计算机科学2004Vol.31№.12

实时监控系统中消息传递和同步算法的研究

胡金初

(上海师范大学计算机系　上海200234)

摘　要　实时系统是计算机应用的一个重要方面,分布式计算技术是解决实时监控系统在系统设计和应用方面的好办法。通过把分布在不同地域的资源整合,构成统一的计算框架来解决以前无法解决的问题,本文提出了在实时监控系统中进行分布式计算所涉及的一些技术和消息传递和同步算法。

关键词　算法,实时监控,进程,分布式计算

The Study of Message Delivery and Synchronization Algorithms in Real-Time Monitoring System

HU Jin-Chu

(Department of Computer Science Shanghai Normal University,Shanghai 200234)

Abstract　The real-time system is an important computer application. Distributed computing technology is a recent trend towards resolving the real-time monitoring system challenge at both system and application levels. By providing a set of services that allow a distributed collection of resources to be tied together into a relatively seamless computing framework,systems can collaborate to solve problems that they could not have attempted before. The paper presents current technologies and algorithms of message delivery and synchronization for distributed computing in real-time monitoring system.

Keywords　Algorithms,Real-time monitoring,Process,Distributed computing

1　引言

实时系统是计算机应用的一个重要方面,例如在电力监控系统中,用电单位需要对用电设备进行监测和控制以及对用电数据进行管理和统计。20世纪90年代后期,出现了多处理机的监控设备。以上海虹桥国际机场电力监控系统为例,它有南北两个变配电站,相隔约1200米,共有四路35kV的高压进线,降压成6000伏和380伏后分88路出线供其他部门使用,在用电规模和用电量上都属于很大的单位。由于机场的特殊性,对计算机系统的可靠性和用电的安全都提出了很高的要求。在使用中各自会产生大量的电压、电流、电功率、功率因数、电能量和负荷率等数据,如何将这些数据及时采集,迅速地处理并根据数据的结果对相应的设备进行快速的控制,以及对大量的历史数据进行有效的管理,是系统设计的主要问题。

2　系统结构和原理

电力监控系统控制的对象多、要处理的数据量大而且类型又很复杂,用一般的微型计算机很难胜任,需要用高性能的计算机、小型计算机,这就使监控系统的成本大幅度攀升,使用户在经济上难以承受。例如虹桥国际机场电力监控系统其分路数达到88路,要对其组成的所有状态空间进行搜索,不管用DFS或BFS方法,其计算量均十分惊人。在实际处理时,往往采用动态规划法,运用最优性原理,对状态树中不能产生最优解的子树不作搜索,从而大大加快了搜索的速度,即使这样计算机仍有相当的计算量。较好的解决办法是采用多处理机技术,用分布式处理代替集中式控制,在实现用户要求的前提下,使系统的总成本下降,性能提高。其系统框图见图1。

目前许多计算机应用系统开始由单机转向并行多处理机、由集中式处理转向分布式处理。应用系统的发展方向表现为:硬件在地理上趋于分散、软件在逻辑上趋于集中。其功能是分布式的,是通过资源分散配置来实现的,把处理功能、存储功能、传输功能分散到各个系统,软件的资源也分散到各个系统上,通过通信网络和软件把它们连成一个整体,真正实现多指令多数据流,并行地执行程序。在一次任务分配过程中,处理器 p_i 的执行时间可以用下式计算:

