网格中基于最小连接块的启发式容错路由算法

陈贵海[1], 杜 鹏[1], 王大进[1,2], 谢 立[1]
(1. 南京大学计算机软件新技术国家重点实验室, 江苏南京 210093; 2. 美国蒙特克莱州立大学计算机科学系, 美国)

摘 要: 矩形无效块模型可以用来解决网格下的容错路由问题, 最小连接块(MCC)模型是它的一个改良模型. 本文在MCC基础上, 建立MCC重叠图, 当发现不存在曼哈顿路径的时候, 给出一套算法, 来计算出一条避免无效块的尽可能短的路径. 模拟试验表明, 通过这种算法找到的路径, 与最短路径相差很小. 比起花费更多的时间去找寻最短路径, 这种启发式容错算法无疑是更好的选择.

关键词: 自适应路由; 矩形无效块模型; 容错性; 网格
中图分类号: TP393 文献标识码: A 文章编号: 0372-2112 (2004) 02-0318-05

Heuristic Fault-Tolerant Routing in Mesh Using Minimal-Connected-Component Fault Blocks

CHEN Gui-hai[1], DU Peng[1], WANG Da-jin[1,2], XIE Li[1]
(1. State Key Laboratory of Novel Software Technology, Nanjing University, Nanjing, Jiangsu 210093, China;
2. Department of Computer Science, Montclair State University, upper Montclair NJ 07043, USA)

Abstract: Rectangular fault block model is designated to solve the problem of fault-tolerant route in mesh and was improved as Minimal-Connected-Component (MCC) model. Based on MCC, we construct an overlapping graph and give a set of algorithm according to the graph to work out the route as short as possible to avoid the appearance of fault block when Manhattan route does not exist. The simulated test shows that the route found by the algorithm mentioned above is nearly the shortest one. Hence compared to other methods costing much more time, this new heuristic fault-tolerant algorithm is of no doubt a better method in finding the shortest route.

Key words: adaptive routing; fault block model; fault tolerance; mesh

1 引言

在使用网格结构的并行计算机中[1~3], 结点间的路由对于获得计算机的高性能起着非常重要的作用, 有许多文献研究网格上的容错处理, 其中一个重要问题就是避开所有的故障结点的自适应路由能力. 以往大多数工作都使用了/矩形无效块模型0[4~6]. 在这种模型中, 所有的故障结点都被划分在不连续的矩形无效块内. 无效块的构造原则上是在保持矩形形状的前提下, 尽可能少的包含非故障结点. 这些无效块内的所有结点, 无论是故障的或是非故障的, 都将被路由跳过. 然而, 针对自适应算法所要追求的最短路径的目的来说, 这种模型并不是最佳选择, 这一点我们将在第二章进行阐述. 在构造矩形无效块时, 一些非故障结点也被不必要的包括在内, 这就减少了路由可选择的结点个数, 因而降低了找到最短路径的可能. 文章[7]提出了一个最小连接块(MCC)模型, 这种模型是对广泛应用的矩形无效块模型的一个改进, 可以使找到路径的机会大大提高.

本文提出一种启发式算法, 这种算法基于MCC模型, 建立MCC重叠图, 在此基础上, 当发现不存在曼哈顿通路的时候, 计算出一条避免无效块的尽可能短的路径. 在所有可能路径中选择一个避免所有故障结点的最短路径是非常耗时的, 所以在路径长度与找到它的时间之间做个折衷是很有意义的. 在多处理器计算机系统中, 特定源/目之间路径或许只会使用很少的几次, 这样, 穷尽寻找最短路径就显得尤其不经济了. 另外, 多处理器计算机系统中, 从一个结点到另一个结点的路由始终都在发生. 在很多种情况下, 更需要的是快速的给出路由以传输消息, 而不是花费过多的时间用于计算出一个绝对最短的路径.

