基于有故障区域的 Mesh 网络目标结点间的最短路由算法

林政宽 王铭成 郭莉莉 杜满意
(苏州大学计算机科学与技术学院 苏州 215006)

摘 要 Mesh 网络是较早研究的且现在仍然是最为重要的、最有吸引力的网络模型之一。因其结构、规则简单及良好的可扩展性,易于 VLSI(超大规模集成电路)的实现,网格(Mesh)网络不仅成为了许多理论研究的基础模型,而且也是许多大型多处理器并行计算机系统所采用的拓扑结构。给出了两种故障情形下的最短路由算法:1)当 Mesh 的行数大于等于 3 且列数大于等于 3、出现一个矩形故障区域时,给出了任意两个无故障结点间的最短路由算法,并且计算出了路径长度;2)当 Mesh 的行数≥3 且列数≥3、某个结点及其 k 跳以内的邻居结点出现故障时,给出了任意两个无故障结点间的最短路由算法,并且计算出了路径长度。

关键词 最短路由算法,容错路由算法,网格网络
中图法分类号 TP393 **文献标识码** A

Shortest Routing Algorithm Based on Target Node in Mesh Network with Faulty Area

LIN Cheng-kuan WANG Ming-cheng GUO Li-li DU Man-yi
(School of Computer Science and Technology, Soochow University, Suzhou 215006, China)

Abstract Mesh network is studied early and it is still one of the most important and attractive network models at present. Because of its simple, regular and scalable structure, and it is useful for the implementation of VLSI (Very Large Scale Integrated Circuit), Mesh network has not only become the basic model of many theoretical studies, but also the topology structure of many large multiprocessor and parallel computer systems. A mesh with m rows and n columns is denoted by $M_{m,n}$. In this paper, we gave the shortest routing algorithm under two different kinds of faulty area. 1) We gave a routing algorithm to find the shortest path between any two fault-free nodes in $M_{m,n}$ with $m \geqslant 3$ and $n \geqslant 3$, and calculated the length of the path obtained by the algorithm when there is a rectangular faulty region. 2) We gave a routing algorithm to find the shortest path between any two fault-free nodes in $M_{m,n}$ with $m \geqslant 3$ and $n \geqslant 3$, and calculated the length of the path given by the algorithm when there exists such a situation that a node and its k-hop neighbours are faulty.

Keywords Shortest routing algorithm, Fault-tolerant routing algorithm, Mesh network

1 引言

随着硬件设备的不断发展,由成千上万个处理器组成的大规模互连网络变得越来越普遍。选择一个合适的互连网络是设计并行处理与分布式系统的一个重要部分。目前,人们已经提出了很多网络拓扑结构,其中超立方体(hypercube[1])有许多优良的特性,例如递归结构、规律性、对称性、直径小、度数低等。这些优点对于大规模并行处理与分布式系统的设计是非常重要的。因此超立方体成为最流行的互连网络拓扑结构之一,且已被用于商用并行计算机系统中,不少学者对它进行了深入的研究[2-4]。为了保持超立方体网络的优点,同时避开因边数过多而引起的成本高、效益低的缺点,人们转而研究超立方体网络的各种变形,例如折叠立方体(folded hypercube[5])、交叉立方体(crossed cube[6])、扭立方体网络(twisted cube[7])。

同样,网格(Mesh)[8]网络因其结构、规则简单及良好的可扩展性,易于 VLSI(超大规模集成电路)的实现,不仅成为许多理论研究的基础模型,而且也是许多大型多处理器并行计算机系统所采用的拓扑结构。因此,基于 Mesh 网络是并行计算机系统中最为重要的网络模型之一,并且许多基于 Mesh 网络的商用或研究所用的并行计算机系统已经问世[9-14],本文以 Mesh 网络为分析模型。

路由是在网络的结点之间传递信息的过程。它的性能和可靠性对于系统网络的表现有着举足轻重的作用。为了有效地通过网络传输数据包,使用路由算法建立信息传递路径是必要的。路由算法为从源结点到目的结点的数据包设置了一条路径[15]。网络中出现故障结点的路由算法是容错路由算法[16-17]。当一个大型多处理器并行计算机系统投入使用之后,结点发生故障是不可避免的。在这种情况下,网络的制造商和使用者最关心的问题是网络中所有的无故障结点之间能否保持通信,即无故障结点中任意给定的源结点和目的结点之间是否存在一条无故障路径[18]。因此,网络的容错路由算

本文受软件新技术与产业化协同创新中心,国家自然科学基金项目(61572340)资助。
林政宽(1976-),男,副教授,CCF 会员,主要研究方向为并行计算系统、交换网络、无线网络、无线传感器网络、算法设计与分析等,E-mail:cklin@suda.edu.cn(通信作者);王铭成(1994-),男,主要研究方向为互联网络,E-mail:wangmc3721@163.com;郭莉莉(1993-),女,硕士生,主要研究方向为图论、并行与分布式计算;杜满意(1991-),男,硕士生,主要研究方向为图论、并行与分布式计算。

法成为一个不可回避的研究课题。近年来，Liu 等人[19]提出一种全新的基于优先级的 Mesh 容错路由算法（fault-tolerant routing in mesh based on priority，PR 算法）；王高才等人[20]提出一种基于局部信息的路由算法。模拟结果表明，该路由算法所构造的路由路径长度非常接近于 2 个结点之间的最优路径长度。

本文所做的工作如下：在 Mesh 网络中，在出现一个矩形故障区域的情况下或某个结点及其 k 跳以内的邻居出现故障时，分别给出了任意两个无故障结点间的最短路由算法和路径长度。

