故障 Torus 网络中的空闲子网搜索方案研究 *

徐 霜[1], 梁家荣[1], 伍华健[2]
(1. 广西大学 计算机与电子信息学院, 南宁 530004; 2. 玉林师范学院 数学与计算机系, 广西 玉林 537000)

摘 要: 为了提高多处理机系统的抗故障能力,对现有的子网搜索算法进行改进,提出了一种新的基于故障节点模式的空闲子网搜索方案。以具有故障节点的二维 Torus 网络为例,详细阐述了方案的具体内容,并给出了相关的算法。该方案是基于集合操作的,能够显著缩小搜索范围并缩短比较时间。实例证明该方法具有可行性。

关键词: 空闲子网; 子网搜索; 故障模式
中图分类号: TP393 **文献标志码**: A **文章编号**: 1001-3695(2009)02-0665-03

Research of submesh searching scheme for Torus networks with faulty nodes

XU Shuang[1], LIANG Jia-rong[1], WU Hua-jian[2]
(1. *School of Computer, Electronics & Information, Guangxi University, Nanning* 530004, *China*; 2. *Dept. of Mathematics & Computer Science, Yulin Normal University, Yulin Guangxi* 537000, *China*)

Abstract: In order to enhance multi-processor system's anti-breakdown ability, this paper proposed a new free submesh-searching scheme. Based on two-dimensional Torus network's with faulty nodes, explained the scheme, and proposed the related algorithm. The scheme was based on manipulating set expressions, with the search space reduced considerably. The experiment proves that this scheme is feasible.

Key words: free submesh; submesh search; faulty mode

0 引言

随着并行计算机互联网络和 VLSI 技术的迅速发展,系统中的并行处理机越来越多,人们对并行计算机互联网络拓扑结构进行了大量的研究,并对其中的一些拓扑结构已研制出了相应的商用和研究用的并行计算机系统[1~4]。Torus 网络是一种完全对称的直连网络拓扑结构,它具有很多优秀的网络特性,如规则对称性、路径多样性以及良好的扩展性。因此,它广泛应用于许多商用系统中。例如,2004 年底评出的全球超级计算机 TOP100 中排名首位的 IBM Blue Gene/L 就采用 Torus 网络;而另一家通信设备制造商,Avici 公司在其推出的世界上第一台太比特路由器中也采用 Torus 网络作为其交换网络拓扑[5]。

通常多处理机系统中都存在一个独立的主处理机,它可以实现任务调度和处理机分配。在并行计算机系统中,请求任务被分配给网络中适当大小的子网执行,该子网可以从一个节点机到整个网络[6,7]。而对处理机进行分配的前提是对网络中的空闲子网进行搜索,在这方面已进行了大量的研究[8]。

随着系统中处理机数目的增加,处理机出现故障的情况是难以避免的,一旦出现故障节点,原有的搜索算法将受到限制:要么不具备完全搜索能力,不能搜索出网络中所有可用的空闲子网;要么需要较高的时间复杂度,从而影响系统的性能。因此研究基于故障节点模式的空闲子网搜索方案是非常必要的[8]。文献[9]中提出的算法能够在具有故障节点的超立方体网络中搜索出所有最大的空闲子立方体;在此基础上文献[10]对具有故障节点的 Torus 网络进行研究,给出了搜索所有最大空闲子网的算法。为了提高系统中处理机的利用率,减少系统碎片的产生,在进行处理机分配前总是希望能够找出所有可用的空闲子网,而不仅仅是最大的空闲子网,为此本文在文献[10]的基础上,以二维故障 Torus 网络为例,给出了一种改进的搜索方案,该方案能够搜索出所有互不包含的空闲子网。

1 相关概念及理论基础[10]

1.1 相关概念

一个二维 Torus 网络用 $T(k_0,k_1)$ 表示,共有 $k_0 \times k_1$ 个节点。k_0 和 k_1 表示在相应维上的节点数。其中 $k_0 \geq 2, k_1 \geq 2$。网络中的每一个节点代表一个处理机,每个节点用坐标(x,y)表示,$0 \leq x < k_0, 0 \leq y < k_1$。

定义 1 在二维 Torus 网络 $T(k_0,k_1)$ 中,对于两个 y 坐标相同的相邻节点 $A(x_u,y_v)$ 和 $B(x_s,y_v)$,如果 $x_s = (x_u + 1) \bmod m$,则节点 B 称为节点 A 的正相邻节点,而从 A 到 B 的方向为正方向;同理,对于两个 x 坐标相同的相邻节点 $A'(x_u,y_t)$ 和 $B'(x_u,y_t)$,如果 $y_t = (y_v + 1) \bmod n$,则 B'称为 A'的正相邻节点,从 A'到 B'的方向为正方向。相反,从 B 到 A(或从 B'到 A')的方向为负方向,所以节点 A 称为节点 B 的负相邻节点(节点 A'称为节点 B'的负相邻节点)。

