JOURNAL OF NATIONAL UNIVERSITY OF DEFENSE TECHNOLOGY
第18卷第1期 1996年3月 Vol. 18 No. 3

SN－PDB 中一种先进的结点容错法*

昌月楼 杨 利 阳国贵

(国防科技大学计算机系 长沙 410073)

摘 要 本文简单介绍了无共享并行数据库(SN－PDB)中几种常见的结点容错方法，重点叙述了链式分布法，给出了此方法的负载平衡算法。

关键词 并行处理，数据库，容错，负载均衡

分类号 TP311.13

An Advanced Fault Tolerance Method for Nodes in SN－PDB

Chang Yuelou Yang Li Yang Guoguei

(Deparement of Computor Science, NUDT, Changsha 410073)

Abstract Several commonly used fault tolerance methods for nodes in shared-nothing parallel database systems(SN－PDB) are presented in the paper. The chained declustering method is focused, and the load balancing algorithm of it is given in the paper.

Key words parallel processing, database, fault tolarence, load balance

并行处理的研究是目前计算机发展中的热门课题之一，吸引了很多的科技人员投身于并行硬件结构、并行软件和并行算法等的研究和开发中，研制出SMP类型和MPP类型的并行机，解决了不少过去由于受限于计算机的能力而不能处理的问题。随之出现并行数据库(PDB)的研究。这是一个具有挑战性的研究领域。由于数据库在各个部门广泛应用，更由于人们对数据库的处理速度和可用性的要求"得寸进尺"，所以尽管研究并行数据库难度很大，仍有许多有志之士在孜孜不倦地耕耘，开发出一些并行数据库原型和商用机上的数据库并行处理功能。前者有MCC的Bubba[1]，UW的Gamma[2]以及Teradata的DBC/1012[3]，等等；后者有Oracle/nCUBE2[4]，Sybase、Informix，等等。Tandem的NonStop SQL已在国内市场上出现，并应用于一些大型的金融信息系统中。

对并行数据库的要求一方面是高性能，即对G量级或T量级的数据提供高效处理(快速的OLTP和快速的DSS支持)和提供大吞吐量；另一方面是系统的高可用性，即要求系统能够提供不停运转，从而要求硬件系统提供容错手段，满足高可靠性。当然软件也要有相应的措施。具体地说，在无共享并行数据库中，要求能够做到在有结点失效的情况下，数据库能继续工作，尽管性能有所下降。这方面的研究是伴随PDB的研究开始的，并出现了不少有效的容错手段。

* 1995年4月28日收稿

1 SN－PDB 中常用的几种结点容错法

1.1 副本方式

1.1.1 数据镜象

图 1 是 Tandem 的 NonStop SQL 所采用的镜象方式，其中假设表 R 只被分割在两个结点上，R_1 和 R_2 为主拷贝，r_1 和 r_2 为副拷贝。

在这种结构下，磁盘容错是很直观的：假设 Diskl 有故障，则马上可以启动 Disk2 继续工作，同时对 Disk1 进行“热”替换。为了做到处理机容错，对于每个结点的磁盘访问有两个进程位于两个处理机上，即 P_1 和 P_2 上都有访问本结点的 I/0进程和他结点的I/0进程，这样，其中一个处理机有故障时另一个可以“接管”它的工作。当然，对于通道故障，也可用双通道备份的方式进行容错。

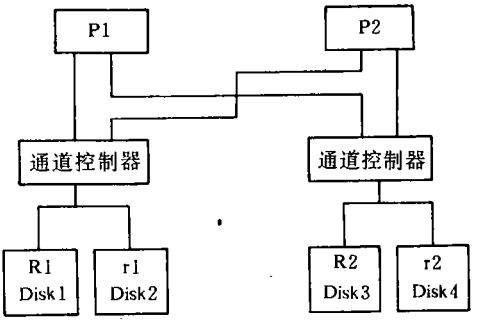

图 1 Tandem NonStop SQL 的镜象磁盘结构

1.1.2 交叉分布

在交叉分布模式中，一个表的主拷贝以某种水平分割方式分散存放在 n 个结点上，而副拷贝的分布则有所变化。某一个结点上的相应于该结点主拷贝的副本再被分成$(n-1)$个部分分散存放在其余$(n-1)$个结点上。每个结点都是如此，形成所谓的“交叉”分布。假设 $n=4$，则表 R 的交叉分布如图 2 所示，其中 R_i 为主拷贝，$r_{i,j}$为副拷贝，即 $r_{i,j}$是 R_i 再分割以后的部分。

