A DTMC Model for Performance Evaluation of Irregular Interconnection Networks with Asymmetric Spatial Traffic Distributions

Daniel Lüdtke[1] and Dietmar Tutsch[2]

[1] German Aerospace Center (DLR)
Simulation and Software Technology
Lilienthalplatz 7, D-38118 Braunschweig, Germany
daniel.luedtke@dlr.de
[2] University of Wuppertal
Automation / Computer Science
Rainer-Gruenter-Str. 21, D-42119 Wuppertal, Germany
tutsch@uni-wuppertal.de

Abstract. Several mathematical models have been proposed to evaluate the performance of interconnection networks used for high-speed connections for supercomputers, switches and routers for local and wide area networks, as well as networks on a chip. Often these models are based on state space reduction by exploiting symmetries of the network and requiring uniform traffic patterns. If an interconnection network is built for a specific application with non-uniform spatial traffic distribution, models that are more general are needed. This paper proposes a mathematical model for performance evaluation of application-specific interconnection networks based on inhomogeneous discrete time Markov chains (DTMC). It supports store and forward routing, irregular network topologies, and asymmetric spatial traffic distributions. The model is described in a generalized way so that it can support arbitrary switching element sizes within the network and its input buffers.

1 Introduction

Interconnection networks with point-to-point links are widely used for high-speed connections in different domains, for example, as networks on a chip (NoC), in supercomputers, and in switches and routers for local and wide area networks. In the space domain, SpaceWire networks [9] are more often used to provide high communication bandwidth onboard spacecraft.

Most interconnection networks have regular or symmetric topologies, because supercomputers or network switches are usually designed for a large variety of applications. However, for application-specific networks, like specialized systems on a chip employing a network on a chip or a spacecraft with many specialized network nodes, irregular topologies are often beneficial.

Additionally, many performance evaluation approaches presume uniform traffic distributions. When planning and optimizing an interconnection network for

The final publication is available at Springer via http://dx.doi.org/10.1007/978-3-319-43904-4_14

a specific application, non-uniform traffic distributions, which match the traffic patterns of the application in question, need to be considered.

For instance, in the research project OBC-NG (Onboard Computer — Next Generation) [6], which was carried out at the German Aerospace Center (DLR), a new reconfigurable distributed computer architecture was developed. The goal of this project and its successor, ScOSA (Scalable On-board computing for Space Avionics), is to provide high performance onboard computing power to support complex future space missions. ScOSA utilizes a reconfigurable SpaceWire network that allows the reconfiguration of the spacecraft computer for different mission phases as well as for error mitigation. Each of these configurations are predetermined for which optimization algorithms are needed to find optimal partitioning and mapping of tasks for each mission phase. These optimizations need to consider the available bandwidth and possible congestion in the network.

The project considers two approaches for the stochastic performance evaluation: simulation and an analytical model. CINSim (Component-based Interconnection Network Simulator) [7], a stochastic discrete event simulation framework, is used as simulator. Additionally, a mathematical model, based on inhomogeneous discrete-time Markov chains (DTMC) is developed, to evaluate different network configurations. This model is presented in this paper. It supports direct networks, where terminals are connected at each switch or switching element as well as indirect networks like Multistage Interconnection Networks (MIN). A discrete-time Markov chain, compared to a continuous-time Markov chain, was chosen to model the synchronous behavior of the switch implementation.

The remainder of this paper is structured as follows: the next section gives a brief overview of related work. Sect. 3 introduces the DTMC model. Followed by some results, where the model results are compared to simulation. Sect. 5 presents the conclusions and an outlook to ongoing and future work is given.

2 Related Work

Performance evaluation models for interconnection networks, especially MINs, have been proposed for many years. Dias and Jump [3] proposed a model for MINs with a buffer at each switch input. Yoon et al. [12] described a model that supports arbitrary buffer lengths and arbitrary switch dimensions. Youn and Mun [13] established a model that confined packet movement from one switch to another in one clock cycle. Atiquzzaman and Akhtar [2] proposed a model to investigate hot spot traffic in MINs. Bin and Atiquzzaman [14] developed a model for output buffered MINs with asymmetric traffic patterns. Tutsch and Hommel [11] proposed a model, which considers multicast traffic within a MIN. This model reduces the state space by exploiting symmetries in the network topology and assuming uniformly distributed traffic.

All mentioned models are based on Markov chains. In recent years, analytical models have been proposed that are based on queuing networks. Moadeli et al. [8] developed a performance model for the spidergon topology for NoCs. Kiasari et al. [5] proposed an analytical latency model for NoCs for arbitrary topologies.

An analytical model for MINs is proposed by Amiri-Zarandi et al. [1]. Hamid et al. [4] introduce an analytical model for a multi-core multi-cluster architecture and Sabbaghi-Nadooshan and Patooghy [10] present a performance model for de Bruijn inspired mesh-based NoCs.

3 Model Description

This section describes the network abstraction, traffic model, and the DTMC model itself.

3.1 Network and Traffic Model

The network model presented in this work is based on CINSim's component-based interconnection network model. It supports combinations of the components source, destination, switch, and buffer. With these atomic components, a great variety of packet-switched interconnection network architectures can be modeled and investigated, especially irregular network topologies.

Sources generate traffic according to a geometric load distribution $r(x)$, with $x \in \{0, 1\}$ (on/off source), which means that a packet is generated at each time step with the probability $r(x)$, and the global spatial distribution ℓ^{g}_{sd}, which defines the probability that a packet at source s is targeted to destination d. Destinations consume packets immediately. Packets move from sources via buffers and switches to their destinations. Switches are components to realize dynamically changing connections between its inputs and outputs. Inputs and outputs are connected according to the requested output of the packet. If multiple inputs contain packets destined to the same output, one of the packet is randomly selected. Switches have input buffers that store packets. It is required that each switch input is connected to a buffer. Input buffers and destinations can only have one preceding component. Switches work in a timeless manner in this model. Only moving through a buffer costs at least one time step. The buffer size can be set individually for each input buffer. Fig. 1 shows a small interconnection network with the described components: sources on the left, input buffers with a capacity of three packets connected to a switch, and two destinations on the right.

The proposed model employs unicast store-and-forward routing, i.e. a packet is the smallest flow control unit in the model. It is based on the local backpressure scheme, which means that packets that enter the network will eventually reach their destination. Packet drops happen only at the sources. *Local* backpressure means that packets can only proceed to the next buffer component during the following time step, if buffer space is available at the current time step. The model implements shortest-path routing. If several shortest paths are available, a path is randomly selected. All network components operate synchronously.

To keep the modeling complexity limited, we assume that all network components operate synchronously, driven by a global internal clock. This assumption is quite realistic for a wide range of applications. For instance, network switches or networks on chips often operate synchronously.