2 最小连接块模型

2.1 基本介绍

建立坐标系来研究二维的网格. 用(x, y)来对应网格上的点. 同时, 把y轴正方向定义为北. 为了方便讨论并不失一般性, 我们定义坐标系原点就是发送消息的原点, 即$(x_s, y_s) = (0, 0)$, 目标结点为第一象限中的点, 即$x_d > 0, y_d > 0$. 在其他象限的话, 我们只需要通过旋转坐标就可以得到相同的结

收稿日期: 2002-02-14; 修回日期: 2003-10-30
基金项目: 国家自然科学基金项目(No. 60073029); 国家973项目(No. 2002CB312002); 教育部高校青年教师奖资助计划

果.

在二维的网格中,一个非边结点(x, y) 有 4 个邻居. (x+ 1, y) , (x- 1, y) , (x, y+ 1) 和(x, y- 1) 分别被称作东邻结点, 西邻结点, 北邻结点和南邻结点.在路由的每一步中,都是从一个结点选择一个方向到它的相邻结点中去.因此, 路径可以表述为一些方向的序列.下面给出曼哈顿路径和最短路径的定义:

定义 1　如果一个路由,它仅仅使用两个方向并且这两个方向为非相反方向进行路由,那么这条路由路径就被称作曼哈顿(Manhattan) 路径.

定义 2　如果一条路径的,它的距离是原点到目标点中所有路径中最短的,那么称这条路径为最短路径.

曼哈顿通路必然是最短通路,但是反之不然.显然,曼哈顿路在没有坏结点的网格中总是存在的,但是有故障结点的网格中,并非任意两点一定存在曼哈顿通路的.文章[8] 提出了一个通过标志网格中不连通矩形无效块(disconnected rectan2 gular block fault model) [9] 的方式来判断它是否存在曼哈顿通路的一个充分条件.一种模型在简化路由算法的同时,却往往不能尽可能地寻找最短路径[1, 4~ 6, 9] .例如,在图 1 中, A, B, C 分别是三个矩形无效块. A 包含了非故障点,但是被认为是无效结点 a, b, c, 等.这整个块被当作一个整体,认为是不可穿越的,所以无论原点和目标结点的位置,块中的非故障点也被认为是没有作用的.然而,如果原点 s 和目标结点 d,利用块中的无效结点,我们可以找到一条曼哈顿通路(s y a y b y c y e y f y d) .为了尽可能的减少无效节点,文[7] 提出了最小连接块 (Minimal Con2 nected Component) 模型,简写为 MCC 模型.

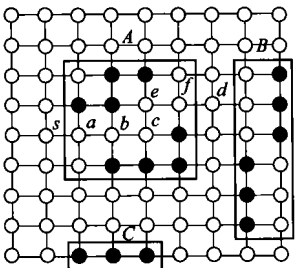

图 1　矩形无效块模型

2.2　MCC 模型

在第一象限内,如果存在曼哈顿通路,那么将只朝东和北方向进行消息传递.形成 MCC 块是一个递归的过程,如表 1.

表 1　算法 1

1) 初始化, 标志所有故障结点为/ faulty0;
2) 如果一个结点(x, y) 是非故障的,但是它的北邻接点和东邻接点是故障的或者是到达无效结点,那么标志该点为/ 到达无效结点0
3) 如果一个结点(x, y) 是无故障的,但是它的南邻接点和西邻接点是故障的或者是不能到达的结点,那么标志该点为/ 不能到达结点0
4) 过程 223 反复运行至不再有新的到达无效结点和不能到达到结点产生

一个结点被称为到达无效结点,是指当路由路径一旦到达了这个结点,它的下一个移动方向必然是西或者是南,使得不可能是曼哈顿通路.一个结点被标记为不能到达结点,是指如果一条路径经过它,那么必然是上一个结点通过西或南访问得到,这也表明这个路由路径不是曼哈顿通路.到达无效结点和不能到达结点一起被称作非故障无效结点.无效结点包括非故障无效结点和故障结点两种.

