文章编号:1001 - 9081(2005)01 - 0052 - 04

Web 环境下 OLAP 对象池的研究与设计

杨庆跃,杨冬青,唐世渭,王腾蛟
(北京大学 计算机科学技术系, 北京 100871)
(qyyang@ db. pku. edu. cn)

摘 要:随着 Web 技术的发展以及 OLAP 服务在各种信息系统中的广泛应用,越来越多的系统允许用户通过 Web 方式访问 OLAP 服务。目前的实现方式存在着性能差、受并发用户数限制、用户状态难于监控等弊端,文中把对象池的概念引入到 OLAP 分析系统,提出了 Web 环境下 OLAP 对象池的设计策略,并给出了一个具体实现。

关键词:对象池; OLAP 服务; Web
中图分类号: TP311 **文献标识码**:A

Research and design of Web-based OLAP object pool technology

YANG Qing-yue, YANG Dong-qing, TANG Shi-wei, WANG Teng-jiao
(Department of Computer Science & Technology, Peking University, Beijing 100871, China)

Abstract: With the development of Web technology, accessing OLAP services through Web is a strong trend. The characteristics of accessing OLAP services based on Web were analyzed. In order to solve the problems of poor performance and license limitation, the concept of object pool was introduced into OLAP service systems. Due to the support of OLAP Object pool, web users can access the OLAP services easily and efficiently.

Key words: object pool; OLAP service; Web

0 引言

随着 Web 技术的快速发展,人们可以更加方便快捷地获取信息,最终用户对信息系统的访问已经不再局限于局域网内的 Client/Server 两层模式,而是可以通过 Web 服务不受地理位置限制地访问资源。特别是在电子商务和企业级应用中,Web 环境已经被广泛应用。

联机分析处理(OLAP)系统是基于多维数据集的分析决策工具,它以灵活的操作方式和多角度的结果展现能力为使用者提供快速应答,允许管理决策人员对数据进行深入观察。几乎所有知名的数据库系统以及数据分析系统的厂商都推出了自己的 OLAP 产品及工具, OLAP 服务已经成为各种信息系统的重要组成部分。

通过 Web 方式访问 OLAP 服务是信息系统的发展趋势,但同时也出现了一些相关的问题。本文分析了 Web 环境下访问 OLAP 服务的特点以及性能瓶颈,提出了在 Web 环境下利用对象池机制访问 OLAP 服务的策略以及实现过程中需要注意的问题,最后通过实例证明了该方式可以显著提高系统性能。

1 对象池的概念以及应用背景

资源管理是开发可伸缩系统时需要考虑的一个重要问题,无论是在简单的单机环境中还是在采用服务器集群的分布式系统中,资源总是有限的。为了有效利用这些有限的资源,对象池(Object Pool)的概念被广泛应用。

对象池的基本思路是:事先建立一定数量的对象并保存在内存中,当调用者申请对象时从池中取出一个空闲对象,使用后立即归还,供其他调用者重复使用,这样可以减少频繁创建、销毁对象所造成的开销。

因为客户端应用程序不需要重复地建立和销毁对象,使用对象池技术会显著提高连接性能。特别是对于占用资源比较严重的对象(例如数据库连接或网络连接),对象池机制可以有效减少建立对象所需的时间。除了可以提高连接性能以外,使用对象池还可以更有效地管理资源,对象池管理器可以根据需要动态调整池中的对象数量,使系统资源得到充分利用。通过对有限资源进行进一步的控制和管理,对象池可以使系统更容易扩展,当用户数量增加时只需要简单地增加池中的可用对象数量。对象池的这些优势对基于 Web 的应用程序来说尤其重要。

目前应用比较广泛的对象池是线程池和数据库连接池:

· 线程池:许多应用服务器,如 Web 服务器、数据库服务器、文件服务器、邮件服务器等都要频繁接受来自客户的短任务请求,如果采用每个请求开启一个进程的方式(thread-per-request)无疑将给服务器造成巨大负担,而利用线程池可以统一对要执行的任务进行合理的线程分配和调度, 有效减少开启过多线程对服务器造成的压力。

