Performance analysis of non-uniform switches in networks on chips

Performance analysis of non-uniform switches in networks on chips

Paul Beekhuizen*† Jacques Resing‡

September 10, 2007

Abstract

Packet switches have been studied extensively as part of ATM and LAN networks under the assumption that the number of input ports N tends to infinity. In networks on chips, N is usually 4 or 5 and asymptotic models lead to inaccurate results. A new approximation of the mean sojourn time has recently been introduced for uniform switches (i.e., each input receives the same load and each destination is equally likely) that leads to much more accurate results for small values of N. Essentially, this approximation allows the switch to be seen as a $Geo/Geo/1$ queue, the discrete-time equivalent of the $M/M/1$ queue.

We devise an accurate approximation of a *non-uniform* switch as a $Geo/Geo/1$ queue. In particular we focus on approximations of throughput, stability conditions, and mean waiting times. The approximations of the stability conditions and throughput are generally accurate within 2% relative error of simulation results, while the mean waiting time approximation is generally within 15%.

1 Introduction and background

Due to the ever increasing complexity of chips, networks on chips have been proposed as a future interconnect of systems on chips [5]. During the design of networks on chips, simulation is often used as a means to analyse performance [12]. While simulation allows for quite a realistic and accurate performance analysis, its use in an optimisation loop is not desirable because it can be time-consuming. A queueing-theoretic analysis of the underlying packet switching network might provide a solution for this problem [11].

This paper is devoted to a queueing-theoretic performance analysis of a packet switch. These are often analysed under the assumption that traffic is uniform (i.e., each input receives the same load and each destination is equally likely) [1, 7]. In addition to this, it is often assumed that N, the number of ports, tends to infinity [7, 8]. In this paper, we devise an accurate approximation that is suitable for non-uniform switches with small N.

One of the most influential papers that study the behaviour of packet switches is the paper by Karol et al. [7]. In this paper the performance of input and output-queued switches is analysed and compared. It is shown that output-queued switches by nature have a much better performance than input-queued switches. Moreover, it is shown that the throughput of input-queued switches is limited to 58.6% due to Head-of-Line blocking. Output-queueing, on the other hand, has the disadvantage that a switch with N input and output ports must be able to either route N packets to the same output port or N packets from the same input port to different output ports in one time slot; the switch must operate N times faster than the links connected to it.

As a solution to these problems, switches with a 'speedup' are often studied (see, e.g., [3,4,13]). A switch with a speedup of s is able to route s packets from an input port or to an output port.

*EURANDOM, P.O. Box 513, 5600 MB, Eindhoven, The Netherlands, e-mail: beekhuizen@eurandom.tue.nl

†Philips Research, Digital Signal Processing group, High Tech Campus 36, 5656 AE, Eindhoven, The Netherlands

‡Eindhoven University of Technology, Department of Mathematics and Computer Science, P.O. Box 513, 5600 MB, Eindhoven, The Netherlands, e-mail: resing@win.tue.nl

An output-queued switch thus has a speedup of N, while an input-queued switch has a speedup of 1. If the speedup is between 1 and N, the switch must have queues at both the inputs and the outputs (Combined Input and Output Queuing, CIOQ).

The problems addressed in studies of CIOQ-switches differ fundamentally from the problems a single input-queued (SIQ) switch poses. For instance, a number of papers deal with the scheduling algorithms a switch uses [9,10,14]; each time slot the switch has to find a matching of input queues and output queues. In a SIQ-switch, the complexity of such a matching is reduced greatly because the switch may only select one packet per output port. Despite the better performance of CIOQ-switches, SIQ-switches are still a valid option for networks on chips because of their lower costs. In this paper, we therefore study a SIQ-switch.

In [7], a SIQ-switch is analysed under the assumption that $N \to \infty$. While this gives good approximations for large switches, it is not particularly useful for networks on chips, since switches in these networks often have only 4 or 5 ports.

An approximation specifically geared towards small uniform switches is devised in [1]. It is also shown that, for small values of N, this approximation outperforms the asymptotic analysis of Karol et al., and a different asymptotic analysis by Kim et al. [8]. The key assumption in the approximation of [1] is that each packet in the first position of the queue has to wait for a geometrically distributed number of packets from other queues with the same destination. The parameter of this geometric distribution can be approximated using an interpolation between 'light traffic' and saturation. The final result is that a single queue of the switch can be approximated by a $Geo/Geo/1$ queue, the discrete-time equivalent of the $M/M/1$ queue.

In this paper we approximate the performance of small non-uniform SIQ-switches. The approximation scheme is also based on the approximation of the switch as a $Geo/Geo/1$ queue and an interpolation between light traffic and saturation, but it is much more widely applicable. In addition to this, the problems encountered in this paper are entirely different from those in [1]. For instance, queues become unstable at different loads. We devise an approximation of stability conditions that allows us to overcome this problem. This in turn yields an approximation of the throughput of a certain queue, given that it is unstable. Finally, together with an independence assumption of the queues, we devise an approximation of the mean waiting time.

In summary, the purpose of our paper is the following: We approximate a non-uniform packet switch as a $Geo/Geo/1$ queue. In particular, we focus on the approximation of stability conditions, throughput of unstable queues, and mean waiting time.

The paper is organised as follows: In Section 2 we describe the model in more detail. Section 3 is devoted to the analysis of a saturated switch as a Markov chain. This analysis is used in an approximation of the stability conditions in Section 4 and an approximation of the throughput of unstable queues in Section 5. The derivation of the (mean) waiting time approximation is given in Section 6. We perform an in-depth numerical analysis of one example in Section 7, and a large-scale numerical analysis of 100 examples in Section 8. Finally, we draw conclusions and discuss our results in Section 9.

2 Model

We consider a single input-queued switch with N input ports and M output ports operating in discrete-time. Each packet in queue i has destination j with probability p_{ij}. Furthermore, each packet has size 1. We also assume that the buffers are infinitely large. A more schematic representation of the model can be found in Fig. 1.

If there are multiple packets with the same destination at the heads of the different queues, contention is said to occur. In this case only one of the contending packets is selected for transmission and the other packets have to try again in the next time slot. Moreover, each of k contending packets is selected with probability $1/k$ (random order), independent of the number of times a packet has already contended for its destination. Because of the similarities between switching and service in ordinary queueing models, we define the service time B_i of a packet in queue i as the time spent in the first position of the queue (*Head-of-Line*, or HoL-position). The service time is

thus equal to the number of successive attempts to reach its destination, including the successful one.

We assume that arrivals take place at the beginning of time slots and departures at the end. Furthermore, a packet arriving at an empty queue is eligible for service in the same time slot.

In this paper we study the situation in which the arrival processes to the different queues are independent Bernoulli processes, which means that in each time slot, independently of what happened in previous time slots, a packet arrives at queue i with probability λ_i. Part of our analysis, however, only depends on the arrival rates and not on the specific process. In Section 9 the dependency on the assumption of Bernoulli arrivals will be discussed in somewhat more detail.

Under the assumption of Bernoulli arrivals, the switch can be modelled as a Markov chain $(Q_1(t), D_1(t), \ldots, Q_N(t), D_N(t))$. Here, $Q_i(t)$ is the number of packets in queue i at time t, and $D_i(t)$ is the destination of the packet in the HoL-position of queue i, or zero if $Q_i(t) = 0$. Due to the intricate nature of the contention that occurs, we are not able to analyse this Markov chain with infinite state space exactly. A work-around would be to numerically analyse a model with finite buffers (and hence with truncated state space), but even for moderate buffer sizes such a state space would already become prohibitively large.

If, however, we assume that every served packet is immediately replaced by a new one, we arrive at a situation that is commonly known as 'saturation'. One could think of saturation as an overload situation, and it could, for instance, be achieved by setting $\lambda_i = 1$. In saturation the process $(D_1(t), \ldots, D_N(t))$ itself constitutes a Markov chain with a finite state space that can be analysed. The results of this analysis are used to devise an approximation of stability conditions, throughput, and service rates in the non-saturated model. In addition to this, it turns out to be possible to analyse the behaviour of the non-saturated model under very low loads (*light traffic* approximation). The results of saturation and light traffic are combined and in the end yield a mean waiting time approximation for *moderate* traffic. In particular, we will focus on throughput and mean waiting times. Part of our analysis is devoted to the mean service time, which is an essential ingredient in an approximation of the mean waiting time.

3 Saturated switch

In this section we study a switch under saturation. In this case, the process $(D_1(t), \ldots, D_N(t))$ is a Markov chain on the state space $\Omega := \{1, \ldots, M\}^N$. The switch is said to be in state $x = (x_1, \ldots, x_N)$ if the packet at queue i has destination x_i.

If we assume that $p_{ij} > 0$ for all i and j, the Markov chain is aperiodic and irreducible, and hence a unique steady-state distribution exists. Also if $p_{ij} = 0$ for some i and j, the Markov chain is still aperiodic and irreducible, but now on a reduced state space. This reduced state space is obtained by removing, for all i and j for which $p_{ij} = 0$, the states x for which $x_i = j$ from the state space Ω.

