计算机科学1999Vol. 26№. 7

超媒体系统中维护引用一致性的可扩缩体系结构

A Scalable Architecture for Maintaining Referential Integrity in Hypermedia Systems

高 尚 赵 宏 常桂然
(东北大学软件中心 沈阳110006)

Abstract This paper presents a scalable architecture for maintaining referential integrity in hypermedia systems and P-flood algorithm for the distribution of update information among the servers. This architecture resolves "dangling links" well in hypermedia space by grouping documents and links into surface types and core types and storing them separately.

Keywords Hypertext, Reference, Integrity, Hypermedia systems, P-flood, Scalability

使用 WWW 或是 Gopher 的网络用户可能遇到过这样的情况：点击一个超链接(WWW)或菜单选项(Gopher)，所指资源不可达。这可能是网络或服务器临时出问题，或是由于以上系统依赖 URL 来访问信息，引用者并不知道这些资源所发生的修改操作。这种现象叫做链接的挂起(dangling links)，引用资源发生修改操作导致一定数量的引用无效，而且这个数量会随着时间的推移有所增长。由于越来越多的文档会变得过时而被移走，这就需要一种手段来支持自动删除挂起链接，或是通知资源的引用者。

在 W3数据模型中，文档通过链接相连，链接直接保存在文档内部。这虽然有利于单一服务器的实现，但由于缺乏一个独立的链接数据库，限制了可链接文档的类型和高级用户接口形式，为维护整个 Web 的完整性带来困难。

Gopher 在本地保持了引用的完整性。删除(移动或修改)一个文档(服务器文件系统中的一个原文件)时，与之相对应的菜单选项随之修改。这种一致性是由底层的操作系统自动完成的，但是对远程服务器的引用仍不可靠 。

一些超媒体系统采用了链接数据库的方法来解决链接挂起问题，如奥地利 Graz 理工大学开发的 Hyper-G 系统，它使用一个数据库引擎来管理文档及其相互关系。文档和链接分别存放，对它们的修改只能通过 Hyper-G 服务器。链接数据库使得链接具有双向性，即可从目标找到源，反之亦然。为了在链接跨服务器时保持这种特性，双方服务器不仅要复制远程文档的抽象信息，还要保存链接信息。服务器双方都要执行对文档和链接的修改，以保持引用一致。因此，系统需要服务器间的信息修改协议。

1 可扩缩性概念

在评估一个分布式算法、系统或协议时，一个重要的因素是可扩缩性。理想情况下，一个可扩缩系统的行为不应该依赖于诸如服务器数量、文档、链接和系统用户数量这些变量。在 Internet 环境下，可扩缩性是系统设计的一个重要方面，因为上述变量的基数已经很大，而且增长的速度很快。系统的可扩缩性可分为四个方面：

1. 性能的可扩缩性　性能(以用户收到的响应时间来衡量)不应依赖于并发的用户数或文档数。一个集中式系统达不到这种要求。在分布式系统中，用户和文档分配给多个服务器，并通过网络相互连接。但是，由于某种原因，一些用户可能访问单一服务器上的一组文档。在这种情况下，分布式系统就象一个集中式系统，负载都加在单个服务器和某个网段上。为了避免这种现象的发生，可以使用复制(Replication)的方法：把特定资源的拷贝放在多个服务器上。应用复制机制的典型例子是 Usenet 的 news 服务。当用户读 news 时，它连接本地的 news 服务器，在服务器上保存着最新的文章。在用户阅读某一篇

高尚　博士研究生，主要研究领域为远程教育，超文本指导技术。赵宏　博士导师，主要研究领域为分布式多媒体及其支撑环境。常桂然　教授，主要研究领域为网络管理，应用及 ATM 技术。

文章时，服务器不必从文章的源地获取。因此，响应时间不依赖于在同一时间有多少用户在阅读同一篇文章（当然，它会依赖于连接到本地 news 服务器的用户数）。

搜索一定数量的文档时，响应时间随着被搜索文档的数量而增加。好的搜索引擎使用具有 O(log n)性能的数据结构：存在一个常数 C，搜索 n 个文档所花费的时间小于 $C * \log n$，即当 n 的值较大时，n 的增加对响应时间 t 的影响不大。所以在评价性能的可扩缩性时，一个具有对数性能的算法是可接受的。

