# The Odd-Even Turn Model for Adaptive Routing

Ge-Ming Chiu, *Member*, *IEEE Computer Society*

**Abstract**—This paper presents a model for designing adaptive wormhole routing algorithms for meshes without virtual channels. The model restricts the locations where some turns can be taken so that deadlock is avoided. In comparison with previous methods, the degree of routing adaptiveness provided by the model is more even for different source-destination pairs. The mesh network may benefit from this feature in terms of communication efficiency. Simulation results show that the even adaptiveness provided by the odd-even turn model makes message routing less vulnerable to nonuniform factors such as hot spot traffic. In addition, this property results in a smaller fluctuation of the network performance with respect to different traffic patterns.

**Index Terms**—Adaptive routing, deadlock, mesh, turn model, wormhole routing.

## 1 Introduction

The mesh topology has become a popular interconnection architecture for constructing massively parallel multiprocessors. In particular, low-dimensional meshes have been used in several academic and commercial machines, such as the DASH multiprocessor [17], Intel TFLOPS supercomputer [19], and Intel Paragon [6].

Processors (or nodes) of a mesh communicate with each other by sending messages through the underlying network. Hence, efficient communication is critical to the performance of a mesh. Recently, the most popular technique for switching packets is *wormhole routing* [20]. With wormhole routing, a packet is divided into *flow control digits* (or *flits*). The flits are routed through the network one after another in a pipeline fashion. The first flit of a packet is designated as the *header flit*, which contains routing information and leads the packet through the network. When the header flit is blocked from advancing due to lack of output channels, all of the flits wait at their current nodes for available channels. Each router only requires small buffer space to store the flits and communication latencies are low with wormhole routing.

Routing algorithms are used to determine the sequence of channels a message packet traverses from the source to the destination. A desirable property of a routing algorithm is freedom from deadlock and livelock. *Livelock* occurs when a message proceeds through the network indefinitely, never arriving at its destination. Livelock is possible only if message routing is adaptive and is nonminimal. *Deadlock* is caused by packets waiting for each other in a cycle. In wormhole routing, the order in which channels are used for a packet must meet certain criteria so that deadlock is prevented. *Adaptiveness* is also an important factor for message routing [16]. Adaptiveness increases the chances that packets may avoid hot spots or faulty components and reduces the chances that packets are continuously blocked.

• *The author is with the Department of Electrical Engineering, National Taiwan University of Science and Technology, Taipei, Taiwan, ROC. E-mail: chiu@optimal.ee.ntust.edu.tw.*

*Manuscript received 10 Feb. 1998; accepted 19 Feb. 2000.*
*For information on obtaining reprints of this article, please send e-mail to: tpds@computer.org, and reference IEEECS Log Number 106313.*

There have been many routing algorithms proposed for meshes that are based on wormhole routing in the literature [2], [3], [5], [13], [15], [18], [22], [23]. Routing algorithms can be generally classified into three categories, depending on the degree of adaptiveness provided by the algorithms. A *nonadaptive* routing algorithm is *deterministic* and routes a packet from the source to the destination along a unique, predetermined path. A *minimal fully adaptive* routing algorithm routes *all* packets through any shortest paths to the destinations. A *partially adaptive* routing algorithm allows multiple choices for routing packets via shortest paths, but it does not allow *all* packets to use any shortest paths. In [8], [9], virtual channels were introduced to assist the design of nonadaptive routing algorithms so that deadlock is avoided. Virtual channels are abstractions that share the same physical channel. Later, several researchers [3], [5], [10], [14], [18], [21], [22], [23] used virtual channels to design partially adaptive and fully adaptive routing algorithms for a variety of network architectures, including meshes. Adding virtual channels allows the design of highly adaptive routing algorithms. In fact, it is impossible to produce a deadlock-free fully adaptive routing algorithm for a mesh without addition of virtual channels [20].

However, adding virtual channels to meshes is not free. It involves adding buffer space and complex control logic to routers, thus communication performance of the network and reliability of the routers may be affected [4], [16]. Furthermore, deadlock-free routing schemes that are not based on adding virtual channels may be used as the basic mechanisms for implementing adaptive routing algorithms that use virtual channels [10]. Hence, the development of deadlock-free routing algorithms that do not use virtual channels is important.

Several routing algorithms that require no virtual channels have been proposed for mesh networks. The *xy routing algorithm* [7] for two-dimensional meshes routes a packet first along $x$ dimension (dimension 0) and then along $y$ dimension (dimension 1). The *xy* algorithm ensures deadlock freedom, but it provides no adaptiveness. Glass and Ni [15], [16] presented an elegant technique, called *turn model*, for designing partially adaptive wormhole routing algorithms that require no virtual channels. The basic idea

of the model is to prohibit the minimum number of turns that break all of the cycles so that deadlock can be avoided. Based on the turn model, three partially adaptive routing algorithms, namely *west-first*, *north-last*, and *negative-first*, were presented for two-dimensional meshes. Unfortunately, the degree of adaptiveness provided by the turn model is highly uneven. Using any of the above-mentioned routing algorithms, at least half of the source-destination pairs are restricted to having only one minimal path, while full adaptiveness is provided for the rest of the pairs. Such uneven adaptiveness not only causes *unfairness* but also curtails the ability of the model in alleviating traffic congestion problem. Performance of the network communication may be affected as a result. The *xy* algorithm can also be considered as achieving deadlock freedom by prohibiting certain turns. In [2], a model called *direction restriction model* was proposed to facilitate the design of partially adaptive routing algorithms. This model is based on dividing a system into two unidirectional networks. Message routing is done in two phases. In the first phase, a packet is routed adaptively to an intermediate node using one unidirectional network. It is then routed adaptively to the destination using the other network in the second phase. The degree of adaptiveness provided by the model is also highly uneven.