· 数据库连接池:事先建立若干个特定参数的数据库连接,放置在内存中,当有数据库访问请求时,不需要执行连接

收稿日期:2004 - 06 - 21;**修订日期**:2004 - 12 - 05
基金项目:国家 973 规划资助项目(G1999032705);国家 863 计划资助项目(2002AA4Z3440)
作者简介:杨庆跃(1975 -),男,硕士研究生,主要研究方向:数据库、信息系统; 杨冬青(1945 -),女,教授,博士生导师,主要研究方向:数据库、信息系统、海量信息集成与应用; 唐世渭(1939 -),男,教授,博士生导师,主要研究方向:数据库、信息系统、海量信息集成与应用; 王腾蛟(1973 -),男,讲师,博士,主要研究方向:数据库、信息系统.

数据库的操作,只需从连接池的空闲队列中取出已经打开的连接;数据库访问完成后,不是关闭连接,而是将连接放回连接池中,供其他数据库操作时复用,从而使整个系统性能得到提高。

2　Web 环境下使用对象池机制访问 OLAP 服务

OLAP 对象是重量级的内存对象,创建时需要与 OLAP 服务器连接、进行相关的验证操作以及初始化环境变量,除此之外,OLAP 服务器还要与数据源进行连接,这些操作都需要消耗系统资源。在传统 C/S 架构中,用户登录系统后新建一个 OLAP 对象,利用该对象保持与多维数据集(Cube)之间的连接并进行分析操作,当整个分析结束后,关闭这个 OLAP 连接并销毁对象。但在 Web 环境中情况发生了变化,用户访问 OLAP 服务的方式与传统访问方式存在较大差异,原因如下:

· 终端用户数量增加:原来 OLAP 系统只供内部管理人员和分析人员使用,但在 Web 环境下,OLAP 系统的使用者可能扩展到合作伙伴甚至是授权用户,如果给每一个用户都在服务器端分配并保持一个 OLAP 对象,不仅会给 OLAP 服务器和 Web 服务器造成过重负担,也容易受到 OLAP 服务的许可证限制。

· 连接协议的限制:用户到 Web 服务器之间采用 Http 协议连接,而 Http 协议不是面向连接的,服务器无法及时确认用户是否断线,这就可能造成 OLAP 对象在内存中空等待的情况,如果不及时清理,将严重影响服务器性能。

· 用户使用模式发生改变:传统的使用方式一般是用户登录后只进行单一的 OLAP 查询操作,多次查询的间隔相对比较密集,而在 Web 环境下,特别是把 OLAP 服务通过 Web 方式集成到企业信息系统中后,OLAP 分析功能与传统的数据查询、报表查询,以及其他各种业务功能同时呈现给用户,用户使用 OLAP 查询的间隔大大加大,极端情况是登录后只进行一两次 OLAP 操作,然后继续使用其他功能。

目前通过 Web 访问 OLAP 服务有两种实现方式:一种是用户登录后,通过 Web 服务器的 Session 机制在服务器端的内存中建立并保持一个 OLAP 对象,该用户的每一次操作都与这个 OLAP 对象相对应;另一种方式是对于每一次用户发出的 OLAP 查询请求都在服务器端建立一个新的 OLAP 对象,执行并返回结果后即关闭连接销毁对象。第一种方式的弊端是当终端用户数量增加时会对服务器造成巨大压力,同时也容易受到 OLAP 服务许可证限制,这使系统的可扩展性大大降低;第二种方式的弊端是用户进行每一次 OLAP 操作都要经过长时间的等待,违背了 OLAP 分析为用户提供快速响应的初衷,同时,频繁的建立、销毁对象也将大大消耗系统资源。

利用对象池机制访问 OLAP 服务可以有效地解决上述问题,但考虑到在 Web 环境中的访问特点,设计和实现对象池时需要注意以下问题:

· 对象池要求池中的对象是无差别的,即要求使用相同的初始参数建立对象,而目前常见的 OLAP 服务器都是针对特定的多维数据集(Cube)进行分析,所以在这种情况下需要根据多维数据集的数量分别建立相应的 OLAP 对象池,用户发出请求时必须在参数部分指定是对哪个多维数据集进行操作,以便对象池管理器为其分配相应的 OLAP 对象。

· 为了保证 OLAP 服务器上的数据安全,同时为了满足许可证限制,Web 环境下的用户一般要映射成为具有一定角色的 OLAP 系统用户,也就是说多个 Web 用户映射成为少数的 OLAP 系统用户进行操作,这就要求在应用的层次上合理控制 Web 用户与 OLAP 系统用户、OLAP 系统用户与多维数据集之间的对应关系。

· 对象池要求池中的对象是无状态的,不允许保留特定用户的环境上下文,所以要保证每次提交的 OLAP 操作请求都必须包含完整的查询条件。

3　Web 环境下 OLAP 对象池的实现

Web 环境下利用对象池机制对 OLAP 服务进行访问的系统结构如图 1,其中最重要的部件是对象池管理器,它负责管理多个 OLAP 对象池,每个对象池中包含若干个被封装起来的 OLAP 对象。为了使 Web 用户能有效地使用对象池,中间处理层也是必不可少的部件,它介于 Web 服务器和对象池管理器之间,负责处理 Web 用户的 OLAP 调用。中间处理层是对象池的实际使用者。

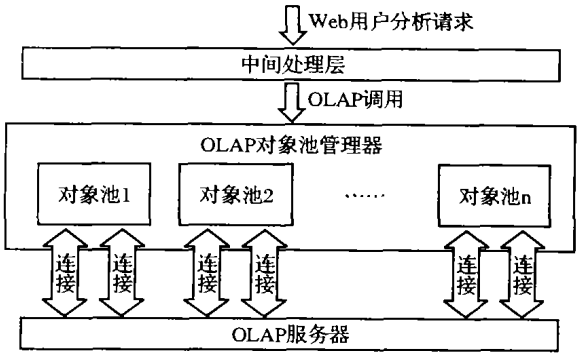

图 1　Web 环境下 OLAP 对象池的运行模式

3.1　对象池管理器 OlapPoolManager

对象池管理器作为一个统一的调用接口,管理多个对象池,每个池中包含性质相同的 OLAP 对象。对象池管理器负责处理调用者的请求,它根据调用者提供的多维数据集以及用户名判断为该调用者分配哪个池中的对象,并把对象的引用交给调用者。同时,对象池管理器负责监控各个对象池中的对象使用情况、动态调整池中对象数量、定时清理与服务器失去连接的对象。

对象池管理器要维护两类重要的数据结构,分别是 OLAP 对象池和 OLAP 对象,其中 OLAP 对象是将 OLAP 服务器中的分析对象进行必要的封装。

OLAP 对象池的主要属性:

· OLAP Pool ID:对象池标识,唯一标识该对象池;

· OLAP Pool Connection Parameters:对象池连接参数,包括每个对象池中对象的 OLAP 系统用户名、密码、所使用的多维数据集等;

· Init Size:对象池中对象初始数目;

· Max Size:对象池中对象数目上限;

· Min Size:对象池中对象数目下限;

· Current Count:对象池中当前的对象数;

· In Use Count:对象池中正在被使用的对象数;

· Wait Time:对象的最长空闲时间,当空闲时间超过这个设置的时候该对象可能与 OLAP 服务器失去连接,这种情

况下对象被强行释放；

· Using Tim：对象的最长使用时间，如果一个对象的状态标记为正在被使用，并且使用时间过长，可能是调用者没有归还该对象或发生其他意外，这种情况下对象也被强行释放；

· OLAP Object Set：OLAP 对象集合，属于该对象池的 OLAP 对象集合。

OLAP 对象的属性：

· OLAP Object ID：对象标识，唯一标识一个 OLAP 对象；

· In Use：是否被使用标志，标识该对象是否正在被使用；

· Begin Time：开始使用时间；

· Free Time：释放时间；

· Object Reference：对象引用，指向 OLAP 系统中的对象。

对象池管理器主要实现下列功能：

1）InitOlapPools 初始化 OLAP 对象池，负责创建不同参数的 OLAP 对象池，对象池之间最典型的差别是用户名和所使用的多维数据集不同。对象池的初始参数从属性文件中读取，属性文件可以根据需要修改，这也增强了对象池的可扩展性。

