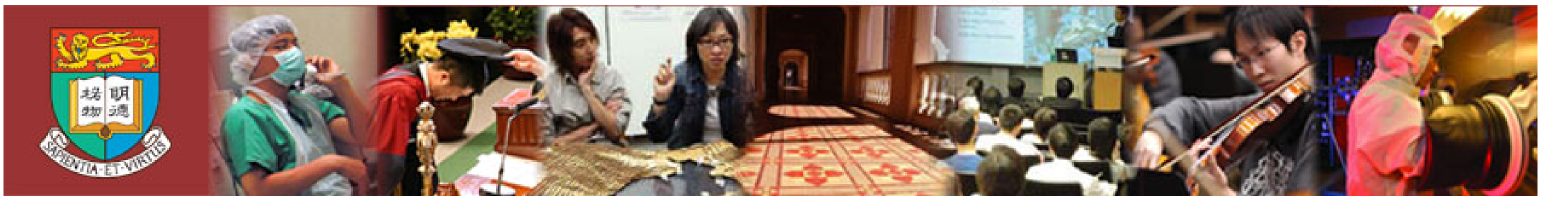

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE GLOBECOM 2005 proceedings.

Traffic Scheduling in Non-Blocking Optical Packet Switches with Minimum Delay

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

***Abstract*—For performance guaranteed OPS switches with reconfiguration overhead, it has been shown that packet delay can be minimized by using N switch configurations (where N is the switch size) to schedule the traffic. However, this usually involves an exorbitant speedup requirement, which makes it impractical under current technology. In this paper, a new minimum-delay scheduling algorithm QLEF (Quasi Largest-Entry-First) is proposed. We prove that QLEF pushes the required speedup bound to the lowest known level. As an example, when N=950, QLEF only requires a speedup of S_{schedule}=21.33 instead of 42.25 for MIN [5] and 30.27 for α^i-SCALE [8]. This gives a 50% improvement over MIN and 30% over α^i-SCALE.**

Keywords-Optical packet switch (OPS); reconfiguration overhead; performance guaranteed scheduling; speedup.

I. Introduction

The explosion of Internet traffic and the rapid progress of optical technology have led to the concept of *optical Internet*, combining IP's flexibility with high efficiency of optical transmission. As a result, optical packet switches (OPS) based on various optical switching technologies [1-4] are developed to meet the ever-increasing demands for larger bandwidth and higher switch port counts.

Despite of many advantages such as scalability, high line rate, huge capacity and low power consumption, an OPS switch usually needs relatively long time to change its cross-connection phase. During this time period (which is called *reconfiguration overhead*), no packet can be transmitted across the switch. This reconfiguration overhead can range from 10ns to several milliseconds [1-4]. In order to achieve *performance guaranteed switching* (i.e. non-blocking with bounded delay), OPS switch fabric needs to transmit packets at an internal rate higher than the external line rate. This *speedup* is to compensate for the reconfiguration overhead and the scheduling inefficiency [5-8] (also refer to Section II).

OPS switch architectures similar to that in Fig. 1 are considered in [5-8]. Particularly, a scalable multi-rack scenario is discussed in [6], where the VOQ/OQ (virtual output queuing/ output queuing) modules in Fig.1 can be regarded as separate line cards locating at different racks. They are connected to the central OPS crossbar switch by optical fibers. An internal speedup S is deployed in the switch to achieve performance guaranteed switching. Compared with traditional cable connections, this setup eliminates the possible electromagnetic

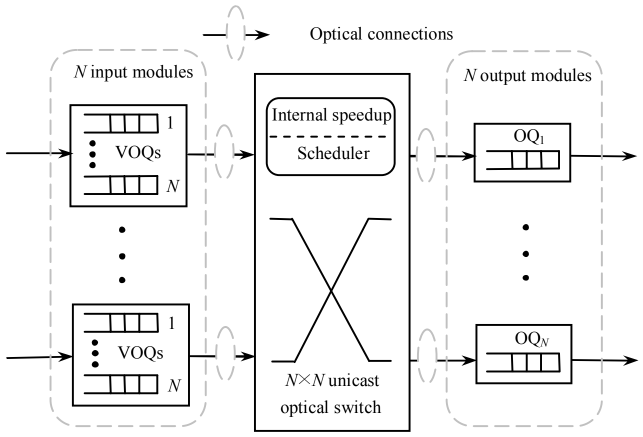

Fig. 1. A scalable high speed optical packet switch.

interference, removes the need of O/E/O conversions at the central switch fabric, and provides extra scalability with lower power consumption in each rack.

