On the Semantics of Multistage Interconnection Networks*

Anna Gambin** and Sławomir Lasota

Fachbereich Informatik LS-2, Universität Dortmund, D-44221 Dortmund , Germany.
Instytut Informatyki, Uniwersytet Warszawski, Banacha 2, Warszawa 02-097, Poland.
{aniag,sl}@mimuw.edu.pl

Abstract. Multistage interconnection networks (MINs) have a number of applications in many areas, for example in communication networks or parallel computer systems. While several performance analyses have been done, as far as we know a formal description of the behaviour of such networks is lacking. In the paper we define Markov chains describing the behaviour of the buffered MIN with a butterfly interconnection structure and 2×2 switching elements. We study several models of packet flow, which appear in the literature and develop technical mathematical tools which allow us to compare them. We prove some equivalence results regarding the models.

1 Introduction

Multistage interconnection networks (MIN) have been widely used for connecting processors in parallel computing systems and in constructing communication networks [7]. It consists (in general) of $(k \times k)$-switching elements (*switches*), which are grouped into several stages. In this paper we are concerned with MINs having the butterfly interconnection structure containing (2×2)-switches with input buffers of capacity one (see fig. 1). They belong to the broader class of *banyan networks*.

A number of papers studied the behaviour of MINs, aiming mainly at evaluation of their performance [1] [3, 5]. In most of these papers some crucial assumptions (relaxations) are made, simplifying the analysis of the network. The reason is that the behaviour of MIN, arising from the dependencies between neighbour switches, is very complex.

In this paper we propose the formal definition of the dynamic behaviour of MIN without making any simplifying assumptions, i.e. in the *full blocking model*. As far as we know, no such definitions have been proposed by now. We describe the network by means of Markov chain, as its behaviour is history-independent. We develop some mathematical machinery which allows us to define the semantics

* This work was performed at the University of Dortmund.
** Research supported by DAAD (Deutscher Akademischer Austauschdienst).
[1] The most frequently evaluated performance measure is *throughput* (expected number of packets reaching their destinations per time step).

of several models of packet flow (including also some relaxed ones [6]). Such a formal approach helps to express differences between models – it appears to us, that so precise comparison was not possible before. Using proposed notation we are able to prove certain equivalence of models.

1.1 Models of a packet flow

MINs are used to route packets appearing in the switches at the first stage (input switches) to appropriate output switches (addresses) at the last stage (see fig. 1). There is exactly one path from every input switch to every output switch, and every packet visits all the stages along its way through the network. We assume, following many related papers ([4, 8]) that an address of a packet is random (it is usually done to make a performance analysis possible), – it means that a packet after entering into each switch chooses its direction randomly. The computation of the network proceeds in synchronized steps more or less simultaneously by all the switches. One step consists of the flow of packets between every pair of neighbour stages. Locally, in one switch, such a step consists of two half-steps:

1) *Flow-out phase*: some packets leave the switch (after resolving conflicts), but only if there is a place in the appropriate buffer of the switch at the next stage.

2) *Input phase*: packets from the previous stage enter to the empty buffers of the switch.

The *ripple model* is the most frequently used in the literature [5, 8]. It has the property that the flow-out phase at the stage happens after the flow-out phase at the next stage has been completed. We can effectively simulate such a model (switches need not to wait). There is an equivalent *odd-even model* (pipelining implementation of the ripple model): the flow-out phases occur simultaneously at the odd stages (while even stages are idle), followed by input phases at the even stages, then in every consecutive step the odd and even stages change their roles. Another natural possibility is offered by the *simultaneous model*, in which the flow-out phase occurs simultaneously at all the stages (similarly input phase which follows flow-out one). In this model switches are never idle.

The paper is organized as follows. In section 2 we define three Markov chains corresponding to the models mentioned above. We start by defining the local behaviour of one switch in section 2.1, then in section 2.2 we define the global-state Markov chains. To this end some mathematical tools are developed – the properties of them are discused in [1], which contains also some general theorems regarding these tools. In section 3 we prove the main result of this paper, giving the equivalence of simultaneous and odd-even models. The formal proof of the key lemma can be find in [1]. We finish with conclusions and a sketch of some directions for the further work in the area in section 4.