In this paper, we propose a novel model for designing partially adaptive, deadlock-free routing algorithms for meshes. The model is not based on adding virtual channels to network topologies. Unlike the previous methods, which rely on prohibiting certain turns in order to achieve deadlock freedom, our model restricts the locations where some types of turns can be taken. The degree of routing adaptiveness provided by the proposed model is more even for different source-destination pairs. The mesh network may benefit from this feature in terms of communication efficiency. Extensive simulations have been conducted to evaluate the performance of the model. The simulation results show that the even adaptiveness provided by our odd-even turn model is important under nonuniform traffic. It makes message routing less vulnerable to nonuniform factors, such as hot spot traffic. Furthermore, this property allows the performance of the network to have smaller fluctuation with respect to different types of traffic.

In the next section, the odd-even turn model for two-dimensional meshes is presented. Simulation results for two-dimensional meshes are shown in Section 3. Section 4 addresses an extension of the model to higher-dimensional meshes. Section 5 draws the conclusions.

## 2 Routing in Two-Dimensional (2D) Meshes

In this section, we present a model, called the *odd-even turn* model, to facilitate deadlock-free routing in two-dimensional (2D) meshes with no virtual channels. In a $K_0 \times K_1$ 2D mesh, a node $X$ is identified by a two-element vector $(x_0, x_1)$, $0 \leq x_0 \leq K_0 - 1$ and $0 \leq x_1 \leq K_1 - 1$, where $x_0$ and $x_1$ are called the coordinates of dimension 0 and dimension 1 of $X$, respectively. To facilitate the presentation, we label the four sides of a 2D mesh as East, West, South, and North. All of the nodes that have the same coordinates of dimension 0 constitute a *column*, and all of the nodes that have the same coordinates of dimension 1 constitute a *row*. *Row channels* refer to channels along dimension 0; that is, a row channel connects two neighboring nodes on the same row. Similarly, *column channels* refer to channels along dimension 1. Further, a column channel is called an SN (respectively, NS) channel if its direction is from South to North (respectively, North to South). Suppose that a channel links node $A$ to node $B$. Nodes $A$ and $B$ are called the *tail* node and the *head* node of the channel, respectively.

**Definition 1.** *A turn consists of a row channel and a column channel such that the tail node of one of the channels is the head node of the other; the common node of the two channels is called the turning node of the turn.*

Essentially, a turn involves a 90-degree change of traveling direction. Unless otherwise specified, a turn is a 90-degree turn in the following description. There are eight types of turns, according to the traveling directions of the associated channels. A turn is called an ES turn if it involves a change of direction from East to South. Similarly, we can define the other seven types of turns, namely EN, WS, WN, SE, SW, NE, and NW turns, where E, W, S, and N indicate East, West, South, and North, respectively. A packet is said to *take a turn* at node $X$ if it traverses the turn and the turning node of the turn is $X$.

### 2.1 The Odd-Even Turn Model

Deadlock in wormhole routing is caused by packets waiting on each other in a cycle. Previous methods, such as the turn model [15] and the $xy$ algorithm, avoid deadlock by prohibiting certain turns. Instead, the odd-even turn model is based on restricting the locations at which certain turns can be taken so that a circular wait can never occur. The proposed model does not eliminate any types of turns for message routing.

**Definition 2.** *In a 2D mesh, a column is called an even (respectively, odd) column if the dimension-0 coordinate of the column is an even (respectively, odd) number.*

For example, in a $K_0 \times K_1$ mesh, the column which consists of all the nodes with addresses $(2, j)$, for $0 \leq j \leq K_1 - 1$, is an even column.

The basic idea of the odd-even turn model is to restrict the locations where some of the turns can occur so that an EN turn and an NW turn are not taken at nodes in the same column, and neither are an ES turn and an SW turn. As shown later, this may avoid deadlock. More precisely, the odd-even turn model is governed by the following two rules:

**Rule 1.** Any packet is *not* allowed to take an EN turn at any nodes located in an even column, and it is *not* allowed to take an NW turn at any nodes located in an odd column.

**Rule 2.** Any packet is *not* allowed to take an ES turn at any nodes located in an even column, and it is *not* allowed to take an SW turn at any nodes located in an odd column.

No other restrictions need to be applied. Note that the even and odd columns in either of the above rules may be interchanged. According to Rule 1, the column channel of

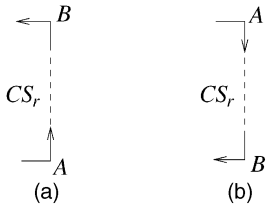


Fig. 1. The rightmost column on the waiting path. (a) SN channels. (b) NS channels.

an EN turn cannot be located in the same column as that of an NW turn. Similarly, according to Rule 2, the column channel of an ES turn cannot be located in the same column as that of an SW turn.

In the following, we show that any routing algorithm, whether minimal or nonminimal, that is based on the odd-even turn model is deadlock free as long as 180-degree turns are prohibited. A routing algorithm is *minimal* if it routes a packet via a shortest path between the source and the destination. The key to deadlock freedom is that the rightmost column segment of a circular waiting path, which is essential for a deadlock state, can never be formed. Consider a set of packets $p_1, p_2, \ldots, p_l$ where $p_i$ waits for $p_{i+1}$ for all $1 \leq i \leq l-1$. The sequence of channels which include the channels traversed by $p_1$, followed by the channels traversed by $p_2$ from the node at which $p_1$ is blocked, then followed by those traversed by $p_3$ from the node at which $p_2$ is blocked, and so on until the current node of $p_l$, is called the *waiting path* of the set of packets. Deadlock is formed if $p_l$ waits for $p_1$, as a circular wait is generated by the packets. The associated waiting path for this deadlocked situation is a circular path; here the waiting path includes the channels that are traversed by $p_1$ from the node at which $p_l$ is blocked, rather than from the source of $p_1$.