Assume that each reconfiguration at the central OPS switch consumes δ time slots (overhead). We hope to use the minimum number of configurations to schedule the traffic in order to minimize the packet delay. For performance guaranteed switching, N (where N is the switch size) is the minimum possible number of configurations required. This is because an $N \times N$ traffic matrix has N^2 entries, and each configuration covers at most N of them [5]. Unfortunately, this minimum-delay scheduling introduces many empty slots in the schedule, which must be overcome by a high speedup [5, 8].

Among all the algorithms proposed in [5-8], MIN [5] and α^i-SCALE [8] can schedule traffic with the minimum number of N configurations. These two algorithms follow the same approach. That is, they both use a scale function to analyze the traffic matrix and calculate only the first $N/4$ configurations to cover the large entries, while using a small constant weight for the other $3N/4$ configurations [5, 8]. Although α^i-SCALE generally outperforms MIN, the speedup bound given by α^i-SCALE may still be too high.

In this paper, we take a totally different approach and a novel QLEF (Quasi Largest-Entry-First) algorithm is proposed to schedule OPS traffic with only N configurations. It has a better design philosophy than MIN and α^i-SCALE, and greatly pushes the required speedup bound to the lowest known level.

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

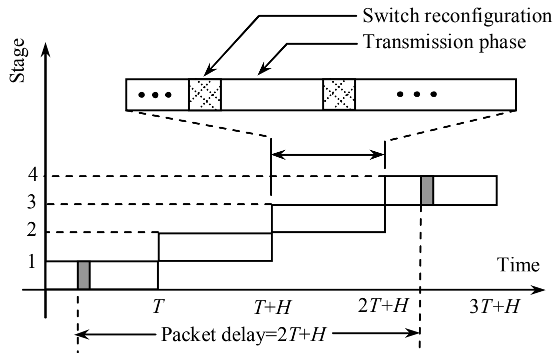

Fig. 2. Optical packet switch scheduling stages.

II. Scheduling Stages

Fig. 2 shows the general OPS scheduling procedure in four stages. In Stage 1, incoming packets are periodically accumulated in the input buffers over T time slots to construct an $N \times N$ traffic matrix $\boldsymbol{C(T)}=\{c_{ij}\}$. Each entry c_{ij} denotes the number of packets received at input i and destined to output j. The scheduling algorithm takes H time slots in Stage 2 to generate N configurations $\boldsymbol{P_n}=\{p^{(n)}_{ij}\}$, $n\in\{1, \ldots, N\}$ with corresponding weights ϕ_n to cover $\boldsymbol{C(T)}$, where "cover" means that $\sum^{N}_{n=1} \phi_n\, p^{(n)}_{ij} \geq c_{ij}$ for any $i, j\in\{1, \ldots, N\}$. $\boldsymbol{P_n}$ is an $N \times N$ permutation matrix with at most a single "1" in each line (row or column). $p^{(n)}_{ij}=1$ indicates that a packet can be sent from input i to output j in one slot; $p^{(n)}_{ij} = 0$ otherwise. $\boldsymbol{P_n}$ is called a *perfect matching* if it has exactly N "1" entries. In Stage 3, the switch fabric is reconfigured according to the N configurations. An internal speedup S is applied to ensure that this stage occupies only T slots. After the speedup is applied, the unit slot time *in the transmission phase of Stage 3* is compressed/ shortened, and the switch fabric holds $\boldsymbol{P_n}$ for ϕ_n *compressed slots* for packet transmission. Finally in Stage 4 packets are sent onto the output lines from output buffers (in T slots).

From the tagged packet in Fig. 2, we can see that the bounded delay of any packet is $2T+H$ slots. Assume each switch reconfiguration takes δ slots and $T>\delta N$. Since δN slots are used to reconfigure the switch for N times in Stage 3, only $T-\delta N$ slots are left for transmitting $\boldsymbol{C(T)}$. Assume that each of the line sums of $\boldsymbol{C(T)}$ is not larger than T. We only consider such *admissible* traffic in this paper. Under this assumption, there are at most T packets waiting at each input port for transmission. Therefore, a speedup factor denoted by $S_{\text{reconfigure}}= T/(T-\delta N)$ is necessary to compensate solely for the idle time caused by reconfigurations. At the same time, the scheduling algorithm may produce many empty slots (i.e. underutilize the bandwidth provided by the configurations [5-8]). As a result, more than T compressed slots are necessary in Stage 3 to transmit all the packets. Therefore another speedup factor