2）GetOlapObj 取得 OLAP 对象，按照调用者提供的参数（OLAP 系统用户名和多维数据集）从相应的对象池中取出 OLAP 对象返回给调用者，必要时新建一个 OLAP 对象。

3）ReturnOlapObj 归还 OLAP 对象，调用者使用完 OLAP 对象后交还给对象池，使该对象可以继续被其他调用者使用。

4）Extend and contract 扩展与收缩，根据系统负载动态增加或减少该池中的对象。

5）Deposit 清理对象池，根据 OLAP 对象使用时间判断该对象是否需要清除。需要考虑两种情况：如果一个对象很长时间未被使用，则可能已经失去与 OLAP 服务器的连接，将该对象销毁并重新建立一个；如果一个对象被某个用户长时间使用，则该用户可能失去与对象的连接，也将该对象销毁并重新建立一个。

6）Destroy 销毁，销毁所有对象，结束连接池管理器的运行。

3.2 中间处理层

中间处理层介于 Web 用户与 OLAP 对象池管理器之间，它主要完成下面的工作。

1）把 Web 用户映射为 OLAP 系统用户。映射规则是由系统管理员事先确定的。Web 环境下的信息系统用户数量众多，一般来说，由于系统资源以及 OLAP 服务器许可证的限制，让每一个 Web 用户对应到一个 OLAP 系统用户是不现实的，这就需要对 Web 用户进行分类，分类原则是该用户可以使用的 OLAP 功能以及可以操作的多维数据集等，每一类 Web 用户对应到一个 OLAP 系统用户。

2）取得一个活动的 OLAP 对象。把 1）中得到的 OLAP 系统用户名和多维数据集作为参数传递给对象池管理器，对象池管理器到相应的对象池中寻找空闲的 OLAP 对象交给调用者使用。

3）执行 Web 用户指定的 OLAP 操作。需要注意的是，在使用 OLAP 对象池的情况下，同一个 OLAP 对象的相邻两次操作很可能是由不同用户发起的，所以 OLAP 对象的环境上下文对调用者没有意义。因此即使是上卷、下钻等操作也都要由应用程序将对应的操作组织为包含维度、度量、过滤条件等信息的完整 OLAP 操作语句，之后再提交给 OLAP 对象执行。

4）将 OLAP 对象返还给对象池。此操作并不是真正销毁 OLAP 对象，而只是将该 OLAP 对象的使用状态标识为空闲，可以供其他用户申请使用。

图 2 是 Web 用户通过对象池管理器调用 OLAP 服务的顺序图。从图中看到，每一次的用户调用都需要中间处理层与 OLAP 对象池管理器的配合才能完成，并且每次使用的是随机得到的 OLAP 对象。但用户不必了解这些过程，他只是在需要进行 OLAP 分析的时候发出操作请求，并且可以在短时间内得到分析结果。

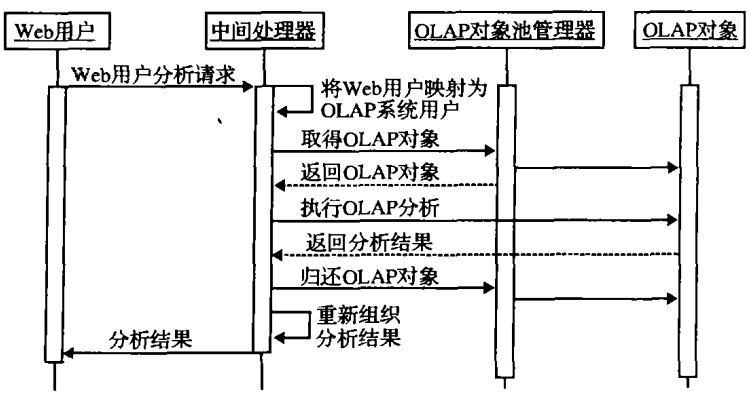

图 2 Web 环境下使用对象池访问 OLAP 服务的顺序图

3.3 一个 OLAP 对象池实例

上面是一个形式化的 OLAP 对象池框架，由于 OLAP 服务的使用环境以及产品之间调用方式的差异，实现通用的 OLAP 对象池还不现实。尽管 Windows 系统中的 COM + 和 J2EE 框架下的 EJB（Enterprise JavaBean）都提出了对象池的概念并且有相应的实现机制，但无法直接应用于 OLAP 服务。

