第30卷 第11期
2007年11月

计 算 机 学 报
CHINESE JOURNAL OF COMPUTERS

Vol. 30 No. 11
Nov. 2007

Mesh网中高效无死锁自适应路由算法

向 东[1)] 张跃鲤[2)]

[1)](清华大学软件学院 北京 100084)

[2)](清华大学计算机科学与技术系 北京 100084)

摘 要 提出了一种新的应用于三维Mesh网中的无死锁路由算法.在当今的商用多计算机系统中,二维和三维的Mesh网是多处理器网络最为常用的拓扑结构之一.在应用于Mesh网的平面自适应路由(Planar Adaptive Routing)算法中,每条物理通道只需三条虚拟通道就可以有效地在三维以及更高维的Mesh网中避免死锁的产生.然而,采用该算法,网络拓扑一维和三维分别有两条和一条虚拟通道始终处于空闲状态.该文所提出的算法针对三维Mesh网,每条物理通道只需两条虚拟通道就可以有效地避免死锁.文中通过充分的模拟数据验证了此算法的有效性.

关键词 容错路由;完全自适应路由;部分自适应路由;平面自适应路由;Mesh网

中图法分类号 TP306

Deadlock-Free Adaptive Routing in Fault-Tolerant Mesh Networks

XIANG Dong[1)] ZHANG Yue-Li[2)]

[1)](*School of Software, Tsinghua University, Beijing* 100084)

[2)](*Department of Computer Science and Technology, Tsinghua University, Beijing* 100084)

Abstract A new deadlock-free routing algorithm is proposed for 3-dimensional meshes. Most of the recent commercial machines are constructed using 2D or 3D meshes. The planar adaptive routing scheme was proposed and effective deadlock avoidance technique use only three virtual channels for each physical channel in 3-dimensional or higher dimensional mesh networks. However, there exist two idle virtual channels for all channels along the first dimension, and one idle virtual channel for channels along the third dimension. A new deadlock avoidance technique is proposed for 3-dimensional meshes using only two virtual channels for each physical channel. Sufficient simulation results are presented to demonstrate the effectiveness of the proposed algorithm.

Keywords fault-tolerant routing; fully adaptive routing; partially adaptive routing; planar adaptive routing; mesh; networks

1 引 言

在当今的实验用及商用的多计算机系统中[1],Mesh网是一种被广泛利用的拓扑结构.k元n维Mesh网是指一个n维的网格结构,在每一维上有k个结点.多计算机系统的性能在最大程度上取决于系统中点到点的通信算法的性能.因此,很有必要提出一些应用于Mesh网的高性能的容错路由算法.

Glass和Ni提出的转弯模型[2]是一个具有重要意义的无死锁路由机制.基于转弯模型,路由算法可以通过限制某些转弯消除通道间的环相关,达到

收稿日期:2006-10-17;最终修改稿收到日期:2007-06-09.本课题得到国家杰出青年基金(60425203)、国家自然科学基金面上基金(60573055)资助.向 东,男,1966年生,教授,博士生导师,主要研究领域为分布式计算、容错计算、集成电路可测试性设计、低成本测试、*BIST*、延迟测试及测试生成.E-mail:dxiang@tsinghua.edu.cn.张跃鲤,1982年生,硕士研究生,主要研究方向为分布式计算、容错计算.

避免死锁的效果，不过这种算法一定程度上牺牲了算法的自适应性.

对于自适应的容错路由算法[3]，通常需要将每条物理通道划分为一定数量的虚拟通道来实现死锁避免机制，通道上的物理资源（如缓存、带宽）会分配到各个虚拟通道之上. 因此，实现死锁避免所需要的虚拟通道个数成为影响路由机制资源利用效率的重要因素，使用更少的虚拟通道，可以使每条虚拟通道分配到更多的资源，提高算法效率，尤其是在物理资源比较有限的情况下；但是，更少的虚拟通道也会使得设计死锁避免机制的难度增加.

Chien 和 Kim[4]提出了一个具有重要意义的部分自适应路由算法——平面自适应路由（Planar Adaptive Routing），此算法将路由限制在一系列连续的平面内，只需三条虚拟通道就可以在任何维数的网络内避免死锁. 然而，在这种方法所应用的块状故障模型中，都会有一部分非故障点像故障点一样失效，这对于高维的 Mesh 网会造成很严重的计算能力的损失.

Gomez[5]等提出了一种通过选择中间节点而实现的两阶段容错路由机制，从源节点到中间节点和从中间节点到目标节点分别使用不同的虚拟子网，在每一子网内部使用 Duato[6]的完全自适应协议，两个子网使用相同的自适应虚拟通道，但使用不同的逃逸通道，也就是说，共需要三条虚拟通道.

块状故障模型是一种最为常用的故障模型，但是这种模型会使一些非故障节点失效. 少数的故障节点可能会使很多甚至全部的非故障节点失效，这种现象在高维的网络中更为明显. 向东[7-8]等提出了局部安全性信息以及扩展局部安全性信息并将其用于指导超立方体和 Mesh 网中的容错路由，在整个网络不安全时仍然可以实现有效的路由，并且在建立故障块时不需要使任何非故障点失效.

Wang[9]提出了最小连通部件（MCC）故障模型的概念，对于平面的 MCC，每个节点在其所在每个平面内有两份安全性信息. MCC 与块状模型相比，有更少的非故障节点失效，可以更有效地指导最短路径路由.

本文的主要工作：(1) 提出了一种新的应用于 Mesh 网的无死锁路由算法，对文献[4]中平面自适应算法中空闲虚拟通道加以有效利用，每条物理通道只需两条虚拟通道；(2) 将该路由算法扩展到有故障的 Mesh 网，利用两条虚拟通道有效支持最短与非最短路径路由；(3) 利用一种新的平面最小连通部件（MCC）故障模型来指导虫孔交换 Mesh 网络中的容错的路由.