$$S_{\text{schedule}} = \frac{1}{T}\sum_{n=1}^{N}\phi_n \qquad (1)$$

is required to compensate solely for the inefficient scheduling. The overall internal speedup S is then given by

$$S = S_{\text{reconfigure}} \times S_{\text{schedule}} = \frac{T}{T-\delta N} S_{\text{schedule}} = \frac{1}{T-\delta N}\sum_{n=1}^{N}\phi_n \,. \qquad (2)$$

Since the values of T, N and δ are predefined, the overall internal speedup S is dominated by S_{schedule}. Thus, our objective is to minimize the sum of all the N weights $\sum^{N}_{n=1}\phi_n$.

III. QLEF Algorithm

A. *LEF (Largest-Entry-First) Procedure*

Given a traffic matrix $\boldsymbol{C(T)}$, we want to cover it by finding a schedule that consists of N configurations $\boldsymbol{P_n}$, $n\in\{1, \ldots, N\}$ with each weighted by ϕ_n. In order to minimize the sum of ϕ_n, intuitively we hope that large entries in $\boldsymbol{C(T)}$ can be scheduled/ covered in the same configuration, so that they can share the same large weight. This also potentially lets other yet-to-be-constructed configurations require smaller weights. In other words, we should always schedule the "largest" entry first.

Be more specific, when an entry in $\boldsymbol{C(T)}$ is selected for scheduling, the corresponding entry in $\boldsymbol{P_n}$ is marked by "1" and this entry is then set to 0 in $\boldsymbol{C(T)}$. For unicast switches, only one entry can be scheduled in each line of each $\boldsymbol{P_n}$. So, we need to "shadow" the corresponding lines of $\boldsymbol{C(T)}$ before selecting the next largest entry in the remaining not-yet-shadowed part. This operation of "shadowing" avoids selecting another entry in the same line for the same configuration. As an example, entry c_{11}=10 of $\boldsymbol{C(T)}$ in Fig. 3 is selected. The corresponding lines, the first row and the first column, are shadowed before c_{11} is set to 0 and c_{22}=9 is selected. We continue this process until no more entries in $\boldsymbol{C(T)}$ can be further scheduled in $\boldsymbol{P_n}$ (i.e. the N entries are selected and all the lines of $\boldsymbol{C(T)}$ are shadowed). At this stage, we un-shadow the whole $\boldsymbol{C(T)}$ (i.e remove all the shadows) and continue to construct the next configuration $\boldsymbol{P_{n+1}}$ in the same way. We call this procedure of constructing configurations as LEF (*Largest-Entry-First*).

Fig. 3 illustrates two possible schedules of a 3×3 $\boldsymbol{C(T)}$. The first one is obtained using the LEF procedure. Entries "10", "9" and "8" are covered in $\boldsymbol{P_1}$ with a weight of 10. The remaining entries are covered by $\boldsymbol{P_2}$ and $\boldsymbol{P_3}$ with (small) weights 2 and 1 respectively. The sum of the weights is 13. The second schedule is generated by some non-LEF procedure. Entries "10", "2" and "1" are covered in $\boldsymbol{P_1}$. As a result, the not-yet-covered large entries "9" and "8" may become the weights for $\boldsymbol{P_2}$ and $\boldsymbol{P_3}$. This gives a very large total weight of 27.

Unfortunately, unlike MIN [5] and α^i-SCALE [8], the above LEF procedure *cannot* guarantee that N configurations are always enough to cover all the N^2 entries of a traffic matrix. This is because LEF cannot prevent *configuration overlap*. Fig. 4 shows such a counterexample. The resulting schedule consists of four configurations instead of the minimum three. The entries that are covered more than once (i.e. configuration overlap) are also illustrated.

