Time Blocking in Time-Driven-Switched Networks

V-T. Nguyen, R. Lo Cigno, Y. Ofek
Università di Trento – Dip. di Ingegneria e Scienza dell'Informazione (DISI)
M. Telek
Budapest University of Technology and Economics – Department of Telecommunications

***Abstract*—This paper presents a closed-form analysis of the time-blocking probability in time-driven switching networks. Time-blocking occurs when transmission resources are available in both input and output, but there is no schedule, i.e., output resources are outside a pre-defined time delay that is allowed for the input. This situation may happen in architectures based on *pipeline forwarding*. The main constraints affecting the schedulability of resources are the load and the maximum delay allowed between input and output. The analysis yields the exact blocking probabilities for all possible scheduling delays and under all load conditions for a single node, as well as initial results for a network of nodes.**

***Index Terms*—optical networks; sub-lambda switching; fractional lambda switching; time-driven switching; blocking probability analysis; combinatorial analysis; switching analysis.**

I. Introduction

Time-Driven Switching (TDS) is a technique where switching decisions are controlled by a common time reference, like the one provided by GPS (Global Positioning System) or Galileo, the European system now under deployment, that delivers UTC (Coordinated Universal Time) everywhere around the globe with high precision [1] and for low cost. TDS operation is based on UTC 1PPS (pulse per second) and does not require high frequency synchronization between switches or input/output ports, thus from this perspective TDS is different and much less demanding than SONET/SDH systems. Furthermore, the TDS operation is completely decoupled from bit synchronization of the serial links, again a major simplification.

A promising example of TDS networks is FλS [4], [5], where the capacity of an optical channel is divided into a large number of sub-channels by using time units, called Time-Frames (TFs), of equal duration. The result is the realization of end-to-end sub-channel pipes that deliver information with minimum buffering and delay and with no delay jitter. A working prototype of FλS networks with TDS is now operational at DISI in Trento [2].

As was mentioned, TDS networks are managed based on the UTC second (i.e., 1PPS) that is divided into Time-Frames (TFs). A group of K contiguous TFs forms a time-cycle (TC); L contiguous TCs are grouped into a super cycle that is equal to one UTC second or 1PPS, as shown in Fig.1. In this example: $K = 1000$ and $L = 80$. In TDS, all TFs are aligned with respect to UTC at the input prior to switching, which constitutes a necessary condition for pipeline forwarding (PF). A path is set up end-to-end by properly configuring the switching time of a TF at all switches along the path.

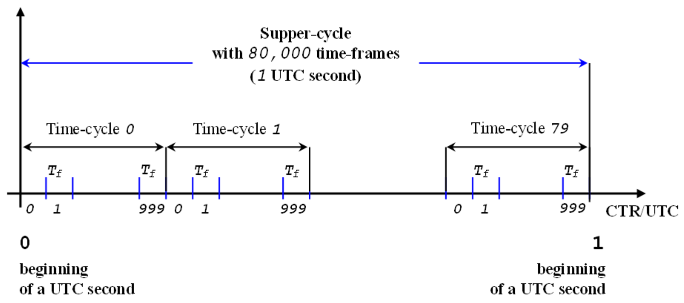

Fig. 1. Cycles and use of UTC in TDS.

One of the key issues in TDS networks is to find the *scheduling* for forwarding TFs (which contain packet flows) along the end-to-end path. Scheduling is a fundamental problem because the TDS architecture aims at minimizing delays, and consequently buffering: if a TF at the input cannot be scheduled for transmission on the output, then the flow using that TF must be denied admission, which means that it is blocked or *time-blocked*.

Indeed, the problem of blocking can be decomposed into three factors: (*i*) there are no resources at the output: this is the traditional problem faced in circuit switching networks normally named 'call blocking'; (*ii*) resources are available at the output, but the internal structure of the switch prevents connecting the input and the output, this is normally called 'space blocking'; (*iii*) resources are available at the output, and the switch can be configured to connect the input and the output, but it is impossible to find a feasible schedule mapping from the input TF to the output TF, this is time-blocking.

Time-blocking is normally solved by buffering, and indeed in TDS switches if K buffers for K TFs per input are used time-blocking is zero; however, in all optical ultra-fast switches buffering is not trivial and with today technology it is limited to only a few TFs.

In the design of a TDS network time-blocking as well as space-blocking must be taken into account. In [6] three different architectures for all optical FλS TDS switches were proposed and analyzed from the complexity and *schedulability* point of view. Schedulability is the ration of the number of schedulable input/output combinations versus the total number of possible combinations. One of the architectures was found technically feasible and non-space-blocking, and for this architecture the time-blocking was computed in the simple case when no delay or buffer is available.

In this work we compute the time-blocking of non-space-blocking time-driven-switches for the general case of both immediate and non-immediate forwarding.

Let $z < K-1$ be the number of buffers available to delay a TF switching from input to output. We define two basic cases: (*i*) *Immediate forwarding (IF):* the case of $z=0$; (*ii*) *Non-immediate forwarding (NIF):* the case of $z \geq 0$.

In essence, our objective is to compute the time-blocking probability as a function of z and the load of the input ports and output ports. Note again that if $z = K-1$ then each and every TF at the output is schedulable for each and every TF at the input and time-blocking is zero.

The contribution of this paper is the exact (closed-form) computation of the time-blocking for all values of z. In addition we provide preliminary results on the methodology to compute the time-blocking in multi-node scenarios. (We remark that the results are mathematically exact for the problem defined, and therefore, validation through simulations is not presented.)

The paper is organized as follows, in Section II the blocking problem is formulated, while providing the analysis approach. Then Section III provides the analysis for the case when $z = 1$, while Section IV provides the analysis for any value of z. Section V formulates and analyzes the multi-hop scenario, while conclusions are discussed in Section VI.

II. Problem Formulation and Analysis Approach

A. Basic assumptions and definitions

We assume independence of each channel (i.e., input and output), thus we examine a single input channel and a single output channel of the switch.

Load definition — The load is defined as the number of busy TFs per TC per channel. For all channels, the busy TFs within each TC is assumed to be distributed uniformly at random. Let b denote the number of busy TFs per TC. The load of a channel is identified by the pair (K, b), where K is the number of TFs in each TC, as was shown in Fig.1.

For the sake of using simple notations we use b identical for all the inputs and the outputs, but this assumption is not required for the analysis and in Sect. IV-C we present results where the load of the input and the output are different.