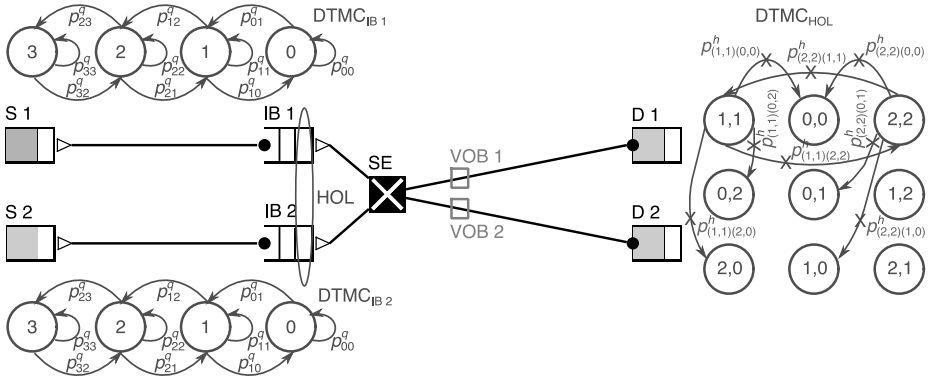

Fig. 1. A 2×2 network with two sources on the left (S), two input buffers (IB), a switch (switching element, SE), and two destinations (D) on the right. The two virtual output buffers (VOB) and the corresponding Markov chains (see Sect. 3.2) are also shown: $\mathrm{DTMC}_{\mathrm{IB}\ 1}$ and $\mathrm{DTMC}_{\mathrm{IB}\ 2}$ for the input buffers and $\mathrm{DTMC}_{\mathrm{HOL}}$ for the Head-of-Line (HOL) Markov chain (only unfeasible state transitions are drawn).

3.2 DTMC Model

For an adequate modeling of non-uniform spatial traffic distributions, the global spatial distribution $\boldsymbol{\ell}^{\mathrm{g}}$ has to be considered at each switch in combination with the actual load distribution of each source. For this, probabilities need to be determined about the packet rate for each input/output combination at every switch. Hence, a local spatial traffic distribution ℓ^{l}_{io} is derived, which represents the probability that a packet at a switch is transmitted from switch input i to switch output o. *Local* means that this distribution refers to a single switch in contrast to the global spatial distribution $\boldsymbol{\ell}^{\mathrm{g}}$. To achieve this, besides the queue length of each input buffer, the Head-of-Line places (HOL) of all buffers are modeled to represent the target outputs of the switch. Additionally, to simplify the equations, virtual output buffers are introduced at the switches. These buffers indicate from which input a packet has arrived in the current time step to adapt ℓ^{l}_{io}. Since the virtual output buffers have no representation in real hardware, the presence of a packet there does not consume time.

It is not feasible to model even small networks with a single Markov chain due to the state space explosion. In particular, no symmetries can be exploited compared to most models mentioned in the related work section due to the requirement to support arbitrary topologies. For example, a simple model of a single 3×3 switch with four buffer places at each input leads to 24,303 states and 261,081 state transitions. Larger networks cannot be modeled in such a way, since the number of states increases exponentially. Thus, a decomposition technique is used that was also applied in previous work (e.g. [11]). However, the proposed model in this paper does not exploit topological symmetries or requires uniformly distributed spatial traffic distributions.

Several Markov chains are setup to reflect different behavioral aspects of the system. Hence, the number of states and state transitions grows only linearly. The connection between individual Markov chains is achieved by dependencies in the state transition probabilities of the individual Markov chains, i.e. the transient state probabilities of an individual Markov chain are considered in the state transition probabilities of dependent other Markov chains. These state transition probabilities change every time step until the network reaches steady-state, leading to inhomogeneous DTMCs.

Fig. 1 shows, next to the network, the corresponding Markov chains and virtual output buffers. For the HOL DTMC, only the unfeasible state transitions are shown, all other transitions are feasible.

State Space of HOL DTMCs. All HOL buffer places in a switch with $i_{\max}$ inputs and input buffers, respectively, as well as $o_{\max}$ outputs and virtual output buffers, respectively, are represented by a HOL Markov chain with a finite state space $S^{\mathrm{h}}(i_{\max}, o_{\max})$. A combined Markov chain for all HOL buffer places is chosen to adequately model the blocking behavior of a switch. The states of $S^{\mathrm{h}}(i_{\max}, o_{\max})$ are denoted by $i_{\max}$-tuples of an $(o_{\max}+1)$-set. Each HOL place of an input buffer i has the possible states $s_i^{\mathrm{h}} \in \{0, 1, \ldots, o_{\max}\}$. The first element of each state represents the state of the HOL place of the first input buffer. A '0' represents the case, where no packet is present; a '1', where a packet wants to move to output 1 etc. s_i^{h} denotes the target output of a packet. For instance, $s_3^{\mathrm{h}} = 2$ means that the HOL packet in the third input buffer of a switch is destined to the second output of the switch.

$S^{\mathrm{h}}(i_{\max}, o_{\max})$ of the HOL Markov chain for a switch is defined by the $i_{\max}$-permutation with repetition (PR) of the set of outputs s_i^{h}:

$$S^{\mathrm{h}}(i_{\max}, o_{\max}) \coloneqq \mathrm{PR}\left(i_{\max}, s_i^{\mathrm{h}}\right) = \{(0, \ldots, 0), \ldots, (o_{\max}, \ldots, o_{\max})\} \ .$$

The number of possible states is given by $\left|S^{\mathrm{h}}(i_{\max}, o_{\max})\right| = (o_{\max}+1)^{i_{\max}}$. Hence, the state probability vector $\boldsymbol{\nu}^{\mathrm{h}}$ of the head-of-line Markov chain has $\left|S^{\mathrm{h}}(i_{\max}, o_{\max})\right|$ components and $\left|S^{\mathrm{h}}(i_{\max}, o_{\max})\right|^2$ elements in its corresponding transition probability matrix $\mathbf{P}^{\mathrm{h}}$.

Initially, it is assumed that all buffers are empty. Thus, $\boldsymbol{\nu}^{\mathrm{h}}_{(0,0)}(0) = 1$ in case of the 2×2 example. To calculate the state probability vector of the second time step $n = 1$, the equation $\boldsymbol{\nu}^{\mathrm{h}}(1) = \mathbf{P}^{\mathrm{h}}(0) \cdot \boldsymbol{\nu}^{\mathrm{h}}(0)$ has to be solved according to the Chapman-Kolmogorov equation.

State Space of Queue Length DTMCs. The finite state space S^{q} of the Markov chain that represents the queue length of input buffer i is constructed by $S^{\mathrm{q}}(m_{\max}(i)) \coloneqq \{0, 1, \ldots, m_{\max}(i)\}$, where $m_{\max}(i)$ denotes the number of buffer places of input buffer i. For example, an input buffer with the capacity for four packets has the state space $S^{\mathrm{q}}(4) = \{0, 1, 2, 3, 4\}$.