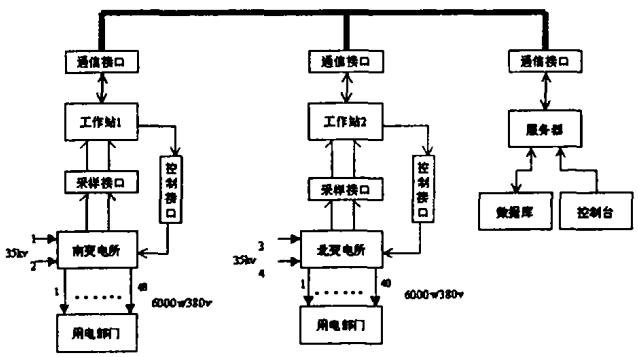

图1　系统框图

$$C(P_i)=\sum_{u\in V} w(u)\,|\,(M(u)=P_i)$$

其中 $M(u)=P_i$ 表示将任务 u 分配在处理器 p_i 上执行。对应处理器 p_i 的通信代价为:

$$T(P_i)=\sum_{(u,v)\in E} w(u,v)\,|\,(M(u)=P_i \cap M(v)\neq p_i)$$

本系统主要由两台工作站和一台服务器组成,一台工作站安放在南变电所内,另一台工作站安放在北变电所,服务器则放在控制室里,组成了在地域上分散的系统。两台工作站主要完成现场数据的采集和控制,服务器则和数据库相连,接受由工作站传送来的数据和处理要求,完成复杂数据的处理和信息的管理,分工合作完成整个系统的监控。

在电力监控系统中会遇到大量的数据库操作，我们采取将原始的数据送到服务器去加工处理，实现远程求值。整个监控系统是一个实时计算机数据处理系统，对数据处理的时间性要求很强，工作站是面向实时处理的。分布式系统中的工作站与服务器之间的进程通信采用基于 Send 和 Receive 原语的消息传递方式，允许程序调用位于其他机器上的过程。消息的传送和 I/O 操作是透明的，这种通信模型双方在不同的机器上，在不同的地址空间执行程序，两机的型号也不同，传送的参数和结果就更复杂。Send 和 Receive 原语将消息传送的细节都隐藏于两个库过程中，就像本地的库函数调用掩盖了系统的中断调用的具体细节一样。消息通常是用消息包或帧的形式发送的，源进程通过执行 Send 操作原语发送消息，宿进程则通过执行 Receive 操作原语获取消息有时候还通过 Reply 操作发给源进程一个回复。

在分布式系统中采用消息传递来完成进程之间的通信，本地的一个进程发送一个请求到远地的另一个进程，当这个请求被接受后就在远程的计算机上运行，然后向本地计算机返回执行的结果。这种客户/服务器方式执行的是请求/回答协议，也可以采用 RPC 来实现。基于消息传递的通信方式，一般采用数据包的通信服务，而不使用虚电路方式，发送方把每个要传输的分组，都给出完整的地址，根据路径算法分别寻址，传至目的地。在单机系统中，进程之间的通信和同步都是基于共享变量的方式进行的，但是在分布式系统中由于没有公共的存储器，基于共享变量的通信模式就不适合在分布式系统中使用。分布式系统是由多台计算机通过通信网络互连而成的，能够在系统范围内对单个问题进行合作，较少地依赖于集中的数据、硬件和过程。由于资源分散在各个不同的地点，因而进程调度、资源分配、系统管理等都必须满足分布处理的要求。各系统的处理机既能自主管理本地(局部)的资源，又能够接受和处理远程(异地)的资源请求，尽可能地达到各资源的均衡利用。

3 消息的传递和同步

当一个进程需要从一个文件中读取数据，而该读操作是一个远程过程时，首先使用一个客户桩模块。在读操作调用之后，客户桩模块把调用的参数包装在一个消息中，并调用远端的服务器桩模块，接着自己就处于阻塞状态，等待响应的到来。当消息到达服务器时，服务器桩模块把消息打开，调用服务器过程，就好象是被客户直接调用一样。服务器执行被请求的工作并把结果返回给调用者，调用完成后服务器重新获得控制，然后把结果打包后返回给客户进程。当消息到达客户端时，激活客户进程，把消息拷贝到缓冲区。客户桩模块解读结果并送给调用者。最后调用进程恢复控制，得到了所要求的数据。

用通信原语 Send 和 Receive 取代基于共享变量进程通信的“读”和“写”操作。目的机接受到一个消息，意味着完成了一次进程的通信，由于消息的发送在前，接受在后，从而实现了进程的同步。一个消息是由消息头和数据区组成。