2. 流量的可扩缩性 流量的增加并不与服务器的个数成线性关系。例如，Usenet 上的每篇文章都要发送到每个 news 服务器上，供本地阅读。复制产生了网络上额外的传输流量。但是，被发送的大多数文章可能没人去读。每个服务器定期向所有其它服务器发送状态信息的解决方案不是可扩缩的，因为它要求发送数量级为 $O(n^2)$ 的消息（n 指服务器的个数）。

3. 健壮性的可扩缩性 健壮性是指系统并不全部依赖于单个服务器或单个网络连接。在某个给定的时间，一般并不认为所有的服务器都是可达的。如果一定比例的服务器毁损并不影响系统的正常运行，该系统就具有可扩缩的健壮性。

4. 管理的可扩缩性 系统的功能不依赖于单个管理实体。例如，Internet 的域名服务（DNS）管理是分布的。截止到1997年1月，Internet 用户已达7100，000，000个，从 Internet 每年100%～200%的增长速度来看，集中管理是不可能的。这要求服务器间通信路径的配置是自动的，与集中式的服务完全不同。

在服务器个数、文档和链接个数方面，Gopher 和 W3的可扩缩性很好。在用户数量方面，存在可扩缩性问题。如果大量用户同时访问一个文档，涉及的网段和服务器就会过载。这种现象被称为瞬间拥挤"flash crowd"现象。

瞬间拥挤现象可通过使用快存服务器得到缓解。它对经常访问的信息保存本地拷贝，对用户请求提供本地拷贝，而不用到源地获取。但它不适合以下两种情况：①当用户访问大型数据时，需要把所复制数据中的 URL 变成 URN（Uniform Resource Names，统一资源命名），这样就可以通过名字识别文档，而不是位置。②当信息被频繁修改时，需要一定的修改协议，确保快存的更新。

2 维护引用一致性的一种体系结构

考虑到以上因素，奥地利 Graz 理工大学的研究人员提出了一种维护引用一致性的可扩缩体系结构，其突出特点是在每个服务器上维护一个链接数据库，其中记录着本地服务器上的所有链接和进出该服务器的链接。如图1所示，超媒体空间被服务器 A、B、C 所分割，跨服务器的链接称为表面链接（surface links），通过这些链接连向其它服务器的文档叫做表面文档（surface documents）。其它的叫做内部链接（core links）和内部文档（core documents）。表面文档和链接的集合要比内部文档和链接的集合小得多。

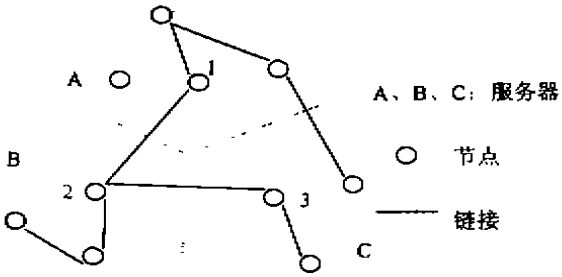

图1 服务器对超媒体空间的分割

表面链接的信息必须存储在所涉及的双方服务器上，以保持双向链接的特性。为了提高性能，服务器保存其它表面文档的元信息（meta-information）。图1中，服务器 A 保存文档1和文档2的抽象信息，以及它们之间的链接；服务器 B 保存文档2和文档1、3的抽象信息，以及1到2、2到3的链接。

这样，不同服务器上的文档就象在单个服务器上一样紧密相连。当要移走文档2时，服务器 B 会通知受影响的服务器 A（文档1）和 C（文档3）。相关的服务器就可以及时地修改受影响的文档和链接。

在弱一致（服务器在修改发生后的一段时间内达到一致）的前提下，可以采用一种扩散（flood）技术，它是信息发现系统 Harvest 的修改发布算法 flood-d 的基础，具有很好的扩缩性，产生的通信量不依赖于引用的对象数量。

Harvest 是一种基于 Internet 的资源发现系统，它支持一种有效的分布式信息收集体系结构："收集者"收集资源的索引信息；"中间人（Broker）"为收集到的信息提供一个索引查询接口，接受来自其它收集者或中间人的信息，不断修改它们的索引。

Harvest 主要依靠复制达到较高的性能。收集者生成的索引定期地复制给中间人。因为索引数量比较大，所以需要高效的信息发布算法。它使用了一种扩散技术：一个收集者并不是把它的索引发送给所有的中间人，它们只发送给 K 个中间人。其后，这