The number of possible states is given by $|S^{\mathrm{q}}(m_{\max}(i))| = m_{\max}(i) + 1$. Hence, the queue length DTMC state probability vector $\boldsymbol{\nu}^{\mathrm{q}}$ has $|S^{\mathrm{q}}(m_{\max}(i))|$

components and a transition probability matrix $\mathbf{P}^{\mathrm{q}}$ with $|S^{\mathrm{q}}(m_{\max}(i))|^2$ elements. $\nu_0^{\mathrm{q}}(n)$ denotes the state probability that the buffer is empty at time step n and $\nu_{m_{\max}(i)}^{\mathrm{q}}(n)$ represents the state probability that the buffer is full, respectively.

In the following, it is assumed that $m_{\max}(i) > 1$. With only one buffer place in an input buffer, a packet cannot simultaneously be sent and received during one time step due to the local backpressure scheme. With $m_{\max}(i) = 1$, the normalized throughput of this connection would maximally be 0.5.

State Space of Virtual Output Buffers. As previously mentioned, virtual output buffers are established to simplify the modeling of the spatial distribution within the network. Packets are only transferred to these buffers if they can be transferred to the next switch or network output in the following time step. This reduces the modeling of the output buffers to simple probabilities that can be derived from the HOL states. The probabilities are updated directly during the same iteration step. Nevertheless, the states of the virtual output buffers in a switch are also denoted as a state probability vector $\boldsymbol{\nu}^{\mathrm{o}}$. The state space of a virtual output buffer is defined by the number of the input buffers of the switch in question: $S^{\mathrm{o}}(i_{\max}) \coloneqq \{0, 1, \ldots, i_{\max}\}$. The number of possible states is given by $|S^{\mathrm{o}}(i_{\max})| = i_{\max} + 1$. Hence, the state probability vector $\boldsymbol{\nu}^{\mathrm{o}}$ has $|S^{\mathrm{o}}(i_{\max})|$ components.

State Transition Probabilities for HOL DTMCs. The state transition probabilities for the HOL DTMCs are shown in Fig. 2. They depend on the competition of packets at the HOL position of the input buffers for an output. Since only a single packet can win such a competition, the losing packets stay at their input buffer. Conflicts are resolved randomly.

Each entry of the state transition matrix $\mathbf{P}^{\mathrm{h}}(n)$ for the HOL DTMCs at time step n is calculated by (1). All state transitions from state s to state t at time step n that are not feasible are set to zero. Infeasible state transitions describe situations, for instance, if two HOL packets destined to the same switch output, could have been successfully sent, which is not possible because only one packet could have won the conflict. For instance, a 5×5 switch has 7,776 possible HOL states and 60,466,176 entries in its $\mathbf{P}^{\mathrm{h}}(n)$ matrix. 22,221,176 state transitions are feasible. Thus, 38,245,000 entries in $\mathbf{P}^{\mathrm{h}}(n)$ are always zero.

(2) determines if a state transition from s to t is feasible. It is feasible if for all HOL packets that are destined to the same switch output o (conflicting inputs, $\mathrm{ci}(o, s^{\mathrm{h}})$, see (4)) only a single input changes from s to t ($|\Delta \mathrm{i}(o, s, t)| \leq 1$, see (5)). This has to be true for all outputs with more than one HOL as target ("courted outputs", $o \in \mathrm{co}(s^{\mathrm{h}})$, see (3)).

The probability for a feasible state transition is the product of the probabilities of a state change from state s to t of the HOL packet for each input. Two general cases have to be distinguished.

First, we consider inputs that are not part of any conflict set at state s. This includes all empty input buffers and HOL packets at s, which have no

$$p_{st}^{\mathrm{h}}(n) = \begin{cases} 0, & \text{if feasible}(s,t) = 0 \\ \prod_{i \in \mathrm{nci}(s)} p_{st}^{\mathrm{noConf}}(i,n) \cdot \prod_{o \in \mathrm{co}(s)} p_{st}^{\mathrm{conf}}(o,n), & \text{if feasible}(s,t) = 1 \end{cases} \tag{1}$$

$$\text{feasible}(s,t) \coloneqq \begin{cases} 1, & \text{if } \forall o \in \mathrm{co}(s) : |\Delta\mathrm{i}(o,s,t)| \leq 1 \\ 0, & \text{else .} \end{cases} \tag{2}$$

$$\mathrm{co}\left(s^{\mathrm{h}}\right) \coloneqq \left\{ o : \; o \in \{1,\ldots,o_{\max}\} \wedge \left|\mathrm{ci}\left(o, s^{\mathrm{h}}\right)\right| > 1 \right\} \tag{3}$$

$$\mathrm{ci}\left(o, s^{\mathrm{h}}\right) \coloneqq \left\{ i : \; i \in \{1,\ldots,i_{\max}\} \wedge s_i^{\mathrm{h}} = o \right\} \tag{4}$$

$$\Delta\mathrm{i}(o,s,t) = \mathrm{ci}(o,s) \setminus \mathrm{ci}(o,t) \tag{5}$$

$$\mathrm{nci}\left(s^{\mathrm{h}}\right) \coloneqq \{1,\ldots,i_{\max}\} \setminus \bigcup_{o \in \mathrm{co}\left(s^{\mathrm{h}}\right)} \mathrm{ci}\left(o, s^{\mathrm{h}}\right) \tag{6}$$

$$p_{st}^{\mathrm{noConf}}(i,n+1) = \begin{cases} 1 - \tilde{q}(i,n), & \text{if } s_i = 0 \wedge t_i = 0 \\ \ell_{it_i}^{\mathrm{l}}(n) \cdot \tilde{q}(i,n), & \text{if } s_i = 0 \wedge t_i \neq 0 \\ \mathrm{r}(s_i,n) \cdot \mathrm{lps}(i,n), & \text{if } s_i \neq 0 \wedge t_i = 0 \\ \mathrm{r}(s_i,n) \cdot \ell_{it_i}^{\mathrm{l}}(n) \cdot \mathrm{nfp}(i,n), & \text{if } s_i \neq 0 \neq t_i \wedge s_i \neq t_i \\ \mathrm{r}(s_i,n) \cdot \ell_{it_i}^{\mathrm{l}}(n) \cdot \mathrm{nfp}(i,n) & \\ \qquad + (1 - \mathrm{r}(s_i,n)), & \text{if } s_i \neq 0 \wedge t_i = s_i \end{cases} \tag{7}$$

$$\mathrm{lps}(i,n) = \begin{cases} \frac{\nu_1^{\mathrm{q}}(i,n) \cdot (1 - \tilde{q}(i,n))}{1 - \nu_0^{\mathrm{q}}(i,n)}, & \text{if } \nu_0^{\mathrm{q}}(i,n) < 1 \\ 0, & \text{if } \nu_0^{\mathrm{q}}(i,n) = 1 \end{cases} \tag{8}$$