2　预备知识

在图论符号定义中，用 G 表示由一个顶点集 V 和一个边集 E 组成的一个简单无向图。任意两个结点 u 和 v，在图 G 中若满足 $(u,v)\in E$，则称 u 和 v 相邻。u 和 v 之间的距离表示为 $d(u,v)$。一个由 m 行和 n 列组成的 Mesh 网络表示为 $M_{m,n}$。网络中的结点 u 可以表示为 (u_i,u_j)，其中 u_i 表示结点 u 所在的行且满足 $0\leqslant u_i\leqslant m-1$，$u_j$ 表示结点 u 所在的列且满足 $0\leqslant u_j\leqslant n-1$。该图的点集表示为 $V(M_{m,n})=\{(i,j)\mid 0\leqslant i\leqslant m-1\ \&\ 0\leqslant j\leqslant n-1\}$，边集表示为 $E(M_{m,n})=\{((i,j),(i+1,j))\mid 0\leqslant i\leqslant m-2\ \&\ 0\leqslant j\leqslant n-1\}\cup\{((i,j),(i,j+1))\mid 0\leqslant i\leqslant m-1\ \&\ 0\leqslant j\leqslant n-2\}$。如果 $V'\subseteq V(M_{m,n})$，则 $M'_{m,n}-V'$ 表示 $M'_{m,n}$ 中删除点集合 V' 后的剩余子图。令 $M_{m,n}$ 的最外面一圈的结点集合为 $Border(M_{m,n})=\{(i,j)\mid i\in\{0,m-1\}\ \&\ 0\leqslant j\leqslant n-1\}\cup\{(i,j)\mid 0\leqslant i\leqslant m-1\ \&\ j\in\{0,n-1\}\}$。若结点属于 $Border(M_{m,n})$，则称该结点为边界点；否则称为内部点。

本文用 $R(x_i:y_i,z_j)$ 表示一条沿行方向的路径，如果 $x_i<y_i$，则路径为 $\langle(x_i,z_j),(x_i+1,z_j),\cdots,(y_i,z_j)\rangle$；如果 $x_i>y_i$，则路径为 $\langle(x_i,z_j),(x_i-1,z_j),\cdots,(y_i,z_j)\rangle$。同理，用 $R(x_i,y_j:z_j)$ 表示一条沿列方向的路径，如果 $y_j<z_j$，则路径为 $\langle(x_i,y_j),(x_i,y_j+1),\cdots,(x_i,z_j)\rangle$；如果 $y_j>z_j$，则路径为 $\langle(x_i,y_j),(x_i,y_j-1),\cdots,(x_i,z_j)\rangle$。令 $P_1=\langle x,\cdots,y\rangle$ 与 $P_2=\langle w,\cdots,z\rangle$，若 $(y,w)\in E(G)$ 且 $V(P_1)\cap V(P_2)=\varnothing$ 或 $y=w$ 且 $V(P_1)\cap V(P_2)=\{y\}$，则用 $Q=P_1+P_2$ 表示 x 经由 P_1 与 P_2 至 z 的路径。

本文用 $\Delta_i(x,y)$ 表示结点 x 和结点 y 之间行的差值，即 $\Delta_i(x,y)=|x_i-y_i|$；用 $\Delta_j(x,y)$ 表示结点 x 和结点 y 之间列的差值，即 $\Delta_j(x,y)=|x_j-y_j|$；用 L 表示路径 P 的长度，即 $L=|V(P)|-1$。

3　网格中存在故障区域的最短路由算法

本节讨论在 $M_{m,n}$ 中存在故障区域时，两个无故障结点间的最短路由算法。本文所讨论的故障区域分为两种：1）故障区域是矩形；2）某结点及其 k 跳之内的邻居结点出现故障所形成的故障区域。

3.1　矩形故障区域的最短路由算法

令 p 与 q 为 $M_{m,n}$ 的任意两个结点且满足 $p_i\leqslant q_i$ 与 $p_j\leqslant q_j$，则令 $F(p,q)=\{(r_i,r_j)\mid p_i\leqslant r_i\leqslant q_i\ \&\ p_j\leqslant r_j\leqslant q_j\}$ 为 $M_{m,n}$ 中的矩形故障区域。算法 RWRF 即为 $M_{m,n}$ 中存在矩形故障区域 $F(p,q)$ 时的最短路由算法。

算法 1　Routing_with_Rectangle_Fault，RWRF（x，y，$F(p,q)$，$M_{m,n}$）

输入：结点 x 与结点 y，矩形故障区域 F(p,q)，$M_{m,n}$

输出：结点 x 与结点 y 之间于 $M_{m,n}-F(p,q)$ 的最短路径

```
1. Begin
2.   If |{x,y}∩F(p,q)|>0 ‖ Δi(p,q)=m && pj>0 && qj<n-1 ‖ Δj(p,q)=n && pi>0 && qi<m-1
3.     Return failure;
4.   Else
5.   {
6.     If |V(R(xi,xj:yj))∩F(p,q)|=0 && |V(R(xi:yi,yj))∩F(p,q)|=0
7.       Return R(xi,xj:yj)+R(xi:yi,yj);
8.     Else If |V(R(xi:yi,xj))∩F(p,q)|=0 && |V(R(yi,xj:yj))∩F(p,q)|=0
9.       Return R(xi:yi,xj)+R(yi,xj:yj);
10.    Else If |V(R(xi,xj:yj))∩F(p,q)|>0 && |V(R(yi,xj:yj))∩F(p,q)|>0
11.    {
12.      If pi=0
13.        Return R(xi:qi+1,xj)+R(qi+1,xj:yj)+R(qi+1:yi,yj);
14.      Else If qi=m-1
15.        Return R(xi:pi-1,xj)+R(pi-1,xj:yj)+R(pi-1:yi,yj);
16.      Else
17.      {
18.        If xi≤yi && Δi(x,p)≤Δi(y,q) ‖ xi>yi && Δi(x,q)>Δi(y,p)
19.          Return R(xi:pi-1,xj)+R(pi-1,xj:yj)+R(pi-1:yi,yj);
20.        Else
21.          Return R(xi:qi+1,xj)+R(qi+1,xj:yj)+R(qi+1:yi,yj);
22.      }
23.    }
24.  Else If|V(R(xi:yi,xj))∩F(p,q)|>0 && |V(R(xi:yi,yj))∩F(p,q)|>0
25.  {
26.    If pj=0
27.      Return R(xi,xj:qj+1)+R(xi:yi,qj+1)+R(yi,qj+1:yj);
28.    Else If qj=n-1
29.      Return R(xi,xj:pj-1)+R(xi:yi,pj-1)+R(yi,pj-1:yj);
30.    Else
31.    {
32.      If xj≤yj && Δj(x,p)≤Δj(y,q) ‖ xj>yj && Δj(x,q)>Δj(y,p)
33.        Return R(xi,xj:pj-1)+R(xi:yi,pj-1)+R(yi,pj-1:yj);
34.      Else
35.        Return R(xi,xj:qj+1)+R(xi:yi,qj+1)+R(yi,qj+1:yj);
36.    }
37.  }
38. }
39. End
```