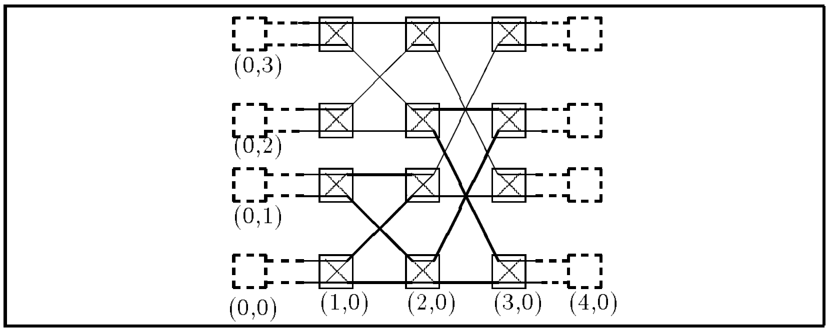

Fig. 1. Butterfly network for $H = 3$ with border stages (dashed line).

2 Semantics of MIN

Throughout the rest of the paper let H denote the number of stages of the butterfly network. The structure of the network can be represented by the graph $G = (I, E)$. The sets of nodes and edges are defined as follows:

$$I = \{1, 2, \ldots, H\} \times \{0, 1, \ldots, W\}, \qquad W = 2^{H-1} - 1,$$

$$E = \{((r, c), (r+1, c')) : 1 \leq r \leq H-1, \quad 0 \leq c \leq W, \quad c' = c \text{ or } c' = \bar{c}^r\}$$

where $\bar{c}^i$ is a number c with its i-th bit negated, $0 \leq c \leq W$, $1 \leq i \leq H-1$. Nodes (r, c) and $(r, \bar{c}^r)$ have common neighbours $(r+1, c)$ and $(r+1, \bar{c}^r)$ at the next $(r+1)$-th stage. In further definitions we will need the notion of c-map :

Definition 1. For finite sets A,B, *c-map* $P : A \rightsquigarrow B$ is a function $P : A \to [0,1]^B$ such that for every $a \in A$, $P(a)$ is a probabilistic distribution, i.e. $\sum_{b \in B} P(a)(b) = 1$.

Note: C-map is precisely the conditional probabilistic distribution. For $A = B$, P defines a transition matrix of a Markov chain.

2.1 Local description of one switch

As a single switching element cannot be described by a Markov chain $S \rightsquigarrow S$ (because its behaviour depends on states of its neighbours), we encode the behaviour of it by means of two c-maps :

$$\boldsymbol{\sigma} : S \times \{0,1\}^2 \rightsquigarrow S \times \{0,1\}^2, \qquad \boldsymbol{\tau} : S \times \{0,1\}^2 \rightsquigarrow S$$

(see tables 1, 2). Such a description of one switch is in the spirit of probabilistic automaton from [6]. Elements of $S = B \times B$ represent states of both buffers of a switch, $B = \{\circ, \downarrow, \searrow\}$. Element "$\circ$" represents empty buffer, and two other elements of B represent two possible directions of a packet staying in the buffer. We introduce "$\swarrow$"="$\searrow$" only to improve a readability. $\boldsymbol{\sigma}(s, (b_1, b_2))$ "returns"

S	in:1, 1	out	in: 0, 1	out	in: 1, 0	out	in: 0, 0	out
$\circ\ \circ$	$\circ\ \circ$	0, 0	$\circ\ \circ$	0, 0	$\circ\ \circ$	0, 0	$\circ\ \circ$	0, 0
$\downarrow\circ$	$\downarrow\circ$	0, 0	$\circ\ \circ$	1, 0	$\downarrow\circ$	0, 0	$\circ\ \circ$	1, 0
$\searrow\circ$	$\searrow\circ$	0, 0	$\searrow\circ$	0, 0	$\circ\ \circ$	0, 1	$\circ\ \circ$	0, 1
$\circ\downarrow$	$\circ\downarrow$	0, 0	$\circ\downarrow$	0, 0	$\circ\ \circ$	0, 1	$\circ\ \circ$	0, 1
$\circ\swarrow$	$\circ\swarrow$	0, 0	$\circ\ \circ$	1, 0	$\circ\swarrow$	0, 0	$\circ\ \circ$	1, 0
$\downarrow\swarrow$	$\downarrow\swarrow$	0, 0	$\frac{1}{2}\downarrow\circ,\ \frac{1}{2}\circ\swarrow$	1, 0	$\downarrow\swarrow$	0, 0	$\frac{1}{2}\downarrow\circ,\ \frac{1}{2}\circ\swarrow$	1, 0
$\searrow\downarrow$	$\searrow\downarrow$	0, 0	$\searrow\downarrow$	0, 0	$\frac{1}{2}\circ\downarrow,\ \frac{1}{2}\searrow\circ$	0, 1	$\frac{1}{2}\circ\downarrow,\ \frac{1}{2}\searrow\circ$	0, 1
$\swarrow\!\!\!\searrow$	$\swarrow\!\!\!\searrow$	0, 0	$\searrow\circ$	1, 0	$\circ\swarrow$	0, 1	$\circ\ \circ$	1, 1
$\downarrow\downarrow$	$\downarrow\downarrow$	0, 0	$\circ\downarrow$	1, 0	$\downarrow\circ$	0, 1	$\circ\ \circ$	1, 1