关于正方向的定义同样适用于 Mesh 网络。在 Mesh 网络中,如果一个节点的所有相邻节点都是它的正邻节点,则该节点称为 Mesh 网络的基节点;相应地,如果一个节点的所有相邻节点都是它的负邻节点,则该节点称为 Mesh 网络的末端节点。

定义 2 在二维 Torus 网络 $T(k_0,k_1)$ 中,一个子网 S 可表

收稿日期: 2008-05-16; **修回日期**: 2008-08-26 **基金项目**: 国家自然科学基金资助项目(60564001);国家教育部优秀人才支持计划专项资助项目(NCET-06-0756);广西自然科学基金资助项目(桂科自 0832286)

作者简介:徐霜(1982-),女,湖北襄樊人,硕士,主要研究方向为并行计算、网络容错(xushuang021@163.com);梁家荣(1966-),男,教授,博士,主要研究方向为粗糙集、数据挖掘;伍华健(1964-),男,副教授,主要研究方向为计算机理论与研究.

示为 $d_0(i:j)d_1(s:t)$。其中:$0\leqslant i,j<k_0$,$0\leqslant s,t<k_1$;d_0 和 d_1 分别代表 x 方向和 y 方向;在 x 维上从 i 到 j 为正方向,在 y 维上从 s 到 t 为正方向。一个空表达式代表整个网络。例如在图 1 中,节点(0,0),(0,1),(0,2),(0,3)所在的子网可表示为 d_0(0,0),由于在 y 方向上跨越整个网络,在该方向上的表达式就可省略。

定义 3　在 Torus 网络中,对于一个给定的节点 D,相对于 D 的候选子网是以 D 为基节点且包括整个网络节点的一个 Mesh 结构的子网。图 2 是相对于节点(1,1)的候选子网。

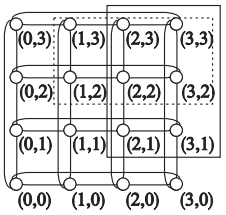

图1　具有两个故障节点的4×4 Torus网络

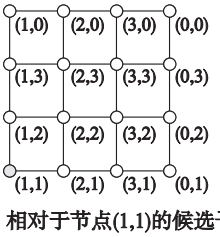

图2　相对于节点(1,1)的候选子网

定义 4　在二维 Torus 网络 $T(k_0,k_1)$ 中,对于给定的节点 $D(x_1,y_1)$,相对于 D 的候选子网中离 D 最远的节点称为 D 的末端节点,末端节点用 $E(x_2,y_2)$ 表示,其坐标值可由以下式子得到:

$$x_2=\begin{cases}x_1-1, & 0<x_1<k_0\\ k_0-1, & x_1=0\end{cases},\quad y_2=\begin{cases}y_1-1, & 0<y_1<k_0\\ k_1-1, & y_1=0\end{cases}$$

如图 1 所示,节点(0,0)的末端节点为(3,3),而节点(1,1)的末端节点为节点(0,0)。

定义 5　在一个具有故障节点的 Torus 网络中,对于一个给定的正常节点 D,相对于 D 的拒绝区域是包括一个故障节点和 D 的末端节点的最小子网。

图 1 中(1,2)和(2,1)为故障节点,对于节点(0,0),它的两个拒绝区域为图中用虚线框标出的子网:$d_0(1:3)d_1(2:3)$ 和 $d_0(2:3)d_1(1:3)$。

定义 6　对于给定的正常节点 D,以 D 为基节点的基本子网满足以下条件:基本子网内的节点都是正常且空闲的,且该子网不被以 D 为基节点的其他任何空闲子网所覆盖。