To formulate the problem, we define the following notations:

- a denotes the number of free TFs per TC, $a = K - b$;
- tf_k denotes a TF k in a TC;
- tf_k^{in} denotes TF k of the *input*, $0 \leq k < K$;
- tf_k^{out} denotes TF k of the *output*, $0 \leq k < K$;
- z denotes the number of buffers (or maximum scheduling delay measured in TFs), $0 \leq z < K$;
- the symbol '`0`' denotes a busy TF;
- the symbol '`1`' denotes an available (or free) TF.

Note that the TF index is periodic, which implies that if $k \geq K$ then $k = (k \bmod K)$ since K TFs are grouped in a TC.

Def. 1 (z-forwarding scheme): — A switch is under *z-forwarding* scheme iff the content of a TF can be buffered arbitrarily for i TFs prior to being forwarded, $i = 0, 1, .., z$.

In other words, for the *z-forwarding* scheme the maximum scheduling delay of a TF is equal to z TFs. $z = 0$ means the immediate-forwarding (IF) scheme or zero scheduling delay.

Def. 2 (A schedulable TF*):* — For a pair of input and output, a TF k of the output (i.e., tf_k^{out}) is schedulable iff tf_k^{out}='`1`' and at least one TF in the set $\{tf_{k-i}^{in} | i = 0, 1, .., z.\}$ is available.

Def. 3 (A blocked TF*):* — For an input output pair, a TF k of the output (i.e., tf_k^{out}) is blocked iff tf_k^{out}='`1`' and all TFs in the set $\{tf_{k-i}^{in} | i = 0, 1, .., z.\}$ are busy. We use a symbol '$\mathtt{1}_b$' to denote the blocked TF, i.e., tf_k^{out}='$\mathtt{1}_b$'.

An example of schedulable and blocked TFs is shown in Fig. 2.

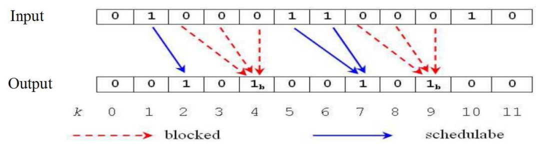

Fig. 2. Illustration when $K = 12, a = 4, z = 2$: tf_4^{out} and tf_9^{out} are blocked; tf_2^{out}, tf_7^{out} are schedulable.

Given an input and output pair of a strictly non space-blocking switch operating under *z-forwarding* scheme, we aim at deriving $p_{z\text{F}}$, the probability that all available TFs of the output are found blocked under a given the load (K, b).

Let C_{blk} be the number of input/output TFs combinations such that all the a available TFs of the output are found blocked, and C_{total} be the total number of input/output TFs combinations. The time-blocking probability is the ratio between C_{blk} and C_{total}:

$$p_{z\text{F}} = \frac{C_{blk}}{C_{total}} \ . \tag{1}$$

B. Run, run-length and blocked positions

Run and run-length: A *run* is defined as a group of equal symbols that are positioned consecutively. For examples, runs of `0`'s are '`0`', '`00`', '`000`' and so on. A number of symbols composing a run is its *run-length*. In between two adjacent runs of `0`'s there is one run of `1`'s, and vice versa.

Because of the periodic nature of TCs, the last TF of a TC and the first TF of the next TC are positioned consecutively. Therefore, in each arrangement the number of runs of `0`'s and the number of runs of `1`'s are equal, excluding the trivial cases of all zeros and all ones.

For instance, the cyclical arrangement of 4 symbols '`1`' and 8 symbols '`0`' shown in the input in Fig. 2, there are 3 runs of `1`'s and 3 runs of `0`'s.

Blocked positions: For a given *z-forwarding* scheme, arrangements of the a available TFs and the b busy TFs in the input may generate positions such that available TFs in the output that are positioned 'beneath' (with reference to the input/output mapping as in Fig. 2) are *blocked*, i.e., tf_k^{out}='$\mathtt{1}_b$'.

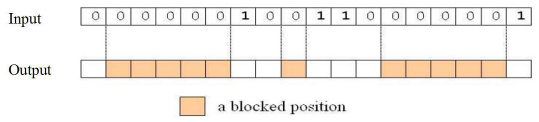

Fig. 3. Example of blocked positions when $z = 1$, given a sample combination of the input.

In order to highlight the idea of blocked positions, which is important in the following analysis, let's consider the following examples:

- For $z = 0$ (i.e., the IF (immediate forwarding) scheme), any arrangement in the input generates b blocked positions. Obviously, if an output's available TF is 'positioned beneath' a busy TF of the input, it is blocked since $z = 0$.
- For $z = 1$, the content of a TF can be delayed at most one TF duration prior to being forwarded. Fig. 3 shows how blocked positions are generated. In fact, for every pair of adjacent '`00`' symbols the right symbol generates a blocked position. Consequently, if there are $l > 1$ consecutive `0`'s, then there are $l - 1$ blocked positions.
- For $z = 2$, a content of TF can be delayed at most two TF durations. Only runs whose run-length is greater than two, such as, '`000`', '`0000`' and so on, generate blocked positions. Consequently, if there are l consecutive `0`'s and $l > 2$, then there are $l - 2$ blocked positions.

Consequently, the number of blocked positions generated by a given arrangement (of available TFs and busy TFs) in the input depends on the specific *z-forwarding* scheme and the given load (K, b) of the input. For a run of `0`'s, there is a relation between the run-length the number of blocked positions generated, and z. Let l_i be the run-length of run i of `0`'s. Let x_i be the number of blocked positions generated by run i, then:

$$x_i = \begin{cases} l_i - z & \text{if } \; l_i \geq z, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

We are interested in run i such that $l_i \geq z$.

Bounds of the number of blocked positions: Given an arrangement in the input, let $x \subset \mathbb{Z}^+$ be the total number of blocked positions generated from all runs of `0`'s in a TC.

Lemma *1 (Bounds of x):* For a given load (K, b), x is bounded by:

$$b - za = x_{min} \leq x \leq x_{max} = b - z \ . \tag{3}$$

Proof: From (2), we yield $x_{max} = b - z$ when all the b busy symbols form a single run in the input, which is obviously the longest possible run.

To compute x_{min}, we further observe that, in a cyclical arrangement, a symbols of '`1`' can split at maximum a runs of `0`'s, where every run has the same length of z (i.e., $l_i = z$ for all i) such that no blocked position is generated according to (2). The remaining number of symbols '`0`' is $(b - za)$. Since no more run of `0`'s can be formed due to running out of symbols '`1`' to split them. Thus, placements of remaining '`0`' symbols generate blocked positions. Therefore, $x_{min} = (b - za)$. ■

In a switch in isolation time-blocking happens only when all available TFs of the output are in blocked positions, since the input TF can be chosen freely.