$$\mathrm{nfp}(i,n) = \begin{cases} \frac{\nu_1^{\mathrm{q}}(i,n) \cdot \tilde{q}(i,n)}{1 - \nu_0^{\mathrm{q}}(i,n)} + \frac{1 - \nu_0^{\mathrm{q}}(i,n) - \nu_1^{\mathrm{q}}(i,n)}{1 - \nu_0^{\mathrm{q}}(i,n)}, & \text{if } \nu_0^{\mathrm{q}}(i,n) < 1 \\ 0, & \text{if } \nu_0^{\mathrm{q}}(i,n) = 1 \end{cases} \tag{9}$$

$$p_{st}^{\mathrm{conf}}(o,n+1) = \begin{cases} \frac{\mathrm{r}(o,n)}{|\mathrm{ci}(o,s)|} \cdot \mathrm{lps}\left(\Delta\mathrm{i}(o,s,t), n\right), & \text{if } t_{\Delta\mathrm{i}(o,s,t)} = 0 \\ \frac{\mathrm{r}(o,n)}{|\mathrm{ci}(o,s)|} \cdot \ell_{\Delta\mathrm{i}(o,s,t) t_{\Delta\mathrm{i}(o,s,t)}}^{\mathrm{l}}(n) & \\ \qquad \cdot \mathrm{nfp}\left(\Delta\mathrm{i}(o,s,t), n\right), & \text{if } t_{\Delta\mathrm{i}(o,s,t)} \neq 0 \\ \frac{\mathrm{r}(o,n)}{|\mathrm{ci}(o,s)|} \cdot \sum_{c \in \mathrm{ci}(o,s)} \left(\ell_{co}^{\mathrm{l}}(n) \cdot \mathrm{nfp}(c,n)\right) & \\ \qquad + (1 - \mathrm{r}(o,n)), & \text{if } \Delta\mathrm{i}(o,s,t) = \emptyset \end{cases} \tag{10}$$

Fig. 2. State transition probabilities $\mathbf{P}^{\mathrm{h}}(n)$ for the head-of-line Markov chains

other competitors for its output (non-conflicting inputs $i \in \mathrm{nci}(s)$, see (6)). The probabilities for these inputs are calculated individually with $p_{st}^{\mathrm{noConf}}(i,n)$, defined in (7).

And second, all HOL packets of inputs that are competing for an output o at state s have to be considered together due to their interdependence and are calculated by $p_{st}^{\mathrm{conf}}(o,n)$, defined in (10). The conflict sets for different outputs are calculated independently.

Non-conflicting Inputs. The state transition probability $p_{st}^{\text{noConf}}(i, n+1)$ for input buffer i from state s to state t for time step $n+1$ is determined by (7). Five cases have to be distinguished for $p_{st}^{\text{noConf}}(i, n+1)$:

- First, if the queue of input buffer i in state s is empty and stays empty in state t, then no new packet comes to i during this state transition with probability $(1 - \tilde{q}(i, n))$, where $\tilde{q}(i, n)$ is the receiving probability of input buffer i in case that buffer space is available. It is determined by the sending probability of the preceding component $r_{\text{Pred}}(o, n)$ normalized by the probability that the buffer is not full: $\widetilde{q}(i, n) = \frac{r_{\text{Pred}}(o,n)}{1-\nu^{\text{q}}_{m_{\max}(i)}(i,n)}$.

  In case the preceding component is a network source, the sending probability r_{Pred} represents the geometric load distribution $r(x)$ of this source (see Sect. 3.1). For switches it is given by the probability that their virtual output buffer is not empty, hence, a packet is sent: $r_{\text{Pred}}(o, n) = 1 - \nu^{\text{o}}_{\text{Pred},0}(o, n)$.
- Second, if the queue was empty in s and has a HOL packet in t, a packet is received during the state transition multiplied with the probability $\ell^{\text{l}}_{it_i}(n)$ that a packet that enters the switch from input i is targeted to the output t_i, i.e., the output that is given by the i-th element of t.
- Third, if the queue of i had only one packet left at s, no new packet were coming ($\text{lps}(i, n)$) and this last packet was successfully sent during the state transition to t because the output was available ($r(s_i, n)$), the new HOL state for i would be $t_i = 0$. The last packet sent probability $\text{lps}(i, n)$ (see (8)) is defined as follows: if i is definitely empty ($\nu^{\text{q}}_0(i, n) = 1$), the probability equals 0; in all other cases, the probability that only one packet is present ($\nu^{\text{q}}_1(i, n)$) is multiplied by the probability that no new packet is received ($1-\tilde{q}(i, n)$). The term has to be normalized by the sum of state probabilities of a non-empty buffer, since $\text{lps}(i, n)$ is only defined for state transitions where the buffer is not empty.
- Fourth, if the HOL packet of i changed to a different target during the state transition from state s to t, then the packet with the output s_i was successfully sent ($r(s_i, n)$) and a new first packet ($\text{nfp}(i, n)$) targeted to t_i is present with the routing probability $\ell^{\text{l}}_{it_i}(n)$. The new first packet probability ($\text{nfp}(i, n)$, see (9)) is defined as follows: in case the input buffer is definitely empty ($\nu^{\text{q}}_0(i, n) = 1$) the probability equals 0.

  In all other cases, the probability for a new first packet is combined by the following two cases: input buffer i had only one packet stored ($\nu^{\text{q}}_1(i, n)$) and a new packet is received ($\tilde{q}(i, n)$). Or i had more than one packet, thus, receiving a new packet does not influence the HOL state ($1 - \nu^{\text{q}}_0(i, n) - \nu^{\text{q}}_1(i, n)$). Since a new first packet only covers the case where the buffer is not empty, the probability has to be normalized by the term $1 - \nu^{\text{q}}_0(i, n)$.
- Finally, if the target of the HOL packet in i at state s is equal to the target at t, then two situations have to be distinguished. Similar to the previous case, a packet could have been sent and the new packet is heading to the same output, or the packet was blocked ($1 - \text{r}(s_i, n)$) due to a filled queue in the destination switch.