**Theorem 1.** *Any routing algorithms that follow the rules of the odd-even turn model are deadlock free as long as 180-degree turns are prohibited.*

**Proof.** We prove the theorem by contradiction. Assume that there exists a set of packets $p_1, p_2, \ldots, p_l,$ that are deadlocked. Thus, the associated waiting path is a circular path. Since 180-degree turns are prohibited, the waiting path must include both row and column channels. Consider any column line segment, $CS_r$, on the waiting path that is located in the rightmost column, i.e., no other column channels on the waiting path are more eastward than $CS_r$. Here a column line segment consists of a sequence of column channels of the same direction in the same column. Let nodes $A$ and $B$ denote the beginning and the end nodes of $CS_r$, respectively. Suppose that the channels of $CS_r$ are SN channels as illustrated in Fig. 1a. In this case, there must be some $p_i$, $1 \leq i \leq l$, that either has already taken or would take an EN turn at node $A$. There must also be some $p_j$, $1 \leq j \leq l$, that either has already taken or would take an NW turn at node $B$. Moreover, both turns are on the waiting path. This can be easily argued by using the fact that $CS_r$ is a rightmost column line segment on the waiting path and 180-degree turns are prohibited. However, according to Rule 1 of the odd-even turn model, an EN turn and an NW turn cannot possibly be taken at nodes in the same column, column of $CS_r$ in this case, and thus contradiction arises. By the same token, we can show that, according to Rule 2 of the odd-even turn model, similar contradiction exists for the case in which the channels of $CS_r$are NS channels, as illustrated in Fig. 1b. Hence, we prove the theorem. □

## 2.2 Implementation of Routing Algorithms

Based on the odd-even turn model, we can design various partially adaptive routing algorithms that are free from deadlocks. In this section, we mainly address the issues involved in the implementation of minimal routing algorithms. Basically, at any intermediate node an odd-even-turn-based routing algorithm must first determine the set of directions toward which a packet may be forwarded for the next hop. Appropriate turn criteria specified by either Rule 1 or Rule 2 must be applied. Consider the case where the destination of a packet is to the west of its source. The packet is prohibited from moving north or south at an intermediate node that resides in an odd column, unless the destination is located in the same column. This is because the packet must otherwise take an NW (respectively, SW) turn at some node in the same column later in order to reach the destination, which is an action prohibited by Rule 1 (respectively, Rule 2). Now, consider the case where the destination of a packet is to the east of its source. Care must be taken when the destination node is located in an even column. The packet must finish routing in dimension 1 before it reaches the column in which the destination is located. This is because an EN or ES turn is not allowed in an even column, according to Rule 1 and Rule 2. Therefore, when the current node is located one column to the west of the destination column, the packet cannot move east, unless it is now in the same row as the destination. The packet can always move north or south, if needed, as long as the current node is in an odd column, as the rules cannot be violated. In addition, if the source node of the packet is in an even column, it is allowed to continuously move north or south, if needed, in the column where the source resides. We show a minimal routing algorithm ROUTE in Fig. 2. In the algorithm, the set $Avail\_Dimension\_Set$ contains dimensions that are available for forwarding the packet.

Minimal routing can always be achieved by algorithm ROUTE for any source-destination pair. In Fig. 3, we show some possible routing paths for four packets in a $9 \times 9$ mesh. $S_i$ and $D_i$ represent the source and the destination nodes of packet $p_i$, $1 \leq i \leq 4$. Packets $p_1$, $p_2$, and $p_3$ use the minimal routing algorithm ROUTE. At node $(2, 3)$, $p_1$ can only move east as an EN turn is not allowed at the column. Consider $p_2$, which is a westbound packet. It cannot turn north at node $(7, 1)$ or node $(5, 1)$ since it is prohibited from taking an NW turn, which is required later for it to reach the destination, in odd columns. Note that, in Fig. 3, the routing path for packet $p_4$ is illustrated to show that nonminimal paths are possible with the odd-even turn model. However, nonminimal paths are not guaranteed to exist.

```
Algorithm ROUTE
/* Source node: (s_0, s_1); destination node: (d_0, d_1); current node: (c_0, c_1). */
begin
  Avail_Dimension_Set ⟵ ∅;
  e_0 ← d_0 − c_0;
  e_1 ← d_1 − c_1;
  if (e_0 = 0 and e_1 = 0)
     Deliver the packet to the local node and exit;
  if (e_0 = 0)     /* currently in the same column as destination */
     if (e_1 > 0)
        Add North to Avail_Dimension_Set;
     else
        Add South to Avail_Dimension_Set;
  else
     if (e_0 > 0)     /* eastbound messages */
        if (e_1 = 0)
           Add East to Avail_Dimension_Set;
        else {
              if (c_0 is odd or c_0 = s_0)
                 if (e_1 > 0)
                    Add North to Avail_Dimension_Set;
                 else
                    Add South to Avail_Dimension_Set;
              if (d_0 is odd or e_0 ≠ 1)    /* odd destination column or ≥ 2 columns to destination */
                 Add East to Avail_Dimension_Set;
            }
     else {     /* westbound messages */
           Add West to Avail_Dimension_Set;
           if (c_0 is even)
              if (e_1 > 0)
                 Add North to Avail_Dimension_Set;
              else
                 Add South to Avail_Dimension_Set;
         }
  Select a dimension from Avail_Dimension_Set to forward the packet;
end
```

Fig. 2. A minimal routing algorithm ROUTE that is based on the odd-even turn model.

### 2.3 Degree of Adaptiveness

One metric for measuring the adaptiveness of a partially adaptive routing algorithm is the degree of adaptiveness [15], which is essentially the number of shortest paths the algorithm allows from the source to the destination. Let $(x_s, y_s)$ and $(x_d, y_d)$ denote the addresses of the source and the destination nodes of a packet, respectively. Also let $P_{algorithm}$ represent the number of shortest paths the *algorithm* allows from the source to the destination for the packet. In the following treatment, $\Delta x$ and $\Delta y$ are defined as $\Delta x = x_d - x_s$ and $\Delta y = y_d - y_s$. To facilitate the following description, we call the packet an NE (respectively, SE, NW, and SW) packet if $\Delta x > 0$ and $\Delta y \geq 0$ (respectively, $\Delta x > 0$ and $\Delta y < 0$, $\Delta x \leq 0$ and $\Delta y \geq 0$, and $\Delta x \leq 0$ and $\Delta y < 0$). A column is called an *allowable* column for an NE (respectively, SE, NW, and SW) packet if and only if an EN (respectively, ES, NW, and SW) turn can be taken in the column according to the rules of the odd-even turn model. In the following, let $d_x = |\Delta x|$ and $d_y = |\Delta y|$.