2　预备知识

2.1　定义与符号

一个 k 元 n 维 Mesh 网有 k^n 个节点，两个节点 $(a_n a_{n-1}\cdots a_2 a_1)$ 和 $(b_n b_{n-1}\cdots b_2 b_1)$ 之间有连接当且仅当两节点坐标有且只有一位不同，并且 $|a_i - b_i| = 1$.

一个 Mesh 网中的非故障节点的安全性信息可通过如下递归方式定义：如果节点在两个不同维上存在故障或不安全的邻居节点，那么该节点为不安全节点；否则该节点为安全节点. 这一定义与文献[4]中的定义有所不同. 故障点与不安全点共同构成矩型的故障区域，不安全点在某些平面内仍然可以被使用. 在本文中，如果源节点到目标节点的路由路径长度等于两点之间的距离，并且路径上的节点均为非故障点，那么我们称此路径为最短可用路径.

二维 Mesh 网中故障点与不安全点集合 F 构成的故障块具有如下特点：(1) 在矩型区域的边界处没有故障点；(2) 矩型区域的内部包含所有的集合 F 中的故障点与不安全点；(3) 矩型区域的内部不包含任何集合 F 之外的节点. 三维以及更高维的 Mesh 网中的故障块可以用类似于文献[4]中的方法建立，唯一的区别在于没有非故障点被标记为失效.

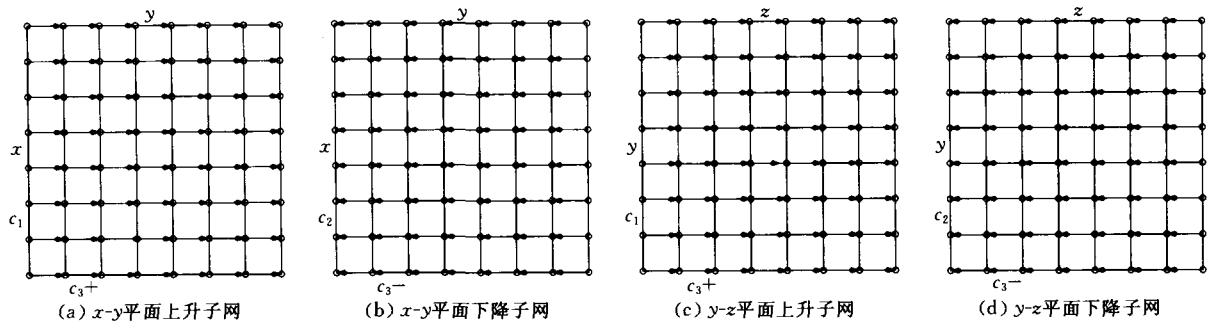
Mesh 网中的节点根据其安全性可以分为故障节点、不安全节点与安全节点三类. 如果一个网络系统内所有的节点均为不安全点或故障点，那么称这个网络系统为不安全的.

需要强调的是，已有的大多数方法都将不安全节点标记为失效，已失效的节点不能作为消息的源节点或目标节点. 而在本文中，不安全节点仍然可以作为源节点或目标节点使用，这也使得路由算法的性能有了很大的提高.

2.2　平面自适应路由算法

平面自适应路由算法[4]给出了一个简单的用于 n 维 Mesh 网的死锁避免机制，每条物理通道需要三条虚拟通道. 下面我们以三维的 Mesh 网为例来说明平面自适应路由算法的机制. Mesh 网首先被划分为一系列的平面，对于三维的 Mesh 网，分为 x-y 和 y-z 两种平面. 路由消息首先在 x-y 平面内路由，当 x 方向的偏移量变为 0 以后，消息跳转至 y-z 平面继续路由. x-y 平面又进一步被分成上升子网（图 1 (a)）与下降子网（图 1(b)）. x-y 平面中的虚拟通道分配方案如下：上升子网中 x 方向使用虚拟通道

c_1，y 方向使用 c_3+；下降子网中 x 方向使用 c_2，y 方向使用 c_3-. 在 y-z 平面，上升子网(图 1(c))中 y 方向使用虚拟通道 c_1，z 方向使用 c_3+；下降子网中 y 方向使用 c_2，z 方向使用 c_3-.



(a) x-y平面上升子网 (b) x-y平面下降子网 (c) y-z平面上升子网 (d) y-z平面下降子网

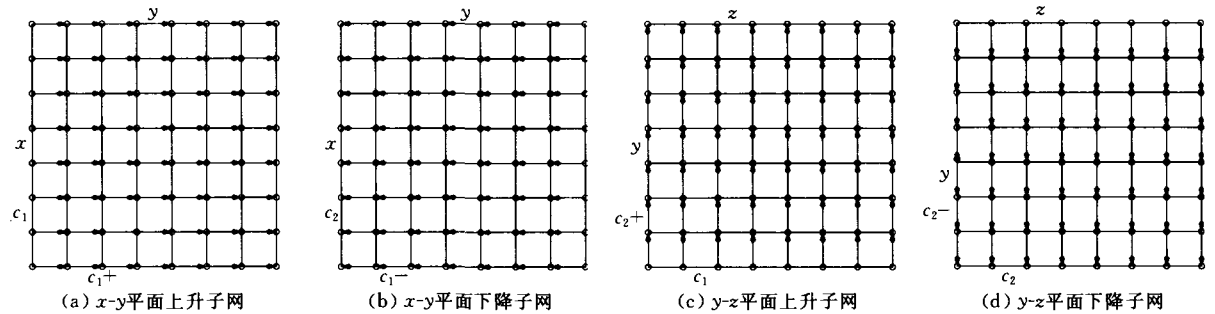
图 1 平面自适应路由的虚拟通道分配

在三维 mesh 中使用以上的虚拟通道分配方案，可以有效的避免死锁. 但是，可以发现，z 方向的 c_1 和 c_2 通道，x 方向的 c_3 通道，并没有得到利用. 这些空闲通道的存在意味着，在三维 mesh 网中避免死锁所需的虚拟通道的个数仍然有可以减少的空间.

