Minimum Delay Scheduling in Scalable Hybrid Electronic/Optical Packet Switches

Bin Wu and Kwan L. Yeung
Dept. of Electrical and Electronic Engineering
The University of Hong Kong
Pokfulam, Hong Kong
E-mail: {binwu, kyeung}@eee.hku.hk

***Abstract*—A hybrid electronic/optical packet switch consists of electronically buffered line-cards interconnected by an optical switch fabric. It provides a scalable switch architecture for next generation high-speed routers. Due to the non-negligible switch reconfiguration overhead, many packet scheduling algorithms are invented to ensure performance guaranteed switching (i.e. 100% throughput with bounded packet delay), at the cost of speedup. In particular, minimum delay performance can be achieved if an algorithm can always find a schedule of no more than *N* configurations for any input traffic matrix, where *N* is the switch size. Various minimum delay scheduling algorithms (MIN, α^i-SCALE and QLEF) are proposed. Among them, QLEF requires the lowest speedup bound. In this paper, we show that the existing speedup bound for QLEF is not tight enough. A new bound which is 10% lower than the existing one is derived.**

Keywords-Minimum delay scheduling; performance guaranteed switching; reconfiguration overhead; speedup bound.

I. INTRODUCTION

The explosion of Internet traffic has led to ever-increasing demands for larger bandwidth and higher port density in the next generation routers. At present, most Internet backbone routers are based on a single-rack solution using a switched backplane. Typically, a standard telecommunication rack is of size 19 inches in width and 7 feet in height. It can accommodate 14-16 line-cards, with aggregate capacity up to 160 Gb/s. To further increase the capacity, multi-rack solution [1] is adopted, where line-cards in different racks are interconnected to/from the central electronic switch fabric by fibers. This architecture defines the 4th generation router, which can offer an aggregate capacity up to 10 Tb/s [2]. Since electronic switch fabric is used, O/E/O conversions are necessary at the central switch rack. As data is handled in electronic domain, power consumption becomes the key constraint [3]. To solve these issues, the 5th generation router is proposed as shown in Fig. 1, where a hybrid electronic/optical switch architecture is adopted. Compared to the 4th generation, the central electronic switch fabric is replaced by an optical one. This not only removes the O/E/O conversions from the switch rack, but also reduces its power consumption. Following this way, the aggregate capacity can be up to 100 Tb/s [3-4].

However, the optical switch fabric needs a non-negligible amount of time to change its configurations, known as *reconfiguration overhead*. During this period, no packet can be transmitted across the switch. Reconfiguration overhead is due to three factors [5]. First, the optical fabric needs time to change its interconnection pattern, which can range from 10 ns to several milliseconds depending on the technology adopted [6-9]. Second, time is required to resynchronize the optical transceivers and the switch fabric in each reconfiguration. Finally, because the arriving time of optical signals varies, the clock and its phase have to be aligned, and extra clock margin has to be considered in order to avoid data loss.

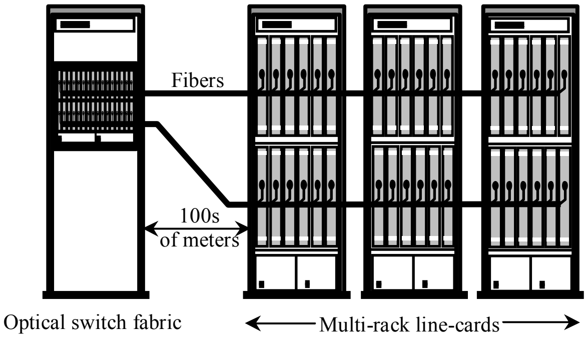


Fig. 1. The 5th generation router architecture.

To achieve *performance guaranteed switching* (i.e. 100% throughput with bounded packet delay) [10-13], the switch fabric must run faster to compensate for both the reconfiguration overhead and the scheduling inefficiency. The required *speedup S* is defined as the ratio of the internal packet transmission rate to the external line-rate ($S\geq 1$).

Assume time is slotted and each time slot can accommodate one fixed-size packet. Several scheduling algorithms are proposed to achieve performance guaranteed switching [10-13]. They all adopt the same four-stage scheduling procedure as shown in Fig. 2. Stage 1 is for traffic accumulation. An $N\times N$ traffic matrix $\boldsymbol{C(T)}=\{c_{ij}\}$ is obtained at the input buffers every T time slots, where N is the switch size. Each entry c_{ij} denotes the number of packets arrived at input i and destined to output j. As a common assumption [10-13], the entries in each line (i.e. row or column) of $\boldsymbol{C(T)}$ sum to at most T. In Stage 2, a scheduling algorithm generates a schedule consisting of (at most) N_S switch configurations in H time slots. Each

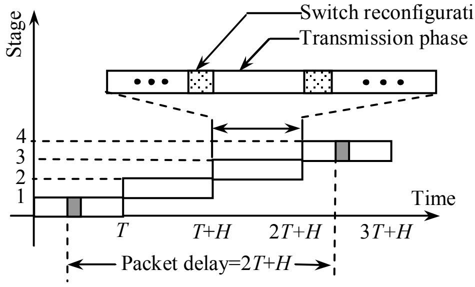


Fig. 2. Timing diagram for packet scheduling.

