A novel look-ahead routing algorithm based on graph theory for triplet-based network-on-chip router

Xu Chen, Feng Shi\textsuperscript{a)}, Fei Yin, and Xiaojun Wang
Beijing Institute of Technology, Beijing 100081, China
\textsuperscript{a)} bitsf@bit.edu.cn

Abstract: The WK-recursive network well conforms to implementation of large distributed systems due to its many favorable properties. In this paper, we employ the concept $S_3$ group to establish an efficient look-ahead routing algorithm for the triplet-based WK-recursive network. Remarkably, we introduce a data flow model to categorize message traffic as six flow models. For the sake of simplicity, we further leverage the permutation group $S_3$ to transform different flow models to one model and hence routing computing can be performed in the same model. It is demonstrated that our proposed design can achieve better network performance.

Keywords: routing algorithm, network-on-chip, look-ahead routing, graph theory, multicore system

Classification: Integrated circuits

References

[1] S. Hu, \textit{et al.}: “Hybrid WK-recursive on-chip network for multi-core system,” Electron. Lett. \textbf{53} (2017) 839 (DOI: 10.1049/el.2017.0441).

[2] D. Wentzloff, \textit{et al.}: “On-chip interconnection architecture of the tile processor,” IEEE Micro \textbf{27} (2007) 15 (DOI: 10.1109/MM.2007.4378780).

[3] W. J. Dally and B. Towles: \textit{Principles and Practices of Interconnection Networks} (Morgan Kaufmann, New York, 2004).

[4] A. Monemi, \textit{et al.}: “Improved flow control for minimal fully adaptive routing in 2D mesh NoC,” International Workshop on Network on Chip Architectures. ACM (2016) 9 (DOI: 10.1145/2994133.2994134).

[5] N. Najib, \textit{et al.}: “Partially adaptive look-ahead routing for low latency network-on-chip,” Research and Development IEEE (2015) 1 (DOI: 10.1109/SCORED.2014.7072956).

[6] A. Kumar, \textit{et al.}: “Toward ideal on-chip communication using express virtual channels,” IEEE Micro \textbf{28} (2008) 80 (DOI: 10.1109/MM.2008.18).

[7] L. S. Peh and W. J. Dally: “A delay model and speculative architecture for pipelined routers,” The Seventh International Symposium on High-Performance Computer Architecture IEEE (2001) 255.

[8] B. Qiao, \textit{et al.}: “A new hierarchical interconnection network for multi-core processor,” 2nd IEEE Conference on Industrial Electronics and Applications (2007) 246 (DOI: 10.1109/ICIEA.2007.4318408).

[9] W. Zuo and S. Feng: “A shortest path routing algorithm in triplet-based network,” Transactions of the Beijing Institute of Technology, School of
1 Introduction

The triplet-based WK-recursive network (TriBA) is a subclass of WK-recursive networks [1], which has many desirable properties such as symmetry, scalability and hierarchy. Whereas the network topology [2] determines the ideal performance of a network-on-chip (NoC), routing is one of the key factors that determines how much of this potential is realized [3]. An efficient routing algorithm [4, 5] renders hop counts of routed packets as few as possible, leading to reduce the overall network latency. Further, a well-designed routing mechanism has an important impact on how to optimize the router microarchitecture [6], e.g., look-ahead routing strategy [7] can be used to optimized the router pipeline stages, resulting in significantly reducing the communication delay.

Previous studies have proposed a number of routing algorithms for TriBA, such as DDRA [8], Min-DDRA [9] and SPORT [10]. DDRA adopts the distributed routing, which computes the output port at each hop based on the relative position of the region in which the target node is located. However, it cannot always provide shortest-path for routing. Min-DDRA creates a routing table for every router to record the intermediate node to the other nodes. Although the shortest path is obtained, there exists a main disadvantage in this table-lookup deterministic routing algorithm. When a TriBA network expands to a larger scale, the routing tables must be updated. Moreover, the routing table stored in each node is more difficult to be obtained especially in large scale TriBA. SPORT is a source routing algorithm which uses recursion to predetermine complete routing path. The computational complexity is relative high. Moreover, the whole routing information should be attached on the head flit, leading to increase the packet size and decrease the network payload.