For a fully adaptive algorithm, we have:

$$P_{\text{fully adaptive algorithm}} = \frac{(d_x + d_y)!}{d_x! d_y!}$$

Now consider the odd-even turn model. Let $h$ and $h'$ be defined as $h = \lceil \frac{d_x}{2} \rceil$ and $h' = \lceil \frac{d_x - 1}{2} \rceil$. Suppose that the packet is an NE or SE packet, i.e., $\Delta x > 0$. We can readily obtain the degree of adaptiveness as follows.

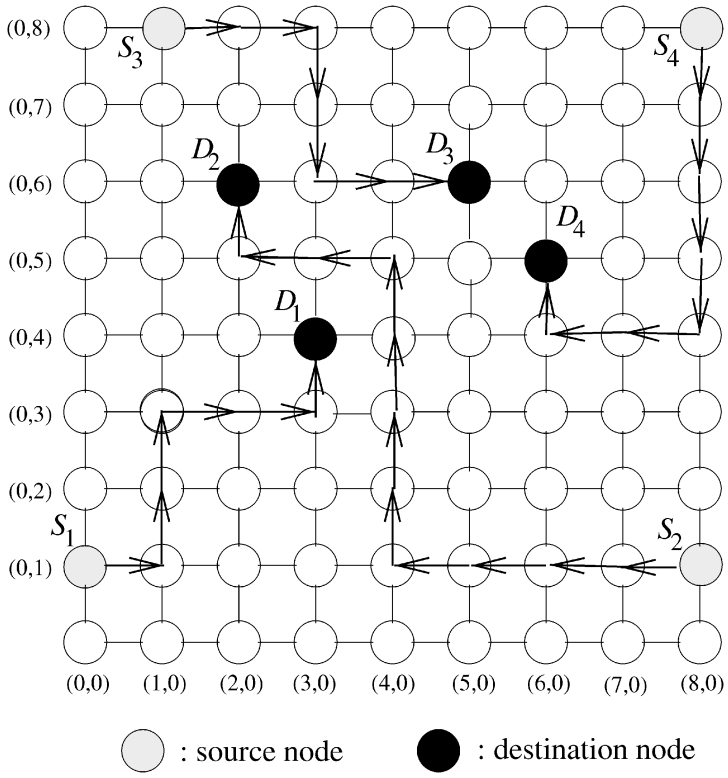


Fig. 3. Routing examples in a $9 \times 9$ 2D mesh.

$$P_{\text{odd-even turn model}} = \begin{cases} \frac{(d_y+h')!}{d_y!h'!} & \text{if column } x_s \text{ is an allowable column and } d_x \text{ is an odd number,} \\ \frac{(d_y+h)!}{d_y!h!} & \text{otherwise.} \end{cases}$$

In case the packet is an NW or SW packet, i.e., $\Delta x \leq 0$, the degree of adaptiveness for the odd-even turn model becomes

$$P_{\text{odd-even turn model}} = \begin{cases} \frac{(d_y+h)!}{d_y!h!} & \text{if column } x_s \text{ is an allowable column or } \Delta x = 0, \\ \frac{(d_y+h')!}{d_y!h'!} & \text{otherwise.} \end{cases}$$

In contrast, the degrees of adaptiveness for the *west-first* and the *negative-first* algorithms that are based on the turn model are given [15] as follows:

$$P_{west-first} = \begin{cases} \frac{(d_x+d_y)!}{d_x!d_y!} & \text{if } x_d \geq x_s \\ 1 & \text{otherwise.} \end{cases}$$

$$P_{negative-first} = \begin{cases} \frac{(d_x+d_y)!}{d_x!d_y!} & \text{if } (x_d \leq x_s \text{ and } y_d \leq y_s) \text{ or } (x_d \geq x_s \text{and } y_d \geq y_s) \\ 1 & \text{otherwise} \end{cases}.$$

It is apparent that the odd-even turn model provides more even adaptiveness than the turn model of [15].

## 3 Performance Evaluation

To evaluate the performance of the odd-even turn model, we have developed an event-driven simulator. The simulations were conducted on a $15 \times 15$ mesh under various traffic patterns. Two unidirectional channels exist between each pair of neighboring nodes. All of the channels have the same bandwidth of 20 flits/$\mu$sec. Each input channel has a buffer the size of a single flit. We compare our algorithm with the nonadaptive *xy* algorithm and the *west-first* and *negative-first* algorithms that are based on the turn model [15]. In the simulation, we consider minimal routing of messages. For the odd-even turn model, algorithm ROUTE is used. When multiple header flits wait for the same available output channel, the *local first-come-first-served* policy [16], which decides in favor of the header flits that arrive at the node first, is adopted. When a header flit has two output channels available for it, our algorithm uses the output channel along dimension 1.

Processors generate messages at time intervals chosen from a negative exponential distribution. Each message is assumed to be a 20-flit packet as commonly used in the literature. Three traffic patterns, namely *uniform*, *transpose*, and *hot spot* [12] are considered in the simulation. With the uniform pattern, a processor sends a message to any other node with equal probability. For the transpose traffic, we have simulated two types of patterns. With the first transpose traffic pattern, a node $(i, j)$ only sends messages to node $(14 - j, 14 - i)$. This traffic pattern is identical to the *matrix-transpose* used in [16]. In the second transpose traffic pattern, a node $(i, j)$ only sends messages to node $(j, i)$. These two transpose patterns correspond to reflections of the source about the lines $y = -x$ and $y = x$, respectively, given a coordinate system through the center of the network. In the hot spot traffic patterns simulated, one or more nodes are designated as the *hot spot nodes*, which receive hot spot traffic in addition to the regular uniform