K 个节点把这些索引发布给其它 K 个，直到最后完成。传送的总索引数保持不变。可以看出，扩散技术有助于均衡网络和服务器的负载。

Harvest 所使用的扩散算法 flood-d 为了使网络消耗和传输时间达到最小，在计算底层物理网络带宽的基础上，寻找"低开销"的 K 连接逻辑拓扑图。这个图无需手工配置，会自动计算和修改。但找到这样一个优化的拓扑图计算耗费很大，尤其当复制的集合很大时。

3 P-flood 算法

flood-d 算法主要针对大量数据的传输，重点在于最小化扩散的开销，但在本文所述的维护引用一致性的可扩缩体系结构下，修改消息的数量并不多，而且强调：①速度：消息要迅速发布，减少不一致的时间。②健壮性：应保护每个消息传递给每个服务器。如果有服务器当时不可达，在它恢复之后，应能收到这个期间所错过的所有消息。③可扩缩性：通知所有相关服务器的时间、产生的通信流量不应过分依赖于服务器的数量。④自动化：自动配置扩散路径。⑤优先级：提供优先级参数，保证可接受的传输延迟和带宽消耗。

P-flood 通过改进 flood-d 算法，满足了上述要求。其中，P 可理解为消息的优先级。图2说明了 P-flood 算法的一个步骤。

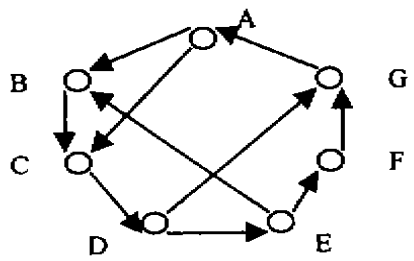

图2 P-flood 的一步

图中，服务器排成环行，每个服务器都知道其它的服务器。我们称环中每个服务器的前后邻居为其前驱和后继，如图服务器 E 的前驱为服务器 D，后继为服务器 F。服务器序列的修改也通过该算法进行发布。为简化分析，在讨论中我们作如下假设：信息服务器所在网络的底层通信协议为 TCP。算法描述如下：

1. 维护本地链接的完整性。

该操作相对简单。如内部文档或链接被移动或更名，可从链接数据库中查出该改动直接影响到哪些相关的文档和链接，然后沿着链接在本地修改这些信息源中与改动有关的引用部分。

2. 维护跨服务器的链接完整性。

修改消息可来自服务器自身（修改表面文档和表面链接的结果），或是来自其它服务器。它的产生依赖于被修改的文档或链接：对于文档，修改消息来自保存该文档的服务器；对于链接，修改消息来自发出该链接的服务器，即链接的源头。

（1）若一个服务器产生修改消息，首先查询其链接数据库，确定该修改操作会涉及到的服务器组，并产生一个相应的服务器通知列表。消息使用产生该消息的服务器序列号作为时戳。

（2）若发送的时间间隔 t 未到，服务器在修改列表中积累修改消息；否则，将修改列表发送给其它 P 个服务器（P 大于等于1）。

（3）当选中的服务器收到修改消息后，它将判断是否是第一次收到该消息。如不是，则抛弃该消息，并不作任何处理；如是，根据本地链接数据库内容和修改信息，进行相关文档或链接的修改。发送消息的服务器在消息成功地传递给后继之后，将它从修改列表中删除。收到的消息使用前一个服务器的序列号作为时戳。

（4）若其后继不是初始服务器，回到（2）步；否则算法结束。

该过程称为算法的一步（step）。如 P＝1，消息直接发送给它的后继。如果 P＞1且 P 为整数，则消息还要发送给其它 P－1个随机选择的服务器。如 P＞1但 P 不是整数，则消息发给它的后继，⌊P⌋－1个随机选择的服务器，再以 P－⌊P⌋的概率发给其它服务器。例如 P＝3.4，消息发送给后继，另两个随机选择的服务器，加上概率为0.2的另一个服务器。

图2表示 P＝1.5的 P-flood 算法的一步。在一步时间内，每个服务器都是并行的，它们的时钟不用同步，每步都有 P＊N 个修改列表发送出去（N 为服务器的数量）。