Conflicting Inputs. The probability $p_{st}^{\text{conf}}(o, n+1)$ for a set of inputs that are competing for output o at state s cannot be calculated independently. Thus, the probability for each conflict set that is identified by the common output o at state s is given by (10). Only one of the inputs can change from state s to t in each conflict set, otherwise, this state transition would not be feasible. Three cases for $p_{st}^{\text{conf}}(o, n+1)$ have to be distinguished:

- First, if the only changed input, identified by $\Delta\text{i}(o, s, t)$ (see (5)), changed its HOL state to $t_{\Delta\text{i}(o,s,t)} = 0$, output o was not blocked ($\text{r}(o, n)$) at time step n, its HOL packet won the conflict in this conflict set ($1/|\text{ci}(o, s)|$, random conflict resolution) and it was the last packet in this buffer ($\text{lps}(\Delta\text{i}(o, s, t), n)$).
- Second, if one HOL packet changes its target from output o to $t_{\Delta\text{i}(o,s,t)}$ during the state transition from s to t, then this input has won the conflict ($1/|\text{ci}(o, s)|$), the output was not blocked ($\text{r}(o, n)$), and a new first packet ($\text{nfp}(\Delta\text{i}(o, s, t), n)$) has the target output $t_{\Delta\text{i}(o,s,t)}$ with the routing probability $\ell^{\text{l}}_{\Delta\text{i}(o,s,t)t_{\Delta\text{i}(o,s,t)}}(n)$.
- Third, if no input changes in the conflict set of output o ($\Delta\text{i}(o, s, t) = \emptyset$) from s to t, two things could have happened: output o was blocked ($1 - \text{r}(o, n)$), so no HOL packet of this conflict set has moved. Alternatively, o was not blocked ($\text{r}(o, n)$), one of the inputs won the conflict ($1/|\text{ci}(o, s)|$), and a new packet moved to its HOL position ($\text{nfp}(c, n)$) with the same output o as target ($\ell^{\text{l}}_{co}(n)$). Since this could be the case for all inputs of the conflict set, the probabilities have to be added up for each winning input.

State Transition Probabilities for Queue Length DTMCs. The state transition matrix for the queue length of input buffer i at time step n is denoted as $\mathbf{P}^{\text{q}}(i, n)$. The element (j, k) of $p^{\text{q}}_{jk}(i, n)$ gives the probability that the queue length of i changes from j to k at n. The queue length can only increase or decrease by one packet from time step n to $n+1$. Thus, only the following elements of $\mathbf{P}^{\text{q}}(i, n)$ are non-zero:

$$p^{\text{q}}_{jk}(i, n) \geq 0, \text{ if } 0 \leq |j - k| \leq 1 \ .$$

To calculate all non-zero elements of $p^{\text{q}}_{jk}(i, n+1)$ for the next time step $n+1$, all possible HOL states have to be considered since the decision whether a packet can leave this buffer depends on the other input buffers as well. Equations (11)–(18) in Fig. 3 show the state transition probabilities for the queue length DTMCs.

The state transition probability of whether the buffer is empty at time step n and no new packet is coming at time step $n+1$ is given by (11).

If the buffer is empty and a new packet is coming, the state transition probability is equal to the receiving probability of this input buffer (see (12)).

All remaining cases for which the state transition probability is not null are given by (13). All states of S^{h} with their state probability $\nu^{\text{h}}_s(n)$ have to be considered except states where no HOL packet is present at input buffer i ($s \in S^{\text{h}} \mid_{s_i \neq 0}$). The resulting probability has to be normalized with the sum

$$p^{\mathrm{q}}_{00}\left(i,n+1\right)=1-\tilde{q}\left(i,n\right) \tag{11}$$

$$p^{\mathrm{q}}_{01}\left(i,n+1\right)=\tilde{q}\left(i,n\right) \tag{12}$$

$$p^{\mathrm{q}}_{jk|_{j>0}}\left(i,n+1\right)=\frac{1}{\sum\limits_{s\in S^{\mathrm{h}}|_{s_i\neq 0}}\nu^{\mathrm{h}}_{s}(n)}\cdot\sum_{s\in S^{\mathrm{h}}|_{s_i\neq 0}}\nu^{\mathrm{h}}_{s}(n)$$
$$\cdot\begin{cases}\tilde{p}^{q}_{m_{\max}(i)m_{\max}(i)}\left(i,s,n\right), & \text{if } j=k=m_{\max}\left(i\right)\\ \tilde{p}^{q}_{jj}\left(i,s,n\right), & \text{if } j=k<m_{\max}\left(i\right)\\ \tilde{p}^{q}_{j(j-1)}\left(i,s,n\right), & \text{if } j<m_{\max}\left(i\right)\wedge k=j-1\\ \tilde{p}^{q}_{m_{\max}(i)(m_{\max}(i)-1)}\left(i,s,n\right), & \text{if } m_{\max}\left(i\right)=j=k+1\\ \tilde{p}^{q}_{j(j+1)}\left(i,s,n\right), & \text{if } k=j+1\wedge j<m_{\max}\left(i\right)\end{cases} \tag{13}$$

$$\tilde{p}^{q}_{m_{\max}(i)m_{\max}(i)}\left(i,s,n+1\right)=1-r\left(s_i,n\right)+r\left(s_i,n\right)\cdot\frac{\left|\mathrm{ci}\left(s_i,s\right)\right|-1}{\left|\mathrm{ci}\left(s_i,s\right)\right|} \tag{14}$$

$$\begin{aligned}\tilde{p}^{q}_{jj}\left(i,s,n+1\right)=&\left(1-r\left(s_i,n\right)\right)\cdot\left(1-\tilde{q}\left(i,n\right)\right)\\&+r\left(s_i,n\right)\cdot\frac{\left|\mathrm{ci}\left(s_i,s\right)\right|-1}{\left|\mathrm{ci}\left(s_i,s\right)\right|}\cdot\left(1-\tilde{q}\left(i,n\right)\right)+r\left(s_i,n\right)\cdot\frac{1}{\left|\mathrm{ci}\left(s_i,s\right)\right|}\cdot\tilde{q}\left(i,n\right)\end{aligned} \tag{15}$$

$$\tilde{p}^{q}_{j(j-1)}\left(i,s,n+1\right)=\left(1-\tilde{q}\left(i,n\right)\right)\cdot\frac{r\left(s_i,n\right)}{\left|\mathrm{ci}\left(s_i,s\right)\right|} \tag{16}$$

$$\tilde{p}^{q}_{m_{\max}(i)(m_{\max}(i)-1)}\left(i,s,n+1\right)=\frac{r\left(s_i,n\right)}{\left|\mathrm{ci}\left(s_i,s\right)\right|} \tag{17}$$

$$\tilde{p}^{q}_{j(j+1)}\left(i,s,n+1\right)=\tilde{q}\left(i,n\right)\cdot\left(\left(1-r\left(s_i,n\right)\right)+r\left(s_i,n\right)\cdot\frac{\left|\mathrm{ci}\left(s_i,s\right)\right|-1}{\left|\mathrm{ci}\left(s_i,s\right)\right|}\right) \tag{18}$$

Fig. 3. State transition probabilities $\mathbf{P}^{\mathrm{q}}\left(n\right)$ for the queue length Markov chains

of all considered state probabilities. Five different situations are distinguished (defined in (14)–(18)):

The probability for a full buffer ($j=m_{\max}\left(i\right)$) at time step n and a full buffer ($k=m_{\max}\left(i\right)$) at $n+1$ is given by (14). Two situations could have happened: the succeeding buffer is not able to receive a packet ($1-r\left(s_i,n\right)$) or it is able to receive ($r\left(s_i,n\right)$) but the packet of this buffer lost the conflict with another HOL packet ($\left(\left|\mathrm{ci}\left(s_i,s\right)\right|-1\right)/\left|\mathrm{ci}\left(s_i,s\right)\right|$).