定义 7　一个空闲子网被称为控制子网,若它不能被其他空闲子网完全覆盖,控制子网之间可以相互重叠。

1.2　理论基础

a)用 + 号表示两个子网的并集,用· 表示两个子网的交集。例如在图 1 中,子网 $d_0(0:0)$,$d_1(0:0)$ 的并集可表示为 $d_0(0:0)+d_1(0:0)$,交集运算符· 有时可以省略。

b)对于一个表达式 $d_i(b:e)$,它的补集可表示为 $\overline{d_i(b:e)}$,且 $\overline{d_i(b:e)}=d_i((e+1)\bmod k_i:(b-1)\bmod k_i)$。

c)得摩根定律:$\overline{A+B}=\overline{A}\cdot\overline{B}$,$\overline{A\cdot B}=\overline{A}+\overline{B}$(在本文中 A、B 代表子网表达式)。

d)化简规则:若 $B\subset A$,则 $AB=B$,$A+B=A$。

2　基于故障节点模式的子网搜索方案

2.1　子网搜索方案的理论依据

引理 1[10]　一个基本子网不包括拒绝区域内的任一节点

引理 2[10]　一个节点如果处于所有的拒绝区域之外,则它必处于一个或多个基本子网之内。

由以上概念以及引理 1、2 可知,对于网络中给定的一个正常节点 D,所有正常节点可以分为两类:a)属于 D 的拒绝区域内的节点(该类节点构成的集合用 R 表示);b)以 D 为基节点的基本子网中的节点(该类节点构成的集合用 P 表示)。例如在图 1 中,对于正常节点(0,0),$R=\{(2,1)(3,1)(1,2)(2,2)(3,2)(1,3)(2,3)(3,3)\}$,$P=\{(0,0)(1,0)(2,0)(3,0)(0,1)(1,1)(1,1)(0,2)(0,3)\}$。

本文提出的搜索方案的基本目标就是当网络中出现故障节点时,在节点恢复正常之前,能够搜索出所有的控制子网。

在 Torus 网络中,对于几个子网的并集,利用得摩根定律可以快速得到不在该并集中的所有节点的地址。根据节点的故障信息以及定义 4 中给出的公式,可以很容易地得到相对于某个正常节点的拒绝区域,而有了拒绝区域就可以利用得摩根定律求出相应的基本子网构成的集合。在图 1 中,正常节点(0,0)的末端节点为(3,3)。故障节点(1,2),(2,1)和节点(3,3)构成的拒绝区域为

$$R=d_0(1:3)d_1(2:3)+d_0(2:3)d_1(1:3)$$

则基本子网集

$$P=\overline{R}=\overline{(d_0(1:3)d_1(2:3)+d_0(2:3)d_1}\ \overline{(1:3))}=$$
$$\overline{(d_0(1:3)+d_1(2:3))}\,\overline{(d_0(2:3)+d_1(1:3))}\text{(利用得摩根定律)}=$$
$$(d_0(0:0)+d_1(0:1))(d_0(0:1)+d_1(0:1))\text{(利用补集公式)}=$$
$$d_0(0:0)+d_1(0:0)+d_0(0:1)d_1(0:1)$$
$$\text{(利用分布律、结合律、化简规则)}$$

对于基本子网集的获取,可以直接调用文献[10]中的算法 A。

2.2　具体的子网搜索方案

搜索方案的基本设计思想如下:首先每个故障节点将其地址信息以广播形式发送给与其相邻的正常节点,收到信息的正常节点再将故障信息发送给与其相邻的正常节点,直到网络中的所有正常节点都收到信息为止。然后每个正常节点根据定义 4 中的公式计算出其末端节点,由末端节点及故障节点的信息就可以得到拒绝区域,拒绝区域的补集就是所有基本子网构成的集合。利用文献[10]中的算法 A 就可得到所有的基本子网,由此得到的基本子网可能存在着这样的情况:一个基本子网可能覆盖一个或多个子网。最后利用下面给出的一个比较算法,即可得到所有的控制子网。综合上面的叙述,给出得到控制子网集合的完整算法:

a)故障节点机向正常节点广播故障节点地址信息;

b)每个正常节点计算其末端节点的地址;

c)每个正常节点根据末端节点以及故障节点地址计算出该节点的拒绝区域;

d)利用得摩根定律对基本子网集进行转换;

e)每个正常节点调用文献[10]中的算法 A,得到其基本子网的集合;

f)网络中的主机调用 compare()算法,得到所有的控制子网。

下面给出 compare()算法的具体描述。用列表 f 存储控制子网,最初 f 为空,每个正常节点按照规定的顺序向主机发送其基本子网的信息。假设当前节点发送来的基本子网的个数为 n,这 n 个子网构成的集合用 p 表示,其中的每一个子网用 p_i($1\leqslant i\leqslant n$)表示,现将 p_i 与 f 中的每个元素进行比较(f 中的元素用 c_i 表示)。如果存在某个子网 c_k 满足 $c_k\subset p_i$,则将 c_k 从 f 中移出;如果存在某个子网 c_k 满足 $c_k\supset p_i$,则终止 p_i 的比较。当全部比较完后,如果没有一个子网 c_j 使得 p_i 满足 $p_i\subset c_j$ 或 $c_j\subset p_i$,则将 p_i 添加到 f 中。具体算法如下:

compare()

a) {if (f 为空)则将 p 中的元素全部添加到 f 中

b)　else

c)　for(i=1; i≤n; i++)//假设 p_i 为 $d_0(x_1,x_2)d_1(y_1+y_2)$, c_j 为 $d_0(x_1',x'_2)d_1(y_1',y_2')$

d)　{ for($j=1;i\leq f$. length; $j++$)

e)　{ flag1 =0; flag2 =0;// flag1、flag2 为两个标记变量

f)　if ($x_1\leq x_1'\leq x_2'\leq x_2$)//在 x 方向上 c_j 处于 p_i 范围内

{ if ($y_1\leq y_1'\leq y'_2\leq y_2 \parallel y_1'\leq y_2'\leq y_2\leq y_1 \parallel y_2'\leq y_2\leq y_1\leq y_1'$)//$p_i$ 包含 c_j　{将 c_j 从 f 中移出;flag1 ++;} }

g)　if (x_1 = =null & & x_2 = =null)

//在 x 方向上 q_i 跨越整个网络

{if($y_1\leq y_1'\leq y_2'\leq y_2$){将 c_j 从 f 中移出;flag 1 ++;}}

h)　if ($x_2'\leq x_2\leq x_1\leq x_1'$)

{if($y_i\leq y_1'\leq y_2'\leq y_2 \parallel y_1'\leq y_2'\leq y_2\leq y_1 \parallel y_2'\leq y_2\leq y_1\leq y_1' \parallel y_2\leq y_1\leq y_1'\leq y_2' \parallel y_1$ = =null){将 c_j 从 f 中移出;flag1 + +;}}

i)　if ($x_1'\leq x_1\leq x_2\leq x_2'$)//在 x 方向上 p_i 处于 c_j 范围内

{if ($y_1'\leq y_1\leq y_2\leq y_2' \parallel y_1\leq y_2\leq y_2'\leq y_1' \parallel y_2\leq y_2'\leq y_1'\leq y_1$)//$p_i\subset c_j$

{ flag2 + +;;返回到 c);}}

j)　if (x_1' = =null & & x_2' = =null)

{if($y_1'\leq y_1\leq y_2\leq y_2'$){ flag2 ++;;返回到 C);} }

k)　if($x_2\leq x_2'\leq x_1'\leq x_1$)

{if ($y_1'\leq y_1\leq y_2\leq y_2' \parallel y_1\leq y_2\leq y_2'\leq y_1' \parallel y_2\leq y_2'\leq y_1'\leq y_1 \parallel y_2'\leq y_1'\leq y_1\leq y_2 \parallel y_1'$ = =null)

{ flag2 ++;;返回到 c); } }

}

l}　if (flag1! =0 && flag2 = =0)将 p_i 添加到 f 中;

}

3　实验

本文用一个仿真实验来验证算法的可行性。利用仿真软件 OPNET 进行仿真,业务源是经典假设的泊松源,仿真网络中共有 17 个节点机。其中 1 个作为主机,并与其他节点直接相连,接收这些节点机发送来的信息,然后执行相应的算法对其进行处理;其他 16 个节点构成 4×4 的二维 Torus 网络结构(图 1),仿真时假设节点(1,2)和(2,1)为故障节点,故障节点可以向其周围节点发送故障信息。当执行完算法 A 后,每个正常节点的基本子网集如表 1 所示。最后执行完 compare()算法后,得到的控制子网的集合为{$d_0(0:0),d_0(0:1)d_1(0:1)$, $d_1(0:0)$, $d_0(3:0)$, $d_0(3:1)d_1(0:1)$,$d_0(2:0)d_1(2:0)$,$d_1(3:0)$,$d_0(0:1)d_1(3:1)$,$d_0(3:1)d_1(3:1)$}共九个控制子网。这与实际情况是完全一致的,达到了预期的目标。

表 1　正常节点的基本子网集

基节点	相应的基本子网	基节点	相应的基本子网
(0,0)	$d_0(0:0),d_0(0:1)d_1(0:0),d_1(0:0)$	(0,2)	$d_0(0,0)$
(1,0)	$d_0(1:1)d_1(0:1),d_1(0:0)$	(2,2)	$d_0(2:0)d_1(2:0)$
(2,0)	$d_1(0:0)$	(3,2)	$d_0(3:0)$
(3,0)	$d_0(3:0),d_0(3:1)d_1(0:1),d_1(0:0)$	(0,3)	$d_0(0:0),d_0(0:1)d_1(3:1),d_1(3:0)$
(0,1)	$d_0(0:0),d_0(0:1)d_1(1:1)$	(1,3)	$d_0(1:1)d_1(3:1),d_1(3:0)$
(1,1)	$d_0(1:1)d_1(1:1)$	(2,3)	$d_1(3:0)$
(3,1)	$d_0(3:0),d_0(3:1)d_1(1:1)$	(3,3)	$d_0(3:0),d_0(3:1)d_1(3:1),d_1(3,0)$