P 的值越高，消息到达所有服务器所需的时间就越短，产生的流量也就越大。可以对不同的消息赋予不同的 P 值。

在一个服务器不可达的情况下，它的前驱会保留相应的消息，直到它恢复。所以一个服务器发生故障的结果是它的前驱的修改列表变得很长。

如果 P＝1，则消息只发给后继，如果某一个服务器出错，会阻碍消息的发送。

4 P-flood 的扩展

为了进一步地提高 P-flood 算法的性能，可针

对不同的应用情况，对它进行扩展。

1. 服务器的组织。P-flood 的一个弱点是对网络随机使用，并不考虑底层的物理网络。它以两种方式选择消息的传播路径：非概率的（直接后继）和概率的（随机选取其它服务器）。随机性的大小由 P 参数控制。对于一个恰当的 P 值，大多数的流量在服务器所组成的静态环中流动，环中服务器的巧妙组织将大大减少网络开销和延迟，而且还保留着随机选择 flood 路径所带来的快速性和健壮性。

使用实际的带宽来衡量和计算最优环路比较困难。可以采用一种启发式的方法：根据 Internet 域名排序，从最后一个字母开始。这样，地点相近的服务器相互连接，节省传输开销。

2. 服务器的添加和删除。当添加或删除一个服务器时，P-flood 以高优先级通知所有的服务器。它们修改相应的服务器列表。当然要用旧的列表进行发布，直到收到后继的响应。

3. 毁灭性事件的恢复。在服务器中，有可能发生服务器毁损的现象。这时，要从信息库中恢复备份。如果备份是 i 天以前的，那么服务器就会失去 i 天以来的修改。

但是，其它服务器有可能有该服务器的表面镜像。在 i 天以内，这个服务器建立了一个指向其它服务器的新文档。现在，文档不存在了，链接也应该删除，以保持一致性状态。换句话说，其它的服务器也得返回到 i 天前的状态。

在这种情况下，该服务器发送一个特殊的消息，它包括自己的所有表面信息，请求其它服务器按照该消息修改有关自己的信息。

4. 修改不一致性，在某些情况下还有可能发生不一致。例如：需要添加一个链接从服务器 A 的文档指向服务器 B 的一个文档，但 B 上的文档以前不是在表面上的，而在添加该链接的消息到达 B 之前，服务器 B 删除了该文档。这样，当来自 A 的消息到达后，B 检测到了不一致，它将把删除的内部文档看作是表面文档，发送一个“文档移走”的消息给服务器 A。

也可以让所有的服务器定期地发送它们的表面信息，这样就可以处理各种不一致的错误。由于这些消息比较长，所以可以用低优先级发送。

总结 本文介绍了超媒体系统中一种可扩缩的体系结构和发布修改信息的 P-flood 算法。P-flood 算法的仿真表明该算法是可扩缩的、快速的，能够处理服务器和网络的暂时和永久错误。它适用于多数的超媒体系统，且已在 Hyper-G 系统中实现。

参 考 文 献

1 Kappe F. A Scalable Architecture for Maintaining Referential Integrity in Distributed Information Systems. Journal of Universal Computer Science, 1995, 1(2)

2 Lee Y_J, et al. A Systematic Approach to Design the Network-based Learning Environment for Home and Office. [Technical report Learn Tech_LCN. ps]. Department of Computer Science in University of Minnesota

3 Danzig P B, et al. Massively Replicating Services in Wide-Area Internetworks: [Technical report]. Computer Science Department, University of Southern California, 1994

4 Andrews K, et al The Hyper-G Network Information System. Journal of Universal Computer Science, 1995, 1(4)

（上接第82页）

基础上，分析了其流控问题以及产生的原因，并提出一种解决方法——无反馈流控法，这种方法既解决了网络流控问题，同时又保证了网络数据实时传输。无反馈流控法已成功地用于一国家重点工程中，并取得了满意的效果。由于1553B 是实时网络的典型代表，因此该文提出的解决1553B 实时网络流控问题的方法对其它实时网络也有很高的参考价值。

参 考 文 献

1 刘松强. 实时计算机系统. 北京：学苑出版社，1994

2 MIL-STD-1553 DESIGNER'S GUIDE. FORTH EDITION ILC DATA DEVICE CORPORATION.

3 Tanenbaum A S. Computer Networks Prentice Hall, 1996

4 Tanenbaum A S. Distributed Operating Systems. Prentice Hall, 1995

5 杨春. 嵌入式实时网络通信管理技术：[学位论文]. 成都：电子科技大学，1996

6 王志平. CRTOS/386R 实时网络软件的研究与实现：[学位论文]. 成都. 电子科技大学，1997