我们把通过算法 1 标志好的块称为 MCC 块.图 2(a) 显示了到达无效结点和不能到达结点,图 2(b) 中显示的是一个 MCC 块的外形,它的形状看起来有点象台阶.不难发现,构造 MCC 块是和原点和目标点的位置相关的,如果目标点在第 Ò , Ó , Ô 象限,其相应的形状如图 2(c) (d) (e) 所示.目标点在第Ñ 和Ó 象限的 MCC 形状是相同的,在Ò 和Ô 象限中的形状也是相同的.所以,我们没有必要对每对原点和目标结点分别求出四种不同的 MCC 块集合,而只需要求出两套 MCC 块集合.为了方便以后的讨论,我们给出一个比较显而易见的属性.

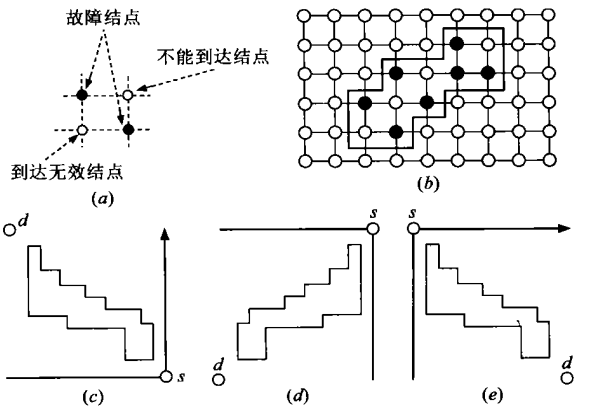

图 2　MCC 块外形

属性 1　任何两个 MCC 至少被分离一个结点距离,也就是说,总是能找到一条路径,穿过两个 MCC.

文章[7] 提出了判断 MCC 模型中是否存在曼哈顿通路的充分条件,同时,还给出了如果存在曼哈顿通路时找到这条路径的算法(用 MIN. ROUTING 表示这个算法) .下面给出存在曼哈顿通路的一个充要条件.

定理 1　在 MCC 模型中,从原点 s= (0, 0) 到目标结点 d = (x_d, y_d) 存在曼哈顿通路的充要条件是不存在以下两种块序列:

(1) 如果存在一个 MCC 块序列(M_1, M_2, , , M_n) ,满足:

M_1 中存在一个结点(0, y_1) ,满足 0< y_1< y_d;

M_n 中包含一个结点(x_d, y_n) ,满足 0< y_n< y_d;

对于每一个 M_i, M_{i+1}, 1[i[n- 1,

min{a: (a, b) I M_{i+1}}- 1[max{u: (u, v) I M_i}[max{x: (x, y) I M_{i+1}}- 1 and max{v: (u, v) I M_i}< max{y: (x, y) I M_{i+1}}.

那么称这个序列为Ñ 类型序列;

(2) 如果存在一个 MCC 块序列(M_1, M_2, , , M_n) ,满足:

M_1 中存在一个结点(x_1, 0) ,满足 0< x_1< x_d;

M_n 中包含一个结点(x_n, y_d) ,满足 0< x_n< x_d;

对于每一个 M_i, M_{i+1}, 1[i[n- 1,

min{b: (a, b) I M_{i+1}}- 1 [max{v: (u, v) I M_i}[max{y: (x, y) I M_{i+1}}- 1and max{u: (u, v) I M_i}< max{x: (x,

y) I M_{i+1}};

那么称这个序列为Ò 类型序列;

关于定理的证明, 可以在文[7]中找到. 图 3 给出的是这两种序列的直观图, 如果存在Ñ 类型或者Ò 类型序列, 表明不存在曼哈顿通路, 那么将寻找其他尽可能短的路径, 来进行路由. 在下一节中, 将提出一种启发式搜索算法, 来处理不存在曼哈顿通路情况下的路由问题.