In this paper, we propose an efficient look-ahead routing algorithm for TriBA NoC (LA4T) with the following attributes. Our proposal always routes the message along the minimal path. Unlike source routing algorithm, it adopt the distributed routing which selects the output port at each hop simply depend on the destination address, the size of the header routing information can be reduced, resulting increasing the network payload. Otherwise, it is orthogonal to the scale of the network and hence can be easily extended to different sizes of TriBA network. Further, with the constraint of shortest-path routing, the output port of the next hop can be easily reasoning. Route look-ahead can be applied to shorten the router pipeline stages. Such a scheme significantly cuts down packet latency and boosts throughput.
2 TriBA NoC

In a TriBA NoC, the number of nodes must satisfy $3^L$. A 3-node complete graph serves as the basic modules and $L$ is the level of expansion. Due to the triangular distribution characters of TriBA, we encode it with the three letter set $T = \{1, 2, 3\}$. As described in Fig. 1, 1, 2 and 3 represent the three positions on the left, right and top respectively. With the expansion of the network size, the number of bits increases. TriBA can be defined in terms of a graph as follows.

![Fig. 1. Coding system of TriBA topology.](image)

**Definition 2.1.** Given a $L$-level TriBA network with $3^L$ cores, denoted as a graph $TG^L = (V(TG^L), E(TG^L))$, consists of a set of nodes and a set of edges:

- $V(TG^L) = \{x_Lx_{L-1} \cdots x_2x_1|x_i \in T, 1 \leq i \leq L\}$
- $E(TG^L) = \{\bar{x}_{L-f+1}ab^{f-1} \leftrightarrow \bar{x}_{L-f+1}ba^{f-1}|a, b \in T, a < b, f \in N, f \leq L\}$

where $\bar{x}_{m-n}$ denotes $x_m x_{m-1} \cdots x_{n+1} x_n$, $x'_L x'_L \cdots x'_1 x'_1$ denotes binary encoding of nodes, $v_1^L$, $v_2^L$ and $v_3^L$ denotes the three vertexes of $TG^L$, respectively. This definition is recursive.

**Proof.** Here we are dealing with the definition of $E(TG^L)$.

Basic Step: we start with the statement $E(TG^1)$ and find that the graph $TG^1$ consists of there edges $1 \leftrightarrow 2$, $2 \leftrightarrow 3$ and $1 \leftrightarrow 3$, and thus we can deduce the following equation:

- $E(TG^1) = \{a \leftrightarrow b|a, b \in T, a < b\}$
  - $= \{\bar{x}_{1-2}ab^0 \leftrightarrow \bar{x}_{1-2}ba^0|a, b \in T, a < b, f \in N, f \leq L\}$
  - $= \{\bar{x}_{1-f+1}ab^{f-1} \leftrightarrow \bar{x}_{1-f+1}ba^{f-1}|a, b \in T, a < b, f \in N, f \leq L\}$

so $E(TG^1)$ is true.

Inductive Step: now we assume the truth of $E(TG^{l-1})$, that is, we assume that

- $E(TG^{l-1}) = \{\bar{x}_{L-1-f+1}ab^{f-1} \leftrightarrow \bar{x}_{L-1-f+1}ba^{f-1}|a, b \in T, a < b, f \in N, f \leq L - 1\}$

is a true statement. From this assumption we want to deduce the truth of $E(TG^L)$. Due to $TG^L$ can be constructed by three interconnected $TG^{L-1}$, $E(TG^L)$ consists of three $E(TG^{L-1})$ and the other three edges $E(plus)$. Using the induction hypothesis $E(TG^{L-1})$, we find that
E(plus) = \{12^{L-1} \leftrightarrow 21^{L-1}, 23^{L-1} \leftrightarrow 32^{L-1}, 13^{L-1} \leftrightarrow 31^{L-1}\}
= \{ab^{L-1} \leftrightarrow ba^{L-1} | a, b \in T, a < b\}
= \{x_{L-f+1}ab^{f-1} \leftrightarrow x_{L-f+1}ba^{f-1} | a, b \in T, f \leq L\}

3E(TG^{L-1}) = \{x_{L}x_{L-1-f+1}ab^{f-1} \leftrightarrow x_{L}x_{L-1-f+1}ba^{f-1} | x, y, z \in T, a < b, f \leq N, f \leq L\}
= \{x_{L-f+1}ab^{f-1} \leftrightarrow x_{L-f+1}ba^{f-1} | a, b \in T, f \in N, f \leq L\}

E(TG^{L}) = 3E(TG^{L-1}) \cup E(plus)
= \{x_{L-f+1}ab^{f-1} \leftrightarrow x_{L-f+1}ba^{f-1} | a, b \in T, a < b, f \in N, f \leq L\}

Consequently, we now know from the principle of mathematical induction that \(E(TG^{L})\) is true.