Table 1. Definition of $\boldsymbol{\sigma}$: the first half of the step (flow out of packets)

S	in: 0, 0	in: 1, 0	in: 0, 1	in: 1, 1
$\circ\ \circ$	$\circ\ \circ$	$\frac{1}{2}\downarrow\circ,\ \frac{1}{2}\searrow\circ$	$\frac{1}{2}\circ\downarrow,\ \frac{1}{2}\circ\swarrow$	$\frac{1}{4}\downarrow\swarrow,\ \frac{1}{4}\searrow\downarrow,\ \frac{1}{4}\swarrow\!\!\!\searrow\ \frac{1}{4}\downarrow\downarrow$
$\downarrow\circ$	$\downarrow\circ$		$\frac{1}{2}\downarrow\swarrow,\ \frac{1}{2}\downarrow\downarrow$	
$\searrow\circ$	$\searrow\circ$		$\frac{1}{2}\searrow\downarrow,\ \frac{1}{2}\swarrow\!\!\!\searrow$	
$\circ\downarrow$	$\circ\downarrow$	$\frac{1}{2}\searrow\downarrow,\ \frac{1}{2}\downarrow\downarrow$		
$\circ\swarrow$	$\circ\swarrow$	$\frac{1}{2}\downarrow\swarrow,\ \frac{1}{2}\swarrow\!\!\!\searrow$		
$\downarrow\swarrow$	$\downarrow\swarrow$			
$\searrow\downarrow$	$\searrow\downarrow$			
$\swarrow\!\!\!\searrow$	$\swarrow\!\!\!\searrow$			
$\downarrow\downarrow$	$\downarrow\downarrow$			

Table 2. Definition of $\boldsymbol{\tau}$: the second half of the step: (input of packets). In empty fields $\boldsymbol{\tau}$ can be arbitrarily defined

an output of packets from a switch in a state s (together with a new state) when (b_1, b_2) describes whether two buffers of neighbours at the next stage are empty, i.e. $b_j = 0$ denotes a free buffer. $\boldsymbol{\tau}(s, (i_1, i_2))$ describes an input of packets into a switch in a state s, assuming that $i_j = 1$ means that a packet is comming into a buffer.

For convenience let us introduce sets $I_r = \{(r, c) : 0 \le c \le W\}$, for $0 \le r \le H + 1$. It represents nodes at the r-th stage. In the next section we will define Markov chains $V \rightsquigarrow V$, describing one step of the computation of the network, where $V = S^{I \cup I_{H+1}}$ is a set of its states, $I = \bigcup_{1 \le r \le H} I_r$. We will also need a set V' of states in a half-step defined as $V' = (S \times \{0,1\}^2)^{I_0 \cup I \cup I_{H+1}}$. Stages 0 and $H + 1$ are introduced only for technical convenience. The stage 0 intuitively represents input of packets into the network. Symmetrically stage $(H+1)$ receives packets which flow out from the network. For further definitions we need two families of auxiliary functions:

$$IN^{r,c} : V \to \{0,1\}^2, \quad \text{for} \quad (r,c) \in I \cup I_0$$

$$in^{r,c} : V' \to \{0,1\}^2, \quad \text{for} \quad (r,c) \in I \cup I_{H+1}$$

$IN^{r,c}(v)$ simply returns an information about buffers of neighbours at the next stage for a switch at node (r,c) (the pair (b_1,b_2) above); similarly $in^{r,c}(v)$ returns an information about packets comming into the switch ((i_1,i_2) above). In the definition of these functions we are "looking through" the edges of graph G:

$$IN^{r,c}(v) = \begin{cases} (is(s{\downarrow_1}), is(\bar{s}{\downarrow_1})) \text{ if } c < \bar{c}^r \\ (is(\bar{s}{\downarrow_2}), is(s{\downarrow_2})) \text{ otherwise} \end{cases} \quad \text{for } 0 < r < H$$

where $s = v(r+1,c) \in B^2$, $\bar{s} = v(r+1,\bar{c}^r) \in B^2$. $(a,b){\downarrow_1} = a$, $(a,b){\downarrow_2} = b$ denote usual projections. Auxiliary function is tells us whether the buffer is occupied; $is : B \to \{0,1\}$, $is(\circ) = 0$, $is(\downarrow) = is(\searrow) = 1$.

$$in^{r,c}(v') = \begin{cases} (o{\downarrow_1}, \bar{o}{\downarrow_1}) \text{ if } c < \bar{c}^{(r-1)} \\ (\bar{o}{\downarrow_2}, o{\downarrow_2}) \text{ otherwise} \end{cases} \quad \text{for } 1 < r < H+1$$

where $o = v'(r-1,c){\downarrow_2} \in \{0,1\}^2$, $\bar{o} = v'(r-1,\bar{c}^{(r-1)}){\downarrow_2} \in \{0,1\}^2$. The definitions of $IN^{0,c}$,$in^{1,c}$, $IN^{H,c}$ and $in^{H+1,c}$ are slightly different, as they describe the "border" of the network:

$$IN^{r,c}(v) = (is(v(r+1,c){\downarrow_1}), is(v(r+1,c){\downarrow_2})), \quad \text{for } r = 0, H$$

$$in^{r+1,c}(v') = v'(r,c){\downarrow_2} \in \{0,1\}^2, \quad \text{for } r = 0, H$$

We define four families of c-maps $\boldsymbol{\sigma^{r,c}}$,$\boldsymbol{\tau^{r,c}}$, $\pi^{r,c}$,$\Omega^{r,c}$, indexed by $(r,c) \in I$, which will be the atomic elements of proper Markov chains in the next section. The c-maps

$$\boldsymbol{\sigma^{r,c}} : V \rightsquigarrow S \times \{0,1\}^2, \qquad \boldsymbol{\tau^{r,c}} : V' \rightsquigarrow S$$

simply apply $\boldsymbol{\sigma}$,$\boldsymbol{\tau}$ to the switch at node (r,c):

$$\boldsymbol{\sigma^{r,c}}(v)(s,o) = \boldsymbol{\sigma}(v(r,c), IN^{r,c}(v))(s,o), \qquad \boldsymbol{\tau^{r,c}}(v')(s) = \boldsymbol{\tau}(v'(r,c), in^{r,c}(v'))(s)$$

On the other hand $\pi^{r,c} : V \rightsquigarrow S \times \{0,1\}^2$, $\Omega^{r,c} : V' \rightsquigarrow S$ describe idle phase of a switch at node (r,c) and are defined by:

$$\pi^{r,c}(v)(v(r,c),(0,0)) = 1, \qquad \Omega^{r,c}(v')(v'(r,c){\downarrow_1}) = 1$$

To unify our notation we will also need some "border" c-maps describing input and output of packets into/from the network:

$$\boldsymbol{\sigma^{0,c}}, \pi^{0,c}, \boldsymbol{\sigma^{H+1,c}}, \pi^{H+1,c}, : V \rightsquigarrow S \times \{0,1\}^2, \qquad \boldsymbol{\tau^{H+1,c}}, \Omega^{H+1,c} : V' \rightsquigarrow S$$

defined as follows:

$$\boldsymbol{\sigma^{0,c}}(v)(s,o) = \boldsymbol{\sigma}((\downarrow,\downarrow), IN^{0,c}(v))(s,o), \quad \pi^{0,c}(v)((\downarrow,\downarrow),(0,0)) = 1,$$

$$\boldsymbol{\sigma^{H+1,c}}(v)((\circ,\circ),(0,0)) = 1, \quad \pi^{H+1,c}(v)(v(r,c),(0,0)) = 1,$$

$$\boldsymbol{\tau^{H+1,c}}(v')(s) = \boldsymbol{\tau}(v'(H+1,c), in^{H+1,c}(v'))(s), \quad \Omega^{H+1,c}(v')(v'(H+1,c){\downarrow_1}) = 1$$