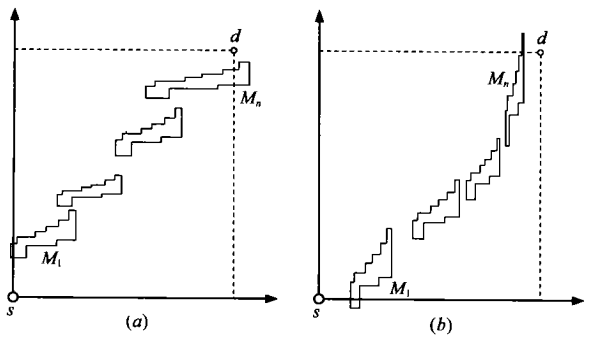

图 3 Ñ 类型和Ò 类型序列外形

Ñ 类型和Ò 类型序列还有以下一个性质, 对于我们下一节的算法有所帮助.

定理 2 对于给定的原点和目标结点, Ñ 类型序列和Ò 类型序列不可能同时存在.

为了证明 2, 我们先给出两个引理.

引理 1 对于给定的原点 s= (0, 0)和目标结点 d= (x_d, y_d)的存在Ñ 类型序列的图中, 不存在点 p= (x, y_d), x I [0, x_d], 使得存在 s 到p 的曼哈顿通路. (Ò 类型序列同)

证明: 反证法, 假设点 p= (x, y_d), x I [0, x_d], 使得存在 s 到 p 的曼哈顿通路R, 它将 s 到d 的矩形分成左右两个部分, 分别标志为 L 和 R. 任何一个 MCC 块必然在 L 或R 中, 不可能和 R 相交.

因为存在Ñ 类型序列, 自下向上对应着 $M_1, M_2, \ldots, M_n$. 则由定义可以知道, M_1 必然在 L 中, M_n 在 R 中. 这样, 必然就将存在两个 M_j, M_{j+1}, 使得 M_j I L, M_{j+1} I R. 由Ñ 序列的定义我们很容易看出, 这是显然不可能的.

定义 3 MCC 块 M_i 中, 一个结点如果它的北邻结点和东邻结点都不属于这个 M_i, 那么这个结点被称作这个 M_i 块的东北结点. 用 U_i 来标志它. 相似的, M_i 中, 一个结点如果它的南邻结点和西邻结点都不属于这个 M_i, 那么这个结点被称作 M_i 的西南结点, 用 L_i 来标志它.

下面用 x(N)来表示 N 的x 坐标, y(N)表示 N 的y 坐标.

定义 4 用 U_i, L_i 来定义M_i 的 Uc_i 和 Lc_i 点: x(Uc_i) = x(U_i)+ 1且 y(Uc_i) = y(U_i)+ 1; x(Lc_i) = x(L_i) - 1 且 y(Lc_i) = y(L_i) - 1

引理 2 对于给定的原点 s= (0, 0)和目标结点 d= (x_d, y_d)的存在Ñ 类型序列的图中, 可以找到一个Ñ 类型序列, 自下向上对应着 $M_1, M_2, \ldots, M_n$, 满足对于任意 M_i 和 M_{i+1}, U_i 到M_{i+1}的下沿没有其他 MCC 块阻隔. (Ò 类型序列同)

证明: 我们按如下方法可以找到满足题设的Ñ 类型序列:

(1)从 s 出发向上, 第一个遇到的MCC 块记为 M_1.

(2)对于任何一个 M_i, 沿 Uc_i 出发, 向上遇到的第一个MCC块记为 M_{i+1}

(3)按步骤 2, 一直找到第一个 M_n, 使得 x(U_n)> x_d

步骤 2 中, Uc_i 一定能向上遇到一个 MCC 块 M_{i+1}, 并且 min{y: (x, y) I M_{i+1}}< y_d. 否则, U_i 向上可以到达一个点(x, y_d), 而从 s 就存在着一条到 Uc_i 的曼哈顿通路(由步骤 1, 2 找 M_i 的过程显然可得), 而那么就是存在着 s 到(x, y_d)的曼哈顿通路, 由引理 1 可知, 这与题设存在Ñ 类型序列矛盾. 因此, 必然可以找到 M_n, M_n 是第一个满足 x(U_n)> x_d, 必然 M_n 中包含一个结点(x_d, y), 满足 0< y< y_d.