C. General form of the time-blocking probability

For a given value of x satisfying (3), let $C(x)$ be the number of arrangements found in the input such that each of these arrangements generate exactly x blocked positions. Given $C(x)$, we have the following result:

Theorem *1:* — For a stand-alone switch, the time-blocking probability for the general *z-forwarding* scheme, $p_{z\text{F}}$ is given by:

$$p_{z\text{F}} = \sum_{x=\max\{a,(b-za)\}}^{b-z} C(x) \binom{x}{a} \Big/ \binom{K}{b}^2 . \tag{4}$$

Proof: Given x blocked positions generated by the input, the number of ways to arrange all a available TFs of the output into blocked positions so that a time-blocking occurs is $\binom{x}{a}$. Thus, the subtotal number of combinations, denoted as C_{sub}, generated by both the input and the output such that time-blocking happens is given by:

$$C_{sub} = C(x) \binom{x}{a} .$$

If $x < a$, then $\binom{x}{a} = 0$. Thus, we only consider $x \geq a$ (i.e., a case where a time-blocking occurs). From Lemma 1, observe that:

- if $(b - za) \geq a \Leftrightarrow K \geq (z+2)a$ then for any combination in the input, we have $x_{min} = (b - za) \geq a$.
- if $(b - za) < a \Leftrightarrow K < (z + 2)a$ then for some x such that $b - za \leq x < a$, we are not interested in. Thus we set $x_{min} = a$.

Combined with (3) we have the range of meaningful x values for computing time-blocking probability:

$$\max\{a, (b - za)\} \leq x \leq b - z \ . \tag{5}$$

The sum of C_{sub} over all meaningful x yields C_{blk}:

$$C_{blk} = \sum_{x=x_{min}}^{x_{max}} C_{sub} = \sum_{x=\max\{a,(b-za)\}}^{b-z} C(x) \binom{x}{a} .$$

Meanwhile, total numbers of combinations at the input and at the output are computed as $\binom{K}{b}$ for each input and output. Thus, we have C_{total}:

$$C_{total} = \binom{K}{b}\binom{K}{b} = \binom{K}{b}^2 .$$

■

The derivation of $C(x)$, i.e., the number of combinations in the input generating exactly x blocked positions, is one of the main contribution of this paper and we dedicate Section III and IV to its computation.

III. Analysis for 1-Forwarding Case

We separate the analysis of the 1-*forwarding* scheme from the general case, because its simpler mathematics allows for descriptions and explanations that will help in deriving the general case. 1-*forwarding* means there is a single position in the buffer: $z = 1$.

Let u be the number of runs of `0`'s. For $z = 1$ all runs satisfy $l_i \geq z$. Summing (2) over all runs yields:

$$\sum_{i=1}^{u} x_i = \sum_{i=1}^{u} (l_i - z) = \sum_{i=1}^{u} l_i - uz \ .$$

Since $\sum_{i=1}^{u} x_i = x$ (total number of blocked positions) and $\sum_{i=1}^{u} l_i = b$ (total number of symbols '`0`'), the equation above becomes:

$$u = b - x \ . \tag{6}$$

(6) holds only for $z = 1$, and it is the reason why this case can be treated differently from the general one. In this case the computation of $C(x)$ can be done in two different ways. The first one, considering a linear disposition of the symbols, which gives the result with a problem decomposition in form of summation. The second one, which will be used also in the general case, considers the cyclic disposition of the symbols and gives the results in form of a multiplicative decomposition that, however, counts the number of possible patterns u times, so that the final result must be divided by u.

A. Additive decomposition

The time cycle patterns may be viewed as non-cyclic patterns with b symbols '`0`' and the a symbols '`1`'. It is easy to identify three possible cases of non-cyclic patterns:
Case 1: the first and the last symbol of the cycle are different, implying that there are u runs of `0`'s and u runs of `1`'s. Case 1 has two obvious and identical (from the combinatorial point of view) sub-cases: the first symbol is '`0`' and the last one is '`1`', or vice versa.
Case 2: both the first and the last symbol of the cycle are '`1`'. so that there are u runs of `0`'s, and $(u + 1)$ runs of `1`'s.
Case 3: both the first and the last symbol of the cycle are '`0`', so that there are $(u + 1)$ runs of `0`'s and u runs of `1`'s.

It is easy to see that the three cases above form a partition of the set of the dispositions, and this is valid for any given x, so that $C(x)$ can be computed as the sum of the three cases.

For $z = 1$, $C(x)$ is given by:

$$\begin{aligned} C(x) &= C_{\text{case 1}} + C_{\text{case 2}} + C_{\text{case 3}} \\ &= \frac{K}{u}\binom{a-1}{u-1}\binom{b-1}{u-1} \ , \end{aligned} \tag{7}$$

where x is implicit in u according to (6).

The following provides the derivation of (7).

Consider case 1: the number of dispositions is the product of the following terms:

- the number of dispositions of the a symbols '`1`' into u distinct runs such that there will be at least one symbol per run. Basic combinatorics (see Chapter 2 of [3]) yields $\binom{a-1}{u-1}$,
- the number of dispositions of the b symbols '`0`' into u distinct runs such that there will be at least one symbol per run, which is $\binom{b-1}{u-1}$,
- a multiplicative factor of 2 reporting of the two subcases:

$$C_{\text{case 1}} = 2\binom{a-1}{u-1}\binom{b-1}{u-1} \ .$$

Following the same counting methods we obtain:

$$C_{\text{case 2}} = \binom{a-1}{u}\binom{b-1}{u-1} = \frac{a-u}{u}\binom{a-1}{u-1}\binom{b-1}{u-1} \ ,$$

$$C_{\text{case 3}} = \binom{a-1}{u-1}\binom{b-1}{u} = \frac{b-u}{u}\binom{a-1}{u-1}\binom{b-1}{u-1} \ .$$

Summing together the three cases leads to (7).