This work is supported by Hong Kong Research Grant Council Earmarked Grant HKU 7032/01E.

1-4244-0357-X/06/$20.00 ©2006 IEEE

configuration is denoted by a permutation matrix $\boldsymbol{P_n}=\{p^{(n)}_{ij}\}$ ($N_S \geq n \geq 1$), where $p^{(n)}_{ij}=1$ means that input port i is connected to output port j (In this case, we also say that $\boldsymbol{P_n}$ covers entry (i, j)). A weight ϕ_n is assigned to each $\boldsymbol{P_n}$, indicating the number of slots that $\boldsymbol{P_n}$ should be kept for packet transmission. The set of N_S configurations generated must *cover* $\boldsymbol{C(T)}$, i.e. $\sum^{N_S}_{n=1} \phi_n p^{(n)}_{ij} \geq c_{ij}$ for any $i, j \in \{1, \ldots, N\}$. Then $\sum^{N_S}_{n=1} \phi_n$ is the number of slots required to transmit all the packets in $\boldsymbol{C(T)}$. Let each reconfiguration take an overhead of δ time slots. Accordingly, sending $\boldsymbol{C(T)}$ requires $\delta N_S + \sum^{N_S}_{n=1} \phi_n$ time slots. This is generally larger than the traffic accumulation time T. Without speedup, 100% throughput is not possible. Stage 3 is for actual packet transmission, where the switch fabric is reconfigured according to the N_S configurations. At a speedup of S, the slot size for a single packet transmission in Stage 3 is shortened by S times. Then 100% throughput is ensured by having

$$\delta N_S + \frac{1}{S}\sum_{n=1}^{N_S}\phi_n = T \, . \tag{1}$$

The values of N_S and $\sum^{N_S}_{n=1} \phi_n$ in (1) are determined by the scheduling algorithm. Note that the total reconfiguration overhead time δN_S cannot be reduced by speedup and thus $T > \delta N_S$. Finally, Stage 4 takes another T time slots to send the packets onto the output lines from the output buffers.

Rearranging (1), we have

$$S = \frac{1}{T - \delta N_S}\sum_{n=1}^{N_S}\phi_n = S_{\text{reconfigure}} \times S_{\text{schedule}}, \tag{2}$$

where $S_{\text{reconfigure}}$ and S_{schedule} are defined as

$$S_{\text{reconfigure}} = \frac{T}{T - \delta N_S} \tag{3}$$

$$S_{\text{schedule}} = \frac{1}{T}\sum_{n=1}^{N_S}\phi_n \tag{4}$$

$S_{\text{reconfigure}}$ is the speedup factor to compensate for the idle time caused by reconfigurations, whereas S_{schedule} is the speedup factor to compensate for the scheduling inefficiency.

In Fig. 2, the packet delay is bounded by $2T+H$ slots where $T > \delta N_S$. With a smaller N_S, T and thus the packet delay bound can be reduced. But N_S must be no less than N. Otherwise, the N_S configurations are not sufficient to cover every entry in $\boldsymbol{C(T)}$ [10-13]. Accordingly, scheduling algorithms that only require $N_S=N$ configurations are called *minimum delay scheduling algorithms*. When $N_S=N$, from (1) T can be made as close to its lower bound δN as possible by minimizing $\sum^{N}_{n=1} \phi_n / S$. Generally, we hope to minimize S for a given packet delay (or equivalently a given T). In minimum delay scheduling, this translates to minimizing S_{schedule} according to (1)～(4).

Recently, several minimum delay scheduling algorithms (MIN [10], α^i-SCALE [12] and QLEF [13]) have been proposed to minimize S_{schedule}. Among them, QLEF (Quasi Largest-Entry-First) requires the lowest speedup bound. In this paper, we derive a new speedup bound for QLEF, which is 10% lower than the one in [13]. Because the same scheduling problem is also involved in other communication networks, such as SS/TDMA [14], TWIN [15] and wireless sensor networks [16], our result also contributes to those systems.

The rest of the paper is organized as follows. In Section II, we review the QLEF algorithm [13]. The new speedup bound is derived in Section III. We conclude the paper in Section IV.

QLEF ALGORITHM

Input:

An $N \times N$ matrix $\boldsymbol{C(T)}$ with maximum line sum not more than T.

Output:

N configurations $\boldsymbol{P_1}, \ldots, \boldsymbol{P_N}$ and weights $\phi_1, \ldots, \phi_N$.

Step 1: Initialization:

Set $0 \rightarrow n$. Initialize $\boldsymbol{P_1}, \ldots, \boldsymbol{P_N}$ to all-zero matrices and the $N \times N$ reference matrix $\boldsymbol{R}=\{r_{ij}\}$ to all 1's.

