分布式系统重构算法的初探
Discussion on Reconstructed Algorithm in Distributed System

吴晓渊
Wu Xiaoyuan

(桂林陆军学院计算机中心 桂林 541002)
(Computer Center, Guilin Military Academy, Guilin, 541002)

摘要 提出一种适于集中式分布式系统的系统重构算法,它根据出发条件分别通过重新分配进程来消除节点失效,或通过转移进程来均衡系统负载。每隔一段较长的时间给出统计总表,对系统硬件的更新换代给出建议。保证了当分布式系统的某些部分失效时,整个系统能够继续正常运行,同时还保证调整系统负载,使CPU负载和网络通信量相对平衡。

关键词 分布式系统 重构 算法
中图法分类号 TP301.6

Abstract A system-reconstructed algorithm for the centralized distributed system is discussed. The algorithm could timely make a need-to-update report to the system in terms of the conditions preset. The system could be reconstructed and kept running through clearing up invalid nodes by reassignment of processing and balancing load by transferring of processing when some parts of the system are out of work.

Key words distributed system, reconstruction, algorithm

近年来,计算机技术的发展日新月异。尤其是高性能微处理器和网络技术的出现,使计算机成为更加廉价和便利的工具,计算机在各项领域的应用有了迅猛的突破。在拥有了丰富的计算机资源后,如何有效利用这些资源便成了一个紧迫的课题,于是分布式处理技术应运而生,成为当今计算机应用技术的前沿课题。

分布式系统是由多个相互连接的处理资源组成的计算机系统,这些系统可以合作执行一个共同的任务,最少依赖于集中的程序、数据和硬件等资源。分布式系统具有很多优点和特性,其中可重构性是其必不可缺的一个部分。可重构性是指当分布式系统的某些部分失效时,能确保整个系统继续正常运行的特性,同时它还保证调整系统负载,使CPU负载和网络通信量相对平衡[1]。

2002-05-28 收稿。

本文的任务就是对于一个假定的分布式系统，讨论在其上实现可重构性的算法。

1 重构目标

一个系统是否具有可重构性，其检验标准如下：

(1)达到了用户需求；

(2)在系统错误(如节点失效)时仍能维持正常运转；

(3)系统中各节点负载较为平均，网络通信量也比较平均。

如果达到了以上标准，则称此系统具有可重构性。当然，以上3点也可作为重构算法的触发条件，即若有一条没达到，则触发重构算法，对系统进行调整。

2 分布式系统假设

我们所要构建的重构算法是基于以下假定的分布式系统：

如图1所示，这是一个集中式分布式系统，具有一个服务器和一个后备服务器，因此服务器崩溃的情况可不考虑。此系统基于以太网构建，所有的消息都是通过广播信道传输的，信道容量假定为10M/bps[2]。

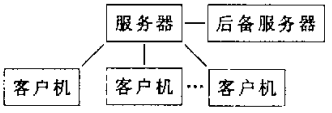

图1 集中式分布式系统图

服务器维护2张表格，一张是系统信息表，它记录了系统中所有节点的状态，格式如表1。

其中负载均衡值是本文中为了衡量一个节点的综合负载所设定的参数，它的计算公式如下：

负载均衡值＝CPU 负载 * 50%＋接口传输率 * 50%/信道容量；

它能较好的反映一个节点的 CPU 负载和网络拥塞率的综合状况。

表1 系统信息表

节点地址	是否有效	接口传输率	CPU 负载	负载均衡值
1	是	4M/bps	33%	36.5%
…	…	…	…	…
255	否	8.3M/bps	58%	70.5%

另外一张是系统进程表，它记录了系统中所有进程的分配状况和进程状态，格式如表2。

表2 系统进程表

进程号	所在节点	进程状态
1	38	未完成
…	…	…
255	235	挂起

进程有3种状态：未完成、完成、挂起，一旦进程完成即从此表中删去。此表保存了系统中所有未完成和挂起的进程信息，以供重构算法使用。

3 重构算法

重构算法主要分为两大部分：信息收集算法和系统重构算法。信息收集算法在服务器和客户端不间断的运行，每隔一段时间便收集整个系统信息，将其存入系统信息表。根据系统信息，若发现前面的三条标准之一没有达到，则触发系统重构算法。

系统重构算法根据出发条件分别进行处理，它将重新分配进程来消除节点失效，或通过转移进程来均衡系统负载，并同时记录各次的触发信息，每隔一段较长的时间给出统计总表，对系统硬件的更新换代给出建议。

3.1 信息收集算法

信息收集算法负责每隔一段时间收集系统信息，更新系统信息表，并检验触发条件是否满足，若满足，则触发系统重构算法。信息收集算法分为服务器算法和客户机算法，分别介绍如下：