Send(A,message) 表示发送一个消息到进程 A；

Receive(B,*message*)表示从进程 B 接受一个消息。

由于封锁原语有延迟性，实时性较差，分布式电力监控系统是典型的计算机实时应用，所以我们采用非封锁原语，满足系统对时间的严格要求。原语在执行过程中不延迟，在发送消息时，同时还可以执行，通过中断方式通知进程，从而提高系统的实时性能。

在集中式系统中，处于两个不同进程中的两个事件的发生顺序是容易确定的，因为在集中式系统中有统一的时钟和公共存储器。但在分布式系统中，各台计算机都是独立的，没有公共的存储器和时钟，所以要确定两个事件哪个先发生，哪个后发生，就必须要有特殊的方法。这里先定义：发生在先关系。

定义1 在同一个进程中的两个事件 a 和 b，事件 a 先执行，事件 b 后执行，则 a 相对于 b 发生在先，表示为 $a \rightarrow b$。

定义2 如果事件 a 是指某个进程发送消息，事件 b 是指另外一进程接收该消息，那么 a 相对于 b 发生在先，表示为 $a \rightarrow b$ 。

定义3 如果 $a \rightarrow b$ 并且 $b \rightarrow c$，那么 $a \rightarrow c$。

如果两个事件 a 和 b 不满足发生在先关系，那么在分布式系统中它们可以按任意一种顺序发生。确定两个事件哪个发生在先，在实时系统中尤其重要，因为许多消息的发送和接收都涉及到数据库的修改。一般数据库的修改都有一定的顺序，不能颠倒。譬如许多系统中的帐户操作，先在帐户 A 内存入￥200元，然后才能转帐￥100元到 B 帐户。

在分布式系统中进程是顺序执行的，在单个进程中执行的所有事件是有序的。同样，一个消息只能够先发送然后才能被接收。分布式系统中没有统一的物理时间，所以只能用近拟的方法，把两台计算机的时钟调整到一致。但是仍然会有几十毫秒的误差，这对于分析分布式系统中两个事件 a 和 b 谁发生在先还是有困难，见图2。图中的黑点表示进程中的事件。另外一般的线性时钟系统也不能区分是由于局部事件引起的时钟变化还是由于进程间的消息交换引起的时钟变化。为了解决这个问题，我们采用向量时钟。

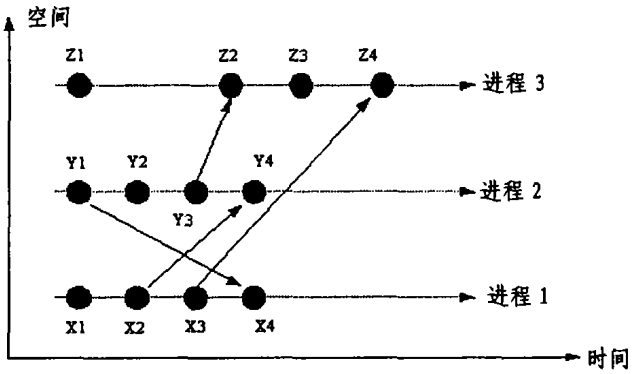

图2 分布式系统的时空图

在向量时间方法中，每个进程 P_i 和一个时间 $C_i[1..n]$ 向量相关联。定义下列操作方法：

方法1 在发生一个事件之前，先更新 C_i：

$C_i[i]:=C_i[i]+\Delta t \quad (\Delta t>0)$

方法2 当接收到一个消息(m,C_j,j)时，P_i 的更新操作为：

$C_i[j]:=\max(C_i[j],C_j[j]),(1\leqslant j\leqslant n)$

$C_i[i]:=C_i[i]+\Delta t$

每个消息都捎带发送方在发送时的向量时钟。C_i 被初始化为初值($\geqslant 0$)，并且它是一个递增的整数序列，进程 P_i 发出的每个消息 m 都被标上 C_i 的当前值和进程的标号 i，形成一个三元组(m,C_i,i)。对于不同的进程 C_i 可以取不同的初值。