Step 2: Determine the first "half" configurations $\boldsymbol{P_{n+1}}$:

a) Un-shadow $\boldsymbol{C(T)}$ and $\boldsymbol{R}$. Set $1 \rightarrow w$.

b) Select the largest entry c_{ij} in the not-yet-shadowed part of $\boldsymbol{C(T)}$. If $w=1$, set $\boldsymbol{P_{n+1}}$'s weight $\phi_{n+1}=c_{ij}$ and $w=0$. Shadow the corresponding lines in both $\boldsymbol{C(T)}$ and $\boldsymbol{R}$, and set c_{ij} and r_{ij} to 0. Set $1 \rightarrow p^{(n+1)}_{ij}$ where $p^{(n+1)}_{ij}$ is the entry (i, j) of $\boldsymbol{P_{n+1}}$. Repeat this step until $N-(2n+1)$ largest entries are selected.

c) Construct a bipartite graph $\boldsymbol{U_G}$ from the remaining not-yet-shadowed part of $\boldsymbol{R}$ and perform maximum-size matching in $\boldsymbol{U_G}$ to get $(2n+1)$ edges. Record the corresponding entries to $\boldsymbol{P_{n+1}}$ by setting $1 \rightarrow p^{(n+1)}_{ij}$. Set these entries of $\boldsymbol{C(T)}$ and $\boldsymbol{R}$ to 0's. Then set $n+1 \rightarrow n$.

d) Repeat *Step 2a)-2c)* until $n=\lceil N/2 \rceil - 1$.

Step 3: Determine the second "half" configurations:

a) Un-shadow $\boldsymbol{C(T)}$ and $\boldsymbol{R}$. Find the largest entry c_{ij} in $\boldsymbol{C(T)}$ and set c_{ij} as the weight for all the subsequent configurations.

b) Find a maximum-size matching in the bipartite graph of $\boldsymbol{R}$ and set the corresponding entries of $\boldsymbol{P_{n+1}}$ to 1. Set these entries to 0 in $\boldsymbol{C(T)}$ and $\boldsymbol{R}$, and then set $n+1 \rightarrow n$. Repeat this step until $n=N$.

Fig. 3. QLEF algorithm.

II. QLEF ALGORITHM

QLEF algorithm [13] is summarized in Fig. 3. It generates N non-overlapping configurations (i.e. the entries covered by any two configurations do not overlap) to cover $\boldsymbol{C(T)}$. It has a time complexity of $O(N^{3.5})$, and the correctness proof can be found in [13]. When the values of T, N and δ are given, S_{schedule} determines the overall speedup S in minimum delay scheduling. Therefore, QLEF aims at minimizing S_{schedule}.

The key idea of QLEF is to cover large entries in $\boldsymbol{C(T)}$ first. A reference matrix $\boldsymbol{R}$ is initialized to all 1's, where a "1" indicates that the corresponding entry of $\boldsymbol{C(T)}$ is not yet covered. Assume that configuration $\boldsymbol{P_{n+1}}$ is being constructed. To avoid configuration overlaps, QLEF only selects $N-(2n+1)$ largest entries from $\boldsymbol{C(T)}$ in its Step 2b, which are called *selected-entries*. The corresponding lines of the selected-entries are *shadowed* in both $\boldsymbol{C(T)}$ and $\boldsymbol{R}$ (see Fig. 4a). Then, QLEF applies maximum-size matching (MSM) [17] to the remaining not-yet-shadowed part of $\boldsymbol{R}$ to get $(2n+1)$ entries in Step 2c, which are called *MSM-entries*. The $N-(2n+1)$ selected-entries combine with the $(2n+1)$ MSM-entries to form $\boldsymbol{P_{n+1}}$. Accordingly, those entries covered by $\boldsymbol{P_{n+1}}$ are set to 0 in both $\boldsymbol{C(T)}$ and $\boldsymbol{R}$. Then both $\boldsymbol{C(T)}$ and $\boldsymbol{R}$ are un-shadowed and QLEF repeats the above process to construct the next configuration.

To ensure $N-(2n+1)>0$, only the first $\lceil N/2 \rceil - 1$ configurations are constructed according to the above mechanism (in Step 2). Then in Step 3, the remaining $N - \lceil N/2 \rceil + 1$ configurations are determined by maximum-size matching [17].

III. SPEEDUP BOUND

In QLEF, the N configurations are sequentially constructed from $\boldsymbol{P_1}$ to $\boldsymbol{P_N}$. We now focus on the first half of them $\boldsymbol{P_1}, \ldots, \boldsymbol{P_n}, \boldsymbol{P_{n+1}}, \ldots, \boldsymbol{P}_{\lceil N/2 \rceil - 1}$. Assume that an entry $c_{ij} \in \boldsymbol{C(T)}$ is covered