结点号	0	1	2	3
主拷贝	R_0	R_1	R_2	R_3
副拷贝		$r_{0,0}$	$r_{0,1}$	$r_{0,2}$
	$r_{1,2}$		$r_{1,0}$	$r_{1,1}$
	$r_{2,1}$	$r_{2,2}$		$r_{2,0}$
	$r_{3,0}$	$r_{3,1}$	$r_{3,2}$	

图 2 交叉分布

在这种模式下，任意结点失效时，同一簇中的其余活跃结点可以均摊失效结点的工作。例如结点 2 失效，由于它的副拷贝 $r_{2,0}$、$r_{2,1}$、$r_{2,2}$分别存放在结点 3、结点 0 和结点 1 上，从而这些结点可以分担结点 2 的 I/0 操作。

2.2 冗余校验方式

这是 RAID(Redundant Array of Inexpensive Disks)磁盘阵列技术中采用的方式。每个结点使用一个磁盘阵列。其中有一个检验盘，其余为数据盘。结点的数据依次存放在数据盘上，而检验盘存放的是各数据盘的校验数据(各数据盘的数据按位异或的结果)。正常 I/0 操作访问的是数据盘。当数据盘中有一个出现故障时，利用活跃的数据盘和检验盘的数据来恢复丢失的数据。

磁盘号	0	1	2	3
BLOCK0	$S_{0,0}$	$S_{0,1}$	$S_{0,2}$	
BLOCK1	$S_{1,0}$	$S_{1,1}$		$S_{1,2}$
BLOCK2	$S_{2,0}$		$S_{2,1}$	$S_{2,2}$
BLOCK3		$S_{3,0}$	$S_{3,1}$	$S_{3,2}$
BLOCK4	$S_{4,0}$	$S_{4,1}$	$S_{4,2}$	

图 3 4 级 RAID 示意图

图 3 是 4 级 RAID 系统示意图，其中每三个扇区组成一块(BLOCK)，$S_{i,j}$表示第 i 块的第 j 扇区。阴影区表示检验数据。

这种方式的优点是没有数据副本，节省磁盘空间；缺点是在正常写操作和故障情况下读操作均需要作异或运算，多用了 CPU 时间。

2 先进的链式分布容错法

2.1 链式分布方法

图 4 给出了 $n=4$ 的链式分布结构，即假设表 R 分布在 4 个

结点号	0	1	2	3
主拷贝	R_0	R_1	R_2	R_3
副拷贝	r_3	r_0	r_1	r_2

图 4 链式分布

结点上。它在4个结点上的主副拷贝的分布，错开一个结点的位置。其名称也由此而来。正常工作使用主拷贝，有结点故障时同时使用主副拷贝。

假设结点1失效，乍一看来，R_1的I/O负载会全部落在结点2上；但是，如果调整一下活跃结点的负载，则可做到负载均匀，

如图5所示。

这样，在任意结点失效的情况下，对于n个结点的系统，其余活跃的每个结点都只增加$1/(n-1)$的负载。在图5的例子中，结点0的I/O操作，除了原来的R_0以外，增加访问$(1/3)*r_3$，结点2的I/O操作是访问$(1/3)*R_2$和r_1，结点3的I/O是访问$(2/3)*R_3$和$(2/3)*r_2$。这样，表R的各部份均能访问到，并能维持并行处理，能做到负载均匀。

结点号	0	1	2	3
主拷贝	R_0		$1/3R_2$	$2/3R_3$
副拷贝	$1/3r_3$		r_1	$2/3r_2$

图5　结点1失效后的负载调节

还要注意的是，一个结点失效后活跃结点负载的调整不需要数据在结点之间进行迁移。只要改变常驻内存控制表中的一些边界值的指针即可。下面将详细讨论这一点。

2.2　负载平衡算法

2.2.1　修改Split表

在SN－PDB中，一般说来，一个查询经优化以后，由调度器根据数据的分布情况和处理机的忙闲情况，进行查询、处理、调度。调度器有一个Split表，它指出一个表的元组在各个结点上的分布情况。不失一般性，下面假设表R按照范围划分的原则分布在4个结点上，其划分属性X的范围是[1，400]，于是有图6所示的Split表。

在正常情况下，调度器根据Split表，将查询要求发往相应结点。查询在主拷贝上进行。精确查询涉及到一个结点，范围查询可能涉及到多个结点。若某结点失效，Split表则发生变化。调度器根据新的Split表向相应结点发出查询指示。

结点号	0	1	2	3
主拷贝 属性范围	R_0 1－100	R_1 101－200	R_2 201－300	R_3 301－400
副拷贝 属性范围	r_0 301－400	r_1 1－100	r_2 101－200	r_3 201－300