3 三维 Mesh 网中的无死锁路由

在这一节我们将提出一种在三维 Mesh 网中只需要两条虚拟通道就可以实现死锁避免的技术. 我们的方法仍然将三维 Mesh 网划分为 x-y 平面与 y-z 平面，虚拟通道的分配如图 2 所示，在 x-y 平面，上升子网的 x 方向使用 c_1，y 方向使用 c_1+，下降子网的 x 方向使用 c_2，y 方向使用 c_1-；在 y-z 平面，上升子网 y 方向使用 c_2+，z 方向使用 c_1，下降子网 y 方向使用 c_2-，z 方向使用 c_2.

基于以上的虚拟通道分配策略，我们提出了新的自适应无死锁路由算法，在一般情况下，路由消息首先在 x-y 平面路由，当 x 方向的偏移量变为 0 时，跳转到 y-z 平面继续路由. 在开始 x-y 平面路由时，根据源节点与目标节点之间在 y 方向的偏移方向，选择将消息插入上升子网或下降子网，并且始终保持在此子网内路由. 在跳转到 y-z 平面时，同样根据 y 方向上的偏移方向选择上升或下降子网并在 y-z 平面内保持在此子网内路由.



(a) x-y平面上升子网 (b) x-y平面下降子网 (c) y-z平面上升子网 (d) y-z平面下降子网

图 2 只需两条虚拟通道的分配方案

图 3～图 5 分别为 x-y 上升子网、x-y 下降子网以及 y-z 平面上升子网中的路由算法描述. y-z 平面下降子网的路由算法与上升子网中的算法类似，y 方向通道由 c_2+ 变为 c_2-，z 方向通道由 c_1 变为 c_2. 值得注意的是，在 x-y 平面内路由，消息优先选择 x 方向进行路由，这样做是为了保证路由的自适应性，而在 y-z 平面，y 方向与 z 方向的路由具有相同的优先级别. 此路由算法具有如下性质.

```
算法. x-y 平面上升子网路由.
输入：当前点坐标(Xcurr, Ycurr, Zcurr),
      目标点坐标(Xdest, Ydest, Zdest)
输出：选择的输出通道 Channel
过程：
  Xoffset = Xdest − Xcurr;  Yoffset = Ydest − Ycurr;
  if Xoffset >0 and X 方向上 c1+通道可用 then
    Channel = X(c1+);
  if Xoffset <0 and X 方向上 c1−通道可用 then
    Channel = X(c1−);
  if Yoffset >0 then Channel = Select(X(c1), Y(c1+));
  if Yoffset =0 and Xoffset >0 then Channel = X(c1+);
  if Yoffset =0 and Xoffset <0 then Channel = X(c1−);
  if Yoffset =0 and Xoffset =0 then Channel = Internal;
```

图 3 x-y 平面上升子网中的无死锁路由算法

```
算法. x-y 平面下降子网路由.
输入：当前点坐标(Xcurr, Ycurr, Zcurr),
      目标点坐标(Xdest, Ydest, Zdest)
输出：选择的输出通道 Channel
过程：
  Xoffset = Xdest − Xcurr;  Yoffset = Ydest − Ycurr;
  if Xoffset >0 and X 方向上 c2+ 通道可用 then
    Channel = X(c2+);
  if Xoffset <0 and X 方向上 c2− 通道可用 then
    Channel = X(c2−);
  if Yoffset <0 then Channel = Select(X(c2), Y(c1−));
  if Yoffset =0 and Xoffset >0 then Channel = X(c2+);
  if Yoffset =0 and Xoffset <0 then Channel = X(c2−);
  if Yoffset =0 and Xoffset =0 then Channel = Internal;
```

图 4　x-y 平面下降子网中的无死锁路由算法

```
算法. y-z 平面上升子网路由.
输入：当前点坐标(Xcurr, Ycurr, Zcurr),
      目标点坐标(Xdest, Ydest, Zdest)
输出：选择的输出通道 Channel
过程：
  Zoffset = Zdest − Zcurr;  Yoffset = Ydest − Ycurr;
  if Zoffset >0 and Z 方向上 c1+ 通道可用 then
    Channel = Z(c1+);
  if Zoffset <0 and Z 方向上 c1− 通道可用 then
    Channel = Z(c1−);
  if Yoffset >0 then Channel = Select(Z(c1), Y(c2+));
  if Yoffset =0 and Zoffset >0 then Channel = Z(c1+);
  if Yoffset =0 and Zoffset <0 then Channel = Z(c1−);
  if Yoffset =0 and Zoffset =0 then Channel = Internal;
```

图 5　y-z 平面上升子网中的无死锁路由算法

引理 1.　使用上述路由算法时，在任何 x-y 平面内，不存在通道的环相关.

证明.　任何一个在 x-y 平面内路由的消息只可能在上升或下降子网中的一个子网内传输. 因此，上升子网与下降子网之间不会存在通道的相关性. 此外，在上升子网中 y 方向只会用到 c_1+ 通道，因此在上升子网内部不可能产生通道的环相关；类似地，在下降子网内 y 方向只用到 c_1− 通道，因此在下降子网内部也不可能产生通道的环相关. 综上，在 x-y 平面内不存在通道的环相关.　　证毕.