Shadow

$$\boldsymbol{C(T)} = \begin{bmatrix} \enclose{circle}{10} & 1 & 1 \\ 1 & \enclose{circle}{9} & 1 \\ 1 & 2 & \enclose{circle}{8} \end{bmatrix} \leq 10\begin{bmatrix} 1 & & \\ & 1 & \\ & & 1 \end{bmatrix} + 2\begin{bmatrix} & & 1 \\ 1 & & \\ & 1 & \end{bmatrix} + \begin{bmatrix} & 1 & \\ & & 1 \\ 1 & & \end{bmatrix}$$

$$\boldsymbol{C(T)} = \begin{bmatrix} \enclose{circle}{10} & 1 & 1 \\ 1 & 9 & \enclose{circle}{1} \\ 1 & \enclose{circle}{2} & 8 \end{bmatrix} \leq 10\begin{bmatrix} 1 & & \\ & & 1 \\ & 1 & \end{bmatrix} + 9\begin{bmatrix} & & 1 \\ & 1 & \\ 1 & & \end{bmatrix} + 8\begin{bmatrix} & 1 & \\ 1 & & \\ & & 1 \end{bmatrix}$$

Fig. 3. "Largest-Entry-First" and "shadow" (the first example).

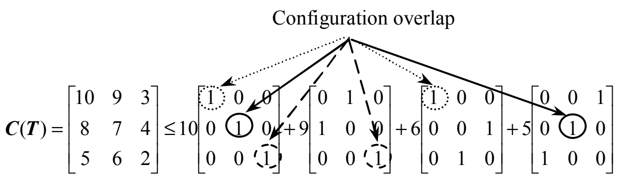
Configuration overlap
$C(T) = \begin{bmatrix}10 & 9 & 3\\ 8 & 7 & 4\\ 5 & 6 & 2\end{bmatrix} \le 10\begin{bmatrix}1&0&0\\0&1&0\\0&0&1\end{bmatrix} + 9\begin{bmatrix}0&1&0\\1&0&0\\0&0&1\end{bmatrix} + 6\begin{bmatrix}1&0&0\\0&0&1\\0&1&0\end{bmatrix} + 5\begin{bmatrix}0&0&1\\0&1&0\\1&0&0\end{bmatrix}$

Fig. 4. Configuration overlap in the LEF procedure.

B. QLEF Algorithm

QLEF algorithm is designed to rectify the above configuration overlap problem. We use a reference matrix $\boldsymbol{R}=\{r_{ij}\}$ to record all the remaining not-yet-scheduled entries in $\boldsymbol{C(T)}$. $r_{ij}=1$ means that c_{ij} is not yet scheduled/covered, and $r_{ij}=0$ otherwise. $\boldsymbol{R}$ is initialized to an all-1 matrix. Let the N configurations be sequentially constructed from $\boldsymbol{P}_1$ to $\boldsymbol{P}_N$. When a configuration is determined, the corresponding entries in both $\boldsymbol{C(T)}$ and $\boldsymbol{R}$ are set to 0*s*. The updated $\boldsymbol{C(T)}$ and $\boldsymbol{R}$ are then used to determine the next configuration.

Without loss of generality, we focus on the construction of the $(n+1)$-th configuration $\boldsymbol{P}_{n+1}$. Assume that both $\boldsymbol{C(T)}$ and $\boldsymbol{R}$ are updated and un-shadowed. The construction process is similar to the original LEF procedure, except that we only select the first $N-(2n+1)$ "largest" entries in $\boldsymbol{C(T)}$ (instead of N in LEF). We call them *selected-entries*. The corresponding lines of the selected-entries are shadowed in *both* $\boldsymbol{C(T)}$ and $\boldsymbol{R}$ (refer to Fig. 5). At this point, we should have shadowed $N-(2n+1)$ rows and $N-(2n+1)$ columns in $\boldsymbol{R}$, and the remaining not-yet-shadowed part of $\boldsymbol{R}$ can form a $(2n+1)\times(2n+1)$ sub-matrix defined as $\boldsymbol{U}$. Construct a bipartite graph [6] $\boldsymbol{U_G}$ from $\boldsymbol{U}$, where all the previously covered entries do not appear in $\boldsymbol{U_G}$ because they have been set to 0*s*. Then find a (partial) perfect matching containing $(2n+1)$ edges by performing maximum-size matching (MSM) [9] in $\boldsymbol{U_G}$ (this point is proved later). This partial perfect matching corresponds to $(2n+1)$ not-yet-covered entries in $\boldsymbol{C(T)}$, called *MSM-entries*. Combining the $(2n+1)$ MSM-entries with the other $N-(2n+1)$ selected-entries, we get the perfect matching $\boldsymbol{P}_{n+1}$.

In the above procedure, the $N-(2n+1)$ selected-entries can always be properly chosen from the not-yet-covered entries according to the LEF procedure. This is because there are $N-n$ not-yet-covered entries in each line of $\boldsymbol{C(T)}$ and we only need to select $N-(2n+1)$ entries from them, where $N-(2n+1)<N-n$.