**Definition 2.2.** In a TriBA NoC, each node has a constant-degree of 3 except the vertex node. As described in Fig. 2, we define the ports in the top, left and right directions as port3, port1 and port2, respectively. The local port is defined as port0. Therefore, when a fit travels from the internal node to nodes A, B and C, the fit will proceed over port 1, port2 and port3, respectively.

![Fig. 2. The definition of router ports.](image)

### 3 Data flow model
In Fig. 3(a), given any source node \(s\) and any target node \(t\), how do we find the shortest path between these two nodes? We aim to address this problem by encapsulating the complex in a simple function: that of finding out the smallest subgraph which includes these two nodes. This allows us to eliminate complex solving process that span the whole network and now solve this problem within the smallest subgraph. Here, we first describe the definition of the two terms ‘Minimum Routing Graph’ (MRG\(^L\)) and ‘Largest SubGraph of MRG’ (LSG\(^I\)).

**Definition 3.1.** MRG\(^L\) represents the smallest subgraph containing the source node and target node, where \(L\) denotes the size of subgraph.

**Definition 3.2.** LSG\(^I\) represents the three largest subgraphs of MRG, Where \(i\) denotes the relative position (left, right and top) of subgraphs in MRG, \(L\) denotes the size of subgraphs.
Theorem 3.3. Assume any source node \( s = a_L a_{L-1} \cdots a_1 \) and any target node \( t = b_L b_{L-1} \cdots b_1 \) in \( L \)-level TriBA network. If the maximal matching of the two ID sequences from left to right ends with \( a_i \) (or \( b_i \)), then the size of MRG of the two nodes is \((i - 1)\) level, and the size of LSG is \((i - 2)\) level.

Example: To illustrate the MRG, we take a 5-level TriBA network for example. Assume two node’s ID sequences, \( s = 32113 \) and \( t = 32213 \). The maximal matching sequence of 32113 and 32213 is 2 at \( a_4 \), then the size of MRG of the two nodes is 3 level, and the size of LSG is 2 level.

According to the direction of data flow, we introduce a Data Flow Model (FM) for classify message traffic into six flow models. As illustrated in Fig. 3(b), assume the source node in LSG\( _1 \) and the target node in LSG\( _2 \), the data flow from LSG\( _1 \) to LSG\( _2 \) is defined as \( FM_0 (FM_{000}) \). And so on, \( FM_1 \)–\( FM_5 \) (\( FM_{000} \)–\( FM_{101} \)) represents the other five data flow models. Given any source node \( s = s_L' s_{L-1}' \cdots s_1' s_0' \) and any target node \( t = t_L' t_{L-1}' \cdots t_1' t_0' \) in TriBA network \( (x_L' x_{L-1}' \cdots x_1' x_0' \) \) is a binary representation of \( x_L \cdots x_1 \), e.g., \( 010101 \) is a binary representation of 111.). The data flow model is labeled as \( FM_{ijk} \), where \( i, j, k \in \{0, 1\} \). Based on Theorem 3.3, the size of LSG can be calculated as \( l \). According to the value of \( s_L' s_i' \) and \( t_L' t_i' \), we can deduce the flow model, as described in the truth Table I.

| \( s_i' \) \( s_j' \) | 01 | 01 | 11 | 10 | 10 | 11 |
|-----------------|----|----|----|----|----|----|
| \( t_i' t_j' \) | 10 | 11 | 10 | 01 | 11 | 01 |
| \( FM_{ijk} \)  | 000| 010| 100| 001| 101| 011|

Based on the above truth Table I, we can conclude the data flow model formula as follows:

\[
FM_{ijk} = \begin{cases} 
  i = s_i' \cdot t_i' \cdot (s_j' \oplus t_j') \\
  j = s_j' \cdot t_j' \cdot (s_i' \oplus t_i') \\
  k = s_i' \cdot t_i' 
\end{cases}
\]
4 The shortest distance from internal nodes to vertices