The transition probabilities $P(x, y)$ of the Markov chain can be determined straightforwardly. In order to do so, we assume that a state transition consists of two steps: the service completions in the first step and the replacements of served packets by new packets in the second step. Furthermore we introduce artificial states that describe the switch immediately after service completions. These states may contain zeros which indicate that a packet at a certain input port has just

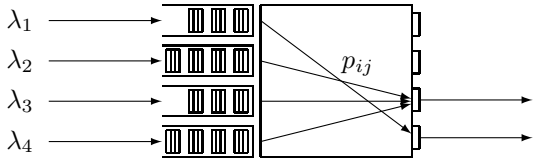

Figure 1: A schematic representation of the model of a SIQ-switch. Packets arrive at rate λ_i to queue i of the switch, and they have destination j with probability p_{ij}.

been served. For instance from state $(1,1,3,3)$ we can go to artificial states $(0,1,0,3)$, $(0,1,3,0)$, $(1,0,0,3)$, or $(1,0,3,0)$, each with probability $1/4$. This is a consequence of the fact that one of the contending packets is selected at random. The second step amounts to replacing a 0 at position i to a j with probability p_{ij}. For instance, from artificial state $(0,1,0,3)$, we go to state $(2,1,3,3)$ with probability $p_{12}p_{33}$. The probability of a transition from $(1,1,3,3)$ to $(2,1,3,3)$ via $(0,1,0,3)$ is thus $p_{12}p_{33}/4$. In addition to this, we also go from $(1,1,3,3)$ to $(2,1,3,3)$ via $(0,1,3,0)$ with probability $p_{12}p_{43}/4$. The sum of these two probabilities is the total probability of going from $(1,1,3,3)$ to $(2,1,3,3)$.

In a more formal setting, we define $A(x)$ to be the set of artificial states reachable from x. Note that each $a \in A(x)$ represents the state of the switch immediately after service completions. Because each service possibility is equally likely, we get that

$$P(x,y) = \frac{1}{|A(x)|} \sum_{a \in A(x)} \prod_{i=1}^{N} \Big(1(a_i = 0) p_{i,y_i} + 1(a_i = y_i) \Big).$$

From $P(x,y)$ we can determine the steady-state distribution $\pi(\cdot)$ of the Markov chain by using the fact that $\pi(\cdot)$ is the unique normalized solution of the system of equations

$$\pi(y) = \sum_{x \in \Omega} \pi(x) P(x,y), \quad y \in \Omega.$$

For our purposes, the most important performance measure of the Markov chain is the steady-state throughput. The throughput of queue i in state x is given by

$$\gamma_i(x) = \left(\sum_{k=1}^{N} 1(x_k = x_i) \right)^{-1}.$$

Here, $\sum_{k=1}^{N} 1(x_k = x_i)$ is the number of packets that have the same destination as the packet at queue i. The probability that the packet in queue i is selected for transmission is given by $\gamma_i(x)$ due to the random order of service. The steady-state throughput of queue i is given by

$$\gamma_i = \sum_{x \in \Omega} \pi(x) \gamma_i(x). \tag{3.1}$$

Because in a saturated switch queues are always non-empty, γ_i can also be interpreted as the steady-state service rate at queue i (i.e., $1/\gamma_i$ is the mean time a packet is in the HoL-position at queue i).

4 Stability conditions

In this section we devise a heuristic that approximates the stability conditions for each queue in a non-saturated switch. Here, we define a queue to be stable if its throughput is equal to its arrival rate and unstable if its throughput is less than its arrival rate. In the derivation of this heuristic, we make use of an idea of Ibe and Cheng [6] who devise a heuristic approach to determine the stability condition of k-limited polling systems with general input, service times and switchover times. Their results were later rigorously proved by Chang and Lam [2], but only for these polling models.

The idea of this approach is the following: During a period $[-T, 0)$, work arrives to a server. After this period, new arrivals are blocked and the work present is processed. A queue is stable if and only if it is able to process the amount of work before an additional period of T time units has elapsed, that is, if it becomes empty before time T. Here, we apply this idea to an arbitrary $N \times M$ switch. However, instead of letting packets arrive, we consider a deterministic fluid approximation where fluid enters (leaves) the buffer at the same rate at which packets would arrive (depart).

At time 0 all queues will be occupied (under the assumption of positive arrival rates), leading to a saturated $N \times M$ switch as analysed in Section 3. We can thus compute the steady-state service rates belonging to this $N \times M$ switch. All queues are then drained at the corresponding rates until the first queue becomes empty. The time at which this happens follows from an easy computation. Since we block new arrivals, this queue remains empty throughout the remainder of the procedure and can therefore be neglected. Moreover, since the other queues still have work present, they constitute a saturated $(N-1) \times M$ switch. This again allows us to determine the steady-state service rates and compute the time at which the next queue becomes empty, and so on.

Like Ibe and Cheng, we consider a queue to be stable if and only if it is empty at time T. However, while their results were rigorously proved later, we have to conclude (see Section 7 and 8) that the approach only yields an approximation of the true stability condition, albeit a very accurate one.

For clarity, we introduce notation before we describe the algorithm more accurately. We define t_n as the time at which the nth queue becomes empty. We let $q_i^{(n)}$ denote the fluid level of queue i at the beginning of step n of the procedure, i.e., at time t_{n-1}. We further denote the service rate of queue i in step n by $\gamma_i^{(n)}$ and e_n as the index of the queue that is the nth to become empty.

Algorithm 4.1.

Initialise $q_i^{(1)} = \lambda_i T$, and let $t_0 = 0$. For $n = 1, \ldots, N$:

- Calculate service rates $\gamma_i^{(n)}$ of queue i in an $(N-n+1) \times M$ saturated switch, constructed by removing queues $e_1, \ldots, e_{n-1}$ from the original switch.
- With $i \neq e_1, \ldots, e_{n-1}$, let

$$e_n = \arg\min_i \left\{ \frac{q_i^{(n)}}{\gamma_i^{(n)}} \right\}, \tag{4.1a}$$

since $q_i^{(n)}/\gamma_i^{(n)}$ is the time it would take to empty queue i if it was allowed to serve packets at rate $\gamma_i^{(n)}$ indefinitely. The time at which queue e_n becomes empty is given by

$$t_n = t_{n-1} + \frac{q_{e_n}^{(n)}}{\gamma_{e_n}^{(n)}} = t_{n-1} + \min_i \left\{ \frac{q_i^{(n)}}{\gamma_i^{(n)}} \right\}, \tag{4.1b}$$

and the remaining fluid in the other queues by

$$q_i^{(n+1)} = q_i^{(n)} - (t_n - t_{n-1})\gamma_i^{(n)}. \tag{4.1c}$$

Remark 4.2. If there are multiple i (say $i = i_1, \ldots, i_k$) for which $q_i^{(n)}/\gamma_i^{(n)}$ is minimal, we can arbitrarily choose one (say i_k). For $i = i_1, \ldots, i_{k-1}$, $q_i^{(n+1)} = 0$, so that $t_{n+k-1} = t_{n+k-2} = \ldots = t_n$.

A schematic representation of the fluid levels of all queues during Algorithm 4.1 can be found in Figure 2.

Lemma 4.3. *Define $\Delta t_n = t_n - t_{n-1}$ for $n \geq 1$ (recall that $t_0 = 0$). We can write Δt_n as follows:*

$$\Delta t_n = T \sum_{m=1}^{n} d_{m,n} \lambda_{e_m}, \tag{4.2}$$

where the constants $d_{m,n}$ are recursively determined by

$$d_{m,n} = \begin{cases} -\sum_{k=m}^{n-1} \frac{\gamma_{e_n}^{(k)}}{\gamma_{e_n}^{(n)}} d_{m,k}, & \text{for } m = 1, \ldots, n-1, \\ 1/\gamma_{e_n}^{(n)}, & \text{for } m = n. \end{cases}$$

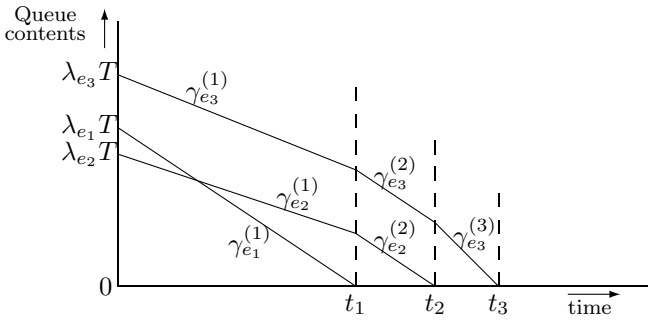

Figure 2: A schematic representation of the queue contents during the stability procedure: All queues start with a fluid level of $\lambda_i T$ and are served at rate $\gamma_i^{(1)}$ until the first queue becomes empty at time t_1. The two remaining queues are served at rates $\gamma_i^{(2)}$ until t_2, which is when e_2 becomes empty. Finally, the last queue is served at rate $\gamma_i^{(3)}$ until time t_3.

Proof. We prove this statement by induction on n. From Eqs. (4.1a) and (4.1b) with $n = 1$ it follows that indeed $\Delta t_1 = T\lambda_{e_1}/\gamma_{e_1}^{(1)}$. Suppose that the statement is true for $1, \ldots, n-1$. From (4.1c) it follows that

$$q_{e_n}^{(n)} = \lambda_{e_n} T - \sum_{k=1}^{n-1} \gamma_{e_n}^{(k)} \Delta t_k.$$

We get

$$\begin{aligned}
\Delta t_n &= \frac{q_{e_n}^{(n)}}{\gamma_{e_n}^{(n)}} = \frac{\lambda_{e_n} T}{\gamma_{e_n}^{(n)}} - \sum_{k=1}^{n-1} \frac{\gamma_{e_n}^{(k)}}{\gamma_{e_n}^{(n)}} \Delta t_k \\
&= T \frac{\lambda_{e_n}}{\gamma_{e_n}^{(n)}} - T \sum_{k=1}^{n-1} \sum_{m=1}^{k} \frac{\gamma_{e_n}^{(k)}}{\gamma_{e_n}^{(n)}} d_{m,k} \lambda_{e_m} \qquad \text{(induction hyp.)} \\
&= T \frac{\lambda_{e_n}}{\gamma_{e_n}^{(n)}} - T \sum_{m=1}^{n-1} \lambda_{e_m} \sum_{k=m}^{n-1} \frac{\gamma_{e_n}^{(k)}}{\gamma_{e_n}^{(n)}} d_{m,k} = T \sum_{m=1}^{n} d_{m,n} \lambda_{e_m}.
\end{aligned}$$

□

Corollary 4.4. *The time at which queue e_n becomes empty is given by:*

$$t_n = \sum_{k=1}^{n} \Delta t_k = T \sum_{k=1}^{n} \sum_{m=1}^{k} d_{m,k} \lambda_{e_m}. \tag{4.3}$$

Approximation 4.5. *We approximate the stability condition of queue e_n by $t_n < T$, or equivalently:*

$$\sum_{k=1}^{n} \sum_{m=1}^{k} d_{m,k} \lambda_{e_m} < 1. \tag{4.4}$$

Note that the stability condition derived in this section is an approximation due to several reasons. First, we look at a process in which we block the arrivals after time 0, while in reality new packets keep on arriving (and interfere with packets that arrived before time 0). Second, we look at a deterministic fluid process instead of a stochastic process in which packets arrive and depart. And third, we assume that if one of the queues becomes empty, the service rate of the other queues instantaneously becomes the steady-state service rate of a saturated switch with one input port less. A numerical evaluation of the accuracy of the Approximation 4.5 can be found in Section 7 and 8, where we compare the stability condition with results obtained from a simulation. It will be shown that the relative error of the approximation is generally within 2%.