Another key issue is to prove that a partial perfect matching containing $(2n+1)$ edges definitely exists in $\boldsymbol{U_G}$. In fact, the following Theorem (taken from Theorem 7 of [5]) can guarantee this, and some further explanation follows.

Theorem: For a bipartite graph $\boldsymbol{G}=(\boldsymbol{X}\cup\boldsymbol{Y}, \boldsymbol{E})$ with $|\boldsymbol{X}|=|\boldsymbol{Y}|=k$, there always exists a perfect matching in $\boldsymbol{G}$ if its minimum degree is greater than $k/2$.

Since we have determined n perfect matchings prior to $\boldsymbol{P}_{n+1}$, there are at most n 0*s* (denoting covered entries) in each line of $\boldsymbol{U}$. Because $\boldsymbol{U}$ is a $(2n+1)\times(2n+1)$ sub-matrix, the minimum degree of its bipartite graph $\boldsymbol{U_G}$ is at least $(2n+1)-n=n+1>(2n+1)/2$. Therefore, a perfect matching containing $(2n+1)$ edges exists in $\boldsymbol{U_G}$ according to the Theorem.

The above discussion provides the foundation for QLEF algorithm. QLEF also has two other features: 1) since $N-(2n+1)>0$ requires $n<(N-1)/2$, we use the above procedure to determine only the first $\lceil N/2\rceil-1$ configurations. This ensures that the $N-(2n+1)$ selected-entries can be properly chosen; 2) after the first $\lceil N/2\rceil-1$ configurations are determined, we find the largest c_{ij} in $\boldsymbol{C(T)}$ and use it as the *constant* weight for each of the subsequent configurations (so as to cover the remaining small entries). Because the bipartite graph of $\boldsymbol{R}$ always remains *regular* after a configuration is determined, each of the subsequent configurations can be obtained by finding maximum-size matching in $\boldsymbol{R}$ [5, 8].

Summarily, QLEF shown in Fig. 6 guarantees that configuration overlap never happens when scheduling $\boldsymbol{C(T)}$. So, $\boldsymbol{C(T)}$ can be covered by using only N configurations. The time complexity of QLEF is dominated by running the $O(N^{2.5})$ maximum-size matching algorithm [9] for N times, resulting

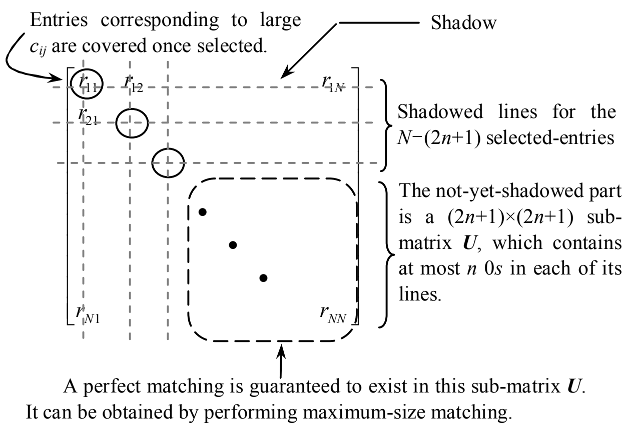
Entries corresponding to large c_{ij} are covered once selected.
Shadow
r_{11}
r_{12}
r_{1N}
r_{21}
r_{N1}
r_{NN}
Shadowed lines for the $N-(2n+1)$ selected-entries
The not-yet-shadowed part is a $(2n+1)\times(2n+1)$ sub-matrix $\boldsymbol{U}$, which contains at most n 0s in each of its lines.
A perfect matching is guaranteed to exist in this sub-matrix $\boldsymbol{U}$.
It can be obtained by performing maximum-size matching.

Fig. 5. Reference matrix $\boldsymbol{R}$ in QLEF algorithm.

QLEF ALGORITHM

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

Output:
$N\times N$ configurations $\boldsymbol{P}_1, \ldots, \boldsymbol{P}_N$ and weights $\phi_1, \ldots, \phi_N$.

Step 1: Initialization:
Set $0\rightarrow n$. Initialize $\boldsymbol{P}_1, \ldots, \boldsymbol{P}_N$ as all-zero matrices and the $N\times N$ reference matrix $\boldsymbol{R}=\{r_{ij}\}$ as all-1.