Before introducing the look-ahead routing, we first discuss how to compute the shortest distance from any internal node to the vertices. We use \( d(x, y) \) to represent the distance from \( x \) to \( y \). As shown in Fig. 4(a), in \( MRG^l \), investigating the relative position between an internal node \( s = x_1 \cdots x_l \) and a vertex \( v^i_l \), there are three cases as follows:

- \( x_l = i \). If \( s \) is located in \( LSG^l_i \), \( s \) and \( v^i_l \) are in the same region. Hence, \( d(x_1 \cdots x_l, v^i_l) = d(x_{l-1} \cdots x_1, v'^{i-1}_l) \)
- \( x_l \neq i \). If \( s \) is not in \( LSG^l_i \), \( s \) and \( v^i_l \) are in different regions. Hence, \( d(x_1 \cdots x_l, v^i_l) = 2^{l-1} + d(x_{l-1} \cdots x_1, v'^{i-1}_l) \)
- \( d(e, v^i_l) = 0. e \) denotes an empty string.

Based on the above analysis, we can derive the recursive formula as follows:

\[
d(x_1 \cdots x_l, v^i_l) = \begin{cases} 
  d(x_{l-1} \cdots x_1, v'^{i-1}_l), & x_l = i \\
  2^{l-1} + d(x_{l-1} \cdots x_1, v'^{i-1}_l), & x_l \neq i 
\end{cases}
\]  

(2)

Assume \( y_k = compare(x_k' x_k, \ell) \), \( 1 \leq \ell \leq l \). When the two values are not equal, the function \( compare() \) returns value 1. Otherwise, the return value is 0. Then, formula 2 can be simplified as follows:

\[
d(x_1 \cdots x_l, v^i_l) = \sum_{k=1}^{l} y_k \times 2^{k-1}
\]  

(3)

Further, we investigate the relationship between the value of \( y_k \) and \( x_k' x_k \), the details are shown in the truth Table II. Through analysis, for \( i = 01 \), we can derive that \( y_k = x_k'' \). And so on, for \( i = 10 \), \( y_k = x_k' \). For \( i = 11 \), \( y_k = x_k'' \oplus x_k' \).

| \( i \) | 01 | 10 | 11 |
|---|---|---|---|
| \( x_k'' x_k' \) | 01 | 10 | 11 |
| \( y_k \) | 0  | 1  | 1  |

Table II. Truth table for computing \( y_k \)

Put the value of \( y_k \) into formula 3, we can get the binary formula 4 as follows:
5 Look-ahead routing in $FM_0$

As shown in Fig. 4(b), in $FM_0$, assume the source node $S = 01s_{l-1}s_{l-1} \cdots s_1 s_1$ in $LSG_1^{l-1}$ and the target node $T = 10t_{l-1}t_{l-1} \cdots t_1 t_1$ in $LSG_2^{l-1}$, the shortest path between these two nodes is one of the two alternatives: PathA ($S \rightarrow A \rightarrow A' \rightarrow T$) and PathB ($S \rightarrow B \rightarrow B' \rightarrow T$). Based on formula 4, we can adopt sectional calculation method to compute them respectively.

$$d(\text{PathA}) = d(S, A) + d(A, A') + d(A', T)$$
$$= d(s_{l-1}^{s_1} s_{l-1}^{s_1} \cdots s_1^{s_1}, v_1^{v_2}) + 1 + d(t_{l-1}^{t_1} t_{l-1}^{t_1} \cdots t_1^{t_1}, v_1^{v_1})$$
$$= s_{l-1}^{s_1} s_{l-1}^{s_1} \cdots s_1^{s_1} + 1 + t_{l-1}^{t_1} t_{l-1}^{t_1} \cdots t_1^{t_1}$$

$$d(\text{PathB}) = d(S, B) + d(B, B') + d(B', T)$$
$$= d(s_{l-1}^{s_1} s_{l-1}^{s_1} \cdots s_1^{s_1}, v_3^{v_3}) + 1 + d(t_{l-1}^{t_1} t_{l-1}^{t_1} \cdots t_1^{t_1}, v_3^{v_1})$$
$$= s_{l-1}^{s_1} s_{l-1}^{s_1} \cdots s_1^{s_1} + 1 + t_{l-1}^{t_1} t_{l-1}^{t_1} \cdots t_1^{t_1}$$

To compare the length of the two paths, we assume $d_{\text{compare}} = d(\text{PathA}) - d(\text{PathB})$. When $d_{\text{compare}} < 0$, select PathA. Otherwise, select PathB. From the above formulas, we can see that the shortest path can be obtained by calculating the binary encoding of source node and target node. Such binary computing will improve the efficiency of computation.