接下来讨论算法 RWRF 的正确性并计算该算法所得到的路径长度。

定理 1 令 $F(p,q)$ 为 $M_{m,n}$ 的矩形故障区域且结点 x 与结点 y 为无故障结点。若 $M_{m,n}-F(p,q)$ 为连通图，则算法 RWRF 会建立结点 x 与结点 y 于 $M_{m,n}-F(p,q)$ 的最短路径。

证明：基于 $M_{m,n}$ 的定义，不失一般性，假设 $x_i \leqslant y_i$ 且 $x_j \leqslant y_j$。首先，RWRF 算法的第(2)行检查结点 x 与结点 y 是否为无故障结点和 $M_{m,n}-F(p,q)$ 是否为连通图。若结点 x 或结点 y 为故障结点，或 $M_{m,n}-F(p,q)$ 为非连通图，则算法将回传失败，即无法找到连接结点 x 与结点 y 的无故障路径。

若 $R(x_i,x_j:y_j)$ 与 $R(x_i:y_i,y_j)$ 不包含故障结点，则算法的第(6)、第(7)行将回传路径 $R(x_i,x_j:y_j)+R(x_i:y_i,y_j)$，回传路径的长度为 $\Delta_i(x,y)+\Delta_j(x,y)$；若 $R(x_i:y_i,x_j)$ 与 $R(y_i,x_j:y_j)$ 不包含故障结点，则算法的第(8)、第(9)行将回传路径 $R(x_i:y_i,x_j)+R(y_i,x_j:y_j)$，回传路径的长度为 $\Delta_i(x,y)+\Delta_j(x,y)$。

接下来将讨论另外两类情况。

情况 1 结点 x 和结点 y 不同行不同列。由于结点 x 与结点 y 为无故障结点，因此不存在 $R(x_i,x_j:y_j)$，$R(x_i:y_i,y_j)$，$R(x_i:y_i,x_j)$ 与 $R(y_i,x_j:y_j)$ 4 条路径同时存在故障结点的情况。故仅需再讨论 $R(x_i,x_j:y_j)$ 与 $R(y_i,x_j:y_j)$ 同时存在故障结点，或 $R(x_i:y_i,y_j)$ 与 $R(x_i:y_i,x_j)$ 同时存在故障结点的情况。

情况 1.1 $R(x_i,x_j:y_j)$ 与 $R(y_i,x_j:y_j)$ 同时存在故障结点。当 $p_i=0$ 时，即 p 属于 $Border(M_{m,n})$，则算法第(12)、第(13)行将回传路径 $R(x_i:q_i+1,x_j)+R(q_i+1,x_j:y_j)+R(q_i+1:y_i,y_j)$，回传路径的长度为 $\Delta_i(x,y)+\Delta_j(x,y)+2\Delta_i(y,q)+2$。当 $p_i\neq 0$ 且 $q_i=m-1$ 时，则算法的第(14)、第(15)行将回传路径 $R(x_i:p_i-1,x_j)+R(p_i-1,x_j:y_j)+R(p_i-1:y_i,y_j)$，算法回传路径的长度为 $\Delta_i(x,y)+\Delta_j(x,y)+2\Delta_i(x,p)+2$。当 $p_i\neq 0$ 且 $q_i\neq m-1$ 时，若 $\Delta_i(x,p)\leqslant\Delta_i(y,q)$，则算法的第(18)、第(19)行将回传路径 $R(x_i:p_i-1,x_j)+R(p_i-1,x_j:y_j)+R(p_i-1:y_i,y_j)$，回传路径的长度为 $\Delta_i(x,y)+\Delta_j(x,y)+2\Delta_i(x,p)+2$；否则，算法第(20)、第(21)行将会回传路径 $R(x_i:q_i+1,x_j)+R(q_i+1,x_j:y_j)+R(q_i+1:y_i,y_j)$，回传路径的长度为 $\Delta_i(x,y)+\Delta_j(x,y)+2\Delta_i(y,q)+2$。

情况 1.2 $R(x_i:y_i,x_j)$ 与 $R(x_i:y_i,y_j)$ 同时存在故障结点。当 $p_j=0$ 时，即 p 属于 $Border(M_{m,n})$，则算法第(26)、第(27)行将回传路径 $R(x_i,x_j:q_j+1)+R(x_i:y_i,q_j+1)+R(y_i,q_j+1:y_j)$，回传路径的长度为 $\Delta_i(x,y)+\Delta_j(x,y)+2\Delta_j(y,q)+2$。当 $p_j\neq 0$ 且 $q_j=n-1$ 时，算法的第(28)、第(29)行将回传路径 $R(x_i,x_j:p_j-1)+R(x_i:y_i,p_j-1)+R(y_i,p_j-1:y_j)$，回传路径的长度为 $\Delta_i(x,y)+\Delta_j(x,y)+2\Delta_j(x,p)+2$。当 $p_j\neq 0$ 且 $q_j\neq n-1$ 时，若 $\Delta_j(x,p)\leqslant\Delta_j(y,q)$，算法的第(32)、第(33)行将回传路径 $R(x_i,x_j:p_j-1)+R(x_i:y_i,p_j-1)+R(y_i,p_j-1:y_j)$，回传路径的长度为 $\Delta_i(x,y)+\Delta_j(x,y)+2\Delta_j(x,p)+2$；否则，算法第(34)、第(35)行将回传路径 $R(x_i,x_j:q_j+1)+R(x_i:y_i,q_j+1)+R(y_i,q_j+1:y_j)$，回传路径的长度为 $\Delta_i(x,y)+\Delta_j(x,y)+2\Delta_j(y,q)+2$。