引理 2.　使用上述路由算法时，在任何 y-z 平面内，不存在通道的环相关.

证明.　证明方法同引理 1，此处从略.

引理 3.　使用上述路由算法时，在 x-y 平面与 y-z 平面之间不存在通道的环相关.

证明.　根据路由算法的定义，消息总是先在 x-y 平面内中路由，然后转到 y-z 平面中路由，因此，会存在从 x-y 平面通道到 y-z 平面通道的依赖关系，但是不可能存在从 y-z 平面通道到 x-y 平面通道的依赖关系. 因此在 x-y 平面通道与 y-z 平面通道之间也不可能存在环相关.　　证毕.

定理 1.　上述用于三维 Mesh 网的路由算法为无死锁的路由算法.

证明.　根据以上三条引理的结论，该路由算法在 x-y 平面内、y-z 平面内以及 x-y，y-z 平面之间都不可能存在通道的环相关. 因此，此路由算法不会产生任何形式的通道环相关，也就是说，此路由算法是无死锁的.　　证毕.

4　平面最小连通部件故障模型

用于二维 Mesh 网的最小连通部件故障模型(MCC)是在文献[9]中最早提出的. 此故障模型的建立无需让节点知道全局的故障分布信息. 每个节点只需根据邻近节点的情况存储两份安全性信息，一份标记在 $x+y+(x-y-)$ 方向上的安全性信息，另一份标记在 $x+y-(x-y+)$ 方向上的安全性信息.

我们以 $x+y+(x-y-)$ 方向为例说明安全性信息的标记过程. 初始状态时，所有非故障点都被标记为安全节点；如果一个安全点在 $x+$ 方向和 $y+$ 方向上的邻居节点都为故障点或不安全点，或者在 $x-$ 方向和 $y-$ 方向上的邻居节点都为故障点或不安全点，那么该节点就被标记为不安全节点；反复进行以上标记过程，直到所有节点的安全性达到稳定状态不再改变.

在文献[10]中，MCC 模型被扩展到三维 Mesh 网. 我们以 $x+y+z+(x-y-z-)$ 方向为例说明其标记过程. 初始状态时，所有非故障点都被标记为安全；如果一个安全点在 $x+$，$y+$，$z+$ 3 个方向(或 $x-$，$y-$，$z-$)的邻居节点均为故障点或不安全节点，那么，该节点就被标记为不安全节点；反复进行以上过程直至所有点安全状态稳定. 三维的 MCC 会将更少的点标记为不安全节点，每个节点需要保存的安全性信息为一个四元组 (a,b,c,d)，对应于三维 Mesh 网 4 条不同的体对角线的方向，$x+y+z+(x-y-z-)$，$x+y-z-(x-y+z+)$，$x+y+z-(x-y-z+)$，$x+y-z+(x-y+z-)$. a,b,c,d 的值可以为安全、不安全或故障.

如图 6 所示，一个 $5\times5\times5$ 的 Mesh 网中有 6 个故障节点，应用三维的 MCC 模型，在 $x+y-z-(x-y+z+)$ 方向上，非故障点(2,1,1)，(2,1,2)，(2,2,1)以及(1,2,2)被标记为不安全节点. 然而，三维的 MCC 模型对于我们提出的平面自适应路由算法并不是完全适用. 比如在图 6 中，(2,3,2)和(2,2,3)两个节点也应该被标记为不安全节点，

因为,如果($x+y-z-$)方向的消息在 y-z 平面(2,*,*)中路由时,如果走到这两个节点处,将会形成一个无法再走最短路径的死角.

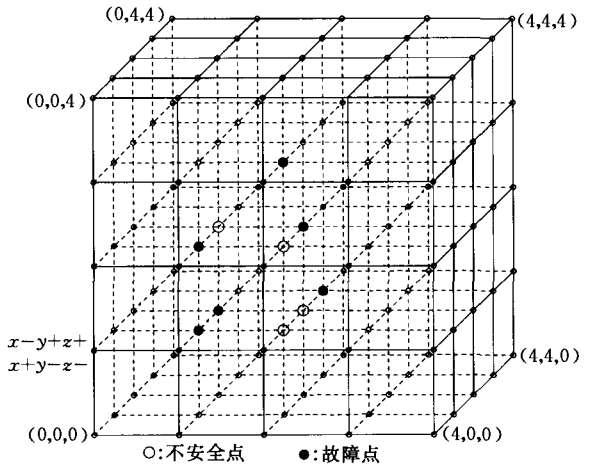


图 6 最小连通部件(MCC)故障模型

我们应用一种新的方法来标记节点的安全性信息.节点的安全性在其所在的 3 个平面内(x-y,y-z,z-x)分别计算,每个平面内计算二维 MCC 模型,节点保存平面内两个方向上的两份安全性信息.这样,每个节点总共需要保存六份安全性信息.