Substituting (7) into (4), replacing $u = b - x$, $z = 1$ and $a = K - b$ yields the time-blocking probability for the 1-*forwarding* scheme:

$$p_{1\text{F}} = \frac{\sum_{x=\max\{a,(b-a)\}}^{b-1} \frac{K}{b-x}\binom{K-b-1}{b-x-1}\binom{b-1}{b-x-1}\binom{x}{K-b}}{\binom{K}{b}^2} \tag{8}$$

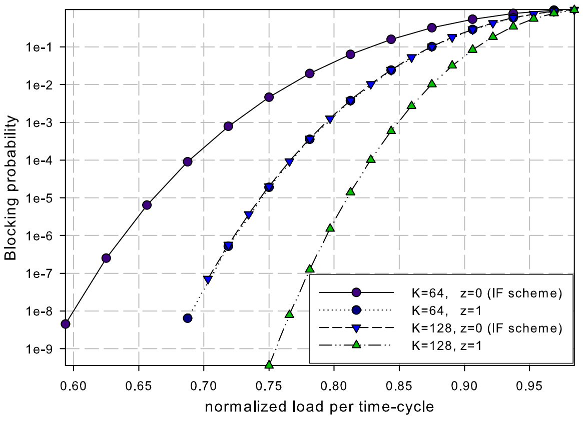

Fig. 4. Examples of numerical result for $z = 0$ and $z = 1$.

Fig. 4 shows numerical examples obtained from (8) for 1-*forwarding* scheme and results for 0-*forwarding* (IF) scheme. In the graph, numerical results for $(K = 64, z = 1)$ and for $(K = 128, z = 0)$ are very close to each other. However, a quick investigation on the actual numbers shows that they are not identical.

IV. Analysis for General z-Forwarding Cases

Equation (6) holds only for $z = 1$, since this is the only case where condition $l_i \geq z$ always holds. If (6) is not valid, the relationship between x, u and b becomes more complex. When condition $l_i \geq z$ is not satisfied by all runs of `0`'s, these runs are divided into two subsets: those that leads to blocking positions and those that do not. Table I summarizes

TABLE I
SUMMARY OF THE NOTATION USED FOR THE GENERAL CASE.

Notation	Explanation
a	number of symbols '1' (i.e. number of free TFs),
b	number of symbols '0' (i.e. number of busy TFs),
z	number of buffers, $1 \leq z < K$,
$p_{z\mathrm{F}}$	blocking probability under z-forwarding scheme,
l_i	run-length of run i,
x_i	number of blocked positions generated by run i,
x	total number of blocked positions generated by all runs of 0's in a given arrangement,
$\mathbb{U}$	set of all runs of 0's such that $l_i \geq z$,
u	number of runs in $\mathbb{U}$, $u = \|\mathbb{U}\|$,
b_u	total number of symbols '0' occupied by all runs in $\mathbb{U}$,
$\mathbb{V}$	set of all runs of 0's such that $1 \leq l_i < z$,
v	number of runs in $\mathbb{V}$, $v = \|\mathbb{V}\|$,
b_v	total number of symbols '0' occupied by all runs in $\mathbb{V}$,
$\mathbb{A}$	set of all runs of 1's, $u + v = \|\mathbb{A}\|$,
$C(u,v)$	number of combinations that generate exactly x blocked positions, given a valid pair of (u,v),
$C(x)$	total number of combinations in the output that generates exactly x blocked positions, for all valid pairs of (u,v).

the notation we use. The 1-*forwarding* case is the special case where $\mathbb{V} = \emptyset$.

One of the key differences between the 1-*forwarding* case and the general case is the presence of non-valid (u,v) couples, i.e., values of u and v that do not satisfy all the constraints of the problem. This fact forces us to separately count for all and any the valid (u,v) couples, while the simple relation (6) allowed for a unique computation. Given this additional complexity, partitioning the set of patterns as we did for $z = 1$ becomes excessively cumbersome, so we resort to the analysis considering the cyclic disposition of TFs.

We now define some general bounds for the parameters of the problem, that will be the upper and lower limits of the indexes used in the formulas derived afterwards. Summing (2) over all runs in $\mathbb{U}$ yields (with some algebraic manipulations):

$$0 < b_u = x + zu \leq b \ . \tag{9}$$

The number of symbols b_v is given by:

$$b_v = b - b_u = b - x - zu \geq 0 \ . \tag{10}$$

While by construction, we have:

$$1 \leq u + v \leq a \ . \tag{11}$$

Lemma *2:* The size of $\mathbb{U}$ is bounded by:

$$1 = u_{min} \leq u \leq u_{max} = \min\Big\{\Big\lfloor\frac{b-x}{z}\Big\rfloor, a\Big\} \ . \tag{12}$$

Proof: When there is only one run of 0's, we have $u_{min} = 1$. From (9) we have $u = \frac{b_u - x}{z}$ and $u = u_{max}$ iff $b_u = b$. $b_u = b$ implies that all symbols '0' of the input are in runs belonging to $\mathbb{U}$ and $\mathbb{V}$=$\emptyset$, $v = 0$. Setting $v = 0$ in (11) yields $u \leq a$ so that $u_{max} \leq \min\{\lfloor\frac{b-x}{z}\rfloor, a\}$.

Note that $u = 0$ is not considered since it means there is one run of 0's with length smaller than z, or $b < z$. In this case we do not have time-blocking. ■

Lemma *3:* For $1 < z < K$, the size of $\mathbb{V}$ is bounded by:

$$\Big\lceil\frac{b_v}{z-1}\Big\rceil = v_{min} \leq v \leq v_{max} = \min\{a-u, b_v\} \tag{13}$$

Proof: We have $v = v_{min} = \lceil\frac{b_v}{z-1}\rceil$ when all runs in $\mathbb{V}$ have the maximum allowed length $l_i = (z-1)$.

The upper bound depends on the ratio between b_v and the number of symbols '1' not used to separate runs in $\mathbb{U}$ that can separate runs in $\mathbb{V}$. That is $a-u$.

- If $b_v > a-u$ then $v_{max} = a-u$.
- If $b_v \leq a-u$, we can split all b_v symbols '0' in runs of length one, so that $v_{max} = b_v$.

Therefore, $v_{max} = \min\{a-u, b_v\}$. ■

A. Deriving $C(x)$

Equations (9)-(13) define the limits of (u,v) for a given value of blocking positions x satisfying (5). Recall that in a time-cycle, tf_{K-1} is adjacent to tf_0.

***Theorem** 2:* Given a valid pair of (u,v), the number of patterns, denoted as $C(u,v)$, that exactly generates x blocked positions, is:

$$C(u,v) = \frac{KC_{uv}C_aC_{b_u}C_{b_v}}{u+v} \ , \tag{14}$$

where x is implicit in b_u, b_v, u, v given the relations (9)-(13). The factors C_{uv}, C_a, C_{b_u}, and C_{b_v} are defined in (15)-(18) of the proof, respectively.