2.2 Semantics of the network

In this section we define three Markov chains $\Gamma_{\text{ripple}}, \Gamma_{\text{sim}}, \Gamma_{\text{odd-even}} : V \rightsquigarrow V$, which correspond to the described models of a packet flow. The Markov chains will be composed of c-maps $\boldsymbol{\sigma}^{\boldsymbol{r},\boldsymbol{c}}$,$\boldsymbol{\tau}^{\boldsymbol{r},\boldsymbol{c}}$,$\pi^{r,c}$,$\Omega^{r,c}$ defined in the previous section by means of the following two operations on c-maps :

Definition 2. For $P : A \rightsquigarrow B$ and $Q : B \rightsquigarrow C$, we define a **sequential composition of** P **and** Q, $P;Q : A \rightsquigarrow C$ as follows: $(P;Q)(a,c) = \sum_{b\in B} P(a,b) \cdot Q(b,c)$.
For $P : A \rightsquigarrow B$ and $Q : A \rightsquigarrow C$, a **parallel composition of** P **and** Q, $P \parallel Q : A \rightsquigarrow B \times C$, is defined as $(P \parallel Q)(a)(b,c) = P(a)(b) \cdot Q(a)(c)$.

Note: Essentially "**;**" corresponds to matrix multiplication, while "$\parallel$" gives a product distribution. Both operations are associative and "$\parallel$" is also commutative. While defining them we were inspired by the operators of CSP ([2]) – for example "$\parallel$" enforces component c-maps to "proceed" simultaneously like the operator of synchronized parallel composition in CSP. In the following "$\parallel$" has a higher priority than "**;**". We sometimes write $\binom{P;}{Q}$ instead of $(P;Q)$ for the sake of simplicity (this allows us to better recognize c-maps concerned with certain stages of the network).

We group c-maps $\boldsymbol{\sigma}^{\boldsymbol{r},\boldsymbol{c}}$ from the r-th stage into $\boldsymbol{\sigma}^{\boldsymbol{r}} : V \rightsquigarrow (S \times \{0,1\}^2)^{I_r}$

$$\boldsymbol{\sigma}^{\boldsymbol{r}} = \|_{0\le c\le W}\ \boldsymbol{\sigma}^{\boldsymbol{r},\boldsymbol{c}}, \text{ for } r = 0,1,\ldots H+1$$

Analogously we define $\boldsymbol{\tau}^{\boldsymbol{r}}$,$\pi^r$,$\Omega^r$ as a parallel composition of appropriate components.

In the following we ignore the subtle difference between sets $A^{B\cup C}$ and $A^B \times A^C$, for example by treating a c-map with a domain equal to the former as if it has a domain equal to the latter. It is useful in definitions below. The three Markov chains are as follows:

$$\begin{aligned}\Gamma_{\text{sim}} = (&\boldsymbol{\sigma}^{\mathbf{0}} \parallel \boldsymbol{\sigma}^{\mathbf{1}} \parallel \boldsymbol{\sigma}^{\mathbf{2}} \parallel \boldsymbol{\sigma}^{\mathbf{3}} \parallel \ldots \parallel \boldsymbol{\sigma}^{\boldsymbol{H-2}} \parallel \boldsymbol{\sigma}^{\boldsymbol{H-1}} \parallel \boldsymbol{\sigma}^{\boldsymbol{H}} \parallel \boldsymbol{\sigma}^{\boldsymbol{H+1}});\\ &(\boldsymbol{\tau}^{\mathbf{1}} \parallel \boldsymbol{\tau}^{\mathbf{2}} \parallel \boldsymbol{\tau}^{\mathbf{3}} \parallel \ldots \parallel \boldsymbol{\tau}^{\boldsymbol{H-2}} \parallel \boldsymbol{\tau}^{\boldsymbol{H-1}} \parallel \boldsymbol{\tau}^{\boldsymbol{H}} \parallel \boldsymbol{\tau}^{\boldsymbol{H+1}})\end{aligned}$$