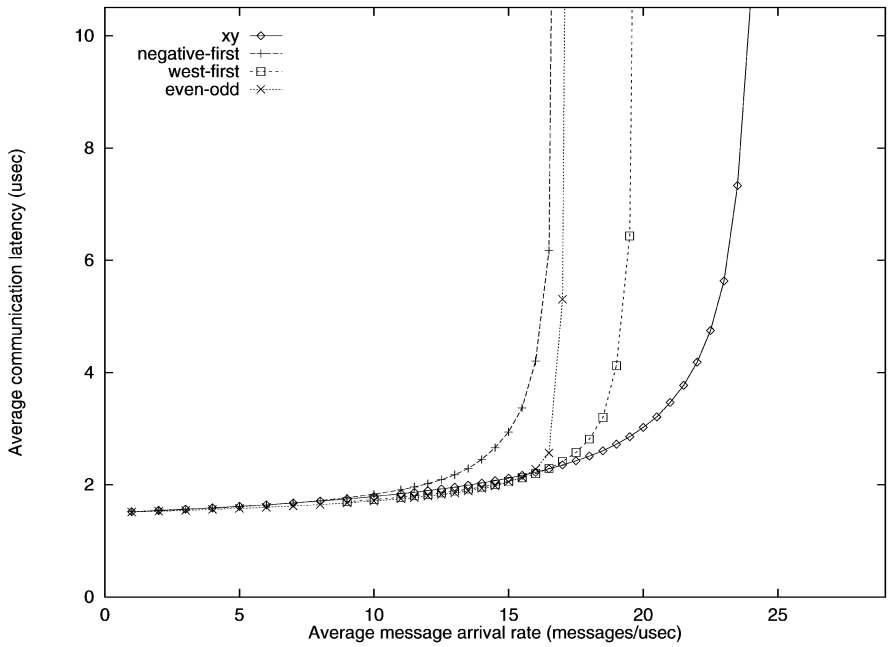


Fig. 4. Performance of the algorithms under uniform traffic.

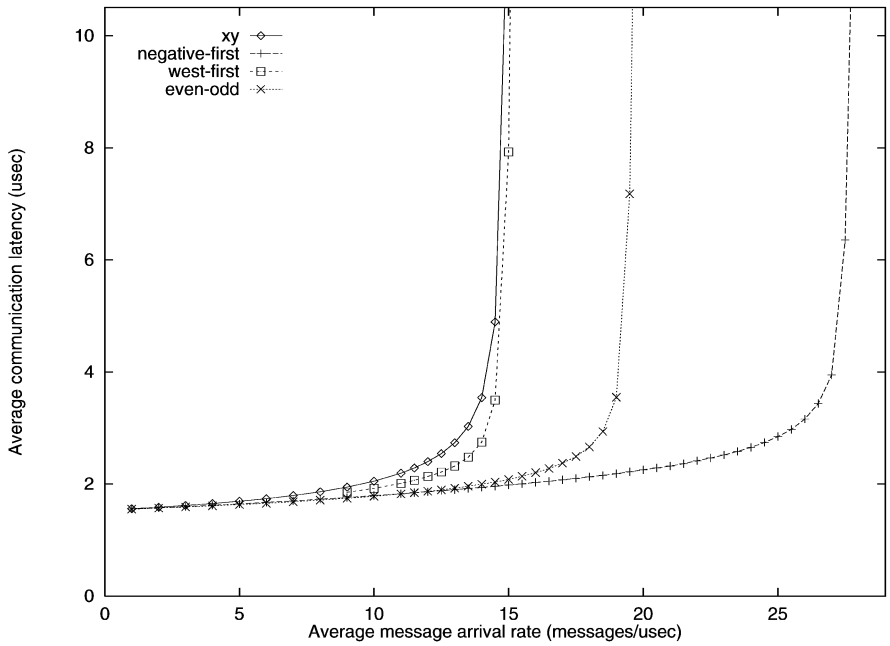


Fig. 5. Performance of the algorithms under the first transpose traffic, $(i, j) \rightarrow (14 - j, 14 - i)$.

traffic. In multiprocessors, these traffic patterns could be representative of computations in which critical sections or shared/replicated data are placed at the hot spot node(s).

In the simulation, we measure the average communication latency of the packets and the average sustainable network throughput under various traffic patterns. For each run of the simulation, the results are averaged over $110,000$ messages. Performance data are not collected for the first $40,000$ messages to allow the system to stabilize. The 95 percent confidence intervals are mostly within 2 percent of the means. Fig. 4 shows the simulation results for the uniform traffic. The results indicate that for uniform traffic, the nonadaptive *xy* algorithm outperforms all three partially adaptive algorithms at high traffic load. At low traffic load, all four algorithms perform about the same with the *west-first* algorithm saturating at a traffic load slightly higher than the odd-even turn model. This is mainly due to that the *xy* algorithm incorporates more global, long-term information about the characteristics of uniform traffic, which may lead to more even distribution of traffic. The *west-first* algorithm routes westbound messages in the same way as the *xy* algorithm, and thus it may benefit from this practice. However, our algorithm performs better than the *negative-first* algorithm.

In Figs. 5 and 6, we show simulation results under the first and the second transpose traffic patterns, respectively.