4　结束语

本文提出了一个基于故障节点模式的空闲子网搜索策略和相关算法,充分考虑二维 Torus 网络中节点故障的发生对系统产生的影响,推广了已有的研究成果,并用实例验证了方案的可行性。这有利于进一步推广 Torus 网络在大规模并行处理机中的应用,同时也为具有故障节点的 Torus 网络中处理机的分配提供了重要的理论基础。

参考文献:

[1] LENOSJI D, LAUDON J, GHARACHORLOO K, *et al*. The Stanford DASH multiprocessor[J]. Computer, 1992, 25(3): 63-79.

[2] GABRANI G, MULKAR T. A quad-tree based algorithm for processor allocation in 2D Mesh-connected multicomputers [J]. Computer Standards & Interfaces, 2005, 27(2): 133-147.

[3] YOO B S, DAS C R. A fast and efficient processor allocation scheme for Mesh-connected multicomputers [J]. IEEE Trans on Computers, 2002, 51(1): 46-60.

[4] CHIU G M, CHEN S K. An efficient submesh allocation scheme for two-dimensional Meshes with little overhead [J]. IEEE Trans on Parallel & Distributed Systems, 1999, 10(5): 471-486.

[5] 顾华玺,刘增基,王琨,等. Torus 网络中分布式自适应路由算法[J]. 西安电子科技大学学报:自然科学版, 2006, 33(3): 352-358.

[6] 张艳,孙世新,彭文钦. 网格多处理机的一种改进的子网分配算法[J]. 软件学报, 2001, 12(8): 1250-1257.

[7] 张艳,孙世新,彭文钦. 一种最佳的 Mesh 中的空闲子网搜索算法[J]. 系统工程与电子技术, 2001, 23(4): 83-86.

[8] ISMIAL I, DAVIS J. Program-based static allocation policies for highly parallel computers[C]//Proc of IPCCC'95. [S.l.]: IEEE Computer Society, 1995: 61-68.

[9] CHEN H L, TZENG N F. A boolean expression-based approach for maximum incomplete subcube identification in faulty hypercubes[J]. IEEE Trans on Parallel & Distributed Systems, 1997, 11(8): 1171-1183.

[10] CHEN H L, HU Shu-hua. Submesh determination in faulty Tori and Meshes[J]. IEEE Trans on Parallel and Distributed Systems, 2001, 12(3): 272-282.

(上接第 664 页)

[3] 黄骥. GLFR:一种新型的基于地理位置信息的 Ad hoc 网络路由算法[D]. 广州:暨南大学, 2007: 26-33.

[4] 唐勇,周明天,张欣. 无线传感器网络路由协议研究进展[J]. 软件学报, 2006, 17(3): 410-421.

[5] 许力,张继东,郑宝玉,等. 移动自组网能量保护策略研究进展[J]. 通信学报, 2004, 25(9): 93-103.

[6] 谭长庚,张芝华,王建新,等. MANET 能量与其他网络性能平衡路由协议[J]. 计算机应用, 2007, 27(5): 1073-1076.

[7] 袁明,张连芳,舒炎泰. Ad hoc 网络路由协议能量消耗分析[J]. 计算机工程与应用, 2003, 15(4): 146-149.

[8] BUTTYAN L, HUBAUX J P. Enforcing service availability in mobile Ad hoc WANs[C]//Proc of IEEE/ACM Workshop on Mobile Ad hoc Networking and Computing. Washington: IEEE Computer Society, 2000: 87-96.

[9] 陆凤山. 排队论及其应用[M]. 长沙:湖南科学技术出版社, 1984: 18-33.

[10] The VINT Project. The UCB/LBNL/VINT network simulator-NS(version 2) [EB/OL]. (2005-11) [2006-02-11]. http://mash.cs.berkeley.edu/ns.

[11] BLACK M, DINI P, KULKARNI P, *et al*. Deploying lightweight queue management for improving performance of mobile Ad hoc networks[C]//Proc of International Conference on Networking and Services. Washington: IEEE Computer Society, 2006: 102-108.

[12] 刘明,窦文华,张鹤颖. 主动队列管理研究综述[J]. 计算机工程, 2006, 32(24): 84-86.