$$\begin{aligned}\Gamma_{\text{ripple}} = &(\pi^0 \parallel \pi^1 \parallel \pi^2 \parallel \ldots \parallel \pi^{H-1} \parallel \pi^H \parallel \boldsymbol{\sigma}^{\boldsymbol{H+1}});\\ &\quad(\Omega^1 \parallel \Omega^2 \parallel \ldots \parallel \Omega^{H-1} \parallel \Omega^H \parallel \Omega^{H+1});\\ &(\pi^0 \parallel \pi^1 \parallel \pi^2 \parallel \ldots \parallel \pi^{H-1} \parallel \boldsymbol{\sigma}^{\boldsymbol{H}} \parallel \pi^{H+1});\\ &\quad(\Omega^1 \parallel \Omega^2 \parallel \ldots \parallel \Omega^{H-1} \parallel \Omega^H \parallel \boldsymbol{\tau}^{\boldsymbol{H+1}});\\ &(\pi^0 \parallel \pi^1 \parallel \pi^2 \parallel \ldots \parallel \boldsymbol{\sigma}^{\boldsymbol{H-1}} \parallel \pi^H \parallel \pi^{H+1});\\ &\quad(\Omega^1 \parallel \Omega^2 \parallel \ldots \parallel \Omega^{H-1} \parallel \boldsymbol{\tau}^{\boldsymbol{H}} \parallel \Omega^{H+1});\\ &\quad\ldots\\ &(\boldsymbol{\sigma}^{\mathbf{0}} \parallel \pi^1 \parallel \pi^2 \parallel \ldots \parallel \pi^{H-1} \parallel \pi^H \parallel \pi^{H+1});\\ &\quad(\boldsymbol{\tau}^{\mathbf{1}} \parallel \Omega^2 \parallel \ldots \parallel \Omega^{H-1} \parallel \Omega^H \parallel \Omega^{H+1})\end{aligned}$$

$$\begin{aligned}\Gamma_{\text{odd-even}} = {} & (\boldsymbol{\sigma^0} \parallel \pi^1 \parallel \boldsymbol{\sigma^2} \parallel \pi^3 \parallel \ldots \parallel \boldsymbol{\sigma^{H-2}} \parallel \pi^{H-1} \parallel \boldsymbol{\sigma^H} \parallel \pi^{H+1});\\ & (\boldsymbol{\tau^1} \parallel \Omega^2 \parallel \boldsymbol{\tau^3} \parallel \ldots \parallel \Omega^{H-2} \parallel \boldsymbol{\tau^{H-1}} \parallel \Omega^H \parallel \boldsymbol{\tau^{H+1}});\\ & (\pi^0 \parallel \boldsymbol{\sigma^1} \parallel \pi^2 \parallel \boldsymbol{\sigma^3} \parallel \ldots \parallel \pi^{H-2} \parallel \boldsymbol{\sigma^{H-1}} \parallel \pi^H \parallel \boldsymbol{\sigma^{H+1}});\\ & (\Omega^1 \parallel \boldsymbol{\tau^2} \parallel \Omega^3 \parallel \ldots \parallel \boldsymbol{\tau^{H-2}} \parallel \Omega^{H-1} \parallel \boldsymbol{\tau^H} \parallel \Omega^{H+1})\end{aligned}$$

Let $\Gamma_1 : V \rightsquigarrow V$ denote the c-map consisting of two first lines of the definition of $\Gamma_{\text{odd-even}}$, $\Gamma_2^1 : V \rightsquigarrow V'$ denote the third line, $\Gamma_2^2 : V' \rightsquigarrow V$ denote the fourth, and $\Gamma_2 = \Gamma_2^1;\Gamma_2^2$. Analogously $\Gamma_{\text{sim}} = \Gamma_{\text{sim}}^1;\Gamma_{\text{sim}}^2$.

Note: The definition of $\Gamma_{\text{odd-even}}$ assumes that H is even (for H odd one should exchange the beginning parts of Γ_1 and Γ_2).

3 Equivalence results

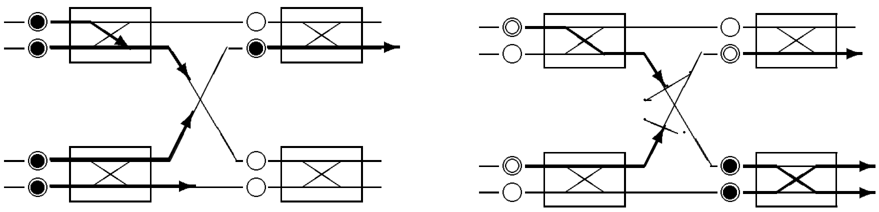

Fig. 2. One even group of switches before and after Γ_1; on the right side black packet moved during Γ_1, while doubled ones did not move.