Further, we delve into the output port at each hop on the two paths. For PathB, a fit will proceed over the output port 3 at each hop on the segment ($S \rightarrow B$), and proceed over the output port 2 at each hop on the segment ($E \rightarrow F$). Accordingly, for PathA, a fit will proceed over the output port 2 at each hop on the segment ($S \rightarrow A$). From this rule, we can deduce the following theorem:

**Theorem 5.1.** $\forall v_i^i \in \{v_1^1, v_2^1, v_3^1\}$, when a flit travels from any node $s$ to a vertex node $v_i^i$ with the constraint of shortest-path routing in $TG^l$, the flit will proceed over the output port $i$ at each hop on the path ($s \rightarrow v_i^i$).

**Proof.** Basic step: for $TG^1$, there are three nodes described as 1, 2, and 3, respectively. We take the case between node 1 and node 3 as an example. When a flit travels from node 1 to node 3, the flit will proceed over the output port 3 at node 1 for the next hop. Similarly, other cases can also be proved. Therefore, for $TG^1$, the theorem is proved to be true.

Inductive step: now we assume the truth in $TG^{l-1}$, that is, we assume that when a flit travels from any node $s$ to the vertex node $v_i^{l-1}$, the flit will proceed over the output port $i$ at each hop on the path ($s \rightarrow v_i^{l-1}$). Due to the TriBA network is a recursive network, $TG^2$ can be constructed by three $TG^{l-1}$. Any node in $TG^l$ must belong to one of the three $TG^{l-1}$. Using the induction hypothesis in $TG^{l-1}$, we find that when a flit travels from any node $s$ to the vertex node $v_i^i$, the flit will proceed over the output port $i$ at each hop on the path ($s \rightarrow v_i^i$) in two $TG^{l-1}$. Therefore, the theorem is proved to be true.
In the same way, we can deduce another two theorems as follows.

**Theorem 5.2.** In FM₀, when a flit arrives at the vertex node v_L¹⁻¹ in LSG¹⁻¹, the flit will proceed over the output port 2 at the next hop v₁⁻¹ in LSG₂⁻¹ with the constraint of shortest-path routing.

**Theorem 5.3.** In FM₀, when a flit arrives at the vertex node v₂⁻¹ in LSG¹⁻¹, we can find that v₁⁻¹ in LSG₂⁻¹ is the next hop with the constraint of shortest-path routing. If the traffic model between v₁⁻¹ and the target node is FM₀, the flit will still proceed over the output port 2 at v₁⁻¹ for the next hop. If the traffic model between v₁⁻¹ and the target node is FM₂, the flit will proceed over the output port 3 at v₁⁻¹ for the next hop.

Based on the above theorems, the output port of the next hop in FM₀ can be easily reasoning. For example, as described in Fig. 4(b), for PathA, a flit will proceed over the output port 2 at each hop in FM₀ based on Theorem 5.1. When the flit arrives at the node A, the output port of the next hop can be reasoning based on Theorem 5.3. Similarly, For PathB, a flit will proceed over the output port 3 at each hop in FM₀. When the flit arrives at the node B, the output port of the next hop can be reasoning as output port 2 based on Theorem 5.2. Thus, the look-ahead routing algorithm in FM₀ can be described as follows:

**Algorithm 1 LA4T in FM₀**

| Input: s = sₗˡₛₖₗ · · · sₜᵢₛₜᵢ, start node; |
| t = tᵢₗᵗₛₖₕ, target node; |
| l, size of MRG; |
| Output: cur-outport, the output port at sᵢₗₛᵢₗ · · · sᵢₜᵢₛᵢₜᵢ; |
| next-outport, the output port at the next hop; |
| 1: d(PathA) = sₗ₋₁ · · · s₁ + t₋₁ · · · tᵢ₁ + 1; |
| 2: d(PathB) = sₗ₋₁ · · · sᵢ₁ + t₋₁ · · · tᵢᵗₛₜᵢ + 2⁻¹ + 1; |
| 3: if d(PathA) ≤ d(PathB) then |
| 4: cur-outport=2; next-outport=reasoning(s,v₁); |
| 5: else |
| 6: cur-outport=3; next-outport=reasoning(s,v₁); |
| 7: end if |
| 8: return cur-outport;next-outport; |