图3表示带有向量时钟的时空图，根据每个事件的向量时钟，可以方便地确定发生在先的关系。物理时钟是连续运行的，而不像逻辑时钟那样以离散的方式计时。为了让时钟 C_i 成为真正的物理的时钟并使不同地点的两个时钟同步，可以定义准确速率和时钟同步两个条件分别如下：

$\forall i|dC_i(t)/dt-1|<a \quad (a<<1)$

$\forall i\forall j|C_i(t)-C_j(t)|<\beta \quad (\beta<<1)$

其时钟同步方法为：

(1)对每个 i，如果 P_i 在物理时间 t 时没有收到消息。则 C_i 在 t 可微并且 $dC(t)/dt>0$；

(2)如果 P_i 在物理时间 t 发送一个消息 m，则 m 包含了 $C_i(t)$。

(3)当在时间 t 收到消息 (m,C_j) 时，进程 P_i 设置 C_i 为 $\max(C_i(t-0),C_j+\mu)$，其中 μ 是预先定义的从一个进程发送消息 m 到另一个进程的最小延迟。

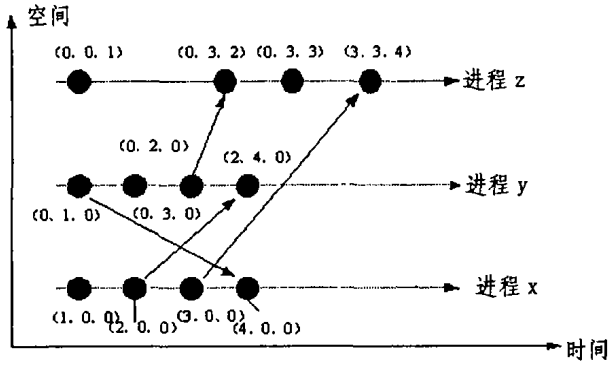

图3　向量时钟原理图

在以上算法中，$t-0$ 表示紧接在 t 之前的时刻，$dC(t)/d(t)>0$ 意味着时钟 C 总是往前走。在许多应用中，例如数据库管理系统，消息按照它们发送的顺序被接收是很重要的。如每个消息是一个数据更新操作，每个负责更新数据的进程按相同的顺序接收更新消息以维护数据库的一致性，要求对消息进行排序。

设 $s(m)$(或者 $r(m)$)表示从进程 P_i 到进程 P_j 的消息 m 的发送(或者接收)事件，$C(e)$ 表示事件 e 被逻辑时钟 C 记录下来的时间，如果下列条件为真，那么消息是有序的。

$C(s(m_1))\rightarrow C(s(m_2))\Rightarrow C(r(m_1))\rightarrow C(r(m_2))$

两个从同一个进程 P_i 发出的消息在接收方会发生乱序，这种情况可以用逻辑时钟解决，通过赋值和初始化，让逻辑时钟的值对应于进程的事件数，当接收方收到一个乱序消息时，它将一直等待直到该消息之前的所有消息到达。

当两个来自不同进程的消息到达接收方时乱序时，来自 P_1 的消息 m_1 比来自 P_2 的消息 m_2 先发出，但是迟收到。向量时钟可以解决这个问题。因为消息 m 中捎带着发送进程的时钟信息，接收者可以方便地区分哪个消息发生在先，接收进程在先收到消息 m_2 的时候，等待消息 m_1 的到来。所以为了保证分布式系统中消息的顺序，基本方法是只有前一个消息递交后才能接收下一个消息。方法如下：

(1)在发送消息之前，P_i 先更新它的 C_i；

(2)当 P_j 收到来自 P_i 的带 C_i 的消息 m 时，不立即递交消息 m，一直等到：① $C_j[i]=C_i[i]-1$；② $C_j[k]\leqslant C_i[k] \quad \forall k\neq i$。