1-4244-0357-X/06/$20.00 ©2006 IEEE

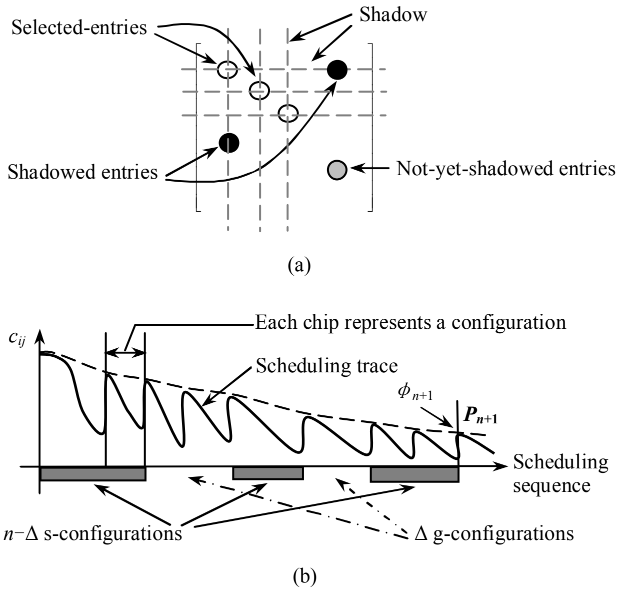


Fig. 4. Conceptual QLEF scheduling procedure.

by $\boldsymbol{P}_{n+1}$. If c_{ij} is shadowed (see Fig. 4a) in the construction of $\boldsymbol{P}_k$ ($n\geq k\geq 1$), then $\boldsymbol{P}_k$ is called an *s-configuration* of c_{ij}. Otherwise, $\boldsymbol{P}_k$ is called a *g-configuration* of c_{ij}.

Fig. 4b shows the *conceptual* QLEF scheduling procedure. We use a "scheduling trace" to represent the trend of c_{ij} values covered. It is usually a *wave* rather than a monotonically decreasing curve, although QLEF always selects the largest entry in the not-yet-shadowed part. Due to the shadowing operation, a large entry may be *shadowed* by an s-configuration and thus can only be covered *later* by another configuration.

For a particular configuration $\boldsymbol{P}_{n+1}$, its weight ϕ_{n+1} also appears as an entry in $\boldsymbol{C(T)}$ and is the *first* selected-entry in $\boldsymbol{P}_{n+1}$. Therefore, in the following discussion, we treat ϕ_{n+1} as an entry in $\boldsymbol{C(T)}$ rather than a weight. Among the n configurations constructed before $\boldsymbol{P}_{n+1}$ (i.e. $\boldsymbol{P}_1 \sim \boldsymbol{P}_n$), we assume that Δ of them are g-configurations of ϕ_{n+1} and the other $n-\Delta$ configurations are its s-configurations, as in Fig. 4b.

A. General Idea

Define the entries larger than or equal to entry ϕ_{n+1} in the *original* $\boldsymbol{C(T)}$ as *larger-entries* (or LEs). To bound the value of ϕ_{n+1}, we count the *minimum* total number of LEs (denoted by M) that are covered by $\boldsymbol{P}_1 \sim \boldsymbol{P}_n$. These M LEs reside in N lines (rows or columns) of $\boldsymbol{C(T)}$, and the line that contains the maximum number of LEs must contain at least the average number of $\lceil M/N \rceil$ LEs. As a result, the smallest LE in this line must be smaller than or equal to the $\lceil M/N \rceil$-th largest entry of this line. Yet it is not smaller than ϕ_{n+1}. Because the maximum line sum of $\boldsymbol{C(T)}$ is T, according to Lemma 1 in [13], we have

$$\phi_{n+1} \leq \frac{T}{\left\lceil \dfrac{M}{N} \right\rceil}. \tag{5}$$

On the other hand, because ϕ_{n+1} is shadowed by $n-\Delta$ s-configurations, from Lemma 2 in [13], we have

$$\phi_{n+1} \leq \frac{T}{\left\lceil \dfrac{n-\Delta}{2} \right\rceil + 1} = \frac{T}{\left\lceil \dfrac{n-\Delta}{2} + 1 \right\rceil}. \tag{6}$$

Combining (5) and (6), we can bound ϕ_{n+1} as follows:

$$\phi_{n+1} \leq \max_{0\leq\Delta\leq n}\left\{ \min_{0\leq\Delta\leq n}\left[\left\lceil \frac{T}{\left\lceil \dfrac{M}{N} \right\rceil} \ ,\ \frac{T}{\left\lceil \dfrac{n-\Delta}{2}+1 \right\rceil} \right\rceil\right]\right\}, \text{ for } 0\leq n<\left\lceil \frac{N}{2} \right\rceil -1. \tag{7}$$

Formula (7) indicates that no matter what is the value of Δ ($0\leq\Delta\leq n$), the bound always holds because we have taken the worst case into consideration (i.e. the max function for Δ).

For the remaining $N-\lceil N/2 \rceil+1$ configurations constructed in Step 3 in Fig. 3, QLEF uses a small constant as the weights. According to QLEF, this constant is not larger than any weight of the first $\lceil N/2 \rceil -1$ configurations (since the weights are monotonically decreasing as shown by the dashed line in Fig. 4b). Therefore, it can be bounded by the weight of the last one among the first $\lceil N/2 \rceil -1$ configurations. That is

$$\phi_{n+1}\Big|_{N\geq n+1 \geq \left\lceil \frac{N}{2} \right\rceil} \leq \phi_{\left\lceil \frac{N}{2} \right\rceil -1}. \tag{8}$$

After the N weights ϕ_{n+1} are bounded by (7) and (8), we can calculate S_{schedule} bound in (4). Note that the key is to count the *minimum* total number of LEs (i.e. M in (7)).