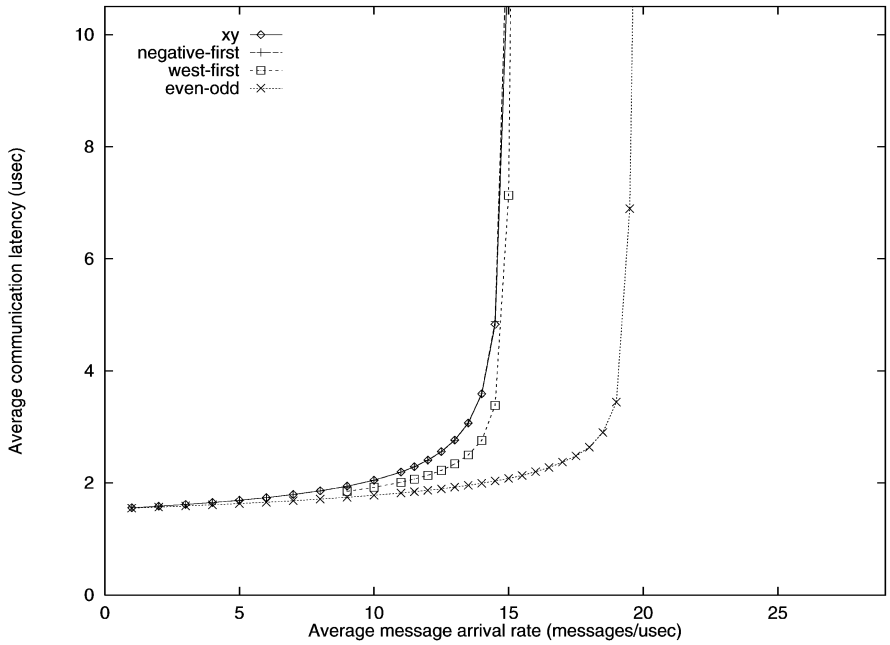


Fig. 6. Performance of the algorithms under the second transpose traffic, $(i, j) \rightarrow (j, i)$.

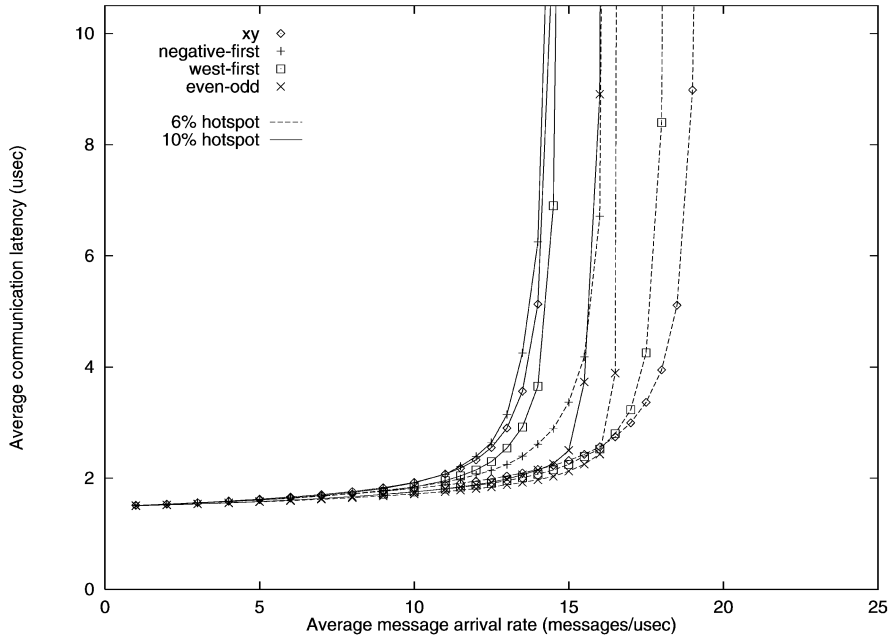


Fig. 7. Performance of the algorithms under the hot spot traffic with a single hot spot node $(7, 7)$. The hot spot percentages are 6 percent and 10 percent.

As expected, the *negative-first* algorithm performs the best with the first transpose traffic, as the pattern allows full adaptiveness for all the packets when the algorithm is used. Our algorithm performs better than the *west-first* and the *xy* algorithms. Under the second transpose traffic pattern, our algorithm is superior to all of the other algorithms. More importantly, from Figs. 5 and 6, one may observe that the odd-even turn model has very close performance under both transpose traffic patterns. This observation exhibits the importance of even routing adaptiveness provided by our model.

Hot spot traffic pattern, in which the hot spot nodes receive hot spot traffic in addition to the regular uniform traffic, is considered a more realistic traffic model [1]. In this paper, we consider various hot spot traffic patterns with different hot spot percentages. In the simulation, given a hot spot percentage of $h$, a newly generated message is directed to each hot spot node with an additional $h$ percent probability. We first simulate hot spot traffic with a single hot spot node. The hot spot node is chosen to be node $(7, 7)$, which is located at the center of the $15 \times 15$ mesh. Fig. 7 shows the simulation results with $6$ percent and $10$ percent hot spot traffic. As the hot spot percentage increases, the

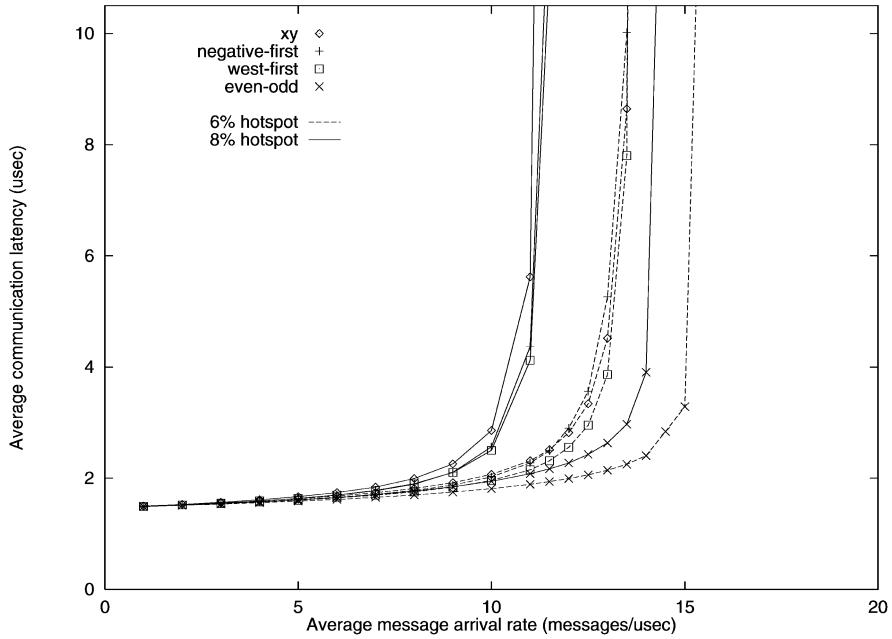


Fig. 8. Performance of the algorithms under the hot spot traffic with four hot spot nodes. The hot spot percentages are 6 percent and 8 percent.

