj, M.; and Morgan, S. "Input versus output queueing on a space division switch," IEEE Trans. Commun., Dec. 1987, vol.COM-35, no.12, p.1347-56.

[2] I. Iliadis and W.E. Denzel, "Performance of packet switches with input and output queueing," in Proc. ICC '90, Atlanta, GA, Apr. 1990. p.747-53.

[3] A.L. Gupta and N.D. Georganas, "Analysis of a packet switch with input and output buffers and speed constraints," in Proc. InfoCom '91, Bal Harbour, FL, Apr. 1991, p.694-700.

[4] Y. Oie; M. Murata, K. Kubota, and H. Miyahara, "Effect of speedup in nonblocking packet switch," in Proc. ICC '89, Boston, MA, Jun. 1989, p. 410-14.

[5] J.S.-C. Chen and T.E. Stern, "Throughput analysis, optimal buffer allocation, and traffic imbalance study of a generic nonblocking packet switch," IEEE J. Select. Areas Commun., Apr. 1991, vol. 9, no. 3, p. 439-49.

Appendix A: Proof of Theorem 1.

A 1: *Speedup Requirement of a Work Conserving Switch that has a Single FIFO Queue at Each Input.*

Theorem 1: *If a $N \times N$ switch maintains a single FIFO queue at each input, then a speedup of N is required for it to be work-conserving.*

To prove this theorem, all we need to show is that such a switch fails to conserve work when $S = N - 1$. An input traffic pattern that causes the switch to fail is shown in Figure 5. Assume the scheduler is optimum, i.e. can look infinitely into the future and make the best scheduling decisions possible in an effort to make the switch conserve work.

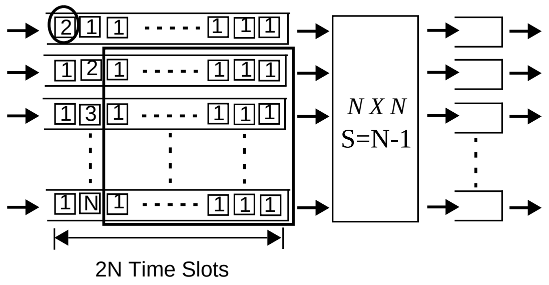

Figure 5: A Counter Example

In Figure 5, each row of packets represents the arrival pattern at an input, with row 1 corresponding to arrivals at input 1, row 2 corresponding to arrivals at input 2, and so on. Each column represents the arrival pattern for all inputs at a single time slot, with the right most column representing arrivals at time slot 1. Packets destined for a particular output are labelled by that output's number. For example, all packets destined to output 1 are labelled 1.

Given the input traffic pattern shown in Figure 5, for the switch to be work conserving at time slot $2N$, it needs to forward $2N$ packets to output 1, two packets to output 2, and one packet to each of the remaining outputs.

In order to conserve work at time $2N - 1$, *any* scheduler must serve inputs *2* through N in each preceding time slot. This will allow it to remove all the 1's marked in the rectangle and thus it will conserve work at time $2N - 1$. However, this leaves a backlog of packets in input 1 for output 1 (all the 1's in the topmost row), and there will be $2N - 1 - (N - 1) = N$ of these 1's at the end of time $2N - 1$.

Therefore, the cell arriving at time $2N$ at input 1, destined for output 2 (shown circled in Figure 5) *cannot* be served, due to HOL blocking.

Hence, *any* schedule is doomed to fail when the speedup is less than *N*, proving the theorem. ❒