我们利用该框架实现了一个运行于 Windows 环境下的 OLAP 对象池。选用 MetaCube ROLAP Option 作为 OLAP 服务器，数据存储采用 IBM-Informix RedBrick Warehouse 数据仓库系统。MetaCube 被实现为自动化服务器，它提供了多种对象用于 OLAP 分析，其中 Metabase 是主对象，它的从属对象包括 queries 和 filters 等，在具体实现中，把 Metabase 作为对象池管理的目标。OLAP 对象池管理器作为系统服务，以 COM 方式实现，中间处理层也以 COM 方式实现，由 ASP 脚本调用。浏览器端采用 OCX 控件向服务器发出请求并展现 OLAP 分析结果。

没有使用 OLAP 对象池之前，当 Web 用户初次使用 OLAP 分析服务时创建一个 Metabase 对象，并利用 Session 的方式保留在内存中，用户以后的分析操作都利用该 Metabase 对象进行，但这时的问题是：第一，初始建立 Metabase 对象的时间过长，在系统负荷较大时用户等待时间超过 15 秒；第二，这种方式非常消耗资源，同时在线的用户数量较多时系统需要建立并保留大量的 Metabase 对象，对 Web 服务器和 MetaCube 分析服务器都造成巨大压力；第三，服务器无法及时判断 Web 用户是否还在使用 OLAP 服务，导致一些失去连接的对象长期占用内存。此外，并发用户数量很容易达到 MetaCube 服

务器的限制。

使用 OLAP 对象池后,上面问题得到解决,系统整体性能显著提高。

1) 由于事先建立了一定数量的 Metabase 对象,用户可以在非常短的时间内得到 Metabase 对象进行 OLAP 分析操作,取得对象的时间从 15 秒的均值减少到 1 秒钟之内完成。

2) 通过共享一定数量的 Metabase 对象,可以满足更多 Web 用户同时使用 OLAP 分析服务,使系统具有良好的可扩展性。

3) 通过调整池中对象数量和定时清理有问题的对象进一步优化了系统性能,使资源得到高效利用。

4　结语

恰当地使用对象池技术,可以有效减少对象生成和初始化时的资源消耗,使多个调用者共享有限的资源,提高系统的运行效率。本文分析了 Web 环境下 OLAP 服务的特点以及性能瓶颈,把对象池机制引入到 OLAP 服务的访问过程中,提出了 Web 环境下 OLAP 对象池的总体设计策略,并给出了一个 Windows 环境下的具体实现。通过使用 OLAP 对象池技术,使系统整体性能得到显著提高。

参考文献:

[1] CODD EF, CODD SB, SALLEY CT. Providing On-line Analytical Processing To User-Analysts : An IT Mandate[R]. EF Codd and Associates, 1994.

[2] IBM Corp . DeveloperWorks : a worldwide resource for developers [EB/OL]. http://www-900.ibm.com.

[3] 刘润东. UML 对象设计与编程[M]. 北京:北京希望电子出版社, 2001.

[4] Application Programmer's Manual MetaCube ROLAP Option for Informix Dynamic Server[Z]. Informix Software Inc., 1998.