5 基于平面 MCC 的容错路由

此节给出一种新的应用于三维虫孔交换 Mesh 网的基于平面 MCC 故障模型的容错路由机制.路由算法如图 7～图 9 所示.路由开始时,根据 3 个方向上的偏移量确定开始路由的平面,如果源节点或目标节点在确定的路由平面内不安全,那么就搜索一个源节点或目标节点在此平面内的安全的邻居节点,作为新的源节点或目标节点,这样可以提高路由的成功率.

```
算法. 平面自适应容错路由.
输入:当前点 curr,目标点 dest
输出:下一跳路由
过程:
  计算 curr 与 dest 之间在 3 个方向上的偏移量,
  根据偏移量选择在各个平面内的 MCC 模型.
  if curr 和 dest 在同一个 x−y(y−z)平面内 then,
    if curr 在 x−y(y−z)平面内不安全 then,
      找到 curr 的一个在此平面内安全的邻居 v,
      将消息路由到 v;
    if dest 在 x−y(y−z)平面内不安全 then,
      找到 dest 的一个在此平面内安全的邻居 v,
      将 v 作为新的目标节点;
  if curr 与 dest 仅在 x 或 y 方向有偏移 then,
    route-in-plane-xy (curr, dest);
  if curr 与 dest 在 x 和 y 方向有偏移 then,
    route-in-plane-xy (curr, dest);
  if curr 与 dest 在 x,y,z 方向上均有偏移 then,
    route-in-plane-xy (curr, dest);
  if curr 与 dest 仅在 z 方向上有偏移 then,
    route-in-plane-yz (curr, dest);
  if curr 与 dest 在 y 和 z 方向上有偏移 then,
    route-in-plane-yz (curr, dest);
```

图 7 Mesh 网中的平面自适应容错路由算法

```
算法. route-in-plane-xy ().
输入:当前点 curr,目标点 dest
输出:下一跳路由
过程:
  if curr 和 dest 在 x 和 y 方向均有偏移 then,
    if curr 在 x 偏移方向的邻居节点安全 then,
      将消息沿 x 偏移方向路由至下一节点;
    if curr 在 x 偏移方向的邻居节点不安全 then,
      将消息沿 y 偏移方向路由至下一节点;
  if curr 与 dest 在 x 方向有偏移,y 方向无偏移 then,
    if curr 在 x 偏移方向的邻居节点安全 then,
      将消息沿 x 偏移方向路由至下一节点;
    if curr 在 x 偏移方向的邻居节点不安全 then,
      将消息沿 y 方向绕行路由,当到目标节点的最短
      路径存在时,恢复到沿 x 偏移方向路由;
  if curr 与 dest 在 y 方向有偏移,x 方向无偏移 then,
    if curr 在 y 偏移方向的邻居节点安全 then,
      将消息沿 y 偏移方向路由至下一节点;
    if curr 在 y 偏移方向的邻居节点不安全 then,
      将消息沿 x 方向绕行路由,当到目标节点的最短
      路径存在时,恢复到沿 y 偏移方向路由;
```

图 8 x-y 平面内的容错路由算法

```
算法. route-in-plane-yz ( ).
输入:当前点 curr,目标点 dest
输出:下一跳路由
过程:
  if curr 和 dest 在 z 和 y 方向均有偏移 then,
    if curr 在 z 和 y 偏移方向的邻居节点均安全 then,
      随机选取 z 或 y 偏方向,将消息路由至下一节点;
    if curr 在只在一个偏移方向的邻居节点安全 then,
      将消息路由至此安全的邻居节点;
  if curr 与 dest 在 z 方向有偏移,y 方向无偏移 then,
    if curr 在 z 偏移方向的邻居节点安全 then,
      将消息沿 z 偏移方向路由至下一节点;
    if curr 在 z 偏移方向的邻居节点不安全 then,
      将消息沿 y 方向绕行路由,当到目标节点的最短
      路径存在时,恢复到沿 z 偏移方向路由;
  if curr 与 dest 在 y 方向有偏移,z 方向无偏移 then,
    if curr 在 y 偏移方向的邻居节点安全 then,
      将消息沿 y 偏移方向路由至下一节点;
    if curr 在 y 偏移方向的邻居节点不安全 then,
      将消息沿 z 方向绕行路由,当到目标节点的最短
      路径存在时,恢复到沿 y 偏移方向路由;
```

图 9 y-z 平面内的容错路由算法

当在 x-y 或 y-z 平面内路由时,当前节点选择在偏移方向上的安全邻居节点作为下一跳路由.如果在当前节点与目标节点只在一个方向有偏移且此方向上的邻居节点为故障或不安全节点时,消息可以在子网内具有双向通道的方向上进行绕行路由,即在非最短路径方向上进行路由,x-y 平面内可以选择 x 方向进行绕行路由,在 y-z 平面可以选择 z 方向进行绕行路由.绕行路由可以使消息绕过故障或不安全节点,可以提高路由的成功率,同时由于路

由仍在上升或下降子网内部进行，因此这种绕行路由不会带来产生死锁的可能性.

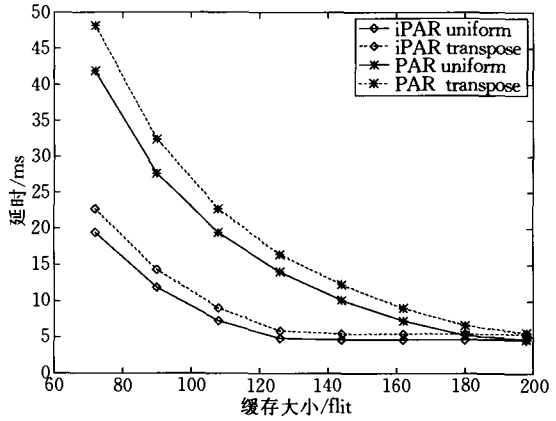
6　模拟数据与性能评估

我们将本文提出的无死锁平面自适应容错路由算法称为 iPAR(improved Planar Adaptive Routing)，并将其在一个微片级的模拟器中加以实现. 同时，我们在同一个模拟器中实现了 Chien 和 Kim[4] 提出的原始的平面自适应路由算法(PAR). 我们通过比较这两种算法在同一个模拟器中体现出来的性能，来验证 iPAR 算法在对高效利用资源实现高性能路由方面的改进与提高. 在模拟数据中，算法性能主要体现在两个参数上，消息路由的平均通信延迟(latency)以及标准化可接受流量(normalized accepted traffic，吞吐率除以网络的饱和负载，比如 8×8×8 Mesh 网的饱和负载为 0.25 flit/node/cycle，16×16×16 Mesh 网的饱和负载为 0.125 flit/node/cycle). 在一定的负载条件下，较低的延时和较高的标准化可接受流量意味着该算法具更好的性能.