B. Speedup Bound Formulation

For entry ϕ_{n+1}, we consider its Δ g-configurations and the other $n-\Delta$ s-configurations (see Fig. 4b). In QLEF, all the selected-entries in the g-configurations are LEs. On the other hand, each s-configuration must cover one LE in the same line as ϕ_{n+1}. However, in addition to this LE, the s-configuration may also cover other LEs in different lines. Assume that a set of consecutive s-configurations $\{\boldsymbol{P}_x\}$ shadow ϕ_{n+1}, and $\boldsymbol{P}_y$ is the first g-configuration after $\{\boldsymbol{P}_x\}$. From Lemma 1 in the Appendix, the number of LEs covered by each $\boldsymbol{P}_x \in \{\boldsymbol{P}_x\}$ is at least half of the number of the selected-entries in $\boldsymbol{P}_y$.

In Fig. 4b, the Δ g-configurations and the $n-\Delta$ s-configurations may queue in any order. From Lemma 2 in the Appendix, in order to minimize M, the $n-\Delta$ s-configurations should *consecutively* locate at either the very beginning or the very end of the configuration sequence $\boldsymbol{P}_1 \sim \boldsymbol{P}_n$.

Case 1: The $n-\Delta$ s-configurations consecutively locate at the very end of the n configurations. In this case, all the selected-entries covered by the Δ g-configurations are LEs, but the number of LEs covered by the $n-\Delta$ s-configurations is trivial and is ignored when counting M. So, $M=(N-1)+(N-3)+\ldots+(N-2\Delta+1)=(N-\Delta)\Delta$. Note that none of the g-configurations shadows ϕ_{n+1}. So, these LEs reside in $N-1$ lines of $\boldsymbol{C(T)}$ instead of N lines. Replacing N in (7) by $N-1$, we have

$$\phi_{n+1} \leq \max_{0\leq\Delta\leq n}\left\{ \min_{0\leq\Delta\leq n}\left[\left\lceil \frac{T}{\left\lceil \dfrac{(N-\Delta)\Delta}{N-1} \right\rceil} \ ,\ \frac{T}{\left\lceil \dfrac{n-\Delta}{2}+1 \right\rceil} \right\rceil\right]\right\}. \tag{9}$$

Mathematically, this is equivalent to (10) below.

$$\phi_{n+1}\Big|_{0\le n<\left\lceil \frac{N}{2}\right\rceil-1} \le \max\left\{ \frac{T}{\left\lceil \frac{n-\Delta}{2}+1\right\rceil}\Bigg|_{\Delta=\frac{(3N-1)-\sqrt{(3N-1)^2-8(N-1)(n+2)}}{4}} \quad , \quad \frac{T}{\left\lceil \frac{n-\Delta}{2}+1\right\rceil}\Bigg|_{\Delta=\frac{2n^2+n+2N}{2N+1}} \right\} \quad , \quad \phi_{n+1}\Big|_{N\ge n+1\ge\left\lceil \frac{N}{2}\right\rceil} \le \phi_{\left\lceil \frac{N}{2}\right\rceil-1} \quad . \tag{13}$$

$$\phi_{n+1} \le \frac{T}{\left\lceil \frac{n-\Delta}{2}+1\right\rceil}\Bigg|_{0\le n<\left\lceil \frac{N}{2}\right\rceil-1 \text{ and } \Delta=\frac{(3N-1)-\sqrt{(3N-1)^2-8(N-1)(n+2)}}{4}} . \tag{10}$$

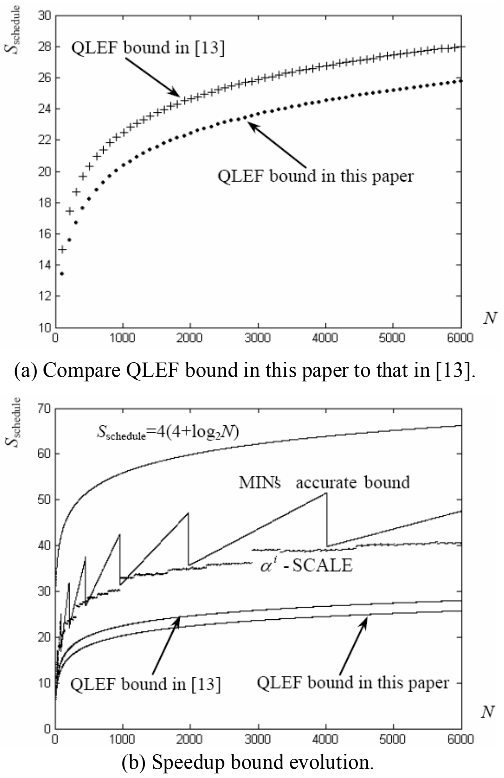
Case 2: The $n-\Delta$ s-configurations consecutively locate at the beginning of the configuration sequence $\boldsymbol{P}_1\sim\boldsymbol{P}_n$. In this case, each s-configuration covers at least $(N-1)/2-(n-\Delta)$ LEs according to Lemma 1 in the Appendix. Taking the LEs covered by the Δ g-configurations into account, we get $M=Nn-n^2-(N+1)(n-\Delta)/2$ by simple calculation. From (7) we have