情况 2 结点 x 和结点 y 同行或同列。不失一般性，假设结点 x 和结点 y 同行(即 $x_i=y_i$)。首先，考虑 $R(x_i,x_j:y_j)$ 存在故障结点的情况。当 $p_i=0$ 时，即 p 属于 $Border(M_{m,n})$，算法第(12)、第(13)行将回传路径 $R(x_i:q_i+1,x_j)+R(q_i+1,x_j:y_j)+R(q_i+1:x_i,y_j)$，回传路径的长度为 $\Delta_j(x,y)+2\Delta_i(x,q)+2$。当 $p_i\neq 0$ 且 $q_i=m-1$ 时，算法的第(14)、第(15)行将回传路径 $R(x_i:p_i-1,x_j)+R(p_i-1,x_j:y_j)+R(p_i-1:x_i,y_j)$，回传路径的长度为 $\Delta_j(x,y)+2\Delta_i(x,p)+2$。当 $p_i\neq 0$ 且 $q_i\neq m-1$ 时，若 $\Delta_i(x,p)\leqslant\Delta_i(x,q)$，则算法的第(18)、第(19)行将回传路径 $R(x_i:p_i-1,x_j)+R(p_i-1,x_j:y_j)+R(p_i-1:x_i,y_j)$，回传路径的长度为 $\Delta_j(x,y)+2\Delta_i(x,p)+2$；否则，算法第(20)、第(21)行将会回传路径 $R(x_i:q_i+1,x_j)+R(q_i+1,x_j:y_j)+R(q_i+1:x_i,y_j)$，回传路径的长度为 $\Delta_j(x,y)+2\Delta_i(x,q)+2$。

类似地，$R(x_i:y_i,x_j)$ 存在故障结点的情况也能同理得出。

综上所述，算法 RWRF 会建立结点 x 与结点 y 于 $M_{m,n}-F(p,q)$ 的最短路径。

很容易看出，RWRF 算法的第(2)行检查结点 x 与结点 y 是否为无故障结点以及 $M_{m,n}-F(p,q)$ 是否为连通图，需要的时间为 $O(1)$。$R(x_i,x_j:y_j)$ 与 $R(x_i:y_i,y_j)$ 不包含故障结点或 $R(x_i:y_i,x_j)$ 与 $R(y_i,x_j:y_j)$ 不包含故障结点时，求出最短路径的时间为 $O(m+n)$。情况 1 中，结点 x 和结点 y 不同行不同列，不同的故障情形下求出最短路径的所需的时间为 $O(m+n)$。情况 2 中，结点 x 和结点 y 同行或同列，求出最短路径的所需时间也为 $O(m+n)$。故 RWRF 算法的时间复杂度为 $O(m+n)$。

引理 1 当 $x_i\leqslant y_i$ 且 $x_j\leqslant y_j$，算法 RWRF 所得到的路径 P 的长度如下：

P 的长度=

(1) $\Delta_i(x,y)+\Delta_j(x,y)$，若 $R(x_i,x_j:y_j)$ 与 $R(x_i:y_i,y_j)$ 或 $R(x_i:y_i,x_j)$ 与 $R(y_i,x_j:y_j)$ 不存在故障结点

(2) $\Delta_i(x,y)+\Delta_j(x,y)+2\Delta_i(y,q)+2$，若 $R(x_i,x_j:y_j)$ 与 $R(y_i,x_j:y_j)$ 同时存在故障结点且有 $p\in Border(M_{m,n})$ 或 $\Delta_i(x,p)>\Delta_i(y,q)$

(3) $\Delta_i(x,y)+\Delta_j(x,y)+2\Delta_i(x,p)+2$，若 $R(x_i,x_j:y_j)$ 与 $R(y_i,x_j:y_j)$ 同时存在故障结点且有 $q\in Border(M_{m,n})$ 或 $\Delta_i(x,p)\leqslant\Delta_i(y,q)$

(4) $\Delta_i(x,y)+\Delta_j(x,y)+2\Delta_j(y,q)+2$，若 $R(x_i:y_i,x_j)$ 与 $R(x_i:y_i,y_j)$ 同时存在故障结点且有 $p\in Border(M_{m,n})$ 或 $\Delta_j(x,p)>\Delta_j(y,q)$

(5) $\Delta_i(x,y)+\Delta_j(x,y)+2\Delta_j(x,p)+2$，若 $R(x_i:y_i,x_j)$ 与 $R(x_i:y_i,y_j)$ 同时存在故障结点且有 $q\in Border(M_{m,n})$ 或 $\Delta_j(x,p)\leqslant\Delta_j(y,q)$

3.2 某结点及其 k 跳之内的邻居出现故障的最短路由算法

用结点 o 表示某故障结点，且把 o 及其 k 跳之内的邻居结点出现故障所形成的故障区域记为 $H(o,k)$，则满足 $H(o,k)=\{u\mid d(u,o)\leqslant k\}$。

令 $a=(a_i,a_j)$ 和 $b=(b_i,b_j)$ 为无故障结点，并且满足 $a=(o_i-k,o_j-k)$ 且 $b=(o_i+k,o_j+k)$。则令 $B(I,J)$ 表示 $F(a,$

$b)-H(o,k)$中划分的 4 个区块，如图 1 所示。其中，I 表示当前区块在 4 个区块中所在的行，J 表示当前区块在 4 个区块中所在的列。这 4 个区块的集合如下：

$B(0,0)=\{u \mid a_i \leqslant u_i < (a_i+b_i)/2 \& a_j \leqslant u_j < (a_j+b_j)/2 \& u \in F(a,b)-H(o,k)\}$；