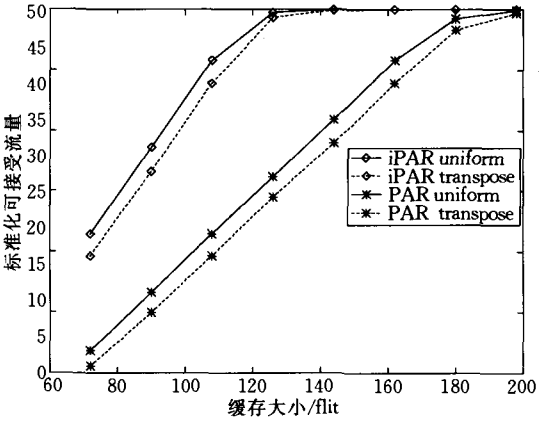
模拟所使用的网络结构为 16×16×16 Mesh 网. 在模拟中，使用两种不同的传输模型：均匀流量模型(uniform)以及对位传输模型(transpose). 所谓对位传输模型是指在 n 维 k 元 Mesh 网中，如果源节点的坐标为 $(i_1, i_2, \cdots, i_n)$，那么目标节点的坐标则为 $(k-1-i_1, k-1-i_2, \cdots, k-1-i_n)$. 路由消息根据负载的大小以特定的速率插入到网络中并随机选取非故障节点作为路由消息的源节点. 在模拟中，消息在节点的启动延迟与接收延迟均设为 0.75μs，相邻节点间的数据传输速率为 320Mbyte/s. 在容错网络中，故障点在网络中随机并且静态分布. 在模拟时，为公平比较两种算法，两者设置的节点缓存大小始终保持一致.

图 10 为在无故障的 16×16×16 Mesh 网中节点缓存变化时 iPAR 与 PAR 的性能比较. 标准化实用负载(normalized applied load)设定为 0.8. 从图中可以看出，当节点缓存较小时，两种算法在延时和标准化可接受流量上都有明显的差别，对于两种消息传输模型，iPAR 的性能均好于 PAR；随着节点缓存的增大，两种算法之间体现出来的性能差别逐渐减小，在节点缓存增加到 200 flit 时，两种算法基本上都得到了足够大的缓存，性能参数也趋向稳定.

图 11 为无故障的 16×16×16 Mesh 网中标准化实用负载(以下简称负载)变化时的性能比较. 节点缓存设定为 126 flit. 从图中可以看出，当负载小于 0.48 时，两种算法性能差别很小. 当负载高于 0.48 时，PAR 算法的延迟迅速增长，标准化可接受流量逐渐停止增长，甚至在负载大于 0.64 后出现下降趋势. iPAR 算法的性能保持稳定，在负载大于 0.8 时，性能才出现较为明显的下降.



(a) 缓存大小对延时的影响
(16×16×16 Mesh: fault number: 0, load: 0.8)



(b) 缓存大小对标准化可接受流量的影响
(16×16×16 Mesh: fault number: 0, load: 0.8)

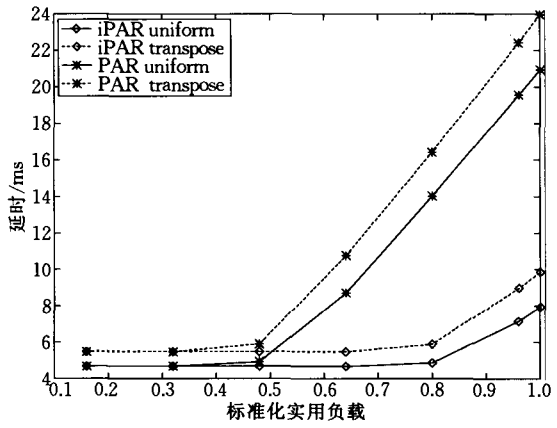
图 10　无故障网络中节点缓存大小对性能的影响

图 12～图 14 为在容错网络中对算法的性能评估. 在这里，我们对原始的 PAR 容错算法作了一些修改，原始的 PAR 容错算法使用三维块状故障模型，这种故障模型会使过多的非故障点失效，从而大大减小其与 iPAR 算法之间的可比性，因此，对于 PAR 算法我们使用文献[8]中平面块状故障模型.

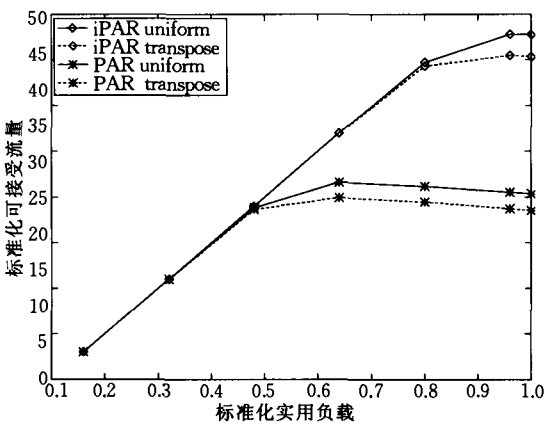
图 12 为 16×16×16 容错 Mesh 网中节点缓存大小变化对性能的影响. 网络中故障点个数设定为 50，标准化实用负载设定为 0.56. 从图中可以看出，在延时和标准化可接受流量两个参数上 iPAR 的性能均好于 PAR. 在延时方面，两种算法在对位传输模型下的差别更为明显. 不同于无故障网络中的比较结果，两种算法在节点缓存约为 200 flit 时仍然有着较为明显的差距.