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*s* in $\boldsymbol{C(T)}$ and $\boldsymbol{R}$, and then set $n+1\rightarrow n$. Repeat this step until $n=N$.

Fig. 6. QLEF algorithm.

in the same overall time complexity of $O(N^{3.5})$ as MIN [5] and α^i-SCALE [8]. Unlike MIN and α^i-SCALE, edge-coloring and partitioning are unnecessary in QLEF.

IV. Speedup Bound

Fig. 7 shows the *conceptual* QLEF scheduling procedure. In Fig. 7b, we use a "scheduling trace" to represent the trend of c_{ij} values covered in the N ordered configurations (from $\boldsymbol{P_1}$ to $\boldsymbol{P_N}$). The scheduling trace is usually a *wave* rather than a monotonically decreasing curve, although QLEF always selects the largest entry in the not-yet-shadowed part of $\boldsymbol{C(T)}$. Due to the shadowing mechanism, a large c_{ij} may be shadowed by other lager entries in the same line, and is therefore scheduled (after some smaller entries) in a later configuration.

We first consider the first "half" ($\lceil N/2\rceil-1$) configurations. When constructing each of them, if c_{ij} is not a selected-entry, then there are two possible cases for c_{ij}: it is shadowed in the current configuration, or it locates in the not-yet-shadowed part (refer to Fig. 7a). If c_{ij} is shadowed in the current configuration, then according to QLEF there must be a larger/equal entry (than c_{ij}) covered in the same line by the current configuration. If c_{ij} is not shadowed, then it must be no larger than all the selected-entries because QLEF always selects the largest entry in the not-yet-shadowed part. In the latter case, we call the selected-entries as *absolutely larger entries* (ALEs) for c_{ij}.

Particularly, we consider the weight ϕ_{n+1} of $\boldsymbol{P_{n+1}}$. Note that ϕ_{n+1} also appears as an entry of the *original* $\boldsymbol{C(T)}$ and is covered in $\boldsymbol{P_{n+1}}$. Among the n configurations prior to $\boldsymbol{P_{n+1}}$, we assume that Δ configurations do not shadow ϕ_{n+1}, and the other $n-\Delta$ configurations shadow it, as shown in Fig. 7b.

For ϕ_{n+1}, what is the minimum *total* number of ALEs in the above scenario? In QLEF, the number of selected-entries becomes less and less in each subsequent configuration. So, the total number of ALEs for ϕ_{n+1} can be minimized only if the Δ configurations that do not shadow ϕ_{n+1} are the last Δ (out of n) configurations prior to $\boldsymbol{P_{n+1}}$. As a result, the minimum total number of ALEs is $(N-2n+1)+(N-2n+3)+(N-2n+5)+\ldots+[(N-2n)+(2\Delta-1)]=(N-2n)\Delta+\Delta^2$. These ALEs distribute over the $N-1$ lines (either rows or columns) of the *original* $\boldsymbol{C(T)}$. They do not go into the same line as ϕ_{n+1}, because any one of them is larger than (or equal to) ϕ_{n+1} but does not shadow it.

So, the line of the *original* $\boldsymbol{C(T)}$ that contains the maximum number of ALEs must contain at least L ALEs, where

$$L=\left\lceil\frac{(N-2n)\Delta+\Delta^2}{N-1}\right\rceil. \tag{3}$$

Furthermore, the smallest ALE in this line must be smaller than or equal to the L-th largest entry of this line. Yet, this smallest ALE is *not* smaller than ϕ_{n+1}. Because each line of $\boldsymbol{C(T)}$ sums up to at most T, from Lemma 1 in the Appendix, we have

$$\phi_{n+1}\le\frac{T}{L}=\frac{T}{\left\lceil\frac{(N-2n)\Delta+\Delta^2}{N-1}\right\rceil}. \tag{4}$$

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

$$\phi_{n+1}\le\frac{T}{\left\lceil\frac{n-\Delta}{2}\right\rceil+1}=\frac{T}{\left\lceil\frac{n-\Delta}{2}+1\right\rceil}. \tag{5}$$

Combining (4) and (5), for $0\le n<\lceil N/2\rceil-1$ we get

$$\phi_{n+1}\le\max_{0\le\Delta\le n}\left\{\min_{0\le\Delta\le n}\left[\frac{T}{\left\lceil\frac{n-\Delta}{2}+1\right\rceil},\ \frac{T}{\left\lceil\frac{(N-2n)\Delta+\Delta^2}{N-1}\right\rceil}\right]\right\}. \tag{6}$$

$$or\quad \phi_{n+1}\le\left.\frac{T}{\left\lceil\frac{n-\Delta}{2}+1\right\rceil}\right|_{\Delta=\left\lfloor\frac{\sqrt{(3N-4n-1)^2+8(N-1)(n+2)}-(3N-4n-1)}{4}\right\rfloor}. \tag{7}$$

Note that (6) & (7) hold for any $0\le\Delta\le n$.