Proof: The goal is computing the total number of possible patterns distributing the b_u symbols '0' into $\mathbb{U}$ runs, the b_v symbols '0' into $\mathbb{V}$ runs, and the a symbols '1' into runs in $\mathbb{A}$. To obtain this we show that there exists a factorization of the problem that counts u+v times the total number of patterns. The factorization starts counting the possible dispositions of the runs themselves given u and v, then counts the dispositions of the symbols in the runs in different sets $\mathbb{A}$, $\mathbb{U}$, and $\mathbb{V}$, finally all possible K cyclic shifts of the above patterns are counted showing that each pattern is counted exactly u+v times.

C_{uv}: number of dispositions of the u runs in $\mathbb{U}$ within the total number of possible runs $u+v$ of $\mathbb{U}\cup\mathbb{V}$. Further combinatorics yields:

$$C_{uv} = \binom{u+v}{u} \ . \tag{15}$$

C_a: number of dispositions of the a symbols '1' into the $(u$+$v)$ distinct runs such that each run has at least one symbol. Again, some combinatorics [3] yields:

$$C_a = \binom{a-1}{u+v-1} \ . \tag{16}$$

C_u: number of dispositions of the b_u symbols '0' into the u distinct runs such that each run has at least z symbols. The counting method consists in first placing $(z-1)$ symbols into every run $\in \mathbb{U}$, then distributing the remaining $b_u - (z-1)u$ symbols in all the u runs such that each run has at least one symbol. Using the same combinatoric result used for C_a we have:

$$C_{b_u} = \binom{b_u - (z-1)u - 1}{u-1} \ . \tag{17}$$

C_{b_v}: number of dispositions of the b_v symbols '0' into the v distinct runs such that each run has at least one symbol and no run has more than $(z-1)$ symbols:

$$C_{b_v} = \begin{cases} \sum_{i=0}^{v} (-1)^i \binom{v}{i} \binom{b_v - i(z-1)-1}{v-1} & \text{if } v > 0, \\ 1 & v=0 \mid b_v = v. \end{cases} \tag{18}$$

Deriving (18) is long and cumbersome and we refer the interested reader to the Technical Report [7].

The time-cycle boundary can be at any TF, thus there are K possible shifts for each disposition counted so far. The total number of possible dispositions given a valid pair (u, v) is then $KC_{uv}C_aC_{b_u}C_{b_v}$. However, each combination is actually counted $u+v$ times and the number $KC_{uv}C_aC_{b_u}C_{b_v}$ must be divided by $u+v$, thus resulting in (14).

Again lack of space forbids to include the proof of multiple counting here, which can be found in [7]. The rationale is that each of the $C_{uv}C_aC_{b_u}C_{b_v}$ can be transformed into exactly $u+v$ other patterns by shifting it circularly of an appropriate number of TFs. ■

***Theorem** 3:* The total number of dispositions $C(x)$ that generates exact x blocked positions is given by:

$$C(x) = \sum_{u=1}^{\min\{\lfloor \frac{b-x}{z} \rfloor, a\}} \left\{ \sum_{v=\lceil \frac{b_v}{z-1} \rceil}^{\min\{a-u, b_v\}} C(u,v)\big|_{u+v \le a} \right\} \ . \tag{19}$$

Proof: A pair of (u, v) is valid iff u and v jointly satisfy (11), (12) and (13). Since $C(u, v)$ is computed through (14) for any valid pair of (u, v), the sum of $C(u, v)$ over all valid pairs of (u, v) leads to the total number of dispositions $C(x)$ in the output that generates exact x blocked positions. ■

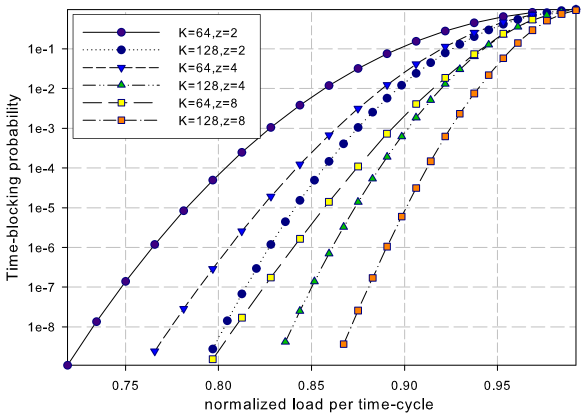

Fig. 5. Examples of numerical result for $z > 1$.

Examples of numerical results for various z and K values are shown in Fig. 5. One interesting property is the reduction in time-blocking probability as K increases for a given normalized load. While it was an easy prediction that time-blocking probability would decrease exponentially increasing the buffering capability z, a similar decrease simply increasing K was not an easy prediction. The phenomenon is similar to the classic result that gives smaller and smaller call blocking probability for a given load as the granularity of the calls decreases.

B. Sanity checks

The result for the general case presented above is rather complex and might be appalling. Here we discuss some limit cases where the exact result can be easily obtained with heuristic reasoning.

1) Immediate forwarding: This is the case when $b \le K-1$, $z = 0$. For any combination in the output, we always have $x = x_{min} = x_{max} = b$, and $C(x) = \binom{K}{b}$. Thus, the equation (4) shrinks to:

$$p_{0F} = \binom{K}{b}\binom{b}{a} \Big/ \binom{K}{b}^2 = \binom{b}{a} \Big/ \binom{K}{b} \ . \tag{20}$$

Trivial combinatorics also reach the same result.

2) 1-forwarding: This is the case when $b \le K-1$, $z = 1$. For $z = 1$, we have $\mathbb{V}=\emptyset$ or $b_v = 0$, and $b_u = b = x + u$. Letting $v = 0$ in the formulas of theorem 2 yields

$$C_{(u,v=0)} = \frac{K}{u} \binom{a-1}{u-1} \binom{b_u - (z-1)u - 1}{u-1} \ , \tag{21}$$

which is equivalent to (7) remembering that $z = 1 \Leftrightarrow b = b_u$; $v = 0$; $x = b - u$.

3) Arbitrary (full) time-frame forwarding: This is the case when $b \le K-1$, $z = K-1$. Replacing $z = K-1$ in (3) yields $x_{max} = b - z = b - (K-1) \le 0$ since $b \le K-1$. This implies that there is no single combination where we can find $x \ge a$. The intuition of zero blocking for this special case is confirmed.