这个方法找到的 $M_1, M_2, \ldots, M_n$, 就是一个满足题设的Ñ 类型序列.

下面给出定理 2 的证明:

如果存在Ñ 类型序列, 我们可以按照引理 2 的方法找到一个 I 类型序列, 自下向上对应着 $M_1, M_2, \ldots, M_n$. 很容易看出, Uc_i 到 Uc_{i+1}都存在曼哈顿通路, 从而 S 到 Uc_{n-1}存在曼哈顿通路. Uc_{n-1}向上, 遇到 M_n 后, 沿 M_n 的下沿向右, 一直到达(x_d, y), 这样, 就找到一个从 s 到(x_d, y)的曼哈顿通路. 由引理 1 可知, 不存在Ò 类型序列.

3 利用 MCC 的启发式自适应路由算法

3l1 MCC 重叠图

路由算法先需要判断两点是否存在曼哈顿通路. 基于定理 2l1, 构建 MCC 重叠图, 在此基础上, 利用传递闭包, 可以方便的判断出来两点是否存在曼哈顿通路问题.

MCC 重叠图是根据 MCC 块来构建的.

给出一个 MCC 块集合, S= ($M_1, M_2, \ldots, M_n$). 在 S 中, 我们来根据Ñ 类型序列来构造一个图 $G_S^{Ñ}$ = (V, $A^{Ñ}$), V= {$v_1, v_2, \ldots, v_n$}. v_i 对应着 MCC 的 M_i, $G_S^{Ñ}$ 的Ñ 上标表示对应的是Ñ 类型序列, $A^{Ñ}$ 定义如下:

如果 M_i 的最东北结点 U_i, 是在 M_j 的东西边的正下方, 且没有被其他MCC 阻隔, 那么就存在一条从 v_i 到 v_j 的有向边. 形式化表示为:

$A^{Ñ}$ = {(v_i, v_j) | M_i, M_j 满足: x(U_i)< x(U_j) and y(U_i)< y(U_j) and v 点 p I M_j, 使得 x(p) = x(U_i)并且 p 点到 U_i 之间的结点都是有效结点}

$G_S^{Ñ}$ 的传递闭包, 我们记为 T($G_S^{Ñ}$), 利用它, 可以直接判断在 $G_S^{Ñ}$ 中两点建是否存在一条通路.

$$T(G_S^{Ñ})(i,j)\begin{cases}1, \text{如果存在一条从 } v_i \text{ 到 } v_j \text{ 的通路}\\ 0, \text{否则}\end{cases}$$

对称地, 我们对于Ò 类型序列同样构造 $G_S^{Ò}$ = (V, $A^{Ⅱ}$), $A^{Ⅱ}$ 定义如下:

如果 M_i 的最东北结点, U_i, 是在 M_j 的南北边的正左方, 且没有被其他 MCC 阻隔, 那么就存在一条从 v_i 到 v_j 的边. 形式化表述为:

$A^{Ñ}$ = {(v_i, v_j) | M_i, M_j 满足: x(U_i)< x(U_j) and y(U_i)<

y(U_j) and (点 p I M_j, 使得 y(p)= y(U_i) 并且 p 点到 U_i 之间的结点都是有效结点}

一旦 T($G_S^{\tilde{N}}$)和 T($G_S^{\grave{O}}$) 被计算出来, 根据定理 1, 是否存在曼哈顿通路的问题, 就可以用它们直接判断出来. 所以, 经过 O(n^3) 时间复杂度(传递闭包的算法时间复杂度为O(n^3)) 的预处理过程后, 以后就只需要一个 O(n) 时间复杂度来解决. n 是 MCC 块的数目, 对比起网格中结点的数目大小, n 要小的多.