条件①保证进程 P_j 已经收到所有来自 P_i 的消息，是 P_i 发出的。条件②保证 P_j 已经收到了所有那些在 P_i 发送消息 m 之前的消息。

用电的监测和管理是保证用电安全和节约用电的重要手段，采用分布式的计算机技术来加强和完善用电管理的工作，已引起人们的高度重视。研制和开发相关的产品也成了人们关注的问题。虹桥国际机场用电监测和管理系统经过一段时间的运行，工作可靠，性能良好。

参 考 文 献

1 Buyya R. High Performance Cluster Computing, Volumeland 2. Prentice-hall, Inc. 1999

2 Wu J. Distributed System Design. CRC Press LLC, 1999. 43～108

3 杨学良，等. 分布式多媒体计算机系统教程. 电子工业出版社，2002. 76～115

(上接第181页)

作系统的热插拔功能是系统高可用技术中重点考虑的问题。

操作系统通过专用函数接口调用 Firmware 监控层函数。其调用一般采用直接物理调用方式实现。

3.4　资源管理系统

用户通过资源管理系统来管理和配置逻辑分区。简单的资源管理系统只是在系统启动时通过配置不同的启动配置文件实现逻辑分区的设定。系统一旦运行后分区结构不可再变化。但在动态逻辑分区中，资源管理系统还要支持分区资源的动态在线配置能力：能够在无须重启操作系统的情况下对分区的 CPU、内存和 IO 设备进行随意的添加和删除，大大增强了逻辑分区的灵活性。动态逻辑分区需要资源管理系统和操作系统的紧密配合。

资源管理系统通过远程监视/控制实体、动态重构管理器和资源管理器实现动态的资源分配和去配。用户应用可通过操作系统提供的分区配置命令或函数来发起一次分区资源的在线重配置，也可由用户在远程监视/控制实体来发起。动态重构管理器收到请求后将要求操作系统释放硬件资源。在成功释放资源后，资源管理器将重新分配资源，并调用相应的操作系统命令将新资源加到正在运行的操作系统中。

发展趋势和存在的问题　同高可用技术相结合的动态逻辑分区技术是分区技术发展的主流趋势。同时，为了满足不同用户应用或同一应用中不同阶段应用模式的不同，于应用相密切结合的目前负载动态平衡的动态逻辑分区技术更是未来分区技术发展的热点。

由于目前的逻辑分区技术普遍采用了虚拟机技术，势必导致性能的损失。同时，大多数的动态逻辑分区技术主要是面向 SMP 系统的，针对 ccNUMA 系统的通信和访存不一致特性的动态逻辑分区优化技术尚不成熟。动态逻辑分区技术对操作系统有或多或少的影响，这对用户的安装使用十分不便。如何尽量少地影响操作系统是目前动态逻辑分区技术要研究的关键。由于高可用技术本身的不成熟，基于此的动态逻辑分区技术也并不成熟。实现一次动态的在线资源重分配性能低下，无法满足灵活变化的负载平衡需求，故针对高效、快捷的动态逻辑分区技术尚有待研究。

随着 SMP 技术和 ccNUMA 技术的不断成熟和完善，计算机系统的规模将变得越来越大，为了满足用户高可用和动态负载平衡，动态逻辑分区技术将越来越重要。

参 考 文 献

1 Dynamic Logical PARtitioning on the AIX-PPC Platform. http://www.research.ibm.com/DLPAR

2 The OpenVMS Galaxy Advantage. http://www.compaq.com/openvms/

3 hp-ux virtual partitions-product brief. http://www.hp.com/go/servicecontrol

4 Disco: Running Commodity Operating Systems on Scalable Multiprocessors. http://www-fash.stanford.edu/Disco

5 Sun Enterprise(tm) 10000 Dynamic Reconfiguration User Guide. http://www.sunmicro.com