5 Throughput approximation

In this section, we study the throughput of each queue of the switch as a function of a single load parameter λ. We divide this load according to a predetermined vector $(\nu_1, \ldots, \nu_N)$, with $\sum_i \nu_i = 1$, such that $\lambda_i = \nu_i \lambda$.

To illustrate how the throughput depends on λ, we show some simulation measurements of the throughput per queue in Figure 3. The input of this figure is precisely the example that will be studied in Section 7. This example will be used as a running example throughout this and the next section, as the characteristics displayed in this plot are typical for non-uniform switches. In the example

$$P = (p_{ij}) = \begin{pmatrix} 0.1 & 0.3 & 0.4 & 0.2 \\ 0.2 & 0.2 & 0.2 & 0.4 \\ 0.2 & 0.3 & 0.4 & 0.1 \\ 0.3 & 0.3 & 0.2 & 0.2 \end{pmatrix},$$

and ν is given by

$$\nu = \begin{pmatrix} 0.35 & 0.3 & 0.2 & 0.15 \end{pmatrix}.$$

Furthermore, the arrivals in this example are governed by Bernoulli processes with parameter $\min\{1, \lambda\nu_i\}$, i.e., each time slot an arrival takes place at queue i with probability $\min\{1, \lambda\nu_i\}$. Here we take $\min\{1, \lambda\nu_i\}$ since $\nu_i\lambda > 1$ if λ is increased far enough. Bernoulli processes are of course not defined for parameters greater than 1, so we have to limit the parameter to 1. Note that this does not have any influence since at most one packet per queue may be served each time slot anyway. The queue is thus saturated and its steady state behaviour no longer depends on its load.

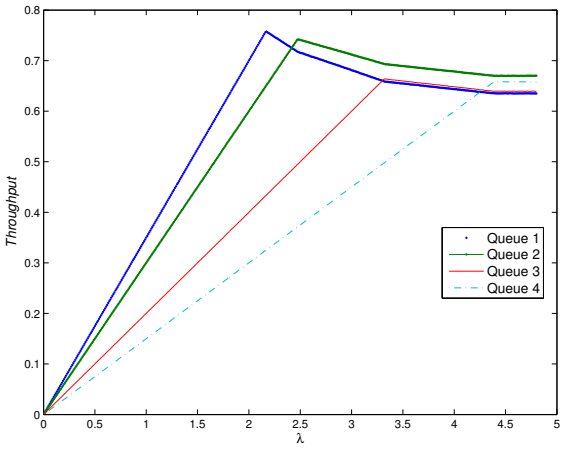

Figure 3: A typical plot of the throughput of the queues.

We see that for each queue the throughput increases linearly in λ up to a certain load. This is caused by the fact that the throughput is initially equal to the arrival rate $\lambda_i = \lambda\nu_i$. The throughput reaches its peak at the load for which it becomes unstable. Beyond this load it decreases, roughly linearly, due to the fact that the load at other, still stable, queues increases (and as a consequence also the throughput of these queues increases). It does so until the next queue becomes unstable, after which it keeps decreasing roughly linearly, but with a different slope. Finally, for loads that are high enough, we see that the throughput is constant. In this case, the load is so high that the switch has become saturated.

We are particularly interested in the region where the throughput is decreasing, which corresponds to the situation where some queues are stable while others are not. It turns out that

the throughput in this region can be approximated using the average amount of fluid that has disappeared from a queue during $[0, T)$ in the process described in Algorithm 4.1. We will describe this approximation in more detail now. A numerical evaluation of its accuracy can be found in Section 7.

We define $\lambda_{sat,i}$ as the value of λ for which queue i becomes unstable in Approximation 4.5. It will be convenient to have a labelling of the queues such that they become unstable in increasing order, i.e., $\lambda_{sat,1} \leq \ldots \leq \lambda_{sat,N}$. By the nature of Algorithm 4.1, the ith queue to become empty is the $(N+1-i)$th queue to become unstable, so $e_1 = N$, $e_2 = N-1$, etc. As a result, we find

$$\lambda_{sat,i} = \lambda_{sat,e_{N+1-i}} = \left(\sum_{k=1}^{N+1-i} \sum_{m=1}^{k} d_{m,k} \nu_{N-m+1} \right)^{-1}. \tag{5.1}$$

Define

$$l_\lambda = \max\{n : \lambda < \lambda_{sat,n}\},$$

where $\max \emptyset := 0$. That is, l_λ is the number of queues that are stable with respect to Approximation 4.5. The amount of fluid that is drained from queue i in $[0, T)$ during the process described in Algorithm 4.1 is given by

$$\sum_{n=1}^{l_\lambda} \gamma_i^{(n)} \Delta t_n + (T - t_{l_\lambda}) \gamma_i^{(l_\lambda+1)}, \tag{5.2}$$

where $\gamma_i^{(n)} = 0$ if $i \in \{e_1, \ldots, e_{n-1}\}$, i.e., the service rate of queue i is 0 after it has become empty in Algorithm 4.1. The rationale behind this equation is that queue i is drained at rate $\gamma_i^{(n)}$ in $[t_{n-1}, t_n)$, as long as $t_n < T$. After time t_{l_λ}, the queue is drained at rate $\gamma_i^{(l_\lambda+1)}$ until time T.

We use the average rate at which fluid is drained as an approximation of the throughput. That is, we divide the expression in (5.2) by T to obtain the throughput approximation of queue i, ϕ_i:

$$\phi_i(\lambda) = \frac{1}{T} \left[\sum_{n=1}^{l_\lambda} \gamma_i^{(n)} \Delta t_n + (T - t_{l_\lambda}) \gamma_i^{(l_\lambda+1)} \right] \tag{5.3a}$$

$$= \frac{1}{T} \left[\sum_{n=1}^{l_\lambda} \gamma_i^{(n)} \Delta t_n - \gamma_i^{(l_\lambda+1)} \sum_{n=1}^{l_\lambda} \Delta t_n \right] + \gamma_i^{(l_\lambda+1)}$$

$$= \lambda \sum_{n=1}^{l_\lambda} (\gamma_i^{(n)} - \gamma_i^{(l_\lambda+1)}) \sum_{m=1}^{n} d_{m,n} \nu_{N-m+1} + \gamma_i^{(l_\lambda+1)}, \tag{5.3b}$$

by Eq. (4.2). Note that ϕ_i is a piecewise linear function. Its slope changes at $\lambda = \lambda_{sat,k}$, since those are the values for which l_λ changes. It can be verified that the above expression indeed gives

$$\phi_i(\lambda) = \lambda \nu_i, \qquad \text{for } \lambda \leq \lambda_{sat,i},$$

i.e., the throughput of a stable queue is given by its arrival rate. Furthermore, if $\lambda > \lambda_{sat,N}$ then $l_\lambda = 0$, in which case Eq. (5.3a) directly implies that

$$\phi_i(\lambda) = \gamma_i^{(1)}, \qquad \text{for } \lambda \geq \lambda_{sat,N},$$

where $\gamma_i^{(1)}$ is the service rate of the saturated $N \times M$ switch, as studied in Section 3.

Remark 5.1. An interesting observation can be made from Eq. (5.3b): Suppose that $\lambda_{sat,k} \leq \lambda < \lambda_{sat,k+1}$, for a certain k. This means that there are k unstable queues, so $l_\lambda = N - k$. For unstable queues i, the term that is constant with respect to λ in $\phi_i(\lambda)$ is given by $\gamma_i^{(N-k+1)}$. The latter quantity is the service rate in a switch without queues $e_1, \ldots, e_{N-k}$, i.e., in a switch *with* queues $1, \ldots, k$. In other words, the constant term of $\phi_i(\lambda)$ in a region where queues $1, \ldots, k$ are unstable, is precisely equal to the service rate of queue i in a saturated switch consisting of only queues $1, \ldots, k$.

Lemma 5.2. *If $\lambda_{sat,i} \leq \lambda < \lambda_{sat,i+1}$ for a certain $i = 1, \ldots, N$ (with $\lambda_{sat,N+1} := \infty$), $\phi_i(\lambda)$ can be rewritten to the following form:*

$$\phi_i(\lambda) = \gamma_i^{(N-i+1)} + \lambda \left(\nu_i - \frac{\gamma_i^{(N-i+1)}}{\lambda_{sat,i}} \right). \tag{5.4}$$

Proof. By Remark 5.1 we know that $l_\lambda = N - i$. From (5.3b) it thus follows that $\phi_i(\lambda) = \gamma_i^{(N-i+1)} + c_i\lambda$ for some constant c_i. Furthermore, we know that $\phi_i(\lambda_{sat,i}) = \nu_i \lambda_{sat,i}$. Substituting $\lambda = \lambda_{sat,i}$ in the first expression and equating the two expressions for $\phi_i(\lambda_{sat,i})$ gives $\nu_i\lambda_{sat,i} = \gamma_i^{(N-i+1)} + c_i\lambda_{sat,i}$. Solving this equation with respect to c_i yields $c_i = \nu_i - \frac{\gamma_i^{(N-i+1)}}{\lambda_{sat,i}}$. □

Note that if $i = N$, this expression reduces to $\phi_N(\lambda) = \gamma_N^{(1)} = \gamma_N$ since $\nu_N \lambda_{sat,N} = \gamma_N$. Here γ_N is the service rate of queue N in the saturated switch consisting of all queues, as analysed in Section 3.