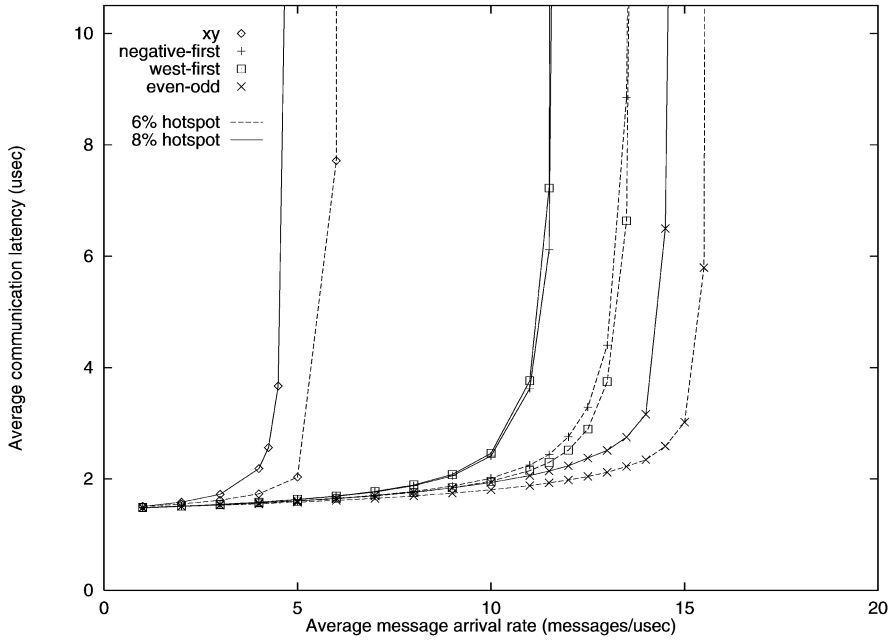


Fig. 9. Performance of the algorithms under the hot spot traffic with five hot spot nodes. The hot spot percentages are 6 percent and 8 percent.

sustainable traffic rate (or throughput) decreases for all four algorithms. However, the odd-even-turn-based routing algorithm is least vulnerable to the hot spot traffic. In fact, with 10 percent hot spot traffic, our routing algorithm outperforms the other algorithms. The performance of the *xy* algorithm is most seriously affected by the hot spot traffic, followed by the *west-first* algorithm.

We then simulate hot spot traffic with multiple hot spot nodes. In Fig. 8, we show simulation results with four hot spot nodes, which are chosen to be nodes $(5,5)$, $(5,9)$, $(9,5)$, and $(9,9)$. The hot spot percentages simulated are 6 percent and 8 percent. Our routing algorithm performs better than the other algorithms for both hot spot percentages. In addition, it is least vulnerable to the hot spot traffic. In Fig. 9, we show simulation results with five hot spot nodes, which are the center node $(7,7)$ and the four previous hot spot nodes. The hot spot percentages simulated are 6 percent and 8 percent. In comparison with the previous case with four hot spot nodes, the performance of the *xy* algorithm is seriously degraded in this case.

We may summarize our observation of the simulation results as follows: Although the *xy* algorithm performs well under uniform traffic, adaptiveness is important for nonuniform traffic. Adaptiveness may alleviate congestion problem caused by the nonuniformity of the traffic. In particular, the even adaptiveness provided by the odd-even turn model makes message routing less vulnerable to nonuniform factors such as hot spot percentage.

Furthermore, this property results in a smaller fluctuation of the network performance with respect to different types of traffic.

## 4 Extension to High-Dimensional Meshes

In this section, we show that the odd-even turn model can be readily extended to high-dimensional meshes. To facilitate the presentation, we will address 3D meshes.

Let the three dimensions of a 3D mesh be denoted by $x$, $y$, and $z$. Further, the two directions associated with $x$ dimension are represented by $x_+$ and $x_-$, which correspond to directions with increasing and decreasing $x$ coordinate, respectively. Similarly, we can define directions $y_+$, $y_-$, $z_+$, and $z_-$. In a 3D mesh, there are 24 90-degree turns. We use $x_+y_+$ to denote a turn that involves a change of direction from $x_+$ to $y_+$. The notations for the other types of turns are used in a similar fashion. A $yz$-plane consists of all the nodes that have the same coordinates of dimension $x$. An even (respectively, odd) $yz$-plane is a $yz$-plane whose dimension-$x$ coordinate is an even (respectively, odd) number. To facilitate the description of our scheme, the following four turns $x_+y_+$, $x_+y_-$, $x_+z_+$, and $x_+z_-$ collectively constitute the class of $x_+ - yz$ turns. Further, the turns $y_+x_-$, $y_-x_-$, $z_+x_-$, and $z_-x_-$ collectively constitute the class of $yz - x_-$ turns. In a 3D mesh, the odd-even turn model is governed by the following rules:

$yz$ **Rule.** *On any $yz$-plane, Rule 1 and Rule 2 adopted for 2D mesh routing are used, with dimension $y$ corresponding to dimension 0 and dimension $z$ corresponding to dimension 1.*

$x$ **Rule.** *A packet is not allowed to take any of the $x_+ - yz$ turns at a node located in an even $yz$-plane, and it is not allowed to take any of the $yz - x_-$ turns at a node located in an odd $yz$-plane.*

Essentially, we do not allow an $x_+ - yz$ turn to be connected to a $yz - x_-$ turn so that the property of deadlock freedom can be achieved. The existence of this property can be readily argued by following the proof of Theorem 1.

## 5 Conclusions

We have presented the odd-even turn model for designing partially adaptive wormhole routing algorithms without adding virtual channels. In comparison with the well-known turn model [15], our scheme provides more even routing adaptiveness. Simulation results demonstrate that communication performance of the meshes may be improved under nonuniform traffics using the proposed model.