C. Load Assumption Relaxation

The results presented in previous sections can be modified to accommodate different loads of the input and the output. Let (K, b_i) and (K, b_o) denote the load for the input and for the output respectively. And let a_i and a_o be the number of available TFs at the input and the output, respectively. Thus, (5) becomes:

$$\max\{a_i, (b_i - za_i)\} \le x \le b_i - z \ . \tag{22}$$

and the modified version of (4) is:

$$p_{z\mathrm{F}} = \left\{ \sum_{x=\max\{a_i, b_i - za_i\}}^{b_i - z} C(x) \binom{x}{a_o} \right\} \Big/ \left\{ \binom{K}{b_o}\binom{K}{b_i} \right\} , \tag{23}$$

$$C(x) = \sum_{u=1}^{\min\{\lfloor \frac{b_i - x}{z} \rfloor, a_i\}} \left\{ \sum_{v=\lceil \frac{b_v}{z-1} \rceil}^{\min\{a_i - u, b_v\}} C(u,v)\big|_{u+v \le a_i} \right\} , \tag{24}$$

where $b_v = b_i - b_u = b_i - x - zu \ge 0$, while (9), (15), (16), (17), (18) and (14) are reused without any change.

V. Multi-hop Scenario Formulation and Analysis

A route from a source to a destination node is connected by $H-1$ consecutive switches. Switches are similar in terms of being strictly non space blocking and having the same *z-forwarding* scheme. Hops and switches are indexed as shown in Fig. 6.

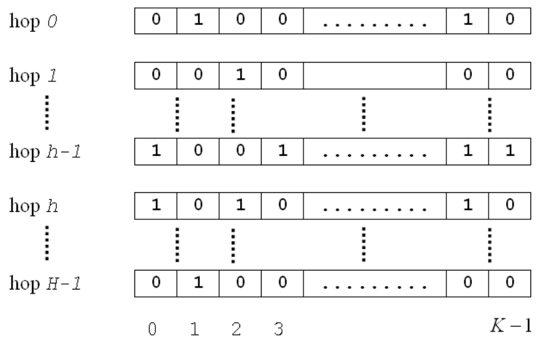

Fig. 6. H hops from source node to destination node with $H-1$ switches in between. There are K TFs per hop and b TFs (out of K) are busy.

We aim at finding the probability that after H hops there is no *schedule* to forward a TF from input to output.

A. Additional Notations and Definitions

Let $[z]$ be the set of integers $\{0, 1, 2, \cdots, z\}$, $tf_k^{(h)}$ be TF k at hop h, '$\texttt{1}_s$' a schedulable TF, and '$\texttt{1}_b$' a blocked TF.

Def. 4 (schedulable TF *at hop h):* — A TF $tf_k^{(h)}$ is said to be schedulable, $tf_k^{(h)}$='$\texttt{1}_s$', iff it is available and at least one TF in the set $\{tf_{k-i}^{(h-1)} | i \in [z]\}$ is schedulable (i.e., $tf_{k-i}^{(h-1)}$='$\texttt{1}_s$' for at least one $i \in [z]$).

A TF $tf_k^{(h)}$ is blocked, $tf_k^{(h)}$='$\texttt{1}_b$', iff it is available and all TFs in the set $\{tf_{k-i}^{(h-1)} | i \in [z]\}$ are either blocked or *busy* (i.e., $tf_{k-i}^{(h-1)}$='$\texttt{1}_b$' or $tf_{k-i}^{(h-1)}$='$\texttt{0}$' for all $i \in [z]$).

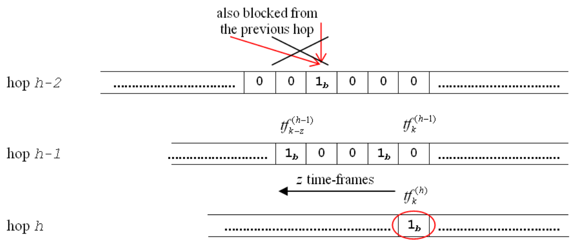

Fig. 7. Illustration of blocked TFs '$\texttt{1}_b$'.

Note that all available TFs of the first hop (i.e., $h=0$) are schedulable. That is if $tf_k^{(0)}$ ='$\texttt{1}$', then $tf_k^{(0)}$ ='$\texttt{1}_s$'.

Blocked TFs are useless in the process of searching for a schedule for setting up a FλP. An illustration of a blocked TF is shown in Fig. 7.

Def. 5: — A TF position in a TC at hop h is said to be *unblocked* position iff it is in a forwarding range of one schedulable TF of the previous hop $h-1$.

In other words, if $tf_k^{(h-1)}$='$\texttt{1}_s$', then all TF positions $tf_{k+i}^{(h)}$, $i \in [z]$, are *unblocked*. If an available TF is placed in an

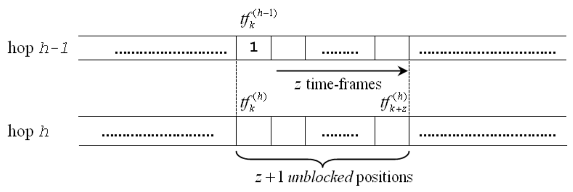

Fig. 8. Example of unblocked positions.

unblocked position, it becomes a schedulable TF, '$\texttt{1}_s$'. If $tf_{k-i}^{(h-1)}$='$\texttt{0}$' or $tf_{k-i}^{(h-1)}$ = '$\texttt{1}_b$' for all $i \in [z]$, then $tf_k^{(h)}$ is a *blocked* position. If an available TF is placed in a *blocked* position, it becomes blocked, '$\texttt{1}_b$'.

Let x be the number of *blocked* positions, and let y be the number of *unblocked* positions:

$$x + y = K \ . \tag{25}$$

Let also, at generic hop h:

- α_{h-1} be the number of schedulable TFs (i.e., symbols '$\texttt{1}_s$') at hop $h-1$;
- β_{h-1} be the number of *unblocked* positions generated by hop $h-1$ for hop h;
- α_h be the number schedulable TFs (i.e., symbols '$\texttt{1}_s$') hop h;
- $Pr(\alpha_h = \tilde{a})$ be the probability that $\alpha_h = \tilde{a}$;
- $Pr(\alpha_{h-1} = \hat{a})$ be the probability that $\alpha_{h-1} = \hat{a}$.