A schematic plot of ϕ_i for a 3×3 switch can be found in Figure 4. Clearly, it has roughly the same characteristics as Figure 3. A more detailed analysis of the accuracy of ϕ_i is conducted in Section 7 and 8.

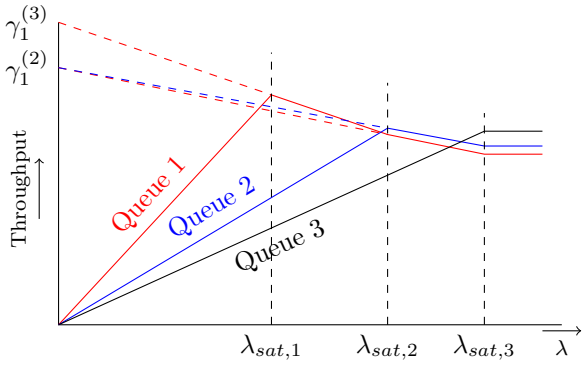

Figure 4: A schematic representation of the throughput approximation $\phi_i(\lambda)$ in a 3×3 switch. If $\lambda \leq \lambda_{sat,i}$, then $\phi_i(\lambda) = \nu_i\lambda$. If $\lambda_{sat,k} \leq \lambda < \lambda_{sat,k+1}$ and $i \leq k$, ϕ_i is also a linear function, whose y-intercept is given by the service rate of queue i in a saturated switch consisting of only the unstable queues $1, \ldots, k$. It can easily be argued that $\gamma_1^{(3)} = 1$ and $\gamma_1^{(2)} = \gamma_2^{(2)}$.

Remark 5.3. In case a uniform switch is considered, i.e., if $\nu = (1/N, \ldots, 1/N)$, and $p_{ij} = 1/N$ for all i and j, our approximation is simplified significantly: The service rates in saturation are the same for all queues (say γ). All queues become saturated at the same load, and the stability condition is given by $\lambda < N\gamma$. Moreover, if $\lambda \geq N\gamma$, the throughput is exactly equal to $\phi_i(\lambda) = \gamma$ because the switch is saturated. Except from notational differences, this is identical to what was found in [1].

6 Waiting time approximation

As stated in the introduction, the waiting time of a uniform switch was successfully approximated by a *Geo*/*Geo*/1 queue in [1]. The arrivals to this switch were assumed to be governed by independent Bernoulli arrival processes with parameter p, and the service time distribution in the switch was approximated by a geometric distribution. The rationale behind this assumption is that if there are k packets with the same destination, one of them is selected with probability $1/k$ independently of what happened in previous time slots. This induces a certain 'lack of memory', which makes the geometric distribution a suitable approximation. The service rate q could be

found in light traffic and saturation, and a quadratic interpolation was proposed between these two values. This led to an approximation $\widehat{q}$ as a function of p (denoted by $\widehat{q}(p)$).

Using known results from the $Geo/Geo/1$ queue (see e.g., Takagi [15]), we get that for a $Geo(p)/Geo(\widehat{q}(p))/1$ queue

$$\mathbb{E}[W] = \frac{p(1-\widehat{q}(p))}{\widehat{q}(p)(\widehat{q}(p)-p)},$$

where W is the steady-state waiting time.

In this section, we will perform a similar approximation. In the analysis of this section it is assumed that arrivals are governed by Bernoulli processes with parameter $\min\{1, \lambda\nu_i\}$. Under the approximation assumption that $B_i \sim \text{Geo}(\mu_i(\lambda))$, given a load of λ, queue i is a $Geo/Geo/1$ queue, with steady-state mean waiting time

$$\mathbb{E}[W_i] = \frac{\lambda\nu_i(1-\mu_i(\lambda))}{\mu_i(\lambda)(\mu_i(\lambda)-\lambda\nu_i)}, \qquad \text{for } \lambda\nu_i < \mu_i(\lambda). \tag{6.1}$$

The nature of our service rate approximation $\mu_i(\lambda)$, however, differs from that in [1]: In our case, queues become unstable for different loads. In order to understand the behaviour of the service rate better, we first show some simulation results in Figure 5. The input of this figure is again the 'running example' (see P. 7).

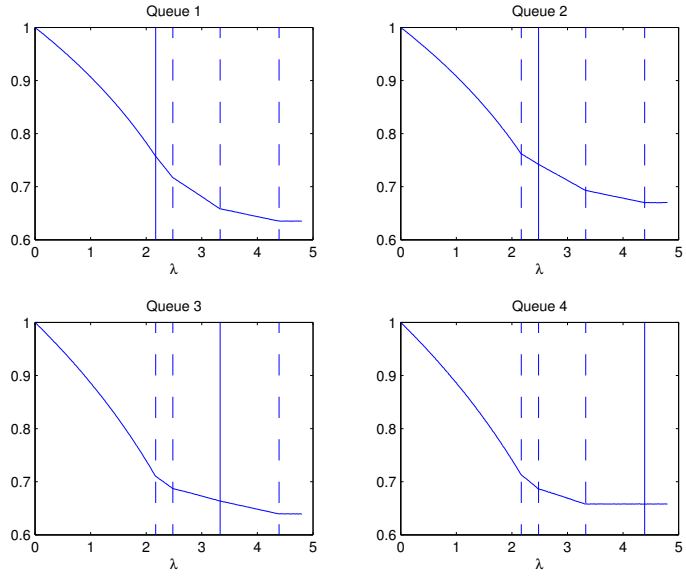

Figure 5: The service rates of all queues of the 'running example'. The vertical lines correspond to the observed saturation loads. In the figure of queue i, the saturation load of queue i itself is indicated by the straight line, while the saturation loads of the other queues are indicated by dashed lines.

We see that up to the first saturation load, the service rate shows a certain curvature. As in [1], we propose a quadratic interpolation for this part. Secondly, we see that the service rates appear to be piecewise linear between the various saturation loads. We propose a linear approximation for these parts. Finally, we observe that if all queues are saturated, the service rates are equal to the service rates of a saturated switch, as analysed in Section 3 and denoted by γ_i.

Recall that the approximated saturation load of queue i is denoted by $\lambda_{sat,i}$. The proposed

service rate approximation can be summarised as follows:

$$\mu_i(\lambda) = \begin{cases} a_i + b_i\lambda + c_i\lambda^2, & \text{for } \lambda < \lambda_{sat,1}, \quad (6.2) \\ \mu_i(\lambda_{sat,k}) + \dfrac{\mu_i(\lambda_{sat,k+1}) - \mu_i(\lambda_{sat,k})}{\lambda_{sat,k+1} - \lambda_{sat,k}}(\lambda - \lambda_{sat,k}), & \text{for } \lambda_{sat,k} \leq \lambda < \lambda_{sat,k+1}, \quad (6.3) \\ \gamma_i, & \text{for } \lambda \geq \lambda_{sat,N}. \quad (6.4) \end{cases}$$

Note that a_i, b_i, c_i, and $\mu_i(\lambda_{sat,k})$, $i = 1, \ldots, N$, $k = 1, \ldots, N$ are still unknown. The precise values of a_i and b_i as well as an expression of c_i in terms of $\mu_i(\lambda_{sat,1})$ follow from a light traffic analysis in Section 6.1. In Section 6.2 we focus on the values of $\mu_i(\lambda_{sat,k})$, $i = 1, \ldots, N$, $k = 1, \ldots, N$.

6.1 Light traffic

In this section we devise an approximation of the service rate for $\lambda < \lambda_{sat,1}$. The approximation proposed here will be a quadratic interpolation between the light traffic service rate and the service rate at $\lambda = \lambda_{sat,1}$.We study the behaviour of the switch as $\lambda \downarrow 0$, while neglecting $\mathcal{O}(\lambda^2)$ terms.

Consider an arbitrary time slot t and an arbitrary (tagged) packet arriving at queue i. Suppose that the tagged packet arrives at a non-empty switch. Because there is at least one packet present from slot $t-1$, there must have been at least two packets present in that time slot. This in turn implies that at some point in time there must have been two simultaneous arrivals. Since this happens with a probability of $\mathcal{O}(\lambda^2)$, we may ignore the situation in which a packet arrives at a non-empty switch.