If the queue length does not change and the buffer is neither full nor empty, the probability is determined by (15). The following three situations result in an unchanged queue length: First, the HOL packet is blocked ($1-r\left(s_i,n\right)$) because the succeeding buffer is full and no new packet is received ($1-\tilde{q}\left(i,n\right)$). Second, the target is available but this HOL packet loses the conflict to another HOL packet. If no competition is present in this HOL state, the term becomes 0 because $\left|\mathrm{ci}\left(s_i,s\right)\right|$ would be 1.

The last case is similar to the previous case but now this HOL packet wins the conflict and a new packet is coming to this buffer.

The state transition probability for a decreasing queue length, if the queue is not full, is calculated by (16). The queue length can only decrease if no new packet arrives $(1 - \tilde{q}(i,n))$, the output is not blocked $(r(s_i,n))$, and the HOL packet of this buffer wins a potential conflict $(1/|\mathrm{ci}(s_i,s)|)$.

Equation (17) also describes a decreasing queue length but for the case of a full buffer. Here, the probability that a new packet is received has to be omitted $(1 - \tilde{q}(i,n))$ due to the local backpressure mechanism. No packet is accepted at a full buffer.

Finally, the state transition probability for an increased queue length is determined by (18). This situation arises if a packet is received $(\tilde{q}(i,n))$ and the output is blocked $(1 - r(s_i,n))$, or the output is not blocked but the HOL packet lost a conflict $((|\mathrm{ci}(s_i,s)| - 1)/|\mathrm{ci}(s_i,s)|)$.

State Probabilities of the Virtual Output Buffers. As mentioned before, the virtual output buffers are a tool to simplify the calculation of the spatial distribution within the network. Their state is derived from the state probability vector of the HOL Markov chain $\boldsymbol{\nu}^{\mathrm{h}}(n)$ at the switch in question and the receiving probabilities of the succeeding component.

The probability for virtual output buffer o to "hold" a packet from input buffer i at time step n is given by

$$\nu^{\mathrm{o}}_{i|_{0<i\leq i_{\max}}}(o,n) = \mathrm{r}(o,n) \cdot \sum_{s \in S^{\mathrm{h}} \wedge s_i = o} \frac{\nu^{\mathrm{h}}_s(n)}{|\mathrm{ci}(o,s)|} \ .$$

A packet can only enter o if the succeeding component is unblocked $(r(o,n))$. This probability is multiplied with the sum of all HOL states where input i has this output o as target $(s_i = o)$ and the probability for winning a possible conflict with other inputs to the same target $(1/|\mathrm{ci}(o,s)|)$.

The probability for an empty virtual output buffer, i.e., no packet will move through this buffer during time step n, can be calculated in the same manner. Either the succeeding component is blocked $(1 - \mathrm{r}(o,n))$ or if the succeeding component is available $(r(o,n))$, then the state probability $(\nu^{\mathrm{h}}_s(n))$ of all HOL states where no HOL packet is targeted to output o $(s \in S^{\mathrm{h}} \wedge o \notin s)$ have to be considered:

$$\nu^{\mathrm{o}}_0(o,n) = (1 - \mathrm{r}(o,n)) + \mathrm{r}(o,n) \cdot \sum_{s \in S^{\mathrm{h}} \wedge o \notin s} \nu^{\mathrm{h}}_s(n) \ .$$

Routing. To support irregular network topologies and non-uniform traffic, spatial traffic distributions and routing have to be regarded. The probability that a packet wants to proceed from source s to destination d of the network is given by the spatial distribution matrix ℓ^{g}_{sd}. These global routing probabilities

are mapped to local ones for each switch. This local matrix $\ell^{\mathrm{l}}_{io}(n)$ denotes the probabilities that a packet at switch input i moves to output o at time step n.

To achieve this, routing tables at each switch are established. With Dijkstra's algorithm, all possible shortest paths from each source to each destination are calculated. Each path is traversed and at every switch along the path, a routing entry is generated. This entry is identified by the source/destination pair as well as the local input/output pair and consists of the number of redundant paths a packet could take and the probability that a packet moves along this specific path (from s to d via i/o at this switch). The probabilities of the routing entries are iteratively set by getting the probability of the related entry from the preceding switches or sources on the routing path and weighted with the probability that a packet leaves the preceding router on this path (given by $\nu^{\mathrm{o}}_{\mathrm{Pred}}(n)$). If a redundant path is available at a switch, the probability for a routing entry is divided by the number of outputs a packet could take (random selection of paths).

The local probability matrices $\boldsymbol{\ell}^{\mathrm{l}}$ are iteratively determined during the fixed-point iteration by summation of the probabilities of the routing entries for each input/output pair.

3.3 Performance Measures

The steady-state performance of a network can be determined by this model since the Markov chains are ergodic; they are aperiodic and positive recurrent. This property is ensured by having time-independent load and spatial distributions as model inputs.

Network performance is usually determined by the normalized throughput, delay times for packets, and queue lengths. These measures can be determined within the network or at the destinations. The normalized throughput λ_o at output o of a switch is given by the probability that a packet moves through its virtual output buffer: $\lambda_o(n) = 1 - \nu^{\mathrm{o}}_0(o,n)$.

The normalized throughput of input buffer i is then given by the sum of the probabilities that this input sends a packet through a virtual output buffer: $\lambda_i(n) = \sum_{o\in\{1,\ldots,o_{\max}\}} \nu^{\mathrm{o}}_i(o,n)$.

If the switch output o is connected to destination d, the throughput $\lambda_d(n)$ is given by $\lambda_d(n) = \lambda_{\mathrm{Pred}(o)}(n) = 1 - \nu^{\mathrm{o}}_{\mathrm{Pred}(o)}(o,n)$.

The weighted sum of state probabilities of the queue length DTMC determines the mean queue length of input buffer i: $\bar{m}(i,n) = \sum_{j=1}^{m_{\max}(i)} j \cdot \nu^{\mathrm{q}}_j(i,n)$. Little's Law calculates the mean delay of a packet residing in input buffer i within the network: $d(i,n) = \frac{\bar{m}(i,n)}{\lambda_i(n)}$. To get the packet's mean delay at the network destinations, each routing entry also holds, besides its probability, the accumulated mean delay of its path. At the destination, these accumulated delays are weighted with their path probability and then summed.