$$\phi_{n+1} \le \max_{0\le\Delta\le n}\left\{ \min_{0\le\Delta\le n}\left[\frac{T}{\left\lceil \frac{2Nn-2n^2-(N+1)(n-\Delta)}{2N}\right\rceil} \quad , \quad \frac{T}{\left\lceil \frac{n-\Delta}{2}+1\right\rceil} \right]\right\}. \tag{11}$$

Again, this is equivalent to

$$\phi_{n+1} \le \frac{T}{\left\lceil \frac{n-\Delta}{2}+1\right\rceil}\Bigg|_{0\le n<\left\lceil \frac{N}{2}\right\rceil-1 \text{ and } \Delta=\frac{2n^2+n+2N}{2N+1}} . \tag{12}$$

Combining (10), (12) and (8), we get the bound for ϕ_{n+1} as shown in formula (13). We can replace ϕ_n in (4) by the bound in (13) to calculate the S_{schedule} bound.



(a) Compare QLEF bound in this paper to that in [13].

(b) Speedup bound evolution.

Fig. 5. Speedup bounds for the minimum delay scheduling problem.

C. *Results*

Fig. 5 shows the new speedup bound we derived for QLEF. As an example, the new S_{schedule} bound gives a gain of 11.29% over the existing bound in [13] for N=200. Fig. 5b shows the S_{schedule} bound evolution for the minimum delay scheduling problem. Particularly, the bound $S_{\text{schedule}}=4(4+\log_2 N)$ is given in [10] for MIN, which is refined in [12] to produce the saw-toothed curve. A tighter bound is then provided by α^i-SCALE [12]. This is then followed by QLEF in [13]. In this paper we further push the QLEF speedup bound to be 10% lower.

IV. Conclusion

Hybrid electronic/optical packet switch provides a scalable switch architecture for huge-capacity backbone Internet routers. Because of the reconfiguration overhead of the optical switch fabric, packet delay is minimized by using N configurations (where N is the switch size) to schedule the traffic. However, this requires a very large speedup to achieve performance guaranteed switching. QLEF (Quasi Largest-Entry-First) is the most efficient minimum delay scheduling algorithm that gives the lowest speedup bound for a given packet delay. In this paper, we derived a new speedup bound for QLEF, which is 10% lower than the existing bound.

Appendix

Lemma 1: Assume that $\{\boldsymbol{P}_x\}$ is a set of consecutive s-configurations of ϕ_{n+1}, and $\boldsymbol{P}_y$ is the first g-configuration of ϕ_{n+1} after $\{\boldsymbol{P}_x\}$. Then, any $\boldsymbol{P}_x\in\{\boldsymbol{P}_x\}$ must cover at least h LEs, where h is half of the number of the selected-entries in $\boldsymbol{P}_y$.

Proof: Since $\boldsymbol{P}_y$ is a g-configuration of ϕ_{n+1}, any selected-entry α covered by $\boldsymbol{P}_y$ is an LE. Because $\boldsymbol{P}_x$ is constructed before $\boldsymbol{P}_y$ and α is not covered in $\boldsymbol{P}_x$, either 1) all the selected-entries covered by $\boldsymbol{P}_x$ are not smaller than α, or 2) α is shadowed in $\boldsymbol{P}_x$ construction.

In case 1), all the selected-entries in $\boldsymbol{P}_x$ are LEs. Since $\boldsymbol{P}_x$ is constructed earlier than $\boldsymbol{P}_y$, it contains more selected-entries than $\boldsymbol{P}_y$. So, the number of LEs covered by $\boldsymbol{P}_x$ is larger than h. In case 2), any α covered by $\boldsymbol{P}_y$ must have been shadowed in $\boldsymbol{P}_x$ construction. Since a selected-entry in $\boldsymbol{P}_x$ can shadow at most two smaller/equal selected-entries in $\boldsymbol{P}_y$ (in row and column, respectively), $\boldsymbol{P}_x$ must cover at least h LEs. Obviously, this is true for the first g-configuration $\boldsymbol{P}_y$ after $\{\boldsymbol{P}_x\}$.

Lemma 2: To minimize M, all the s-configurations of ϕ_{n+1} should be consecutively located at either the very beginning or the very end of the configuration sequence $\boldsymbol{P}_1\sim\boldsymbol{P}_n$.