先用 Manhattan 算法判断是否存在曼哈顿通路, 如果存在, 我们将直接调用在文[7] 中介绍的 MIN. ROUTE 算法来实现曼哈顿通路. 如果不存在, 将调用下节的算法, 找寻另一条非曼哈顿路径.

312　非曼哈顿通路的路由算法

首先定义一种/ 回头路0 (Backward) 方法, 如图 4(a) 所示, 路径向上走, 直到 M_j, 然后它沿着 M_j 的下沿(优先向左, 其次向下) 走到 Lc_j. 我们给出一个在$\tilde{N}$ 类型序列中的启发式路由算法如表 2:

表 2　算法2

(1) MCCFlag= {1, 2, 3, , n- 1, n}　//用来判断 M_i 是否被访问过, n 是所有 MCC 的数目
(2)From node (x_s, y_s), go up until an MCC M_i is reached;
(3)for all j I MCCFlag such that T($g_s^{\tilde{N}}$)(i, j)= 1 do
{ //让 j 从小到大进行搜索, 这样可以使得第一个遇到的存在曼哈顿通路的 Lc_j 所在的MCC块是最近的
¹ Call Manhattan(Lc_j, (x_d, y_d))
º If Manhattan(Lc_j, (x_d, y_d))= YES
{J z j; break for2 loop; goto 4 }
» nodeFlag= nodeFlag2{j}　}
goto 5
(4)//进入这一步, 表明存在 j, 使得从 Lc_j 开始, 存在一条曼哈顿通路到目标结点
¹ Go along the lower boundary of MCC sequence, until M_j is reached
º Take the backward route (toward southwest) until Lc_j is reached
» From 结点 Lc_j, 调用算法 MIN. ROUTING 找寻一条最短路径到目标结点(x_d, y_d)
¼ 找到路径, goto 6
(5)//进入这一步, 表明不存在任何的 M_j, 使得从 Lc_j 到目标结点存在曼哈顿通路
¹ Go along the backward route (toward southwest) of M_i, until Lc_i is reached with the Backward Method.
º (x_s, y_s) z Lc_i;
» if (x_s, y_s) is out of the Mesh boundary then { can. t find the route path, goto 6} else goto 2
(6) end.

Step2 直接向上推进, 遇到一个 MCC 块 M_i. Step4 的目的是利用 MCC 重叠图, 来判断是否存在只需要/ 回头路0 一次到达结点 Lc_j, 并且 Lc_j 到目标结点存在曼哈顿通路. 如果存在, 就转入 Step 4, 直接调用 MIN. ROUTE 方法, 找寻曼哈顿通路到达目标结点; 否则, 如果 Step3 没有找到所希望的 Lc_j, 则用 Step5, 其主要思想见图 4(b). 集合 MCCFlag 是避免重复判断. 当一个 Lc_j 被判断过不存在曼哈顿通路到达目标结点, 那么以后将没有必要再次来考虑它. 由定理 2 可知, 如果存在$\tilde{N}$ 序列, 就不会存在Ò 序列. 而且我们可以容易发现, 如果 L_j 不存在曼哈顿通路到目标结点, 必然是存在$\tilde{N}$ 序列. 这就使得没有必要再次去判断是$\tilde{N}$ 还是Ò 序列而直接继续用跳转到 step2 去继续执行.

我们可以看出, 在最坏情况下, 所有 MCC 块都被访问一次, 上述算法的时间复杂度仅仅就是 O(n). 由图 4 可以知道, 对比起网格的规模, n 要小的多. 同时, 我们在下面一章的性能分析中将看到, 在绝大多数的情况下, Step3 将只执行一遍.