(上接第 51 页)

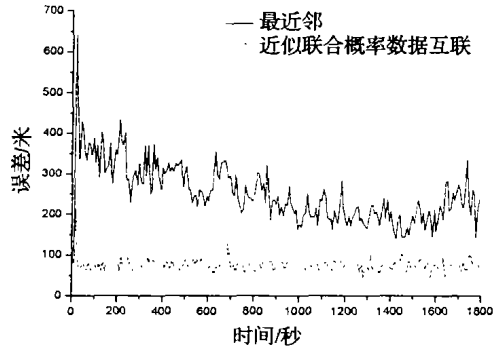

图5　20 批目标 4 部雷达两种算法误差比较

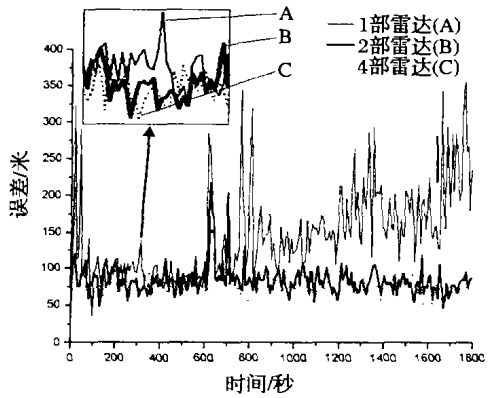

图6　20 批目标 AMSJPDA 法误差比较

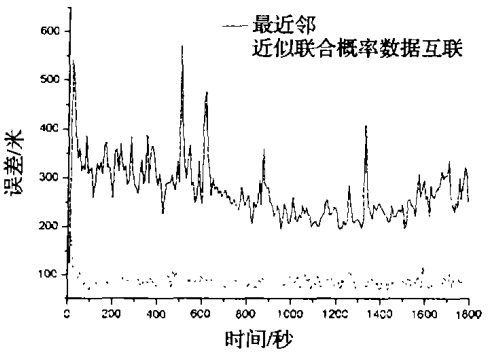

图7　60 批目标 4 部雷达两种算法误差比较

从仿真试验结果可以看出,如图 5 和图 7 以及表 1,AMSJPDA 比最近邻法要精确;从图 6 和图 8 以及表 2 中可以看出,多部雷达的融合效果要比单部雷达的融合效果好的多,但是单纯依靠提高雷达等传感器的数目来增加融合的精确性的做法其效果是有限的。由试验可以看出,在目标批次高,回波密度比较大的情况下,近似联合概率互联算法依然可以获得较好的融合效果。

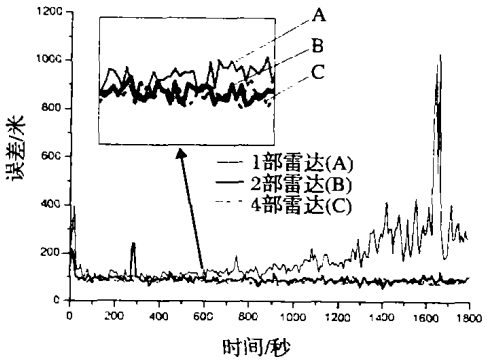

图8　60 批目标 AMSJPDA 法误差比较

4　结语

在真实战场情况下,很多因素对目标跟踪性能有影响。使用传统的 MSJPDA 算法在理论上可以得到较好的融合效果,但是,当战场环境较为复杂、目标回波密度很高时,传统的 MSJPDA 算法的关联计算量将大幅度提高,不能满足工程使用的要求。AMSJPDA 可以简化密集情况下的关联计算量,实时性较好,相比最近邻法融合精度也提高较多,适用于工程应用。

参考文献:

[1] ZHOU B, BOSE NK. Multitarget Tracking in Clutter: Fast Algorithms for Data Association[J]. IEEE Trans on Aerospace and Electronic Systems, 1993, AES-29(2): 352 - 363.

[2] PAO LY, FREI CW. A comparison of parallel and sequential implementations of a multisensor multitarget tracking algorithm[A]. Proceedings of the American Control Conference Seattle[C]. American Control Conference. Washington, 1995, vol 3. 1683 - 1687.

[3] 何友,王国宏,陆大绘,等. 多传感器信息融合及应用[M]. 北京:电子工业出版社, 2000.

[4] 康耀红. 数据融合理论与应用[M]. 陕西:西安电子科技大学出版社, 1997.