$B(0,1)=\{u \mid a_i \leqslant u_i < (a_i+b_i)/2 \& (a_j+b_j)/2 < u_j \leqslant b_j \& u \in F(a,b)-H(o,k)\}$；

$B(1,0)=\{u \mid (a_i+b_i)/2 < u_i \leqslant b_i \& a_j \leqslant u_j < (a_j+b_j)/2 \& u \in F(a,b)-H(o,k)\}$；

$B(1,1)=\{u \mid (a_i+b_i)/2 < u_i \leqslant b_i \& (a_j+b_j)/2 < u_j \leqslant b_j \& u \in F(a,b)-H(o,k)\}$。

算法 RWKHF **即为** $M_{m,n}$ **中存在** k **跳故障区域** $H(o,k)$ **时的最短路由算法**。

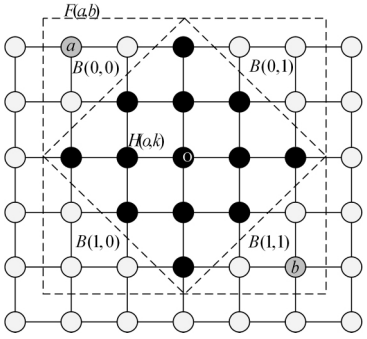

图 1　$F(a,b)-H(o,k)$ **中** $B(I,J)$ **分布图**

算法 2　Routing_with_K_hop_Fault, RWKHF(x, y, $H(o,k)$, $M_{m,n}$)

输入：结点 x **与结点** y，k **跳故障区域** H(o,k)，$M_{m,n}$

输出：结点 x **与结点** y **之间于** $M_{m,n}-H(o,k)$ **的最短路径**

```
1. Begin
2.   Border(M_{m,n})={(i,j)|i∈{0,m-1} & 0≤j≤n-1}∪{(i,j)|
     0≤i≤m-1 & j∈{0,n-1}};
3.   set B=B(0,0)∪B(0,1)∪B(1,0)∪B(1,1);
4.   If |{x,y}∩H(o,k)|≠0 || |Border(M_{m,n})∩H(o,k)|≠0
5.     Return failure;
6. Else
7.   {
8.     If x∉B && y∉B
9.       Return RWRF(x,y,F(a,b),M_{m,n});
10.    Else If |{x,y}∩F(a,b)|=1
11.    {
12.      If x∈B && y∉B
13.          Return OIOO(x,y,H(o,k),M_{m,n});
14.      Else
15.      {
16.          P=OIOO(y,x,H(o,k),M_{m,n});
17.          Return P^{-1};
             //令 Path P=⟨x_1,x_2,…,x_k⟩,
             则 Path P^{-1}=⟨x_k,x_{k-1},…,x_1⟩。
18.      }
19.    }
20.    Else If x∈B && y∈B
21.    {
22.      set I and J being indices such that x∈B(I,J);
23.      set I' and J' being indices such that y∈B(I',J');
24.      If I=I' && J=J'
25.      {
26.        If |V(R(x_i,x_j:y_j))∩H(o,k)|=0 && |V(R(x_i:y_i,
           y_j))∩H(o,k)|=0
27.          Return R(x_i,x_j:y_j)+R(x_i:y_i,y_j);
28.        Else
29.          Return R(x_i:y_i,x_j)+R(y_i,x_j:y_j);
30.      }
31.      Else If I≠I' || J≠J'
32.      {
33.        P_1=OIOO(x,(a_i-1,y_j),F(a,b),M_{m,n})+R(a_i-1:y_i,
           y_j);
34.        P_2=OIOO(x,(b_i+1,y_j),F(a,b),M_{m,n})+R(b_i+1:y_i,
           y_j);
35.        P_3=OIOO(x,(y_i,a_j-1),F(a,b),M_{m,n})+R(y_i,a_j-1:
           y_j);
36.        P_4=OIOO(x,(y_i,b_j+1),F(a,b),M_{m,n})+R(y_i,b_j+
           1:y_j);
37.        If I'=0 && J'=0
38.          Return SP(P_1,P_3);
39.        Else If I'=0 && J'=1
40.          Return SP(P_1,P_4);
41.        Else If I'=1 && J'=0
42.          Return SP(P_2,P_3);
43.        Else If I'=1 && J'=1
44.          Return SP(P_2,P_4);
45.      }
46.    }
47.  }
48. End
```

算法 3　One_In_One_Out, OIOO(p, q, $H(o,k)$, $M_{m,n}$)

输入：结点 p **与结点** q(**且** p **在** F(a,b)**中**，q **在** $M_{m,n}-F(a,b)$**中**)，k **跳故障区域** H(o,k)，$M_{m,n}$

输出：结点 p **与结点** q **之间于** $M_{m,n}-H(o,k)$ **的最短路径**

```
1. Begin
2.   If p∈B(0,0)
3.   {
4.     P_1=R(p_i:a_i-1,p_j)+RWRF((a_i-1,p_j),q,F(a,b),M_{m,n});
5.     P_2=R(p_i,p_j:a_j-1)+RWRF((p_i,a_j-1),q,F(a,b),M_{m,n});
6.   }
7.   Else If p∈B(0,1)
8.   {
9.   P_1=R(p_i:a_i-1,p_j)+RWRF((a_i-1,p_j),q,F(a,b),M_{m,n});
10.  P_2=R(p_i,p_j:b_j+1)+RWRF((p_i,b_j+1),q,F(a,b),M_{m,n});
11.  }
12.  Else If p∈B(1,0)
13.  {
14.    P_1=R(p_i:b_i+1,p_j)+RWRF((b_i+1,p_j),q,F(a,b),M_{m,n});
15.    P_2=R(p_i,p_j:a_j-1)+RWRF((p_i,a_j-1),q,F(a,b),M_{m,n});
16.  }
17.  Else If p∈B(1,1)
18.  {
19.   P_1=R(p_i:b_i+1,p_j)+RWRF((b_i+1,p_j),q,F(a,b),M_{m,n});
20.   P_2=R(p_i,p_j:b_j+1)+RWRF((p_i,b_j+1),q,F(a,b),M_{m,n});
21.  }
22.  Return SP(P_1,P_2);
23. End
```