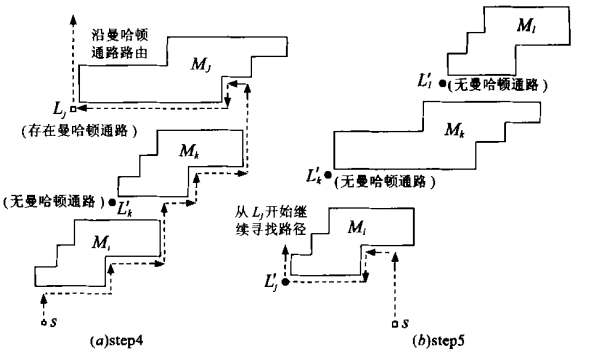

图 4　路由算法示意图

4　实验结果和讨论

测试两点间存在通路, 但是不存在曼哈顿通路的出现比例. 随机地产生了一些故障结点, 来观察 MCC 分布情况以及路由的情况. 网格分别是 50@50, 60@60, 70@70, 80@80, 90@90. 图 5 显示的是 80@80 的结果. 其他边长也有相同的趋势. 当故障率小于 0121 时, 1000 组统计表明, 利用 MCC 模型, 所有情况都能找到一条曼哈顿通路. 当故障率逐渐增高, 存在曼哈顿通路的几率就会逐渐降低. 当不存在曼哈顿路由通路的时候, 有两种可能性: 一种是的确不存在路径, 另一种是存在路径, 但是它不是曼哈顿通路. 例如, 当故障率到达 27% 的时候, 1000 组数据中, 将有 897 组存在曼哈顿通路, 98 组根本没有通路, 还有 5 组就是不存在曼哈顿通路, 但是存在其他通路. 这些数据表明, 在 MCC 结构中, 不存在曼哈顿通路却存在其他通路的情况很少, 要不就不存在通路, 要不就存在曼哈顿通路. 这个事实解释了为什么要在不存在曼哈顿通路之后, 再

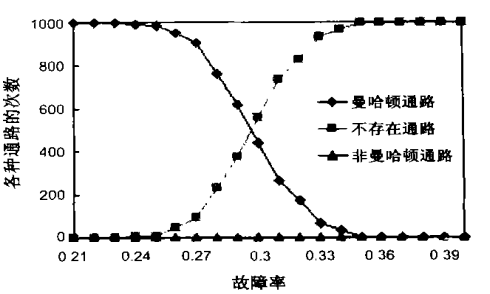

图 5　80@80 的网格中出现非曼哈顿通路的比率图

调用启发式算法来寻找通路.

再对算法在存在通路但不存在曼哈顿通路的网格中进行了模拟测试,用这种算法产生的路由长度与最短路由长度相比.图 6 给出了故障率为 0125 时 500 次不存在曼哈顿通路但存在非曼哈顿通路的情况下,运行的平均数的结果,其他故障率情况下的结果也遵循相同的模式.图中标志为三角的数据为源/ 目之间的曼哈顿路径长度并不存在,标志出来供对比参考.菱形和正方形的数据,分别是在不同边长的网格下,最短路由长度以及我们通过启发式算法 得到路由的长度.从图中可以看出,在所有情况下,启发式路由的长度与最短路由长度相差甚小.这是因为,算法 2 中的 Step2 是来寻找第一个有效的左下结点,并且它存在一条到达目标结点的曼哈顿通路.由于存在路径后,存在曼哈顿通路的几率就非常高,所以,我们很快可以找到到目标结点存在曼哈顿通路的 L 结点.这就意味着,/ 回头路0 通常只会发生一次.

虽然这样的算法并不能保证是最优的,但是在大部分情况求得的是最优解,即使不是最优解,相差也不是很大.如果每次保证求得最优解,将要更多时间代价,这实际上是非常不值得的.