图6　表R的Split表

下面给出修改Split表的公式。

修改Split表就是在原表的基础上修改各结点上划分属性值的范围。假设表R分布在M个结点上，它在结点i的主拷贝属性值的下限值为$W_l(R_i)$，上限值为$W_h(R_i)$；又设失效结点号为s，则修改后结点i的属性值范围是：

$$[W_l(Ri),\ W_h(Ri) - f(i,s)*(W_h(Ri) - W_l(Ri) + 1)] \tag{1}$$

其中，$f(i,s)=[M-(i-s+1)]\text{Model}M/(M-1)$，(1)式中有＊号的项往上取整数。

设表R在结点i的副拷贝属性值的下、上限值分别为$W_l(ri)$和$W_h(ri)$；则当结点s失效时，修改后结点i的副拷贝属性值范围是：

$$[W_l(ri) + g(i,s)*(W_l(ri) - W_l(ri) + 1),\ W_h(ri)] \tag{2}$$

其中，$g(i,s)=[M-(s-i+1)]\text{Model}\ M/(M-1)$，(2)式中有＊号的项往下取整数。

下面以图6为例，假设结点1失效来修改Split表。这时，$s=1$，$M=4$，于是可得：

$$f(0,1) = [4-(0-1+1)]\text{Model}\ 4/3 = 0$$

$$f(2,1) = [4-(2-1+1)]\text{Model}\ 4/3 = 2/3$$

$$f(3,1) = [4-(3-1+1)]\text{Model}\ 4/3 = 1/3$$

从而，对于主拷贝R_0、R_2和R_3，其属性范围分别为

$$[1,\ 100-0*100]=[1,\ 100]$$

$$[201,\ 300-(2/3)*100]=[201,\ 233]$$

[301，400－(1/3)＊100]=[301，366]

同样可得：$g(0,1)=2/3$，$g(2,1)=0$，$g(3,1)=1/3$.从而，对于副拷贝 r_0，r_2 和 r_3，其属性范围分别为：[367，400]，[101，200]，[234，300]，于是得出修改后的 Split 表如图 7 所示。

由于在一个结点失效的情况下，各活跃结点的主副拷贝都要参与查询工作。从图 7 可以看出，在查询要求均匀分布的情况下，各活跃结点的负荷都增加 1/3。这与图 5 的结论是一致的。

结点号	0	1	2	3
主拷贝 属性范围	R_0 1－100		R_2 201－233	R_3 301－366
副拷贝 属性范围	r_0 367－400		r_2 101－200	r_3 234－300

图 7　结点 1 失效后的新 Split 表

2.2.2　修改选择谓词

为简单计，下面假设查询条件中涉及的属性正好是数据划分所依赖的属性。由于 Split 表的变化，因而各结点的工作范围有变化，即它们负责查询的属性范围有变化。在正常情况下，各结点的工作范围是各主拷贝的划分范围；然而，在有一个结点失效的情况下，各结点的主副拷贝都用来查询，负责查询的属性范围有调整，于是，查询条件的选择谓词要作适当调整。调度器不能按正常情况下的 Split 表作调度，只能按照修改后的新 Split 表作调度。

这一工作是很直观的。对精确查询来讲，调度器按照新的 Split 表直接将查询要求发往指定结点；对范围查询来讲，调度器将各结点的工作属性范围与输入查询要求的选择属性范围进行“逻辑与”。若结果非空，则按结果修改查询谓词，并将查询要求发往相应结点。

下面以图 6 和图 7 来举例说明。

(1)精确查询

假设查询谓词是“$R.x=150$ OR $R.x=240$ OR $R.x=395$”。在正常情况下，调度器按照图 6 将查询要求发往结点 1、结点 2 和结点 3，在主拷贝上进行；在结点 1 失效后，调度器按照图 7 将查询要求发往结点 2、结点 3 和结点 0，在副拷贝上进行。

(2)范围查询

假设查询谓词是“$R.x>150$ AND $R.x<250$”在正常情况下，调度器按照图 6 将查询要求发往结点 1、结点 2，在主拷贝上进行；在结点 1 失效后，调度器按照图 7 和上述规则将查询要求分解为：($R.x>150$ AND $R.x<=233$) OR ($R.x>=234$ AND $R.x<250$)。将前者发往结点 2，在其主、副拷贝上进行；将后者发往结点 3，在副拷贝上进行。