图 13 为 16×16×16 容错 Mesh 网中负载变化时的性能比较. 网络中故障点个数设定为 50，节点缓存大小设为 180 flit. 从图中可以看出，在延时和标准化可接受流量两个方面，iPAR 算法的性能均好于 PAR，并且两种算法的差距随着负载的增加而增大.

图 14 体现了 16×16×16 容错 Mesh 网中故障点个数对系统性能的影响. 节点缓存大小设定为 216 flit，标准化实用负载设定为 0.2，故障点个数变化区间为 50～300. 从图中可以看出，在两种传输模型下，使用平面 MCC 故障模型的 iPAR 算法的性能始终好于使用平面块状故障模型的 PAR 算法. 从均匀传输模型的标准化可接受流量上看，PAR 算法在故障点个数大于 150 时性能明显下降，而 iPAR 算法在故障点个数大于 250 时才出现明显下降.

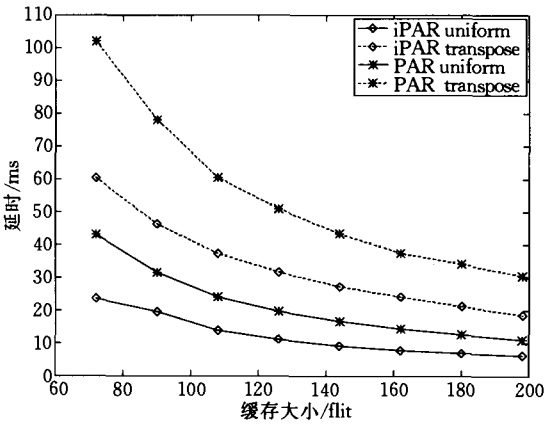


(a) 标准化实用负载对延时的影响
(16×16×16 Mesh: buffer: 126, fault: 0)

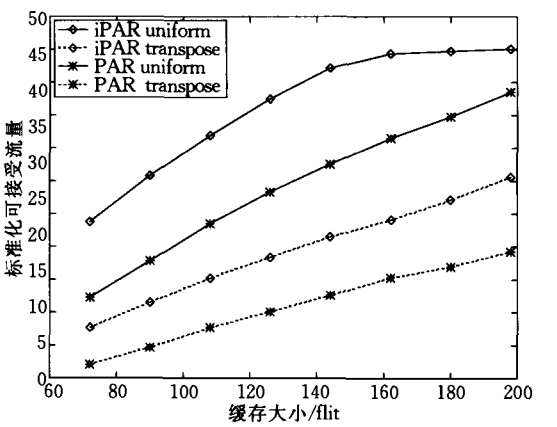


(b) 标准化实用负载对标准化可接受流量的影响
(16×16×16 Mesh: buffer: 126, fault: 0)

图 11 无故障网络中标准化实用负载对性能的影响

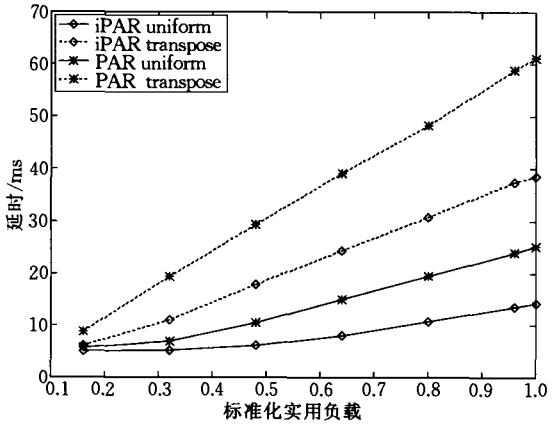


(a) 缓存大小对延时的影响
(16×16×16 Mesh: fault number: 50, load: 0.56)

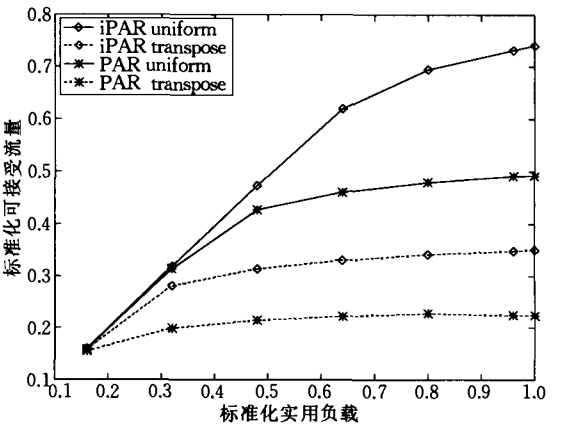


(b) 缓存大小对标准化可接受流量的影响
(16×16×16 Mesh: fault number: 50, load: 0.56)

图 12 容错 Mesh 网中节点缓存大小对性能的影响

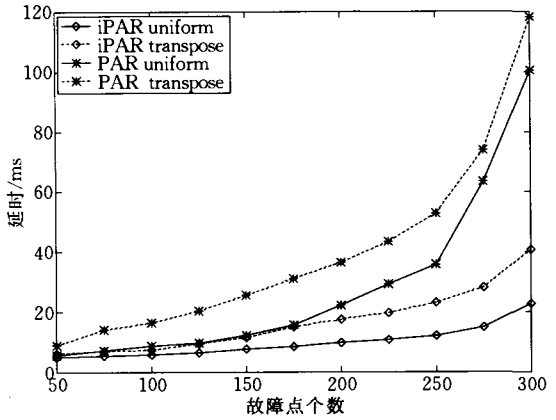


(a) 标准化实用负载对延时的影响
(16×16×16 Mesh: buffer: 180, fault: 50)