So we consider a tagged packet arriving at an empty switch. Suppose that it has destination j. The probability that it suffers from contention is equal to the probability that another packet with destination j arrives in the same time slot. There is another arrival at queue $k \neq i$ with destination j with probability $\lambda_k p_{k,j} = \lambda \nu_k p_{k,j}$. The probability that there is another arrival at any of these queues is given by

$$\mathbb{P}(\text{contention}|\text{arrival at } i \text{ with dest. } j) = \lambda \sum_{k \neq i} \nu_k p_{k,j} + \mathcal{O}(\lambda^2).$$

The probability that there is contention is thus

$$\mathbb{P}(\text{contention}|\text{arrival at } i) = \beta_i \lambda + \mathcal{O}(\lambda^2), \quad (6.5)$$

where

$$\beta_i = \sum_{j=1}^{N} p_{i,j} \sum_{k \neq i} \nu_k p_{k,j}.$$

Because we ignore events that happen with probability $\mathcal{O}(\lambda^2)$, we know that if there is contention, the tagged packet is selected for service with probability $1/2$. As a result, the tagged packet is always served in the time slot in which it arrives, except if it loses contention:

$$\mathbb{P}(B_i = 1) = 1 - \frac{1}{2}\beta_i \lambda + \mathcal{O}(\lambda^2).$$

Suppose that the tagged packet loses the contention. In this case, it has to wait until time slot $t+1$, in which new arrivals potentially take place. If there are no arrivals in slot $t+1$, the tagged packet is served, which leads to a service time of 2. If there is another arrival in this time slot, this induces another factor λ, which implies that $\mathbb{P}(B_i = k)$ is $\mathcal{O}(\lambda^2)$ for $k > 2$. We conclude

$$B_i = \begin{cases} 1 & \text{w.p. } 1 - \frac{1}{2}\beta_i \lambda + \mathcal{O}(\lambda^2), \\ 2 & \text{w.p. } \frac{1}{2}\beta_i \lambda + \mathcal{O}(\lambda^2), \end{cases} \quad (6.6)$$

and consequently

$$\frac{1}{\mathbb{E}[B_i]} = 1 - \frac{1}{2}\beta_i\lambda + \mathcal{O}(\lambda^2).$$

We demand that our service rate approximation is exact in light traffic and continuous at $\lambda_{sat,1}$, i.e.,

$$\mu_i(\lambda) = 1 - \frac{1}{2}\beta_i\lambda + \mathcal{O}(\lambda^2), \qquad \text{and}$$
$$\mu_i(\lambda) \to \mu_i(\lambda_{sat,1}), \qquad \text{as } \lambda \uparrow \lambda_{sat,1}.$$

Note that in the latter condition $\mu_i(\lambda_{sat,1})$ will only be defined in Section 6.2. It is easily verified by substitution that the only quadratic function that satisfies these conditions is given by

$$\mu_i(\lambda) = 1 - \frac{1}{2}\beta_i\lambda + \frac{-1 + \frac{1}{2}\beta_i\lambda_{sat,1} + \mu_i(\lambda_{sat,1})}{\lambda_{sat,1}^2}\lambda^2, \tag{6.7}$$

for $\lambda < \lambda_{sat,1}$. This expression provides us with values of a_i, and b_i, and an expression for c_i in terms of $\mu_i(\lambda_{sat,1})$.

6.2 Saturation loads

Now that we have determined the quadratic part of our approximation, we give an algorithm that can be used to find the service rate approximation $\mu_i(\lambda_{sat,k})$, for $i = 1, \ldots, N$, $k = 1, \ldots, N$. We first present the approximation in a concise form, and then we derive the underlying equations. For the description of the approximation we introduce the following notation: For a set $J \subseteq \{1, \ldots, N\}$, γ_i^J will be the service rate of queue i in a saturated switch consisting precisely of queues from J.

Approximation 6.1. *For all k:*

If $i \leq k$,

$$\mu_i(\lambda_{sat,k}) = \phi_i(\lambda_{sat,k}). \tag{6.8}$$

If $i = k + 1$,

$$\mu_i(\lambda_{sat,k}) = \nu_i\lambda_{sat,k} + \left(1 - \frac{\lambda_{sat,k}}{\lambda_{sat,k+1}}\right)\gamma_i^{\{1,\ldots,i\}}. \tag{6.9}$$

If $i \geq k + 2$, $\mu_i(\lambda_{sat,k}) = 1/b_i$, where b_i is a solution of the following set of equations:

$$b_i = \sum_{J: i \in J} \frac{1}{\gamma_i^J} \prod_{\substack{j \in J \\ j \neq i}} \rho_j \prod_{j \notin J} (1 - \rho_j), \tag{6.10}$$

where

$$\rho_j = \begin{cases} 1 & \text{for } j \leq k, \\ \nu_j\lambda_{sat,k}/\mu_j(\lambda_{sat,k}), & \text{for } j = k + 1, \\ \nu_j b_j \lambda_{sat,k}, & \text{for } j > k + 1. \end{cases}$$

Remark 6.2. For general $N \geq 4$ we cannot guarantee that a unique solution to the set of equations of (6.10) exists. However, in all our numerical examples we found precisely one solution between 1 and N. Here, the maximal value is N since in the worst case N packets have the same destination and one of them is selected for service at random. For $N = 4$ there are two linear equations in two unknowns. We can show that the existence of a unique solution is a weaker statement than $I \subseteq J \Rightarrow \gamma_i^I \geq \gamma_i^J$, for all I and J, i.e., if we remove a queue from the switch, the throughput of the remaining queues increases. Although this statement is intuitively very plausible, we cannot prove it. If $N \geq 5$, we have $N - 2$ equations involving cross products of variables, for which we cannot prove the existence of a unique solution.

The rationale behind Equation (6.8) is that the service rate of an unstable queue is given by its throughput. Due to the ordering of the queues we know that queue i is unstable if $i \leq k$, which explains why $\mu_i(\lambda_{sat,k}) = \phi_i(\lambda_{sat,k})$ if $i \leq k$. Since ϕ_i and μ_i are both piecewise linear functions, this implies that $\mu_i(\lambda) = \phi_i(\lambda)$ for all $\lambda \geq \lambda_{sat,i}$.

For Equation (6.9), i.e., $i = k + 1$, we can make an interesting observation from Figure 5. We see that when a queue reaches its saturation load, the slope of the service rate does not change. That is, the service rate keeps changing in the same way at any of the solid vertical lines. This phenomenon can be explained intuitively by the following reasoning: We are interested in the service rate of queue i, which means that we look at the switch knowing that there is at least one packet present in queue i. If we increase the load, the most important cause of the change in the service rate of queue i is the fact that the load at other, stable, queues is increased. These queues take away capacity from queue i which leads to a lower service rate. The capacity that the other queues take away from queue i changes in a constant fashion, until one of the *other* queues becomes unstable; regardless of whether queue i becomes unstable in the meantime.

This reasoning implies that the slope of the service rate of queue i does not change at its saturation load. Given that we apply a piecewise linear approximation, the only approximation that takes this into account is the one that takes the slope of $\mu_i(\lambda)$ the same for $\lambda \in [\lambda_{sat,i-1}, \lambda_{sat,i})$ and $\lambda \in [\lambda_{sat,i}, \lambda_{sat,i+1})$, for $i = 2, \ldots, N$ (recall that $\lambda_{sat,N+1} := \infty$). In Lemma 5.2, we have shown that for $\lambda_{sat,i} \leq \lambda < \lambda_{sat,i+1}$,

$$\phi_i(\lambda) = \gamma_i^{(N-i+1)} + \lambda \left(\nu_i - \frac{\gamma_i^{(N-i+1)}}{\lambda_{sat,i}} \right) = \nu_i \lambda + \left(1 - \frac{\lambda}{\lambda_{sat,i}} \right) \gamma_i^{(N-i+1)}.$$

In Remark 5.1, it is furthermore argued that $\gamma_i^{(N-i+1)}$ is equal to the service rate in a saturated switch consisting only of queues $1, \ldots, i$. With the notation introduced in this section, we get $\gamma_i^{(N-i+1)} = \gamma_i^{\{1,\ldots,i\}}$, so that

$$\mu_i(\lambda) = \phi_i(\lambda) = \nu_i \lambda + \left(1 - \frac{\lambda}{\lambda_{sat,i}} \right) \gamma_i^{\{1,\ldots,i\}},$$

for $\lambda \in [\lambda_{sat,i}, \lambda_{sat,i+1})$. Taking the slope of $\mu_i(\lambda)$ the same for $\lambda \in [\lambda_{sat,i-1}, \lambda_{sat,i})$ yields

$$\mu_i(\lambda_{sat,i-1}) = \nu_i \lambda_{sat,i-1} + \left(1 - \frac{\lambda_{sat,i-1}}{\lambda_{sat,i}} \right) \gamma_i^{\{1,\ldots,i\}},$$

which is precisely (6.9).

In order to derive Equation (6.10), we consider the mean service time conditional on the event that a set of queues J is occupied. We obtain:

$$\mathbb{E}[B_i] = \sum_{J : i \in J} \mathbb{E}[B_i | J \text{ occupied}] \mathbb{P}(J \text{ occupied}).$$

The sum is restricted to sets J for which $i \in J$ because service of queue i implies that queue i itself is occupied.

The probability that a single queue j is occupied is approximately given by ρ_j, as defined in Approximation 6.1. The probability that a set J is occupied is approximated by treating the queues as independent:

$$\mathbb{P}(J \text{ occupied}) \approx \prod_{\substack{j \in J \\ j \neq i}} \rho_j \prod_{j \notin J} (1 - \rho_j).$$

The quantity $\mathbb{E}[B_i | J \text{ occupied}]$ is approximated by $1/\gamma_i^J$. That is,

$$\mathbb{E}[B_i | J \text{ occupied}] \approx 1/\gamma_i^J,$$

which yields

$$\mathbb{E}[B_i] \approx \sum_{J:i\in J} \frac{1}{\gamma_i^J} \prod_{\substack{j\in J\\ j\neq i}} \rho_j \prod_{j\notin J} (1-\rho_j).$$

Since $\mu_i(\lambda_{sat,k})$ is an approximation of $1/\mathbb{E}[B_i]$ we substitute $\mathbb{E}[B_i]$ by $b_i = 1/\mu_i(\lambda_{sat,k})$; the approximation of Equation (6.10) follows.