Although fault tolerance is not addressed in this paper, the odd-even turn model may be useful for designing fault-tolerant routing algorithms. The relative locations where restrictions on the individual turns are placed is an issue that requires further study in this respect. In addition, nonminimal routing and 180 degree turns must also be considered to avoid faulty components [11]. This is an interesting subject for future work.

## Acknowledgments

The author thanks the referees for their insightful suggestions and comments. The author is grateful to Mr. Chun-Long Chen for helping him develop the simulator. This work was supported in part by the National Science Council of the Republic of China under grants NSC-87-2213-E011-045 and NSC-87-2213-E011-046.

## References

[1] R.V. Boppana and S. Chalasani, "A Comparison of Adaptive Wormhole Routing Algorithms," *Proc. Int'l Symp. Computer Architecture,* pp. 351–360, May 1993.
[2] Y.M. Boura and C.R. Das, "A Class of Partially Adaptive Routing Algorithms for $n$-Dimensional Meshes," *Proc. 1993 Int'l Conf. Parallel Processing,* pp. III–175–III–182, 1993.
[3] Y.M. Boura and C.R. Das, "Efficient Fully Adaptive Wormhole Routing in $n$-Dimensional Meshes." *Proc. Int'l Conf. Distributed Computing Systems,* pp. 589–596, 1994.
[4] A.A. Chien "A Cost and Speed Model for $k$-Ary $n$-Cube Wormhole Routers," *Proc. Hot Interconnects '93,* Aug. 1993.
[5] A.A. Chien and J.H. Kim, "Planar-Adaptive Routing: Low-Cost Adaptive Networks for Multiprocessors," *J. ACM,* vol. 42, no. 1, pp. 91–123, Jan. 1995.
[6] Intel Corporation, *Paragon XP/S Product Overview,* 1991.
[7] Intel Corporation, *A Touchstone DELTA System Description,* 1991.
[8] W.J. Dally, "Virtual-Channel Flow Control," *IEEE Trans. Parallel and Distributed Systems,* vol. 3, no. 2, pp. 194–205, Mar. 1992.
[9] W. J. Dally and C. L. Seitz, "Deadlock-Free Message Routing in Multiprocessor Interconnection Networks," *IEEE Trans. Computers,* vol. 36, no. 5, pp. 547–553, May 1987.
[10] J. Duato, "A New Theory of Deadlock-Free Adaptive Routing in Wormhole Networks," *IEEE Trans. Parallel and Distributed Systems,* vol. 4, no. 12, pp. 1,320–1,331, Dec. 1993.
[11] J. Duato, "A Theory of Fault-Tolerant Routing in Wormhole Networks," *IEEE Trans. Parallel and Distributed Systems,* vol. 8, no. 8, pp. 790–802, Aug. 1997.
[12] M.L. Fulgham and L. Snyder, "Performance of Chaos and Oblivious Routers under Non-Uniform Traffic," Technical Report UW-CSE-93-06-01, Univ. of Washington, July 1993.
[13] C.J. Glass and L.M. Ni, "Adaptive Routing in Mesh-Connected Networks," *Proc. Int'l Conf. Distributed Computing Systems,* pp. 12–19, 1992.
[14] C.J. Glass and L.M. Ni, "Maximally Fully Adaptive Routing in 2D Meshes," *Proc. 1992 Int'l Conf. Parallel Processing,* pp. 101–104, 1992.
[15] C.J. Glass and L.M. Ni, "The Turn Model for Adaptive Routing," *Proc. 19th Ann. Int'l Symp. Computer Architecture,* pp. 278–287, May 1992.
[16] C.J. Glass and L.M. Ni, "The Turn Model for Adaptive Routing," *J. ACM,* vol. 41, no. 5, pp. 874–902, Sept. 1994.
[17] D. Lenoski, J. Laudon, K. Gharachorloo, W.-D. Weber, A. Gupta, J. Hennessy, M. Horowitz, and M.S. Lam, "The Stanford Dash Multiprocessor," *Computer,* vol. 25, no. 3, pp. 63–79, Mar. 1992.
[18] D.H. Linder and J.C. Harden, "An Adaptive and Fault-Tolerant Wormhole Routing Strategy for $k$-Ary $n$-Cubes," *IEEE Trans. Computers,* vol. 40, no. 1, pp. 2–12, Jan. 1991.
[19] T.G. Mattson and G. Henry, "An Overview of the Intel TFLOPS Supercomputer," *Intel Technology J.* vol. 1, pp. 1–12, 1998.
[20] L.M. Ni and P.K. McKinley, "A Survey of Wormhole Routing Techniques in Direct Networks," *Computer,* vol. 26, no. 2, pp. 62–76, Feb. 1993.
[21] L. Schwiebert and D.N. Jayasimha, "Optimal Fully Adaptive Minimal Wormhole Routing for Meshes," *J. Parallel and Distributed Computing,* vol. 27, pp. 56–70, 1995.
[22] C.-C. Su and K.G. Shin, "Adaptive Deadlock-Free Routing in Multicomputers Using Only One Extra Virtual Channel," *Proc. Int'l Conf. Parallel Processing,* pp. 227–231, 1993.

[23] J. Upadhyay, V. Varavithya, and P. Mohapatra, "A Traffic-Balanced Adaptive Wormhole Routing Scheme for Two-Dimensional Meshes," *IEEE Trans. Computers,* vol. 46, no. 2, pp. 190–197, Feb. 1997.

**Ge-Ming Chiu** received the BS degree from the National Cheng-Kung University, Taiwan, in 1976, the MS degree from the Texas Technical University in 1981, and the PhD degree from the University of Southern California in 1991, all in electrical engineering. He is currently a professor in the Department of Electrical Engineering at the National Taiwan University of Science and Technology, Taipei, Taiwan. His research interests include distributed computing, fault-tolerant computing, and parallel processing. Dr. Chiu is a member of the IEEE Computer Society.