3.1.1 服务器算法

服务器算法是一个如下的大循环：首先根据系统信息计算出一个时间间隔，然后延迟此时间段，随后对系统中每个节点发送信息查询消息，将接收到的信息填入系统信息表，如果某一节点超时没有返回消息，则认为此节点失效，将信息表中此节点对应的项置为无效。最后检查触发条件，如下条件之一若满足，则触发系统重构算法：

(1)用户敲入重构命令，或改变系统要求；

(2)有新增的无效节点；

(3)max {$(Li - Lj)|Li,Lj$ 为信息表中任意节点的CPU负载}>50%；

(4)max {$(Ni - Nj)$/ 信道容量 $|Ni,Nj$ 为信息表中任意节点的网络接口传输率}>50%。

若因为条件(2)、(3)、(4)没满足而触发系统重构算法，则将 i 作为参数传递过去。

3.1.2 客户机算法

客户机算法是一个如下循环：等待接收一条消息，若此消息不是传给自己的则丢弃，若是，则处理此消息。装配一个应答消息，检测自己的网络接口传输率及CPU负载填入此消息，最后将此消息发送给服务器。

3.2 系统重构算法

系统重构算法负责处理节点失效以及调整系统负载，并给出硬件升级方案，具体可分为3个部分。

3.2.1 节点失效处理算法

当检测到节点失效时，查进程表，找出所有此节点上未完成的进程，将其终止。然后检查系统信息表，根据负载均衡值对每个有效节点按从小到大排序，从中删去CPU负载或网络接口占空比(占空比＝接口传输率/信道容量)大于75%的节点，若还有剩余的节点，则将进程依次分配给这些节点；若没有节点剩余，则直接将进程分配给未删之前的节点序列。最后重新填写进程表。

3.2.2 处理系统负载不均的算法

若是节点 i 引起CPU负载不均，则检查进程表，找出 i 上所有进程，随机选出其中的一半。然后检查系统信息表，根据CPU负载值对每个有效节点按从小到大排序，从中删去CPU负载大于75%的节点，若还有剩余的节点，则将进程依次转移给这些节点；若没有节点剩余，则直接将进程转移给未删之前的节点序列。最后重新填写进程表。

若是节点 i 引起网络接口负载不均，则检查进程表，找出 i 上所有进程，随机选出其中的一半。然后检查系统信息表，根据网络接口负载对每个有效节点按从小到大排序，从中删去网络接口负载大于75%的节点，若还有剩余的节点，则将进程依次转移给这些节点；若没有节点剩余，则直接将进程转移给未删之前的节点序列。最后重新填写进程表。

3.2.3 长期重构目标算法

这要求系统维护一张系统重构触发条件表，格式如表3。

表3 系统重构触发条件

节点地址	失效次数	接口传输率过大次数	CPU负载过重次数	总次数
1	3	4	1	8
…	…	…	…	…
255	0	2	8	10

此算法要求每次系统重构算法完成后，根据触发条件重填此表。一段时间后根据此表列出所有信息，给出失效次数节点排序、CPU负载过重节点排序、网络接口传输率过大节点排序、整体最不稳定节点排序。以此作为硬件升级建议。

4 结束语

本文提出了一种分布式系统的重构算法，应用该算法能够使分布式系统达到可重构性。它保证了当分布式系统的某些部分失效时，整个系统能够继续正常运行，同时还保证调整系统负载，使CPU负载和网络通信量相对平衡。本文对于研究分布式系统有很好的参考价值。

参考文献

1 朱海滨. 分布式系统原理与设计. 长沙：国防科技大学出版社，1997. 15.

2 Andrew S. Tanenbaum. 计算机网络. 北京：清华大学出版社，1998. 209.

（责任编辑：黎贞崇）

（上接第156页）

* *

_

* *

_。

如此重复移位、比较，最多移至只剩8个二进制位，平均出现匹配的概率只有1/256。当出现匹配时再取部分低位指印数值做低位比较，这样可以跳过部分字符，提高搜索比较速度。

算法复杂性分析：

如果不考虑在指印匹配时字符匹配比较所需时间，时间复杂性应是$O(n+m)$，n为正文长度。本算法对KR算法中指印函数的改进提高了指印数值的生成速度，片段匹配可跳过部分字符，对搜索速度的贡献约为1倍。

参考文献

1 Richard M. Karp, Michael O. Rabin, Efficient randomized pattern-matching algorithms, IBM Research and Development, 1987, 31(2): 249～260.

2 苏德富，钟 诚. 计算机算法设计与分析. 北京：电子工业出版社，2001.

（责任编辑：蒋汉明）