Remark 6.3. In case a uniform switch is considered, the entire approximation specialises to that of [1]. As argued in Remark 5.3, the stability conditions of all queues are the same and they are given by $\lambda < N\gamma$ where γ denotes the service rate in a saturated switch. Furthermore, if $\lambda \geq N\gamma$, we have $\phi_i(\lambda) = \gamma$. Straightforward substitution yields $\beta_i = (N-1)/N^2$, so that for all i,

$$\mu_i(\lambda) = \begin{cases} 1 - \frac{1}{2}\frac{N-1}{N}\frac{\lambda}{N} + \frac{-1+\frac{1}{2}\frac{N-1}{N}\gamma+\gamma}{\gamma^2}\left(\frac{\lambda}{N}\right)^2, & \text{for } \frac{\lambda}{N} \leq \gamma, \quad (6.11) \\ \gamma, & \text{for } \frac{\lambda}{N} > \gamma. \quad (6.12) \end{cases}$$

Despite the differences in notation (i.e., in [1], $p = \lambda/N$, $\widehat{q}(p) = \mu(\lambda)$ and $T_{sat} = \gamma$), this expression is the same as in [1]. The performance of this approximation was extensively analysed in [1] and it was established that this approximation outperforms known approximations for uniform switches if N is small.

Remark 6.4. If we consider a switch without contention, for instance with $p_{ij} = 1$ if $i = j$ and 0 otherwise, we simply have N independent $Geo/D/1$ queues. In this case our approximations are exact: The approximate stability conditions reduce to $\lambda_i < 1$ and our service rate approximation $\mu_i(\lambda) = 1$ for all λ and i. In other words, each queue is approximated by a $Geo/Geo/1$ queue with service rate 1, which is in fact a $Geo/D/1$ queue.

7 Analysis of the running example

In this section we illustrate the accuracy of the approximations devised in Section 4, 5, and 6, by a comparison with simulation. The example we study in detail in this section is the running example of the paper. In Section 8, we conduct a larger numerical study of 100 examples, where we draw more general conclusions.

Recall that the arrivals are governed by independent Bernoulli processes with parameter $\min\{\nu_i\lambda, 1\}$ and that for the running example, p_{ij} is as follows:

$$P = (p_{ij}) = \begin{pmatrix} 0.1 & 0.3 & 0.4 & 0.2 \\ 0.2 & 0.2 & 0.2 & 0.4 \\ 0.2 & 0.3 & 0.4 & 0.1 \\ 0.3 & 0.3 & 0.2 & 0.2 \end{pmatrix}.$$

Furthermore, the vector ν is given by

$$\nu = \begin{pmatrix} 0.35 & 0.3 & 0.2 & 0.15 \end{pmatrix}.$$

This section is divided into two parts: In 7.1 we study the accuracy of the throughput and the stability condition approximations, and in 7.2, we study the accuracy of the service rate and mean waiting time approximations.

7.1 Throughput and stability conditions

Recall that we defined a queue to be stable if its throughput is equal to its arrival rate. Because of this definition, we can look at throughput measurements from a simulation and if they start to

deviate from the arrival rate, we know that the queue is unstable. Recall furthermore that the value of λ for which a queue becomes unstable is called the 'saturation load' of that queue. During the entire numerical analysis, we make a distinction between the approximated saturation load of Approximation 4.5 and the saturation load observed via simulation.

Each simulation run consists of 10^7 time slots, and measurements of the first 10^5 time slots were discarded. Each run was repeated for 10 times, after which λ was increased by 0.01. In Table 1 we find throughput measurements of queue 1, for loads close to its saturation load. The fourth column of this table shows the standard deviation of the measurements. Note that this is the standard deviation of a single measurement, so in order to find the confidence intervals of the mean, one has to divide it by $\sqrt{10}$. In the last column it can be seen that, up to a load of 2.16, the simulated throughput is within 0.5 times the standard deviation (i.e., within $0.5\sqrt{10} \approx 1.6$ times the standard deviation of the mean). If $\lambda = 2.17$ the deviation of the throughput from the arrival rate is clearly significant, which means that the queue is unstable. We will call this load the saturation load. The fact that the queue is already unstable for this load implies that the true saturation load lies somewhere between 2.16 and 2.17, whereas Approximation 4.5 gives a saturation load of $\lambda_{sat,1} = 2.1470$. We emphasise that we call $\lambda = 2.17$ the observed (or simulated) saturation load, even though the queue is in fact already unstable for this load.

Table 1: Throughput measurements obtained via simulation.

λ	Arr. rate ($\nu_1\lambda$)	Avg. throughput($\overline{\phi}$)	St.dev.(σ)	$(\nu_1\lambda - \overline{\phi})/\sigma$
2.13	0.7455	0.7455	$1.4 \cdot 10^{-4}$	-0.09
2.14	0.7490	0.7489	$1.1 \cdot 10^{-4}$	0.47
2.15	0.7525	0.7525	$0.8 \cdot 10^{-4}$	0.50
2.16	0.7560	0.7560	$1.8 \cdot 10^{-4}$	-0.23
2.17	0.7595	0.7574	$2.0 \cdot 10^{-4}$	10.35
2.18	0.7630	0.7560	$1.6 \cdot 10^{-4}$	44.55
2.19	0.7665	0.7548	$0.9 \cdot 10^{-4}$	128.48

An overview of the observed saturation loads can be found in Table 2. We see that the approximation is more accurate for queues that become unstable for higher loads. Nevertheless, in all cases, the relative error is limited to 1%. So although the stability condition found is a little off in some cases, it gives a very accurate approximation of the true stability condition, especially for queues 3 and 4.

Table 2: An overview of the stability conditions found.

	Saturation load	
	Simulation	Alg. 4.1
Queue 1	2.17	2.1470
Queue 2	2.48	2.4669
Queue 3	3.33	3.3199
Queue 4	4.39	4.3869

Now that we have investigated the accuracy of the stability condition approximation, we move on to the analysis of the throughput approximation. In Section 5 we saw that the throughput approximation is piecewise linear in λ between the approximative saturation loads. It thus makes sense to look at the throughput approximation in these points, and compare them with simulation values. In order to achieve a fair comparison, we compare the values of the throughput approximation in the approximated saturation loads to the throughput measurements in the observed saturation loads, even though these are not the same. For example, we compare the throughput approximation of queue 1 with load 2.1470 to the simulated throughput with load 2.17 rather than 2.1470.

The comparison of the throughput approximation and the simulated throughput can be found in Table 3. The diagonal of the table corresponds to the throughput of the queues at their saturation loads. The lower left triangle corresponds to the throughput of the queues if they are already unstable and another queue becomes so too. The upper right triangle of the table would correspond to the throughput of stable queues. However, in Section 5 it was shown that the throughput approximation is exact for stable queues. As a result, it would make no sense to compare the throughput approximation to measurements for different loads. The measurements for stable queues have therefore been omitted from the table.

Again we conclude from the table that the throughput approximation is in general very accurate, especially if more queues are unstable. In all cases the relative error is limited to 1%. The standard deviation of all measurements in the table is roughly between $1 \cdot 10^{-4}$ and $2 \cdot 10^{-4}$.

Table 3: The accuracy of the throughput approximation.

		Throughput			
	λ	Queue 1	Queue 2	Queue 3	Queue 4
Appr.	2.1470	0.7515			
Sim.	2.17	0.7571			
Appr.	2.4669	0.7144	0.7401		
Sim.	2.48	0.7170	0.7419		
Appr.	3.3199	0.6588	0.6933	0.6640	
Sim.	3.33	0.6586	0.6931	0.6638	
Appr.	4.3869	0.6532	0.6700	0.6395	0.6580
Sim.	4.39	0.6354	0.6700	0.6394	0.6580

7.2 Service rate and mean waiting time

In this subsection we study the accuracy of the service rate and mean waiting time approximations. In Figure 6 the service rate approximation is plotted together with simulation values. It is clear that our service rate approximation is indeed very accurate.

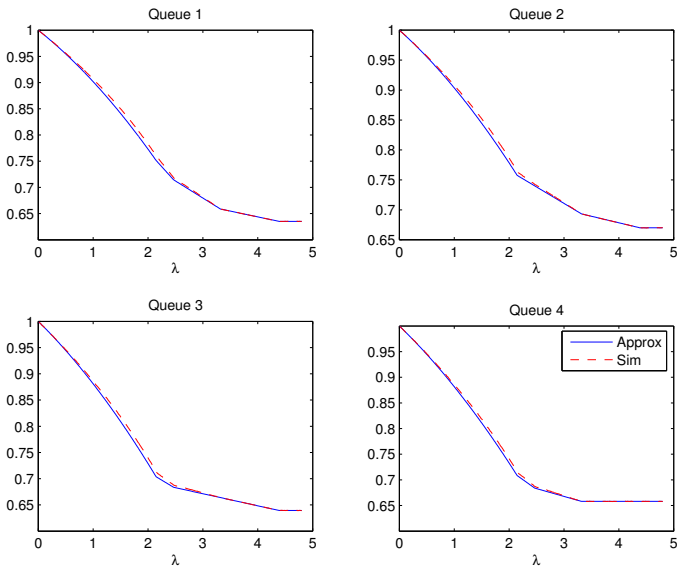

Figure 6: The service rate approximation.

Even though the service rate approximation is accurate, our mean waiting time approximation

is not necessarily accurate too. After all, an additional error might be induced by the assumption that the service time is geometrically distributed. A plot of the mean waiting time, together with its approximation, can be found in Figure 7. The figure shows that our approximation is quite accurate in this case, especially for queues 1 and 2.

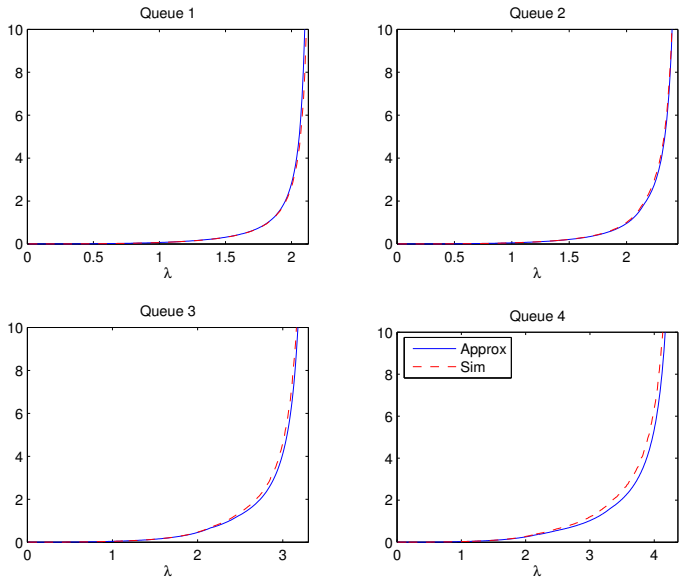

Figure 7: The mean waiting time approximation.