From the above discussion, it is clear that any weight ϕ_{n+1} of the first $\lceil N/2\rceil-1$ configurations $\boldsymbol{P_{n+1}}$ ($0\le n<\lceil N/2\rceil-1$) can be bounded by (6) & (7). For the remaining "half" configurations, QLEF uses a small constant as the weight. 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 in the dashed line in Fig. 7b). Consequently, it can be bounded by the weight of the last

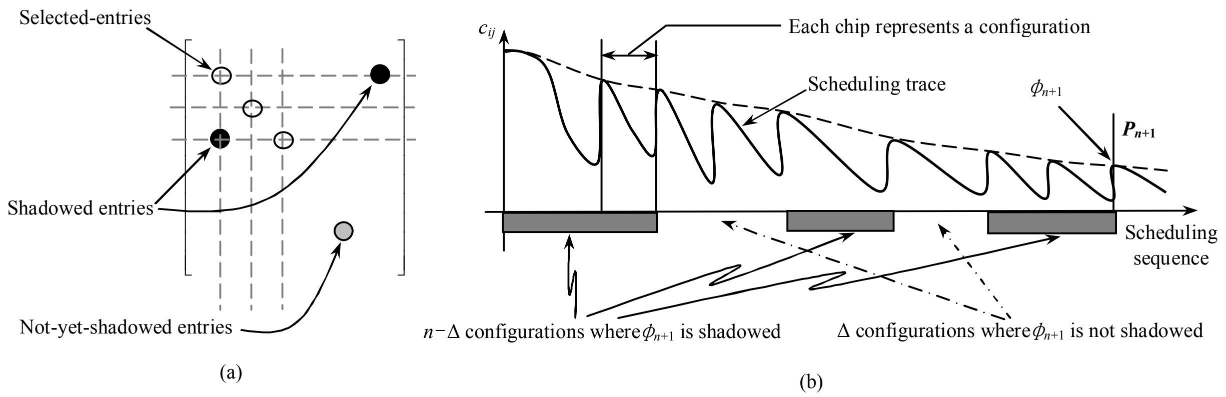
Selected-entries
Shadowed entries
Not-yet-shadowed entries
(a)
c_{ij}
Each chip represents a configuration
Scheduling trace
ϕ_{n+1}
P_{n+1}
Scheduling sequence
$n-\Delta$ configurations where ϕ_{n+1} is shadowed
Δ configurations where ϕ_{n+1} is not shadowed
(b)

Fig. 7. Conceptual QLEF scheduling procedure.

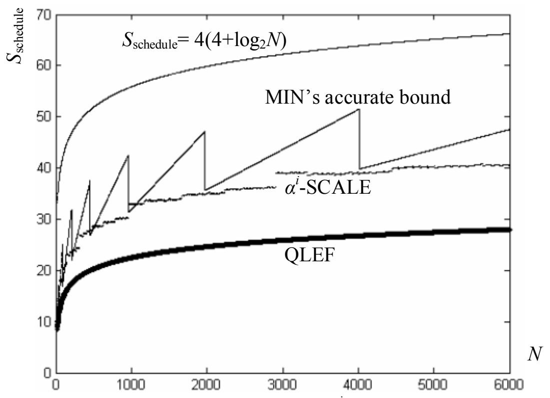

Fig. 8. Speedup bounds of MIN, α^i-SCALE and QLEF.

configuration in the first $\lceil N/2 \rceil - 1$ configurations. That is

$$\phi_{n+1}\Big|_{N \ge n+1 \ge \left\lceil \frac{N}{2} \right\rceil} \le \phi_{n+1}\Big|_{n+1=\left\lceil \frac{N}{2} \right\rceil -1} . \qquad (8)$$