These steady-state performance measures of the network are calculated by the same iterative method which has been applied to similar models (e.g. [3]): start with an empty network and iterate over the above equations and transition tables until all performance measures reach steady-state. Each time step is

divided into smaller steps to calculate the different equations presented above. These calculations are executed concurrently for all network components within the following minor steps:

1. The State transition probabilities for the queue length of the input buffers $\mathbf{P}^{\mathrm{q}}$ and the state probabilities for the HOL DTMCs $\mathbf{P}^{\mathrm{h}}$ are determined for the next time step.
2. The state vectors $\boldsymbol{\nu}^{\mathrm{q}}$ and $\boldsymbol{\nu}^{\mathrm{h}}$ are calculated for the input buffers and the HOL Markov chain as well as the receiving probabilities of the destinations.
3. During this minor step, the state vectors of the virtual output buffers $\boldsymbol{\nu}^{\mathrm{o}}$ are derived and the mean queue lengths $\bar{m}$ are calculated.
4. Based on the the results of the former steps, the normalized receiving probability $\tilde{q}$ at input buffers in case space is available, the new routing probabilities for each routing entry, and the mean delays are determined.
5. With the results of the former step, the local routing tables $\boldsymbol{\ell}^{\mathrm{l}}$ and the updated delays for the routing entries are calculated.
6. Finally, the delays for each router output are computed.

These minor steps are necessary to support the concurrent calculations without evaluating the topology of the network and some equations are dependent to equations in preceding or succeeding network components.

The intermediate results during iteration show the transient behavior of the network. Up to now, the existence of the fixed point has not been proven. However, all of our tests show that it exists. We have tested the model with a large variety of networks with regular and irregular topologies and traffic distributions ranging from 1×2 to 32×32 networks.

4 Results

The DTMC model is implemented in a tool, called CINAn (Component-based Interconnection Network Analysis). It uses caches and precalculates feasible state transitions at setup to reduce computation time dramatically during fixed-point iteration. During the iteration steps, only non-dynamic data structures are used and the calculation of the state transition probabilities is parallelized. CINAn uses the same file format for network descriptions as CINSim. Thus, CINSim's Graphical User Interface (GUI) (see Fig. 4) can be used for model description and the results of the DTMC model can easily be compared to simulation results.

As an example, we evaluate a 8×8 bidirectional MIN with three stages (shown in Fig. 4). This example shows the support of different switch sizes (4×4 in the first two stages and 2×2 in the last stage) and different route length in the network. The input buffers have a queue length of four packets. This example has eight HOL DTMCs with 625 states each, four HOL DTMCs with nine states each and forty queue length DTMCs with 5 states each.

The spatial distribution of each source is set to $\ell^{\mathrm{g}}_{sd} = \frac{d}{36}$ for Destinations $\{1, \ldots, 8\}$, i.e., Destination 2 is targeted with the probability $\frac{2}{36} = \frac{1}{18}$ and Destination 8 with $\frac{8}{36} = \frac{2}{9}$. The offered load is increased at each run. The results of

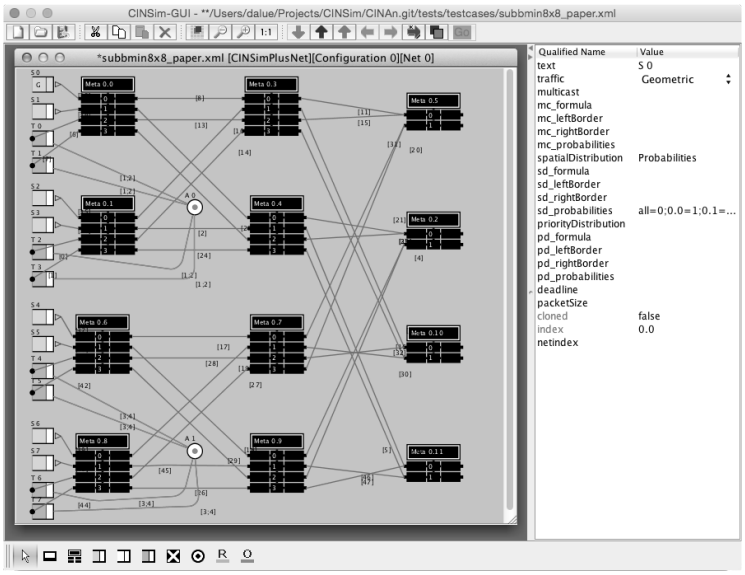

Fig. 4. Screenshot of the GUI of CINSim with a three-stage 8×8 bidirectional MIN

the proposed model are compared with simulation runs and depicted in Fig. 5. The normalized throughput and the mean delay are measured for Destinations 1–4 and Destinations 5–8 separately. The simulation termination criteria was set to a confidence level of 99% and a precision of 0.1.

The results indicate that the normalized throughput of the proposed model matches very well with the simulation. The mean delay also matches very well until the network reaches saturation. In the part of the network where the load is very high (Destinations 5–8), the DTMC model underestimates the mean delay when the network load increases. This is typical for these kind of models and has been reported before (e.g. [3,11]). We expect that these errors are caused by the decomposition of the model.

For Destinations 1–4 another effect predominates. In case a conflict happens at a switch, the DTMC model does not consider a possible redundant routing path for packets that have lost the conflict. CINSim, however, tries to find an alternative path for packets that have lost a conflict. Hence, the DTMC model estimates the mean delay higher if a network has redundant paths.

Fig. 6 shows the finite horizon performance measures of the same network for an offered load of 0.5 for the first fifty time steps. Again, the throughput is close to the simulation and with some more differences for the delay.

The observed errors of the DTMC model are representative to other scenarios we evaluated, for instance, asymmetric topologies, larger networks, etc.

Table 1 shows the runtime and memory usage of the DTMC model and the simulation for the steady-state (Fig. 5) and the finite horizon (Fig. 6) case with an offered load of 0.5, respectively. The results were obtained on a Linux machine

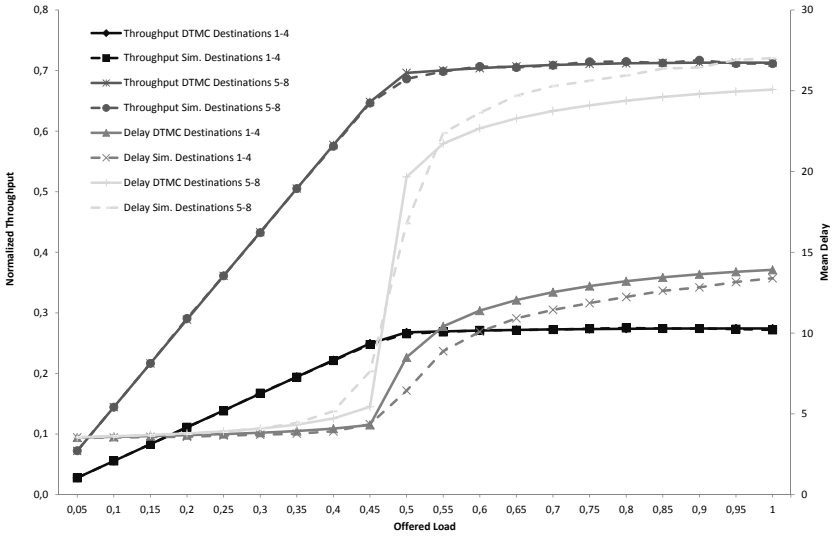

Fig. 5. Steady-state normalized throughput and mean delay of an 8×8 bidirectional MIN shown in Fig. 4 with an asymmetric spatial traffic distribution Results obtained by the DTMC model and simulation.