For the first hop (i.e., $h=0$), we have:

$$Pr(\alpha_0 = \tilde{a}) = \begin{cases} 1 & \text{if } \ \tilde{a} = a_0, \\ 0 & \text{otherwise.} \end{cases} \tag{26}$$

In general, at hop $h-1$, $\hat{a}$ is bounded by:

$$0 \leq \hat{a} \leq a \ . \tag{27}$$

Note that if α_{h-1}=0, then β_{h-1}=0. For α_{h-1}=$\hat{a}$> 0, on one extreme case, when all $\hat{a}$ schedulable TFs at hop $h-1$ form a unique run, then we obtain the minimum value of *unblocked* positions:

$$y_{min} = \min\{\hat{a} + z, K\}.$$

On the other extreme case, when each schedulable TF forms one run, and two consecutive schedulable TFs are split by an interval of at least z TFs, then we obtain the maximum value of *unblocked* positions:

$$y_{max} = \min\{(z+1)\hat{a}, K\}.$$

Thus at hop $h-1$, given $\alpha_{h-1} = \hat{a} > 0$, y (the number of *unblocked* positions) is bounded by:

$$\min\{\hat{a} + z, K\} \leq y \leq \min\{(z+1)\hat{a}, K\} \ . \tag{28}$$

An example of multi-hop time-blocking is shown in Fig. 9. In the example, $H=4$ and each switching node use 2-*forwarding* scheme. Each hop contains one channel, where $K=12$ TFs. In the example, until the third hop, there are two schedulable TFs. However, all available TFs of the fourth hop are blocked.

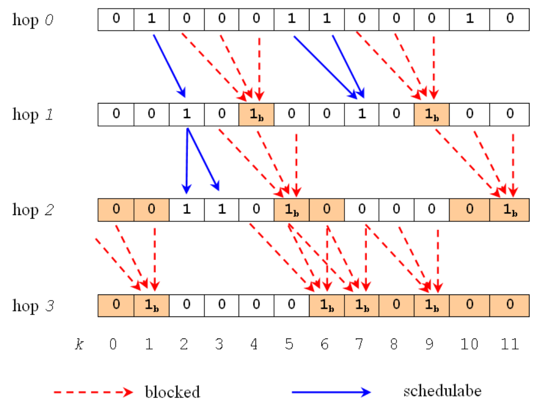

Fig. 9. Example where there is blocking after 4 hops.

B. *Exact Solution of the Multi-Hop Case*

Even if the multi-hop case is much more complex than the single node in isolation, the problem can still be formulated with an exact solution, that we sketch in the following, referring the reader interested in more details and numerical results to the Technical Report [8].

C. *Zero scheduling delay*

For this special case, we have the following:

Lemma *4:* For $z = 0$, the probability that there are $\tilde{a}$ schedulable TFs at hop h is given by:

$$Pr(\alpha_h = \tilde{a}) = \sum_{\hat{a}=0}^{a} \frac{\binom{\hat{a}}{\tilde{a}}\binom{K-\hat{a}}{a-\tilde{a}}}{\binom{K}{a}} \; Pr(\alpha_{h-1} = \hat{a}) \;. \tag{29}$$

Proof: Let $Pr(\alpha_h = \tilde{a}|\alpha_{h-1} = \hat{a})$ be the conditional probability that there are $\tilde{a}$ schedulable TFs at hop h, given $\hat{a}$ schedulable TFs at hop $h-1$.

An available TF of hop h becomes a schedulable TF if it is positioned "below" a schedulable TF of hop $h-1$. To generate exactly $\tilde{a}$ schedulable TFs for hop h ($\tilde{a} < \hat{a}$), $\tilde{a}$ available TFs are distributed "beneath" the $\hat{a}$ possible positions, obtaining the number $\binom{\hat{a}}{\tilde{a}}$.

The other $a - \tilde{a}$ available TFs must be placed under the $(K - \hat{a})$ busy TFs of hop $h-1$ so that no more schedulable TF is generated, yielding the number $\binom{K-\hat{a}}{a-\tilde{a}}$.

Meanwhile, without any constraint, the total number of ways to distribute a available TFs into K TF positions is $\binom{K}{a}$. Thus, we obtain:

$$Pr(\alpha_h = \tilde{a}|\alpha_{h-1} = \hat{a}) = \frac{\binom{\hat{a}}{\tilde{a}}\binom{K-\hat{a}}{a-\tilde{a}}}{\binom{K}{a}} \;. \tag{30}$$

Taking the sum over all possible $\hat{a}$ different from 0 we obtain (29). ■

With the initial condition of $Pr(\alpha_0 = \tilde{a})$ given in (26), (29) is used recursively to obtain the time-blocking probability for the IF scheme.

D. *Nonzero scheduling delay*

The analysis for zero scheduling delay is rather straightforward. The idea is that we can derive the quantity $Pr(\alpha_h = \tilde{a})$ using a hop-based computation. In this section, we apply this approach to analyze cases of nonzero scheduling delay schemes (i.e., $z \geq 1$).

Let us introduce two following conditional probabilities:

- $Pr(\beta_{h-1} = y|\alpha_{h-1} = \hat{a})$ is the probability that β_{h-1}=y, given that α_{h-1}=$\hat{a}$.
- $Pr(\alpha_h = \tilde{a}|\beta_{h-1} = y)$ is the probability that α_h=$\tilde{a}$, given that β_{h-1}=y.

It follows that the next two coupled equations can be recursively used to compute $Pr(\alpha_h = \tilde{a})$ for any hop $h \geq 1$:

$$Pr(\beta_{h-1} = y) = \sum_{\hat{a}=0}^{a} Pr(\beta_{h-1} = y|\alpha_{h-1} = \hat{a}) \;\; Pr(\alpha_{h-1} = \hat{a}) \;, \tag{31}$$

$$Pr(\alpha_h = \tilde{a}) = \sum_{y=0}^{\min\{(z+1)a,K\}} Pr(\alpha_h = \tilde{a}|\beta_{h-1} = y) \;\; Pr(\beta_{h-1} = y) \;. \tag{32}$$

We can obtain the time-blocking probability if we are able to compute two conditional probabilities $Pr(\alpha_h = \tilde{a}|\beta_{h-1} = y)$ and $Pr(\beta_{h-1} = y|\alpha_{h-1} = \hat{a})$. More specifically, the time-blocking probability after H hops is computed by (32) for $\alpha_{H-1} = 0$ where

$$Pr(\alpha_h = \tilde{a}|\beta_{h-1} = y) = \begin{cases} \frac{\binom{y}{\tilde{a}}\binom{K-y}{a-\tilde{a}}}{\binom{K}{a}} & \text{if } \tilde{a} \leq y \;\&\; a - \tilde{a} \leq K - y, \\ 0 & \text{otherwise.} \end{cases} \tag{33}$$

In order to have $\tilde{a}$ schedulable TFs at hop h, we distribute $\tilde{a}$ available TFs among y *unblocked* positions generated by hop $h-1$, which yields $\binom{y}{\tilde{a}}$. To block the other $(a - \tilde{a})$ available TFs, they must be arranged among $(K - y)$ *blocked* positions, which yields $\binom{K-y}{a-\tilde{a}}$. Meanwhile, without any constraint, the total number of dispositions is $\binom{K}{a}$. Thus, we derive $Pr(\alpha_h = \tilde{a}|\beta_{h-1} = y)$ as in (33).