From (1), (7) and (8), S_{schedule} can be bounded by

$$S_{\text{schedule}} = \frac{1}{T}\sum_{n=1}^{N}\phi_n \le \sum_{n+1=1}^{\lceil N/2 \rceil -1} \frac{1}{\left\lceil \frac{n-\Delta}{2}+1 \right\rceil}\Bigg|_{\Delta=\left\lfloor \frac{\sqrt{(3N-4n-1)^2+8(N-1)(n+2)}-(3N-4n-1)}{4} \right\rfloor} +$$

$$\frac{\left\lfloor \frac{N}{2} \right\rfloor +1}{\left\lceil \frac{n-\Delta}{2}+1 \right\rceil}\Bigg|_{\Delta=\left\lfloor \frac{\sqrt{(3N-4n-1)^2+8(N-1)(n+2)}-(3N-4n-1)}{4} \right\rfloor \text{ and } n+1=\left\lceil \frac{N}{2} \right\rceil -1} . \qquad (9)$$

Fig. 8 plots the S_{schedule} bounds for QLEF, MIN [5] and α^i-SCALE [8]. As pointed out in [8], the bound S_{schedule}= 4(4+log$_2N$) originally given in [5] is not accurate for MIN, and MIN's accurate bound is represented by the saw-toothed curve. From Fig. 8, it is clear that QLEF achieves the lowest speedup bound. For example, when N=950, QLEF only requires S_{schedule} =21.33 instead of 42.25 for MIN and 30.27 for α^i-SCALE. This gives a 50% improvement over MIN and 30% over α^i-SCALE.

V. Conclusion

In this paper, a novel minimum-delay scheduling algorithm QLEF (Quasi Largest-Entry-First) was proposed to minimize the required speedup bound for a performance guaranteed OPS switch. We proved that QLEF pushes the speedup bound to the lowest known level. On the other hand, QLEF also provides a new method to decompose an $N \times N$ matrix into only N non-overlapping permutation matrices. This technique may also be used in other communication research such as SS/TDMA and WDM networks.

Appendix

Lemma 1: If the set of N entries $\boldsymbol{E}$={e_1, e_2, …, e_N} sum to at most T and e_k is the k-th largest entry in $\boldsymbol{E}$, then $e_k \le T/k$.

Proof: Assume that the N entries in $\boldsymbol{E}$ are sorted in a monotonically decreasing order as $e_1 \ge e_2 \ge \ldots \ge e_N$. Because $\sum^{N}_{i=1} e_i \le T$ and $\boldsymbol{E}$ is assumed to be monotonically decreasing, e_k can reach its maximum possible value only when e_{k+1}= e_{k+2}=…= e_N=0 and e_1= e_2=…= e_k. We then have

$$e_k = \frac{1}{k}\sum_{i=1}^{k} e_i = \frac{1}{k}\sum_{i=1}^{N} e_i \le \frac{T}{k} .$$

Lemma 2: In QLEF scheduling, if any not-yet-covered entry c_{ij} of $\boldsymbol{C(T)}$ is shadowed by k configurations, we have

$$c_{ij} \le \frac{T}{\left\lceil \frac{k}{2} \right\rceil + 1} .$$

Proof: In QLEF, any not-yet-covered entry c_{ij} can only be shadowed by another larger (or equal) selected-entry in the same line. Because c_{ij} is shadowed by k configurations, those configurations collectively cover at least k larger (or equal) selected-entries in the same line as c_{ij}.

Among these k larger/equal selected-entries, some may locate in row i while others locate in column j (because a line may refer to either a row or a column). Without loss of generality, we assume that k' out of the k selected-entries locate in row i and the other $k-k'$ locate in column j. As a result, c_{ij} is (at most) the (k'+1)-th largest entry in row i and the ($k-k'$+1)-th largest entry in column j. Because both row i and column j sum to at most T, according to Lemma 1 we have

$$c_{ij} \le \frac{T}{k'+1} \text{ and } c_{ij} \le \frac{T}{k-k'+1},$$

$$or \quad c_{ij} \le \min\left\{ \frac{T}{k'+1} \ , \ \frac{T}{k-k'+1} \right\} \le \frac{T}{\left\lceil \frac{k}{2} \right\rceil + 1} .$$

References

[1] 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.

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

[3] 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.

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

[5] 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.

[6] Xin Li and Hamdi, M., "On scheduling optical packet switches with reconfiguration delay", *IEEE Journal on Selected Areas in Communications*, vol. 21, issue 7, pp. 1156-1164, Sept. 2003.

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

[8] 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.

[9] 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.