Proof: In Fig. 6, let y-axis denote the number of selected-entries covered in each configuration, and x-axis denote the scheduling sequence. Without loss of generality, assume there are three sets of consecutive s-configurations of ϕ_{n+1}, denoted by $\boldsymbol{A}_1$, $\boldsymbol{A}_2$ and $\boldsymbol{A}_3$ (others are g-configurations). Particularly, $\boldsymbol{A}_1$ and $\boldsymbol{A}_2$ contain x_1 and x_2 s-configurations respectively, and $\boldsymbol{A}_3$ locates at the very end of the configuration sequence $\boldsymbol{P}_1\sim\boldsymbol{P}_n$.

1-4244-0357-X/06/$20.00 ©2006 IEEE

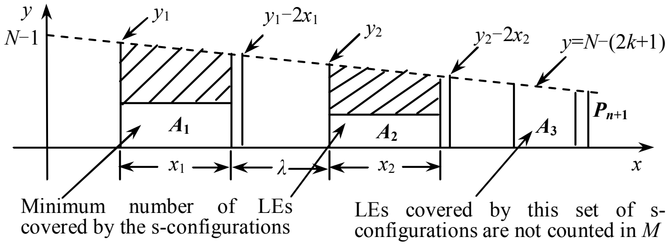

y
N−1
y1
y1−2x1
y2
y2−2x2
y=N−(2k+1)
A1
A2
A3
Pn+1
x1
λ
x2
x
Minimum number of LEs covered by the s-configurations
LEs covered by this set of s-configurations are not counted in M

Fig. 6. s-configurations also cover a considerable number of LEs.

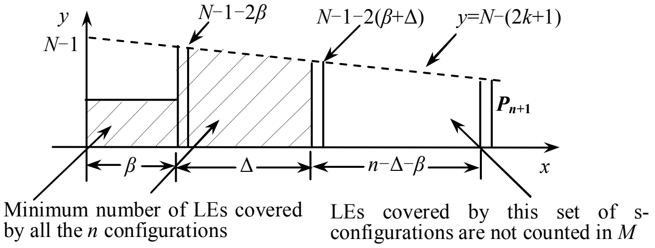

y
N−1
N−1−2β
N−1−2(β+Δ)
y=N−(2k+1)
Pn+1
β
Δ
n−Δ−β
x
Minimum number of LEs covered by all the n configurations
LEs covered by this set of s-configurations are not counted in M

Fig. 7. The $n-\Delta$ s-configurations locate at the both ends of $\boldsymbol{P_1}\sim\boldsymbol{P_n}$

The first s-configuration in $\boldsymbol{A_1}$ covers y_1 selected-entries, and the first g-configuration after $\boldsymbol{A_1}$ covers (y_1-2x_1) selected-entries. Similarly, the first s-configuration in $\boldsymbol{A_2}$ covers y_2 selected-entries, and the first g-configuration after $\boldsymbol{A_2}$ covers (y_2-2x_2) selected-entries. From Lemma 1, each s-configuration in $\boldsymbol{A_1}$, $\boldsymbol{A_2}$ covers at least $(y_1-2x_1)/2$, $(y_2-2x_2)/2$ LEs. We do not count any LEs in $\boldsymbol{A_3}$ because there is no g-configuration after it. Although each s-configuration in $\boldsymbol{A_3}$ covers one LE in the same line as Φ_{n+1}, it is trivial and is ignored when counting M.

We first consider $\boldsymbol{A_1}$ and $\boldsymbol{A_2}$. Since all selected-entries covered by g-configurations are LEs, minimizing M is equivalent to maximizing the number of selected-entries in the shadowed areas in $\boldsymbol{A_1}$ and $\boldsymbol{A_2}$. That is

$$\max\left\{\left[x_1\left(\frac{y_1-2x_1}{2}+2\right)+\frac{x_1(x_1-1)}{2}\times 2\right]+\left[x_2\left(\frac{y_2-2x_2}{2}+2\right)+\frac{x_2(x_2-1)}{2}\times 2\right]\right\}$$

$$\text{or}\quad \max\left\{\frac{(y_1+2)x_1+(y_2+2)x_2}{2}\right\}.$$

Let λ be the number of g-configurations between $\boldsymbol{A_1}$ and $\boldsymbol{A_2}$. We have $y_2=y_1-2(x_1+\lambda)$. So, the above formula is equivalent to

$$\max\left\{\frac{(y_1+2)(x_1+x_2)-2x_2(x_1+\lambda)}{2}\right\}.$$

To maximize the above formula for any given x_1 and x_2, it is necessary that y_1 takes the maximum possible value and $\lambda=0$. This entails that $\boldsymbol{A_1}$ and $\boldsymbol{A_2}$ should be located consecutively at the very beginning of the configuration sequence $\boldsymbol{P_1}\sim\boldsymbol{P_n}$. It is easy to see that this point can be generalized to the case where more sets of consecutive s-configurations are involved.