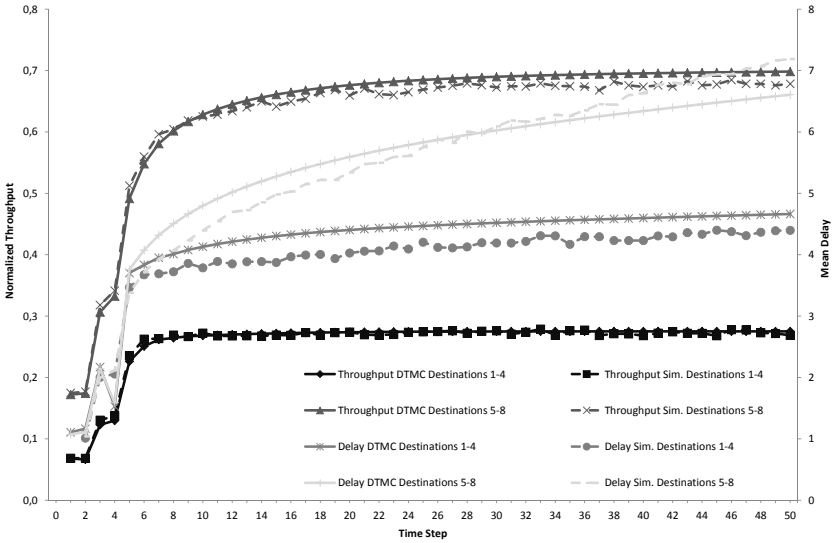

Fig. 6. Normalized throughput and mean delay for time step 0–50 of the 8×8 bidirectional MIN with an asymmetric spatial traffic distribution shown in Fig. 4 with an offered load of 0.5

with two Intel Xeon E5 processors with four cores each with a clock frequency of 3.3 GHz.

Table 1. Runtime and memory usage for results of Fig. 5 and Fig. 6

	Runtime DTMC	Runtime Sim.	Memory DTMC	Memory Sim.
Steady-state	13 s	18 s	195 MiB	73 MiB
Time-depended	2 s	82 s	158 MiB	74 MiB

5 Conclusions

In this paper we presented a new analytical model, based on inhomogeneous discrete time Markov chains, for the performance evaluation of interconnection networks. The model supports arbitrary topologies and asymmetric spatial traffic distributions. It provides results that are close to a simulation of the network. The model was implemented in a tool that reuses the GUI of the simulation framework CINSim.

Due to the complexity of the model, the runtime of the fixed-point iteration is comparable to the simulation, depending on the selected precision of the simulation and the network topology. However, the proposed model directly delivers the time-dependent behavior of the network until it reaches its steady-state in a single run in contrast to simulation, where the simulation restarts repeatedly to gather enough samples for each time step under investigation.

With this model, two approaches exists for performance evaluation during the design process of future onboard computers. This increases confidence in the results. The implementation of the SpaceWire network switches coincides well with the synchronous behavior, which is the basis of the model presented in this paper. Future work will compare actual measurements of the SpaceWire network in different configurations with the results of the presented model.

Currently, the DTMC model is extended to support Wormhole routing by adding an additional Markov chain to keep track of the part of the packet currently at the HOL position at a switch. This model will then be used for optimizations of configurations for future spacecraft onboard computers. Furthermore, other routing algorithms can easily be added to the model, e.g., west-first or xy routing, by replacing Dijkstra's algorithm during model setup.

References

1. Amiri-Zarandi, M., Safaei, F., Roozikhar, M.: Performance evaluation of generic multi-stage interconnection networks with blocking and back-pressure mechanism. The Journal of Supercomputing 71(3), 1038–1066 (March 2015)
2. Atiquzzaman, M., Akhtar, M.: Performance of buffered multistage interconnection networks in a nonuniform traffic environment. Journal of Parallel and Distributed Computing 30(1), 52 – 63 (1995)
3. Dias, D., Jump, J.: Analysis and simulation of buffered delta networks. Computers, IEEE Transactions on C-30(4), 273–282 (April 1981)
4. Hamid, N., Walters, R.J., Wills, G.B.: An analytical model of multi-core multi-cluster architecture (MCMCA). Open Journal of Cloud Computing (OJCC) 2(1), 1–12 (2015)

5. Kiasari, A., Lu, Z., Jantsch, A.: An analytical latency model for networks-on-chip. Very Large Scale Integration (VLSI) Systems, IEEE Transactions on 21(1), 113–123 (Jan 2013)
6. Lüdtke, D., Westerdorff, K., Stohlmann, K., Börner, A., Maibaum, O., Peng, T., Weps, B., Fey, G., Gerndt, A.: OBC-NG: Towards a reconfigurable on-board computing architecture for spacecraft. In: Aerospace Conference, 2014 IEEE. pp. 1–13 (March 2014)
7. Lüdtke, D., Tutsch, D.: The modeling power of CINSim: Performance evaluation of interconnection networks. Computer Networks 53(8), 1274–1288 (Jun 2009)
8. Moadeli, M., Shahrabi, A., Vanderbauwhede, W., Ould-Khaoua, M.: An analytical performance model for the spidergon noc. In: Advanced Information Networking and Applications, 2007. AINA '07. 21st International Conference on. pp. 1014–1021 (May 2007)
9. Parkes, S., Armbruster, P.: SpaceWire: a spacecraft onboard network for real-time communications. In: Real Time Conference. 14th IEEE-NPSS. pp. 6–10 (2005)
10. Sabbaghi-Nadooshan, R., Patooghy, A.: Analytical performance modeling of de bruijn inspired mesh-based network-on-chips. Microprocessors and Microsystems 39(1), 27 – 36 (2015)
11. Tutsch, D., Hommel, G.: Generating systems of equations for performance evaluation of multistage interconnection networks. Journal of Parallel and Distributed Computing (JPDC) 62(2), 228–240 (2002)
12. Yoon, H., Lee, K.Y., Liu, M.T.: Performance analysis of multibuffered packet-switching networks in multiprocessor systems. IEEE Transactions on Computers 29(3), 319–327 (Mar 1990)
13. Youn, H.Y., Mun, Y.: On multistage interconnection networks with small clock cycles. Parallel and Distributed Systems, IEEE Transactions on 6(1), 86–93 (Jan 1995)
14. Zhou, B., Atiquzzaman, M.: Efficient analysis of multistage interconnection networks using finite output-buffered switching elements. Computer Networks and ISDN Systems 28(13), 1809 – 1829 (1996)