We have plotted the relative error of our mean waiting time approximation in Figure 8. This figure reveals that the relative errors of queue 1 are generally within 5%, those of queue 2 and 3 generally within 10%, and the relative error of queue 4 takes values up to roughly 15%. Note that if λ is close to 0, the relative error varies greatly. This is caused by the fact that we divide two numbers close to 0.

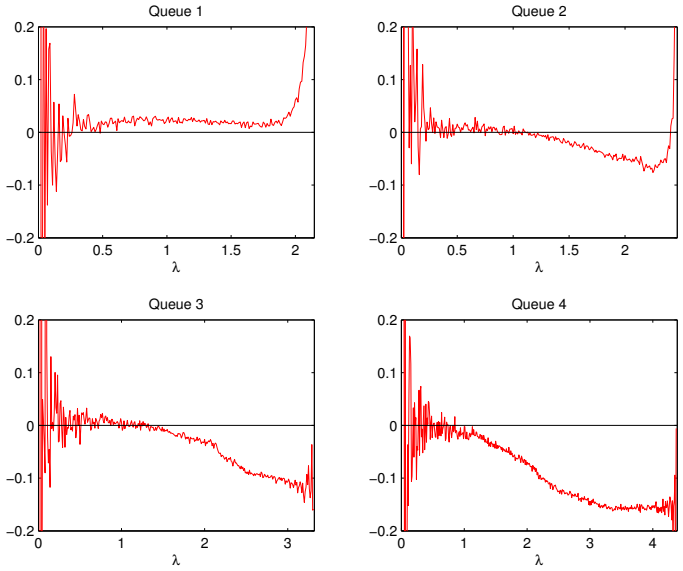

Figure 8: The relative error of the mean waiting time approximation

8 Numerical analysis

In this section, we study the performance of our approximation on a much larger scale. We introduce ten matrices (p_{ij}) and ten vectors ν, and study the performance of the approximation of the 100 possible combinations. Out of these ten matrices and vectors, five of each have been chosen and five have been generated randomly.

The following five matrices have been chosen: First,

$$(p_{ij}) = \begin{pmatrix} 0.25 & 0.25 & 0.25 & 0.25 \\ 0.25 & 0.25 & 0.25 & 0.25 \\ 0.25 & 0.25 & 0.25 & 0.25 \\ 0.25 & 0.25 & 0.25 & 0.25 \end{pmatrix},$$

because this is the matrix analysed in [1] (although there it was combined with *one* specific vector ν). Second,

$$(p_{ij}) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix},$$

since it corresponds to the situation in which all traffic has the same destination. This could, for instance, occur if a single memory is shared among multiple processors. Third,

$$(p_{ij}) = \begin{pmatrix} 0.5 & 0.5 & 0 & 0 \\ 0.5 & 0.5 & 0 & 0 \\ 0.5 & 0.5 & 0 & 0 \\ 0.5 & 0.5 & 0 & 0 \end{pmatrix},$$

which corresponds to the situation where two outputs are equally likely. Fourth, the matrix of the running example of the paper is included,

$$(p_{ij}) = \begin{pmatrix} 0.1 & 0.3 & 0.4 & 0.2 \\ 0.2 & 0.2 & 0.2 & 0.4 \\ 0.2 & 0.3 & 0.4 & 0.1 \\ 0.3 & 0.3 & 0.2 & 0.2 \end{pmatrix},$$

and fifth,

$$(p_{ij}) = \begin{pmatrix} 0.7 & 0.1 & 0.1 & 0.1 \\ 0.1 & 0.7 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.7 & 0.1 \\ 0.1 & 0.1 & 0.1 & 0.7 \end{pmatrix},$$

where most traffic (i.e., a 0.7 fraction) has its own destination, and occasionally it deviates from this destination.

The following five ν vectors were chosen manually:

$\nu = (0.25, 0.25, 0.25, 0.25)$,
$\nu = (0.4, 0.3, 0.2, 0.1)$,
$\nu = (0.6, 0.2, 0.1, 0.1)$,
$\nu = (0.7, 0.1, 0.1, 0.1)$,
$\nu = (0.8, 0.1, 0.05, 0.05)$.

The first vector is that of a uniform switch. The other vectors were chosen such that the first queue receives an increasingly higher load.

For each random matrix, we generated elements from a uniform distribution on $[0, 1]$. To ensure that the rows sum to 1, each element was divided by its row sum and rounded off to two decimals (while preserving the row sum of course). The following matrices resulted from this procedure:

$$(p_{ij}) = \begin{pmatrix} 0.15 & 0.32 & 0.24 & 0.29 \\ 0.27 & 0.24 & 0.13 & 0.36 \\ 0.05 & 0.38 & 0.42 & 0.15 \\ 0.40 & 0.14 & 0.31 & 0.15 \end{pmatrix},$$

$$(p_{ij}) = \begin{pmatrix} 0.28 & 0.29 & 0.15 & 0.28 \\ 0.34 & 0.03 & 0.15 & 0.48 \\ 0.15 & 0.21 & 0.30 & 0.34 \\ 0.40 & 0.04 & 0.01 & 0.55 \end{pmatrix},$$

$$(p_{ij}) = \begin{pmatrix} 0.20 & 0.15 & 0.34 & 0.31 \\ 0.25 & 0.50 & 0.14 & 0.11 \\ 0.09 & 0.33 & 0.20 & 0.38 \\ 0.24 & 0.16 & 0.36 & 0.24 \end{pmatrix},$$

$$(p_{ij}) = \begin{pmatrix} 0.11 & 0.31 & 0.02 & 0.56 \\ 0.18 & 0.48 & 0.26 & 0.08 \\ 0.55 & 0.41 & 0.01 & 0.03 \\ 0.37 & 0.03 & 0.23 & 0.37 \end{pmatrix},$$

$$(p_{ij}) = \begin{pmatrix} 0.26 & 0.25 & 0.30 & 0.19 \\ 0.01 & 0.05 & 0.62 & 0.32 \\ 0.02 & 0.59 & 0.14 & 0.25 \\ 0.10 & 0.32 & 0.23 & 0.35 \end{pmatrix}.$$

The elements of the five random ν vectors were also drawn from a uniform $[0, 1]$ distribution and scaled in the same way. The vectors were subsequently sorted in descending order. This procedure resulted in the following vectors:

$$\nu = (0.37, 0.30, 0.25, 0.08),$$
$$\nu = (0.34, 0.24, 0.23, 0.19),$$
$$\nu = (0.38, 0.32, 0.20, 0.10),$$
$$\nu = (0.40, 0.28, 0.17, 0.15),$$
$$\nu = (0.31, 0.29, 0.27, 0.13).$$

Each simulation of these 100 possible combinations ran for 10^7 time slots, and measurements of the first 10^5 time slots were discarded. The load λ was increased in steps of 0.01, until all queues became unstable. After the simulation, we renumbered the queues such that queue 1 is the first queue to become unstable, queue 2 the second, etc., similar to the convention of Section 5. We will give an overview of the errors of the saturation load and mean waiting time approximation.

An overview of the absolute value of the relative errors made in the saturation load approximation can be found in Table 4, and a graph of the empirical cumulative distribution in Figure 9. The absolute value of the relative error is the largest for queue 1, which we also saw in the previous section. On average, the error made for this queue is 1%, while for queues 2, 3, and 4 the error is even smaller. For queue 1, 95% of our simulations gave results within 2.2% error, and for the other queues within 0.87%.

Table 4: The saturation load approximation.

	Queue 1	Queue 2	Queue 3	Queue 4
Avg. |rel. error|	0.010	0.0037	0.0024	0.0022
90% error quantile	0.020	0.0068	0.0047	0.0040
95% error quantile	0.022	0.0087	0.0063	0.0046

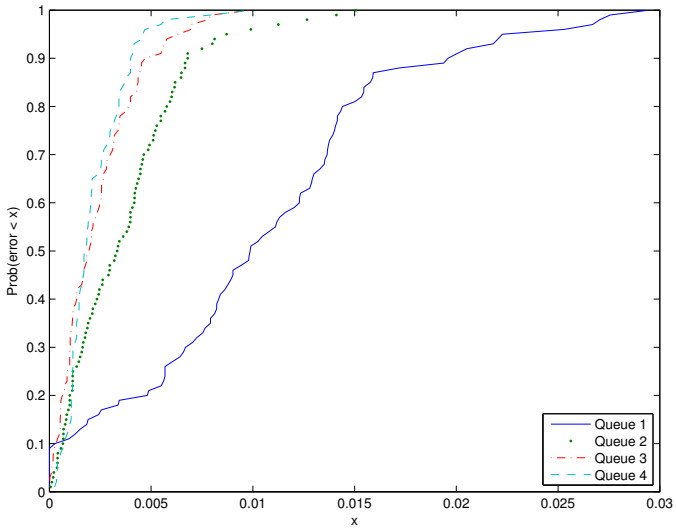

Figure 9: The empirical cumulative distribution of the errors of the saturation load approximation.

Note that even in cases where the approximation is exact, errors of up to 0.01 occur since this is the difference between consecutive simulation loads. For example, if the exact saturation load is 1.001, the simulated saturation load would be either 1.00 or 1.01, depending on the precise values of p_{ij} and ν_i.

Our approximation gives an underestimate of the saturation load in 94% of the cases. Furthermore, if our approximation overestimates the saturation load, it has an error of at most 0.0025, which is considerably less than 0.01. We therefore conjecture that our saturation load approximation is a lower bound for the true saturation load.