算法 4 Short_Path,SP(P_1,P_2)

输入:两条路径 P_1,P_2

输出:P_1,P_2中长度最短的一条路径

1. Begin
2. If $|V(P_1)|\geqslant|V(P_2)|$ Return P_2;
3. Else Return P_1;
4. End

接下来讨论算法 RWKHF 的正确性并计算该算法所得到的路径长度。

定理 2 令 $H(o,k)$为 $M_{m,n}$中结点 o 的 k 跳内故障区域且结点 x 与结点 y 为无故障结点。若 $M_{m,n}-H(o,k)$为连通图且 $Border(M_{m,n})\cap H(o,k)=\varnothing$,则算法 RWKHF 会建立结点 x 与结点 y 于 $M_{m,n}-H(o,k)$的最短路径。

证明:RWKHF 算法第(4)行首先检查结点 x 与结点 y 是否为无故障结点,$M_{m,n}-H(o,k)$是否为连通图以及 $Border(M_{m,n})\cap H(o,k)$是否为空集。若结点 x 或结点 y 为故障结点,或 $M_{m,n}-H(o,k)$为非连通图,或 $Border(M_{m,n})\cap H(o,k)\neq\varnothing$,则算法将回传失败,即无法找到连接结点 x 与结点 y 的无故障路径。

接下来将讨论以下 3 类情况。

情况 1 结点 x 和 y 都不在 $F(a,b)$中。不失一般性,假设 $x_i\leqslant y_i$ 且 $x_j\leqslant y_j$,则算法的第(8)、第(9)行将回传路径 $RWRF(x,y,F(a,b),M_{m,n})$,回传路径的长度分为以下几种情况:

(1)当 $R(x_i,x_j:y_j)$与 $R(x_i:y_i,y_j)$不存在故障结点,或者 $R(x_i:y_i,x_j)$与 $R(y_i,x_j:y_j)$不存在故障结点时,$L=\Delta_i(x,y)+\Delta_j(x,y)$;

(2)当 $R(x_i,x_j:y_j)$与 $R(y_i,x_j:y_j)$同时存在故障结点时,若 $\Delta_i(x,a)\leqslant\Delta_i(y,b)$,则 $L=\Delta_i(x,y)+\Delta_j(x,y)+2\Delta_i(x,a)+2$;否则,$L=\Delta_i(x,y)+\Delta_j(x,y)+2\Delta_i(y,b)+2$;

(3)当 $R(x_i:y_i,y_j)$与 $R(x_i:y_i,x_j)$同时存在故障结点时,若 $\Delta_j(x,a)\leqslant\Delta_j(y,b)$,则 $L=\Delta_i(x,y)+\Delta_j(x,y)+2\Delta_j(x,a)+2$;否则,$L=\Delta_i(x,y)+\Delta_j(x,y)+2\Delta_j(y,b)+2$。

情况 2 结点 x 和结点 y 一个在 $F(a,b)$中而另一个在 $M_{m,n}-F(a,b)$中。不失一般性,假设 x 在 $F(a,b)$中,y 在 $M_{m,n}-F(a,b)$中,则算法的第(12)、第(13)行将回传路径 $P=OIOO(x,y,H(o,k),M_{m,n})$。因此调用 OOIO 算法,不失一般性,假设 $I=0$ 且 $J=0$,定义两条路径 P_1 和 P_2 满足条件 $P_1=R(x_i:a_i-1,x_j)+RWRF((a_i-1,x_j),y,F(a,b),M_{m,n})$ 和 $P_2=R(x_i,x_j:a_j-1)+RWRF((x_i,a_j-1),y,F(a,b),M_{m,n})$,随后算法回传路径 $SP(P_1,P_2)$,回传路径的长度分为以下几种情况:

(1)当 $y_i<a_i$ 或者 $y_j<a_j$ 时,$L=\Delta_i(x,y)+\Delta_j(x,y)$;

(2)当 $a_i<y_i<b_i$ 且 $y_j>b_j$ 时,$L=\Delta_j(x,y)+\Delta_i(x,a)+\Delta_i(y,a)+2$;

(3)当 $y_i>b_i$ 并且 $a_j<y_j<b_j$ 时,$L=\Delta_i(x,y)+\Delta_j(x,a)+\Delta_j(y,a)+2$;

(4)当 $y_i>b_i$ 并且 $y_j>b_j$ 时,算法所回传的路径长度为 $L=\min\{(\Delta_j(x,y)+\Delta_i(x,a)+\Delta_i(y,a)+2),(\Delta_i(x,y)+\Delta_j(x,a)+\Delta_j(y,a)+2)\}$。

情况 3 若结点 x 和 y 都在 $F(a,b)$中。当结点 x 和结点 y 在同一个无故障区块 $B(I,J)$中,若 $R(x_i,x_j:y_j)$与 $R(x_i:y_i,y_j)$的结点均为无故障结点,则算法的第(26)、第(27)行将回传路径 $R(x_i,x_j:y_j)+R(x_i:y_i,y_j)$,回传路径的长度为 $\Delta_i(x,y)+\Delta_j(x,y)$;否则算法的第(28)、第(29)行将回传路径 $R(x_i:y_i,x_j)+R(y_i,x_j:y_j)$,回传路径的长度为 $\Delta_i(x,y)+\Delta_j(x,y)$。当结点 x 和结点 y 不在同一个无故障区块中时,定义 4 条路径 P_1,P_2,P_3,P_4,分别满足:

$P_1=OIOO(x,(a_i-1,y_j),F(a,b),M_{m,n})+R(a_i-1:y_i,y_j)$

$P_2=OIOO(x,(b_i+1,y_j),F(a,b),M_{m,n})+R(b_i+1:y_i,y_j)$

$P_3=OIOO(x,(y_i,a_j-1),F(a,b),M_{m,n})+R(y_i,a_j-1:y_j)$

$P_4=OIOO(x,(y_i,b_j+1),F(a,b),M_{m,n})+R(y_i,b_j+1:y_j)$

P_1 表示根据 OIOO 算法从结点 x 路由到结点(a_i-1,y_j),再从结点(a_i-1,y_j)通过路由到达结点 y。同理,P_2,P_3,P_4 也是类似的路由。不失一般性,假设结点 y 位于 $B(1,1)$中,则返回路径 $SP(P_2,P_4)$。回传路径的长度分为以下几种情况:

(1)当结点 x 位于 $B(0,1)$中时,$L=\Delta_i(x,y)+\Delta_j(x,b)+\Delta_j(y,b)+2$;

(2)当结点 x 位于 $B(1,0)$中时,$L=\Delta_j(x,y)+\Delta_i(x,b)+\Delta_i(y,b)+2$;

(3)当结点 x 位于 $B(0,0)$中时,则算法所回传的路径长度为 $L=\min\{(\Delta_i(x,y)+\Delta_j(x,y)+2\Delta_i(x,a)+2\Delta_j(y,b)+4),(\Delta_i(x,y)+\Delta_j(x,y)+2\Delta_j(x,a)+2\Delta_i(y,b)+4)\}$。

依据结点 x 与结点 y 于 $M_{m,n}-H(o,k)$所处的位置不同,由定理 2 的证明可以得到引理 2、引理 3 与引理 4 的结论。

引理 2 当 $x\notin F(a,b)$,$y\notin F(a,b)$,且满足 $x_i\leqslant y_i$,$x_j\leqslant y_j$,则算法 RWKHF 所得到的路径 P 的长度如下:

P 的长度=

$$\begin{cases}(1)\Delta_i(x,y)+\Delta_j(x,y),\text{若 } R(x_i,x_j:y_j)\text{与 } R(x_i:y_i,y_j)\\ \quad\text{或 } R(x_i:y_i,x_j)\text{与 } R(y_i,x_j:y_j)\text{不存在故障结点}\\ (2)\Delta_i(x,y)+\Delta_j(x,y)+2\Delta_i(y,b)+2,\text{若 } R(x_i,x_j:y_j)\\ \quad\text{与 } R(y_i,x_j:y_j)\text{同时存在故障结点且 }\Delta_i(x,a)>\Delta_i(y,b)\\ (3)\Delta_i(x,y)+\Delta_j(x,y)+2\Delta_i(x,a)+2,\text{若 } R(x_i,x_j:y_j)\\ \quad\text{与 } R(y_i,x_j:y_j)\text{同时存在故障结点且 }\Delta_i(x,a)\leqslant\Delta_i(y,b)\\ (4)\Delta_i(x,y)+\Delta_j(x,y)+2\Delta_j(y,b)+2,\text{若 } R(x_i:y_i,x_j)\\ \quad\text{与 } R(x_i:y_i,y_j)\text{同时存在故障结点且 }\Delta_j(x,a)>\Delta_j(y,b)\\ (5)\Delta_i(x,y)+\Delta_j(x,y)+2\Delta_j(x,a)+2,\text{若 } R(x_i:y_i,x_j)\\ \quad\text{与 } R(x_i:y_i,y_j)\text{同时存在故障结点且 }\Delta_j(x,a)\leqslant\Delta_j(y,b)\end{cases}$$

引理 3 当 $x\in F(a,b)$的 $B(0,0)$且 $y\notin F(a,b)$,算法 RWKHF 所得到的路径 P 的长度如下:

P 的长度=

$$\begin{cases}(1)\Delta_i(x,y)+\Delta_j(x,y),\text{若 } y_i<a_i\text{ 或 } y_j<a_j\\ (2)\Delta_j(x,y)+\Delta_i(x,a)+\Delta_i(y,a)+2,\text{若 } a_i<y_i<b_i\\ \quad\text{且 } y_j>b_j\\ (3)\Delta_i(x,y)+\Delta_j(x,a)+\Delta_j(y,a)+2,\text{若 } y_i>b_i\\ \quad\text{且 } a_j<y_j<b_j\\ (4)\min\{(\Delta_j(x,y)+\Delta_i(x,a)+\Delta_i(y,a)+2),(\Delta_i(x,y)+\\ \quad\Delta_j(x,a)+\Delta_j(y,a)+2)\},\text{若 } y_i>b_i\text{ 且 } y_j>b_j\end{cases}$$

引理 4　当 $x\in F(a,b)$ 且 $y\in F(a,b)$ 的 $B(1,1)$，算法 RWKHF 所得到的路径 P 的长度如下：

P 的长度＝

$$\begin{cases}(1)\Delta_i(x,y)+\Delta_j(x,y), & \text{若 } x\in B(1,1)\\ (2)\Delta_i(x,y)+\Delta_j(x,b)+\Delta_j(y,b)+2, & \text{若 } x\in B(0,1)\\ (3)\Delta_j(x,y)+\Delta_i(x,b)+\Delta_i(y,b)+2, & \text{若 } x\in B(1,0)\\ (4)\min\{(\Delta_i(x,y)+\Delta_j(x,y)+2\Delta_i(x,a)+2\Delta_j(y,b)+4),\\ \quad(\Delta_i(x,y)+\Delta_j(x,y)+2\Delta_j(x,a)+2\Delta_i(y,b)+4)\}, & \text{若 } x\in B(0,0)\end{cases}$$

首先分析算法 3 的性能：无论哪种情形，计算 P_1 所需的时间为 $O(m+n)$，同理计算 P_2 所需的时间也为 $O(m+n)$，故算法 3 的时间复杂度为 $O(m+n)$。