**5.1 Flow model transformation using S₃ group**

In order to simplify the calculation, we find that the calculations in (FM₁–FM₅) can be transformed into FM₀ due to the rotational symmetric of TriBA topology. Therefore, the calculations can be conducted in the same model. Fig. 5 illustrates the equivalent transformation from FM₃ to FM₀. Fig. 5(a) can be transformed to Fig. 5(b) after 120 degrees anti-clockwise rotation. Specifically, the routing calculation from s = 323 to t = 132 in FM₃ can be converted to the routing calculation from s' = 131 to t' = 213 in FM₀.
Definition 5.4. The permutation group $S_3$ has six elements: $\tau_0 = (1)$, $\tau_1 = (12)$, $\tau_2 = (23)$, $\tau_3 = (123)$, $\tau_4 = (13)$, $\tau_5 = (132)$. For any node described as $x_Lx_{L-1}\cdots x_2x_1$ in the TriBA graph, we define that $\tau_i(x_Lx_{L-1}\cdots x_2x_1) = \tau_i(x_L)\tau_i(x_{L-1})\cdots \tau_i(x_2)\tau_i(x_1)$, $0 \leq i \leq 5$.

Theorem 5.5. For a TriBA graph $TG^L$, the automorphism group of $TG^L$ is the permutation group $S_3$.

Proof. The three letter set $T = \{1, 2, 3\}$ used in our coding system has the same element as the $S_3$ group. Therefore, $S_3$ is a automorphism group of $T$. For a element $\tau \in S_3$, $\forall x_i \in T$, $\exists y_i \in T$ such that $\tau : x_i \mapsto y_i$.

Based on Definition 5.4, we can deduce that $\tau_i(x_Lx_1) = \tau_i(x_L)\cdots \tau_i(x_1) = y_L\cdots y_1$. Thus, $\forall x_{L-1}\cdots x_1 \in T$, $\exists y_L\cdots y_1 \in T$ such that $\tau : x_L\cdots x_1 \mapsto y_L\cdots y_1$. It means that there is a bijection relationship between the nodes in $TG^L$.

For an edge described as $x_L-f+1ab/-1$ in the TriBA graph, since it has a bijection relationship between the nodes $\tau(x_L-f+1) = \tilde{y}_{L-f+1}$, $\tau(a) = c$ and $\tau(b) = d$, the two endpoints of the edge can be mapped to image points $\tilde{y}_{L-f+1}cd/-1$ and $\tilde{y}_{L-f+1}de/-1$. Based on Definition 1, the two image points are adjacent to each other. Similarly, for the edge $\tilde{y}_{L-f+1}cd/-1$ to $\tilde{y}_{L-f+1}de/-1$, there exists $\tau^{-1}(\tilde{y}_{L-f+1}cd/-1) = \tilde{x}_{L-f+1}ab/-1$ and $\tau^{-1}(\tilde{y}_{L-f+1}de/-1) = \tilde{x}_{L-f+1}de/-1$. The two image points are also adjacent to each other. Such that, there is a bijection relationship between the edges in $TG^L$. So for a TriBA graph $TG^L$, the automorphism group of $TG^L$ is the permutation group $S_3$. 

Based on Theorem 5.5, we can leverage the cyclic permutation of group $S_3$ to transform different flow models ($FM_1$–$FM_5$) into $FM_0$, e.g., assume the source node is 323 and the target node is 132. Based on the data flow model, we can deduce that the current flow model is $FM_3$. Then we use the rule of $\tau_3 = (123)$ to perform cyclic permutation (1 $\rightarrow$ 2, 2 $\rightarrow$ 3, 3 $\rightarrow$ 1) on the coding of the source node and the target node, and thus 323 can be transformed into 131, and 132 can be transformed into 213. Therefore, the flow model $FM_3$ is transformed to $FM_0$. Table III plots the equivalent transformation from $FM_1$–$FM_5$ to $FM_0$ using different elements of $S_3$ group.

| Current mode | $FM_1$ | $FM_2$ | $FM_3$ | $FM_4$ | $FM_5$ |
|--------------|--------|--------|--------|--------|--------|
| Cyclic permutation | (12)  | (23)  | (123) | (13)  | (132) |
| Transformed mode | $FM_0$ |
ent elements of $S_3$ group. The look-ahead routing algorithm in $FM_1–FM_5$ can be described as Algorithm 2.