We conclude that the approximation is more accurate for the queues that become unstable at higher loads. This can be understood if we recall that queue 4 is the first to become empty in Algorithm 4.1, queue 3 the second, and so on. It appears that we slightly underestimate the time at which the first queue becomes empty, the difference between this time and the time at which the second queue becomes empty, etc. All these errors combined entail that the saturation load approximation becomes less accurate for the queues that become unstable early on. In addition to this, we suspect that the algorithm becomes less accurate if N becomes larger.

We will now focus on the relative error (not its absolute value) of the mean waiting time approximation. Since each simulation has its own saturation loads, we scale the loads in order to compare the various relative errors with each other. We do so by looking at $\rho_i := \lambda/\overline{\lambda}_{sat,i}$, where $\overline{\lambda}_{sat,i}$ is the simulated saturation load. We then compare the relative error of the approximation for each value of ρ_i between 0 and 1. The mean of this error and the 5% and 95% quantile as functions of ρ_i are plotted in Figure 10.

Generally, the mean of the relative errors is reasonably close to 0, which indicates that our approximation is accurate on average. Furthermore, for ρ_i roughly up to 0.8, the error quantiles are well within -20% and 20%, except for queue 1. If ρ_i approaches 1, our approximation clearly becomes less accurate and the difference between the error quantiles increases.

An interesting question that arises naturally from this analysis is which example is particularly bad. From the data we inferred that the matrix

$$(p_{ij}) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

performs particularly bad with respect to the mean waiting approximation. An explanation for this

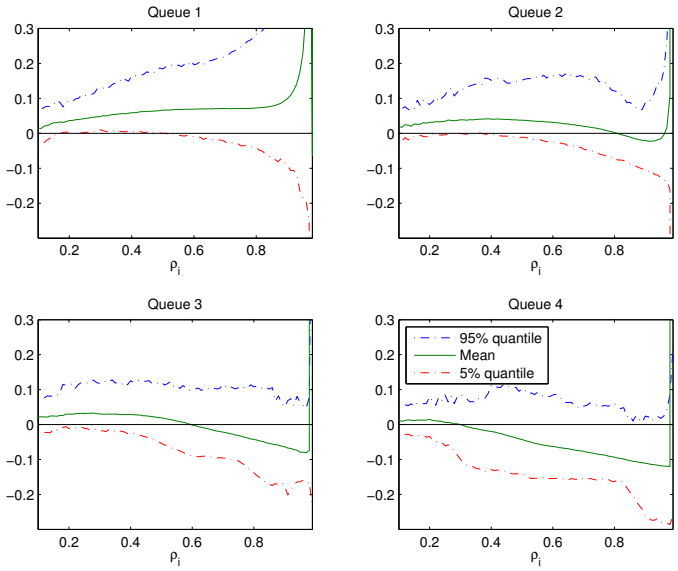

Figure 10: The relative error of the mean waiting time approximation. Note that the horizontal axis starts at $\rho_i = 0.1$. This has been done because for small ρ_i we have to divide two values close to 0, which is essentially meaningless.

phenomenon is that our approximation is based on an independence assumption for the queues. With the matrix mentioned above, there is only one output port that is shared by all queues, which means that we have strong dependence between all queues. The approximation performs worst if this matrix is combined with $\nu = (0.25, 0.25, 0.25, 0.25)$. If one queue receives a greater part of the total load, the approximation becomes better. Apparently the dependence between the queues is the strongest if the loads are uniformly distributed.

Furthermore, in contrast to the fact that the mean waiting time approximation with this p_{ij} and ν is the least accurate, it is striking that the approximation of the saturation loads is exact. This is easily seen if we observe that all queues together constitute a $Geo/D/1$ queue with load λ and unit service times. All queues are thus stable if and only if the $Geo/D/1$ queue is stable, which it is if $\lambda < 1$. Algorithm 4.1 gives precisely this value.

A second example that performs particularly bad is the example with

$$(p_{ij}) = \begin{pmatrix} 0.28 & 0.29 & 0.15 & 0.28 \\ 0.34 & 0.03 & 0.15 & 0.48 \\ 0.15 & 0.21 & 0.30 & 0.34 \\ 0.40 & 0.04 & 0.01 & 0.55 \end{pmatrix},$$

and $\nu = (0.7, 0.1, 0.1, 0.1)$. We found that this example had the largest error in the saturation load approximation ($\sim 3\%$). While we cannot explain this error in particular, it seems very likely that a large error in the saturation load approximation generally also induces a large error in the mean waiting time approximation.

9 Conclusion

The approach devised in this paper gives rise to the approximation of an arbitrary queue in a switch as a $Geo/Geo/1$ queue. A fluid approximation was used to approximate the throughput and saturation loads of each queue, and these were used to approximate the service rate of the approximating $Geo/Geo/1$ queue.

In Remark 5.3, we also argued that our approximation reduces to that of [1] in case the switch is uniform. In [1], the performance of this approximation was extensively tested and shown to be more accurate than asymptotic models. In addition to this, we showed that our approximation is exact if $p_{ii} = 1$ and $p_{ij} = 0$ for $j \neq i$. Note that this example implies that all queues behave as independent queues with fixed service times equal to 1.

In order to investigate the performance of the approximation quantitatively, we performed a large numerical study in Section 8. It was shown that the saturation load approximation is very accurate in general, but less accurate for the queues that are the first to become unstable. Furthermore, we observed that the approximation generally leads to an underestimation of the true saturation load.

The mean waiting time approximation was also shown to be quite accurate. Especially for utilisation rates up to 0.8 its accuracy should be sufficient for practical purposes. We also found two causes for a less accurate mean waiting time approximation. First, due to the independence assumption in Approximation 6.1, our approximation performs worse if there is a strong dependency between queues. This is especially true if the load is uniformly distributed over all input ports. Second, our approximation performs worse if there is a comparatively large error in the saturation load approximation. Pinpointing the causes for such an error is, however, beyond the scope of this paper.

The entire analysis of this paper was performed under the assumption of Bernoulli arrival processes. However, the approximations of the throughput and stability conditions, as well as the approximation of $\mu_i(\lambda_{sat,k})$ do not depend on the specific arrival processes, but only on their rates. The light traffic approximation in its present form depends on the assumption of Bernoulli arrivals, but it can be extended to more general arrival processes. The approximation of the mean waiting time also depends on this assumption, since results on the $Geo/Geo/1$ queue were used. If another arrival process is assumed, the switch has to be approximated by a $G/Geo/1$ queue. The accuracy of our approximation probably depends on the arrival process as well, but this is beyond the scope of the current paper.

Even though we use a piecewise linear approximation, the service rate is in reality probably not linear. It is quite likely that higher order terms play a role, but their effect is insignificant; a linear interpolation is already very accurate. Furthermore, should a more accurate approximation be required, we believe that it is more likely that this can be achieved by a better approximation of $\mu_i(\lambda_{sat,k})$ (i.e., replace (6.9) and (6.10) by something better), rather than taking higher order terms into account.

10 Acknowledgements

We would like to thank Onno Boxma and Dee Denteer for their helpful suggestions. Their suggestions have considerably improved the paper.

References

[1] P. Beekhuizen, T.J.J. Denteneer, and I.J.B.F. Adan, *Analysis of a tandem network model of a single-router network-on-chip*, Annals of Operations Research (To appear).

[2] R.K.C. Chang and S. Lam, *A novel approach to queue stability analysis of polling models*, Performance Evaluation **40** (2000), 27–46.

[3] S-T. Chuang, A. Goel, N. McKeown, and B. Prabhakar, *Matching output queueing with a combined input output queued switch*, IEEE Journal on Selected Areas in Communications **17** (1999), no. 6, 1030–1039.

[4] J.G. Dai and B. Prabhakar, *The throughput of data switches with and without speedup*, Proc. of IEEE INFOCOM, 2000, pp. 556–564.

[5] W.J. Dally and B. Towles, *Route packets, not wires: on-chip interconnection networks*, DAC proceedings, 2001, pp. 684–689.

[6] O.C. Ibe and X. Cheng, *Stability conditions for multiqueue systems with cyclic service*, IEEE Transactions on Automatic Control **33** (1988), no. 1, 102–103.

[7] M.J. Karol, M.G. Hluchyj, and S.P. Morgan, *Input versus output queueing on a space-division packet switch*, IEEE Transactions on Communications **35** (1987), no. 12, 1347–1356.

[8] H. Kim, K. Kim, and Y. Lee, *Derivation of the mean cell delay and cell loss probability for multiple input-queued switches*, IEEE Communications Letters **4** (2000), no. 4, 140–142.

[9] N. McKeown, A. Mekkittikul, V. Anantharam, and J. Walrand, *Achieving 100% throughput in an input-queued switch*, IEEE Transactions on Communications **47** (1999), no. 8, 1260–1267.

[10] N.W. McKeown, *Scheduling algorithms for input-queued cell switches*, Ph.D. thesis, Univ. California, Berkeley, 1995.

[11] U.Y. Ogras and R. Marculescu, *Analytical router modeling for networks-on-chip performance analysis*, DATE Proceedings, 2007, pp. 1096–1101.

[12] S. González Pestana, E. Rijpkema, A. Rădulescu, K. Goossens, and O. Gangwal, *Cost-performance trade-offs in networks on chips: A simulation based approach*, DATE Proceedings, 2004, pp. 764–769.

[13] B. Prabhakar and N. McKeown, *On the speedup required for combined input and output queued switching*, Automata **35** (1999), no. 12, 1909–1920.

[14] D. Shah and D. Wischik, *Optimal scheduling algorithms for input-queued switches*, Proc. of IEEE INFOCOM, April 2006.

[15] H. Takagi, *Queueing Analysis: A Foundation of Performance Evaluation*, vol. 3: Discrete-time systems, North-Holland, Amserdam, 1993.