Finding $Pr(\beta_{h-1} = y|\alpha_{h-1} = \hat{a})$ is more challenging. Increasing the number of hops, schedulable TFs tend to form "batches" rather than being uniformly distributed among K positions. The exact computation of $Pr(\beta_{h-1} = y|\alpha_{h-1} = \hat{a})$ depends not only on a certain *z-forwarding* scheme but also on the distribution of the position of the $\hat{a}$ schedulable TFs, which is no longer uniform.

In fact, an exact solution for time-blocking probability is possible if we are able to compute probabilities associated with all possible distribution of $\hat{a}$ schedulable TFs. We demonstrate the process to obtain the exact time-blocking probability by examining following example.

TABLE II
ALL POSSIBLE PATTERNS AND y VALUES FOR $K = 6, a = 2, z = 1$.

State	$\hat{a}$	Pattern	y
s_0	0	000000	0
s_2	1	100000	2
s_3	2	110000	3
s_{41}	2	101000	4
s_{42}	2	100100	4

1) An example for small K: We perform the exact solution for a set of small parameters: $K = 6, a = 2, z = 1$. Following (27) and (28) we have $0 \leq \hat{a} \leq 2$ and $y \in \{0, 2, 3, 4\}$.

For each possible $\hat{a}$, we consider all possible patterns taking into account the ordering of run-lengths of `0`'s and of `1`'s. For example, by allowing shifting the two following patterns are interchangeable: $101000 \equiv 100010$. All patterns and their corresponding $\hat{a}$ and y are given in Table II. The number of patterns is small. Transition probabilities between patterns can be easily computed as shown in Table III.

TABLE III
TRANSITION PROBABILITY BETWEEN PATTERNS.

	s_0	s_2	s_3	s_{41}	s_{42}
s_0	1	0	0	0	0
s_2	6/15	8/15	1/15	0	0
s_3	3/15	9/15	2/15	1/15	0
s_{41}	1/15	8/15	3/15	2/15	1/15
s_{42}	1/15	8/15	2/15	2/15	2/15

Let $\mathbf{s}_h$ be the vector associated with the probability that hop h is at the state s_*, $\mathbf{s}_h \doteq \langle Pr(s_*) \rangle$. Let $\mathbf{S}$ be the transition matrix given in Table III. We have $\mathbf{s}_0 \doteq \langle 0, 0, \frac{3}{15}, \frac{6}{15}, \frac{6}{15} \rangle$.

And the vector $\mathbf{s}_h$ is computed by:

$$\mathbf{s}_h = \mathbf{s}_0 \times \mathbf{S}^h \ .$$

The above equation is used to compute $\mathbf{s}_{H-2}$ then (32) is applied to compute the blocking probability.

Unfortunately this exact approach is computationally unfeasible as soon as K grows, which means for any meaningful real system. However, the existence of an exact solution is the starting point to obtain approximations and bounds. We are now working on deriving bounds that can be used for the design of TDS based networking systems. Early results, can be found in [8].

VI. CONCLUSIONS

TDS is a novel networking technique that is based on pipeline forwarding for performance optimization and for service guarantees. The technique is especially suited to support ultra-fast switching with extremely large capacities (multi-terabit/s). Under such conditions, however, buffering information at switches is difficult and costly. Consequently, in TDS there is associated with a novel *time-blocking* problem, arises when a switch cannot be scheduled to exploit available resources on both an input and an output.

This paper presented an exact combinatorial analysis of the time-blocking probability of a single switch. Results from this analysis allow the design of systems taking into account the tradeoffs between cost (the amount of buffering) and performance (time-blocking).

The extension of the exact combinatorial analysis to a network of switches, i.e., a sequence of h switches in series, was also proved feasible, but, due to correlations introduced by recursive buffering, requires the computation of a set of conditional probabilities that grows geometrically with the number of time-frames K, making its use unfeasible in any meaningful scenario. Yet, the availability of theoretical results enables further research to establish (hopefully tight) bounds to be used in networks design.

REFERENCES

[1] T. E. Parker, D. Matsakis, “Time and Frequency Dissemination: Advances in GPS Transfer Techniques,” *GPS World*, pp. 32-38, November 2004

[2] D. Agrawal, M. Baldi, M. Corrà, G. Fontana, G. Marchetto, V.T. Nguyen, Y. Ofek, D. Severina, T.H. Truong, O. Zadedyurina, “Scalable Switching Testbed not ’Stopping’ the Serial Bit Stream,” *Proc of IEEE ICC’07*, Glasgow, Scotland, June 24–28, 2007.

[3] A. Tucker, *Applied Combinatorics*, John Wiley & Sons, 1980.

[4] M. Baldi and Y. Ofek, “Fractional lambda switching,” *Proc. of the IEEE ICC*, vol. 5, pp. 2692–2696, 2002.

[5] D. Grieco, A. Pattavina, and Y. Ofek, “Fractional lambda switching for flexible bandwidth provisioning in WDM networks: principles and performance,” *Photonic Network Communications*, vol. 9, no. 3, pp. 281–296, 2005.

[6] V.T. Nguyen, R. LoCigno, and Y. Ofek, “Design and Analysis of Tunable Laser-based Fractional λ Switching (FλS),” *Proc. of the IEEE INFOCOM*, 2006. (An extended version is to appear in IEEE Tr. on Communications).

[7] V-T. Nguyen, R.Lo Cigno, Y. Ofek, “Time-Driven Switching Blocking Analysis: A Single Node Case,” *DISI Technical Report DIT-07-045 – Ver. 1.0*, University of Trento, 2006, URL:http://dit.unitn.it/locigno/preprints/DIT-07-045.pdf.

[8] V-T. Nguyen, R.Lo Cigno, Y. Ofek, M. Telek, “Multi-hop Time-blocking Anaysis for TDS,” *DISI Technical Report DIT-07-046 – Ver. 1.0*, University of Trento, 2007, URL:http://dit.unitn.it/locigno/preprints/DIT-07-046.pdf.