Algorithm 2 $LA4T$ in $FM_1–FM_5$

Input: $s = s'_0 s'_1 \cdots s'_l s'_1$, start node;
      $t = t'_0 t'_1 \cdots t'_l t'_1$, target node;
      $l$, size of MRG;

Output: cur-outport, the output port at $s'_0 s'_1 \cdots s'_l s'_1$;
         next-outport, the output port at the next hop;

1: $i \leftarrow FM(s'_0 s'_1 \cdots s'_l s'_1, t'_0 t'_1 \cdots t'_l t'_1);$  
2: $\tilde{u}_{EQ} \leftarrow \text{Convert}(s'_0 s'_1 \cdots s'_l s'_1);$  
3: $\tilde{v}_{EQ} \leftarrow \text{Convert}(t'_0 t'_1 \cdots t'_l t'_1);$  
4: return $S_3^{-1}(LA4TinFM_0(\tilde{u}_{EQ}, \tilde{v}_{EQ}, l));$

6 Performance evaluation

6.1 Experimental methodology

We compare the network latency and network throughput of our proposal LA4T routing algorithm to the previous routing algorithms (SPORT and DDRA). We perform our evaluations using a modified version of Noxim [11], a cycle-accurate network-on-chip simulator. We assume 45 nm technology with an on-chip voltage of 0.8 V and a frequency of 3 GHZ. We simulate a 27-core TriBA NoC. The performance evaluation is conducted under four traffic patterns (uniform, bitreversal, transpose, and hotspot) with different injection rate. The simulator was run for 10,000 cycles to warm up and the average performance was measured over another 100,000 cycles. Table IV lists the important simulator parameters used in the experiments.

| Parameter | Technology/Vdd | Frequency | Network size | Buffer size | Flit size |
|-----------|----------------|-----------|--------------|-------------|-----------|
| 45 nm/0.8 V | 3 GHZ | 27-core | 4 flits | 32 bits |

6.2 Results

Fig. 6 describes the average network latency variation under different traffic patterns. Across all four traffic patterns, our proposed LA4T algorithm can provide 15.6%–38.3% improvement in average network latency compared to DDRA and SPORT. The reasons for the lower latency of LA4T are as follows: (1) LA4T can provide a minimal path for routing packets compared with DDRA. (2) Since LA4T adopts look-ahead strategy, it reduces the router pipeline stages and thus cuts down the packet latency. Overall, these algorithms reach a saturation point earlier in uniform traffic than other traffic patterns. The reason is that the uniform traffic pattern leads to message traffic spread evenly across the network from global and long-term perspective. However, all these algorithms select channels based on local, short-term information and thus they can not maintain the evenness of
uniform traffic. The highest latency improvement is seen in bitreversal traffic. This is mainly because bitreversal traffic represents global communication in which the average path length requires more hops. The smallest improvement can be expected seen in hotspot traffic due to this traffic pattern generates some significant hotspots which dominate the network latency.

Fig. 7 plots the throughput variation under different traffic patterns. LA4T routing algorithm can achieve 15.2%–21.8% network throughput improvement compared to DDRA and SPORT. This is principally because LA4T reduces the average hop counts of packets compared to DDRA. Flits travel to the destination across the shorter path, which eventually decreases the congestion of the network. In contrast to SPORT, LA4T adopts distributed routing and thus cuts down the routing information of the header of a packet, resulting in improving the network payload.
7 Conclusion

Routing algorithm plays a vital role in ensuring the performance of NoC. In this paper, we propose a express look-ahead routing algorithm for TriBA NoC, which not only provides the shortest-path for routing but also reasonings the output port of the next hop at each hop. Using an extend cycle-accurate simulator Noxim, we compare the performance of our proposed LA4T algorithm against the previous algorithms. The results show that LA4T algorithm can achieve better network performance. In the future work, we will leverage the look-ahead routing strategy to optimize the router microarchitecture, especially virtual channel which is one of the most important components of the area penalty.

Acknowledgments

The authors would like to thank Dr. Yan Su for providing the Noxim code of the method. The authors would also like to thank the anonymous referees for their valuable comments and helpful suggestions. This work is supported by National Natural Science Foundation of China (NO. 6300011, 6100010).