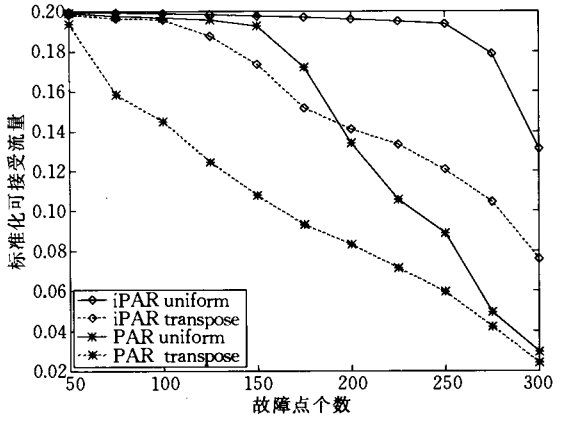


(b) 标准化实用负载对标准化可接受流量的影响
(16×16×16 Mesh: buffer: 180, fault: 50)

图 13 容错 Mesh 网中标准化实用负载对性能的影响



(a) 故障点个数对延时的影响
(16×16×16 Mesh: buffer: 126, load: 0.2)



(b) 故障节点个数对标准化可接受流量的影响
(16×16×16 Mesh: buffer: 126, load: 0.2)

图 14　容错 Mesh 网中故障节点个数对性能的影响

7　结　论

在当今的商用与实验用多计算机系统的网络设计中，经常用到的是二维和三维的 Mesh/Torus 拓扑结构，更高维的网络拓扑几乎很少有应用. 在文献[4]中提出的平面自适应路由机制中，每条物理通道只需要三条虚拟通道就可以在 n 维的 Mesh 网中避免死锁，但是在最低维和最高维上存在着空闲的虚拟通道. 本文提出了一种专门用于三维 Mesh 网的新的平面自适应路由机制，每条物理通道只需两条虚拟通道就可以避免死锁的产生. 同时，本文将此算法扩展为具有容错性，使用了平面最小连通部件(MCC)故障模型来指导虫孔交换网络中的容错路由. 大量的模拟实验数据表明，本算法与原始的平面自适应算法相比，无论在无故障网络或是容错网络中，都具有更好的性能表现.

参　考　文　献

[1] Allen F et al. Blue gene: A vision for protein science using a petaflop supercomputer. IBM Systems Journal, 2001, 40(2): 310-327

[2] Glass C J, Ni L M. The turn model for adaptive routing. Journal of ACM, 1994, 40(5): 874-902

[3] Duato J, Yalamanchili S, Ni L. Interconnection Networks: An Engineering Approach. Piscataway, NJ: IEEE Press, 1997

[4] Chien A A, Kim J H. Planar adaptive routing: Low-cost adaptive networks for multiprocessors. Journal of ACM, 1995, 42(1): 91-123

[5] Gomez M E, Nordbotten N A, Flich J, Lopez P, Robles A, Duato J, Skeie T, Lysne O. A routing methodology for achieving fault tolerance in direct networks. IEEE Transactions on Computers, 2006, 55(4): 400-415

[6] Duato J. A new theory of deadlock-free adaptive routing in wormhole networks. IEEE Transactions on Parallel and Distributed Systems, 1997, 4(12): 1320-1331

[7] Xiang D. Fault-tolerant routing in hypercube multicomputers using local safety information. IEEE Transactions on Parallel and Distributed Systems, 2001, 12(9): 942-951

[8] Xiang D, Sun J G, Wu J, Thulasiraman K. Fault-tolerant routing in meshes/tori using planarly constructed fault blocks//Proceedings of the 34th International Conference on Parallel Processing. Oslo, Norway, 2005: 577-584

[9] Wang D. A rectilinear-monotone polygonal fault block model for fault-tolerant minimal routing in mesh. IEEE Transactions on Computers, 2003, 52(3): 310-320

[10] Jiang Z, Wu J, Wang D. A new fault information model for fault-tolerant adaptive and minimal routing in 3-D meshes// Proceedings of the 34th International Conference on Parallel Processing. Oslo, Norway, 2005: 500-507

XIANG Dong, born in 1966, professor, Ph. D., Ph. D. supervisor. His research interests include design and test of digital systems, including design for testability, testing, test pattern generation, and built-in self-test, fault-tolerant computing, distributed computing, and computer networking.

ZHANG Yue-Li, M. S.. His research interests include fault-tolerant computing, distributed computing and computer networks.

Background

Mesh-connected networks have been widely used in recent experimental or commercial multi-computers. A mesh network has an n-dimensional grid structure with k nodes in each dimension (called mesh for short). Performance of a multi-computer highly depends on the routing algorithm. It is quite possible for one or more of the system nodes or links between any pair of nodes to become faulty in a large scale system. An effective fault-tolerant routing algorithm in a mesh network is essential for a high-performance multi-computer system.

In this paper, a new deadlock-free adaptive routing algorithm is proposed for meshes with only two virtual channels per physical channel. The number of virtual channels per physical channel required for deadlock-free routing is important for cost-effective and high-performance system design. The planar adaptive routing scheme is an effective deadlock avoidance technique using only three virtual channels for each physical channel in 3-dimensional or higher dimensional mesh networks. However, there exist idle virtual channels for all channels along the first dimension, and one idle virtual channel for channels along the last dimension in a mesh network. A new deadlock avoidance technique is proposed for 3-dimensional meshes using only two virtual channels, where a virtual channel can be shared by two consecutive planes without any cyclic channel dependency. The deadlock-free adaptive routing scheme is then modified to a deadlock-free adaptive fault-tolerant routing scheme based on a planarly constructed minimum connected components (MCCs) fault model.