Lemma 3. $\Gamma_1;\Gamma_{sim} = \Gamma_1;\Gamma_2$.

The formal proof, which is involved, can be found in [1]. Here we present some intuitive arguments. All the switches can be divided into groups of four (nodes (r, c), $(r, \bar{c}^r)$, $(r+1, c)$, $(r+1, \bar{c}^r)$). We call a group *even* or *odd*, depending on parity of r. Each switch belongs to precisely two groups (fig. 2 shows an even group). During Γ_1 even groups make their work, while odd ones stay idle. During Γ_2 role changes: only odd groups perform their actions. Odd groups during Γ_{sim} do precisely the same as during Γ_2. We only should observe that even groups after Γ_1, during Γ_{sim} do nothing more. This can be established by careful analysis of possible cases. As in Γ_1 groups operate independently, we will consider one group. We should only think about non-empty buffers at r-th stage switch. Let us consider one such a buffer. The packet staying in it could not have come during Γ_1 as odd groups were idle. There are two possible causes why it did not move forward during Γ_1. If the reason was stage blocking, i.e. another packet staying in the appropriate buffer at the next stage, then this packet still remains in its buffer (the argument as before). If the reason was a collision in the switch, then our packet is now blocked by the winner. In both cases our packet can not move forward.

The proof of the following lemma is analogous to the proof of lemma 3.

Lemma 4. $\Gamma_2;\Gamma_{sim} = \Gamma_2;\Gamma_1$.

Theorem 5. $\Gamma^k_{odd\text{-}even} = \Gamma_1;\Gamma^{2k-1}_{sim}$.

Proof. We apply interchangeably lemmas 3 and 4 to obtain the required equality. $\Gamma^k_{\text{odd-even}} = (\Gamma_1;\Gamma_2)^k \overset{L.3}{=} (\Gamma_1;\Gamma_2)^{k-1};\Gamma_1;\Gamma_{\text{sim}} \overset{L.4}{=} \ldots \overset{L.4}{=} \Gamma_1;\Gamma^{2k-1}_{\text{sim}}$

4 Conclusions

In this paper we have presented the formal description of several models of packet flow in the butterfly MINs. By varying the definitions of c-maps $\boldsymbol{\sigma}$ and $\boldsymbol{\tau}$ one can obtain the number of different models, without introducing any changes in the main definitions. For example the relaxed model of [6] can be easily obtained in this way. It is interesting to observe how the correspondence between ripple, odd-even and simultaneous changes when we modify $\boldsymbol{\sigma}$ and $\boldsymbol{\tau}$.

The other possible continuation of the work is to examine the usefulness of the presented notation in the analysis of the performance of MIN, what is a challenging problem, because there are no results in this field for the full blocking MIN model with finite buffers.

Acknowledgement. Thanks are due to Martin Dietzfelbinger for helpful discussions and his comments.

References

1. M. Dietzfelbinger, A. Gambin, S. Lasota *On Different Models for Packet Flow in Multistage Interconnection Networks*, Dortmund University, Technical Raport Nr. 625, 1996
2. C.A.R. Hoare, "Communicating Sequential Processes," Prentice – Hall, 1985.
3. R.R. Koch, *Increasing the Size of a Network by a Constant Factor Can Increase Performance by More Than a Constant Factor*, Proc. 28th Ann. Symposium on Foundations of Computer Science (1988), 221-231.
4. T.F. Leighton, "Parallel Algorithms and Architectures: Arrays, Trees, Hypercubes," Morgan Kaufmann, 1992.
5. Y. Mun, H. Yong, *Performance Analysis of Finite Buffered Multistage Interconnection Networks*, IEEE Trans. on Computers, vol. C-43, 1994, pp. 152-162
6. R. Rehrmann, B. Monien, R. Lüling, R. Diekmann, *On the Communication Throughput of Buffered Multistage Interconnection Networks*, Proc. 8th ACM Symposium on Parallel Algorithms and Architectures, 1996.
7. R. Rooholaminiight, *Finding the Right ATM Switch for the Market*, IEEE J. Comp., 1994, pp. 16-28
8. T.H. Theimer, E.P. Rathgeb, M.N. Huber, *Performance Analysis of Buffered Banyan Networks* IEEE Trans. on Communication, vol. C-39, 1991, pp. 269-277