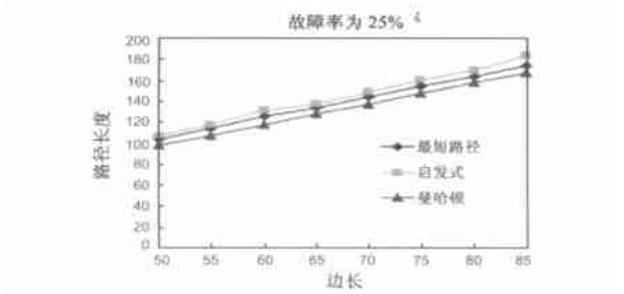

图 6 启发式路由与最短路由的比较

5 结论

提出了一个启发式算法,利用 MCC 模型,来解决网格中容错路由问题.它首先形成 MCC 块集合,然后基于 MCC 重叠图,很容易判断两点是否存在一个曼哈顿通路.如果存在,直接调用 Manhattan 通路算法,否则,调用非 Manhattan 通路启发式容错算法.它的时间复杂度在最坏情况下是 $O(n^3)$, n 为 MCC 块的数目,它比网格的结点数目要小得多.

因为从所有可能的路径中寻找最短的,是一个非常耗时的过程,并且在一个并行计算机系统中,不存在曼哈顿路径而需要寻找非曼哈顿路径的情况出现的几率非常的低,因此花费过量的搜索工作来找寻这种最短路径是非常不必要的.从一个点到另一个点的路由在并行处理机中时刻发生,在大多数情况下,相比起花费大量的时间来找寻一条最短路径,它更需要的是尽快地找到一条路由路径来传递信息.本文提出的算法为容错路由提供了重要的选择.

参考文献:

[1] Dally W J. The J2machine: System support for actors [A]. Actors Knowledge2Based Concurrent Computing [C]. Hewitt and Agha, eds, MIT Press, 1989.

[2] Lillevik S L. The touchstone 30 gigaflop DELTA prototype [A]. Proc. 6th Distributed Memory Computing Conf [C]. Portland, OR, 1996. 671 - 677.

[3] Seitz C L. The architecture and programming of the Amete Series 2010 multicomputer [A]. Proc. 3rd Conf. Hypercube Concurrent Computers and Applications [C]. Pasadema, CA, 1988. I33- I36.

[4] Boppana R V, Chalasani S. Fault2tolerant wormhole routing algorithms for mesh networks [J]. IEEE Trans. Comput, 1995, 44(7): 848- 864.

[5] Boura Y M, Das C R. Fault2tolerant routing in mesh networks [A]. Proc. of Int. l Conf. on Parallel Processing [C]. Urbana2Champion, IL, 1995. I106- I109.

[6] Chien A A, Kim J H. Plana2adaptive routing: Low cost adaptive net2works for multiprocessors [A]. Proc. of the 19th Int. l Symp. on Com2puter Architecture[C]. Queensland, Australia, 1992. 268- 277.

[7] Wang D. A rectilinear2monotone polygonal fault block model for fault2tolerent minimal routing in mesh [J]. IEEE Trans. Comput. 2003, 52(3): 310- 320.

[8] Wu J. Fault2tolerant adaptive and minial routing in mesh2connected multicomputers using extended safety levels [J]. IEEE Trans. Parallel and Distributed Systems, 2000, 11(2): 149- 159.

[9] Su C C, Shin K G. Adaptive fault2tolerant deadlock2free routing in meshes andhypercubes [J]. IEEE Trans. Comput, 1996, 45(6): 666- 683.

作者简介:

陈贵海 男,1963 年 3 月生于广西贵县,教授,博导,1984 年获南京大学计算机软件专业学士学位,1987 年获东南大学计算机应用专业硕士学位,1997 年获香港大学计算机理论博士学位,主要研究方向为并行与分布式系统,网格计算,组合数学等,已在 IEEE Transactions on Parallel & Distributed Systems, The Computer Journal, Computer Communications, Information Systems, International Journal of Foundations of Computer Science 等国际刊物与会议上发表国际论文四十余篇.

杜 鹏 男,1977 年 5 月生于江西南昌,2001 年南京大学计算机系硕士毕业,现为美国纽约州立大学石溪分校博士生,研究方向为并行与分布式系统,算法分析,图形设计.