接下来分析算法 2 的性能：RWKHF 算法第(4)行检查结点 x 与结点 y 是否为无故障结点，$M_{m,n}-H(o,k)$ 是否为连通图以及 $Border(M_{m,n})\cap H(o,k)$ 是否为空集，需要的时间为 $O(1)$。情况 1 中，结点 x 和 y 都不在 $F(a,b)$ 中，调用 RWRF 算法，求出最短路径所需的时间为 $O(m+n)$。情况 2 中，结点 x 和结点 y 一个在 $F(a,b)$ 中而另一个在 $M_{m,n}-F(a,b)$ 中，调用 OIOO 算法，求出最短路径所需的时间为 $O(m+n)$。情况 3 中，结点 x 和 y 都在 $F(a,b)$ 中，当结点 x 和结点 y 在同一个无故障区块 $B(I,J)$ 中，显然求出最短路径所需的时间为 $O(m+n)$，否则需要求出 P_1,P_2,P_3,P_4。调用 OIOO 算法计算每条路径所需的时间也为 $O(m+n)$，故 RWKHF 算法的时间复杂度为 $O(m+n)$。

4　算法仿真

本文采用 OPNET 仿真平台对 RWRF 算法和王高才等人[20]提出的 Mesh 网络容错单播路由算法(k-Mesh 路由算法)进行仿真。仿真是在 2DMesh 网络上进行的，网络规模为 16×16。当在 16×16 的 Mesh 网络中有一个 9×9 的子 Mesh 发生故障时，随机选择了 10 组无故障结点，利用两种算法分别求出了结点间的最长路径的长度，实验结果如图 1 所示。

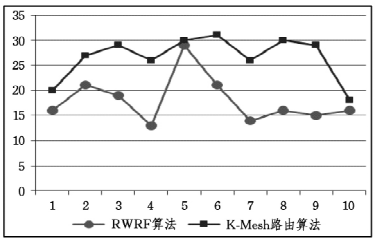

图 1　RWRF 算法和 k-Mesh 路由算法的实验结果对比

结束语　根据故障区域的不同，本文提出了两种情形下的 Mesh 网路的最短路由算法：1)当 Mesh 的行数大于等于 3 且列数大于等于 3、出现一个矩形故障区域的情况时，给出了任意两个无故障结点间的最短路由算法，并且计算出了路径长度；2)当 Mesh 的行数大于等于 3 且列数大于等于 3、某个结点及其 k 跳以内的邻居结点出现故障时，给出了任意两个无故障结点间的最短路由算法，并且计算出了路径长度。

下一步可以讨论其他网络结构，例如 Torus 网络、Cube 网络、Hypercube 网络、Twisted 立方网等网络在有故障区域下的最短路径，或考虑其他不同的故障区域类型。

参 考 文 献

[1] HARARY F, HAYES J P, WU H J. A survey of the theory of hypercube graphs[J]. Computers and Mathematics with Applications, 1988, 15(4): 277-289.

[2] 杨靖，徐迈，赵伟，等. 传感器网络中一种能量高效的数据收集算法[J]. 系统工程与电子技术，2011，33(3)：650-655.

[3] 朱永利，于永华，李丽芬. 数据收集传感器网络的多模层次网络构建[J]. 计算机工程，2011，37(2)：111-113.

[4] 潘文虎，张瑞华. WSN 中基于移动 sink 的高效数据收集算法[J]. 计算机工程，2011，37(18)：94-96.

[5] WANG D. Embedding Hamiltonian cycles into folded hypercubes with faulty links[J]. Journal of Parallel and Distributed Computing, 2001, 61(4): 545-564.

[6] EFE K. The crossed cube architecture for parallel computation [J]. IEEE Transactions on Parallel and Distributed Systems, 1992, 3(5): 513-524.

[7] ABRAHAM S, PADMANABHAN K. The twisted cube topology for multiprocessors: A study in network asymmetry[J]. Journal of Parallel and Distributed Computing, 1991, 13(1): 104-110.

[8] LOURI A, SUNG H. An optical multi-mesh hypercube: a scalable optical interconnection network for massively parallel computing[J]. Journal of Lightwave Technology, 1994, 12(4): 704-716.

[9] LILLEVIK S L. The Touchstone 30 Gigaflop DELTA Prototype [C]// Proceedings of the Sixth Distributed Memory Computing Conference. 1991: 671-677.

[10] LENOSKI D, LAUDON J, GHARACHORLOO K, et al. The Stanford DASH Multiprocessor[J]. IEEE Computer, 1992, 25(3): 63-79.

[11] AGARWAL A, BIANCHINI R, CHAIKEN D, et al. The MIT alewife machine: architecture and performance[J]. International Symposium on Computer Architecture, 1998, 23(12): 2-13.

[12] Intel. A touchstone DELTA system description[R]. Intel Advanced Information, Intel Corporation. Portland, Oregon, Technical Report, 1991.

[13] SEITZ C L, ATHAS W C. The architecture and programming of the Ametek series 2010 multicomputer[C]// Conference on Hypercube Concurrent Computers & Applications: Architecture. ACM, 1988: 33-37.

[14] 孙凝晖，徐志伟. 曙光 2000 超级计算机系统软件的设计[J]. 计算机学报，2000，23(1)：9-20.

[15] YU Z G, WANG X Y, SHEN K L, et al. A general methodology to design deadlock-free routing algorithms for mesh networks [M]// Algorithms and Architectures for Parallel Processing. Springer International Publishing, 2015: 478-491.

[16] 胥大成，樊建席，张书奎. 基于 2D-Mesh 的容错路由算法[J]. 计算机科学，2012，39(3)：113-117.

[17] 袁利永，朱艺华，邱树伟. 无线传感 mesh 网络的分段地址分配策略及其路由[J]. 计算机科学，2016，43(6)：116-121，155.

[18] 王高才，陈建二，陈松乔，等. Mesh 网络路由算法容错性的概率分析[J]. 计算机学报，2004，27(3)：319-327.

[19] 刘家俊，顾华玺，王长山. mesh 优先级容错路由[J]. 计算机工程与应用，2009，45(4)：105-107.

[20] 王高才，王国军，陈建二，等. Mesh 网络容错单播路由算法[J]. 中南工业大学学报，2003，34(6)：657-660.