However, we still need to consider $\boldsymbol{A_3}$ in Fig. 6. In fact, the n configurations before $\boldsymbol{P_{n+1}}$ may also locate as shown in Fig. 7, where the Δ g-configurations are in the middle and the $n-\Delta$ s-configurations are at the both ends (the β s-configurations locate consecutively at the beginning of $\boldsymbol{P_1}\sim\boldsymbol{P_n}$ as argued above). In Fig. 7, minimizing M is equivalent to maximizing the number of selected-entries in the not-shadowed areas, i.e.

$$\max_{0\le\beta\le n-\Delta}\left\{\left[\left(\frac{N-1-2\beta}{2}+2\right)\beta+\frac{\beta(\beta-1)}{2}\times 2\right]+\right.$$
$$\left.\left[(n-\Delta-\beta)(N-2n+1)+\frac{(n-\Delta-\beta)(n-\Delta-\beta-1)}{2}\times 2\right]\right\}$$

$$\text{or}\quad \max_{0\le\beta\le n-\Delta}\left\{\beta^2+\frac{4\Delta-N+1}{2}\beta+(n-\Delta)(N-n-\Delta)\right\}.$$

Because the above formula is a quadratic function of β, it can be maximized only if $\beta=0$ or $\beta=n-\Delta$. From Fig. 7, obviously all the $n-\Delta$ s-configurations should be consecutively located at either the very beginning ($\beta=n-\Delta$) or the very end ($\beta=0$) of the configuration sequence $\boldsymbol{P_1}\sim\boldsymbol{P_n}$. The specific location is determined by the values of N, n and Δ.

References

[1] H. J. Chao, "Next generation routers", *Proceedings of the IEEE*, vol. 90, issue 9, pp. 1518-1558, Sept. 2002.

[2] N. Mckeown, "Do optics belong in internet core routers?" *keynote, opticomm 2001*, Denver, Colorado, http://tiny-tera.stanford.edu/~nickm/talks/opticomm_2001.ppt.

[3] K. Isaac et. al, "Scaling internet routers using optics", *Computer Communication Review*, vol. 33, no. 4, pp. 189-200, Oct. 2003.

[4] M. N. Islam, "100 Tb/s aggregate capacity router using an optical switching core", http://www.eecs.umich.edu/OSL/Islam/100Tbs-Router.pdf.

[5] K. Kar, D. Stiliadis, T. V. Lakshman, and L. Tassiulas, "Scheduling algorithms for optical packet fabrics", *IEEE Journal on Selected Areas in Communications*, vol. 21, issue 7, pp. 1143-1155, Sept. 2003.

[6] J.E Fouquet et. al, "A compact, scalable cross-connect switch using total internal reflection due to thermally-generated bubbles", *IEEE LEOS Annual Meeting*, pp. 169-170, Dec. 1998.

[7] L. Y. Lin, "Micromachined free-space matrix switches with submilli-second switching time for large-scale optical crossconnect", *OFC'98 Tech. Digest*, pp. 147-148, Feb. 1998.

[8] O. B. Spahn, C. Sullivan, J. Burkhart, C. Tigges, and E. Garcia "GaAs-based microelectromechanical waveguide switch", *Proc. 2000 IEEE/LEOS Intl. Conf. on Optical MEMS*, pp. 41-42, Aug. 2000.

[9] A. J. Agranat, "Electroholographic wavelength selective crossconnect", *1999 Digest of the LEOS Summer Topical Meetings*, pp. 61-62, Jul. 1999.

[10] B. Towles and W. J. Dally, "Guaranteed scheduling for switches with configuration overhead", *IEEE/ACM Trans. Networking*, vol. 11, no. 5, pp. 835-847, Oct. 2003.

[11] Bin Wu and Kwan L. Yeung, "Minimizing internal speedup for performance guaranteed optical packet switches", *IEEE GLOBECOM '04*, vol. 3, pp. 1742-1746, 29 Nov.-3 Dec. 2004.

[12] Bin Wu and Kwan L. Yeung, "Scheduling optical packet switches with minimum number of configurations", *IEEE ICC '05*, vol. 3, pp. 1830-1835, May 2005.

[13] Bin Wu and Kwan L. Yeung, "Traffic scheduling in non-blocking optical packet switches with minimum delay", *IEEE GLOBECOM '05*, vol. 4, pp. 2041-2045, Dec. 2005.

[14] Y. Ito, Y. Urano, T. Muratani, and M. Yamaguchi, "Analysis of a switch matrix for an SS/TDMA system", *Proc. of the IEEE*, vol. 65, no. 3, pp. 411-419, 1977.

[15] K. Ross, N. Bambos, K. Kumaran, I. Saniee and I. Widjaja, "Scheduling bursts in time-domain wavelength interleaved networks", *IEEE Journal on Selected Areas in Communications*, vol. 21, issue 9, pp. 1441-1451, Nov. 2003.

[16] Hai Liu, P. Wan, C.-W. Yi, Xiaohua Jia, S. Makki and N. Pissinou, "Maximal lifetime scheduling in sensor surveillance networks", *IEEE INFOCOM '05*, vol. 4, pp. 2482-2491, Mar. 2005.

[17] J. E. Hopcroft and R. M. Karp, "An $n^{5/2}$ algorithm for maximum matching in bipartite graphs", *Soc. Ind. Appl. Math. J. Comput.*, vol. 2, pp. 225-231, 1973.