2.3　与其他方法的比较

这里主要从两个方面将链式分布与第 1 节中给出的各种容错法进行比较。这两个方面是：可用性(Availability)和负载均衡(Loadbalancing)。

(1)可用性

可用性的高低反映容错能力的大小。从图 1 可以看出，在数据镜象中，若将 $2n$ 个节点按照该图的方式构成 n 个镜象对，将一个表分布在这 n 个镜象对中，只有同一镜象对中的两个结点失效时才会引起系统不可用。从图 2 可以看出，在交叉分布中，同一簇中任意两个结点失效将导致系统不可用。从图 4 可以看出，在链式分布中，只有当相邻两个结点失效时，才会丢失表中的数据，从而使系统不可用。除此以外的任意两个结点失效均不会如此，因为表的各部份数据都还完整地保留在活跃结点中。冗余校验方式不能提供处理机容错，因而可用性是最差的。

进一步分析一下。在目前工艺水平下，磁盘的 MIBF 是 3～5 年。假定磁盘故障的出现是一平稳泊松流，均值是 3 年，又假定排除磁盘故障恢复使用的时间是 5 小时，那么由于任意两个磁盘出现故障而造成系统不可用的概率，即相邻两个磁盘故障的时间不超过 5 小时的概率可用如下公式计算：

$$P=1-e^{-\lambda}$$

其中 $\lambda=1/(3*365*24)$，$t=5$。设 M 为磁盘个数，当 $M=2$ 时，$P\approx0.0002$；当 $M=8$、32、250 和 1，000时，P 分别约是 0.0014、0.0062、0.05 和 0.2。然而，在链式分布中，只有相邻两个结点的磁盘出现故障时才会造成系统不可用。这个概率是 0.0004，与结点个数无关。由此可见，链式分布系统的可用性大大优于交叉分布的系统，结点越多，优越性越大。

(2)负载均衡

负载是否均衡影响系统的性能。在数据镜象模式中，在一个磁盘失效对全系统性能的影响并不明显，而在一个结点(处理机)失效时全系统性能的影响是很糟的，因为另一结点承担了双倍的工作，引起了严重的负载不均。在交叉分布模式中，任意结点失效时，族内其余活跃结点可以均摊失效结点的工作。在冗余校验方式中，一个磁盘失效时系统性能的影响很小。从第 2.2 节可以看出，链式分布在一个结点失效时不但能做到负载均衡，而且这种负载的调整是动态的。动态平衡的好处是可以考虑处理机忙闲的情况。

对比之下，交叉分布模式在可用性和负载均衡方面都优于其他模式。

3 结 语

从上面的分析看出，链式分布无疑是 SN－PDB 中一种先进的结点容错方式。然而，还有一些值得深入研究的地方。首先是查询的非均匀性问题。第 2.2.1 节中给出的算法的前提是，假定对表中各元组的查询是均匀分布的。如果不是这样(实际上很可能)，又怎样修改算法使它动态地自适应实际的查询分别情况，做到总是保持各结点负载均衡，这是个很有意义和很有趣的问题。当然，系统首先要能够积累关于查询分布的统计，这是另外的问题。与负载均衡有关的另一个问题是数据迁移。频繁的数据迁移可能会由于通信瓶颈影响系统性能；但是，当一个查询只引起少数几个结点工作而多数闲着时，有必要使闲着的结点也忙起来，提高并行度。这就带来了数据在结点间的迁移。这时，必须修改上面的算法以便自动适应这种情况。最后一个问题是 PDB 中普遍关心的问题，即划分属性和查询属性不一致时的算法问题。这个问题也许最难解决，因为查询属性是变化多端的，而且可能是复合属性。黑龙江大学的李建中教授提出的 CMD 多维属性划分方法为解决最后一个问题找到了一个较好的途径[6]，但在多维划分方式下的结点容错技术值得进一步研究。笔者认为，对上述问题的解决不能看作是“锦上添花”，而是“雪中送炭”。

参考文献

1 Boral Haran Alexander William Clay Larry Prototyping Bubba, A Highly Parallel Database System. IEEE transactions on Knowledge and data engineering, 1990, 2(1)

2 Dewitt D I, Ghandeharizadehv S. The Gamma Database Machine Project, IEEE transactions on Knowledge and data engineering, 1990, 2(1)

3 Page J. High Performance Database for Client/Server Systems. Applied Imformation Technology 13, Parallel Processing and Data Management, 1992

4 Costicoglou S, Podgorng M, Choudhary A. On Benchmarking the ORACLE Parallel Server on NCUBE2. Advances in Parallel & Distributed System, October 6, 1993

5 杨利，周兴铭，郑若忠. 并行数据库系统的体系结构. 计算机科学，1994，21(4)

6 Li J Z, Srivastara J, Rotem D. CMD:A Multidimensional Declustering Method for parallel Database systems, In Proceeding of the 18st VLDB Conference, Canada, 1992

(责任编辑　潘生)