文章编号:1008-0570(2006)06-3-0042-03

线程池技术在网络游戏服务器中的应用

Application of Thread Pool Technique in The Network Game Server

(1.上海大学;2.西安电子科技大学)张　渊[1]　崔　滨[2]　余小清[1]　万旺根[1]
Zhang,Yuan　Cui,Bin　Yu,Xiaoqing　Wang,Wangen

摘要:在网络服务器的开发过程中,Pool (池) 的概念已经被广泛地应用。使用池技术可以明显地提高应用程序的速度,改善效率和降低系统资源的开销。正因为此,池的设计成为服务器设计中的重要组成部分。其中,使用最多,最为广泛的就是线程池。本文提出了一个线程池的设计模型和实现方法,并讨论了服务器程序利用线程池技术所能带来的好处。

关键词: 线程池; 网络游戏服务器; 并发服务
中图分类号: TP311.52　　**文献标识码**:A

Abstract :The idea of pool has been widely used in network server applications. The usage of pool technology can largely improve the server performance, reduce the cost of the system resource. So, the design of the pool has become an important part of the network server. Among them, the thread pool is most widely used. This paper explains the module of the thread pool design, and discuss the benefit of the thread pool.

Key words :thread pool ,game network server , parallel service

技术创新

1　引言

对于网络游戏运营商而言,在硬件上最大的成本投入就是庞大而又昂贵的服务器群了。由于接入的用户数量极其庞大,所以也需要大量的服务器来支撑网络游戏的运营。而如何能够用最少的硬件设备来处理同样多的用户请求就成了运营商最为关心的问题。本文针对如何能尽可能地发挥处理器的工作能力提出了一个线程池的模型,通过采用线程池模型,可以有效提高计算机的处理能力。

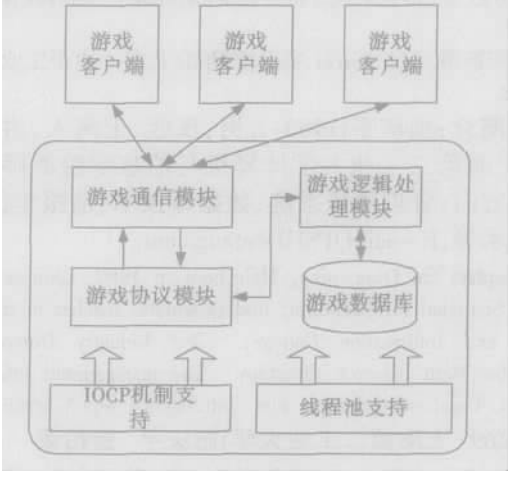

图 1　服务器结构

张渊:硕士
基金项目:上海市科委重大科技攻关项目
基金编号:NO.045115014

如图 1 所示,线程池模型贯穿于整个网络服务器结构之中,通过线程池模型加快数据包的处理速度,提高整个服务器的性能。

线程模型的动机是为了生成一个简单的操作系统抽象,能够让多个实体执行同一个程序,使用相同的文件和设备。线程替代了传统概念上的进程,是可调度的计算单位。一个进程可以有多个线程,兄弟线程间通过同一个进程相关联——即共享进程的程序和资源。线程是一个执行实体,使用的程序和资源属于相关联的进程。

对于单线程的处理结构,计算机执行某项操作而暂时没有所必需的资源时,计算机将会一直处于等待状态直到所需要资源准备好为止。而在多线程处理结构中,计算机可以将这段等待时间执行其他的线程操作。一些比较简单的服务器程序采用单线程的方式来处理大量的数据包,假设有 N 个数据包要被处理,每个数据包各自的执行时间为 a,b,c,d,e,那么在一个单线程的程序里,总的执行时间就是 t=a+b+c+d+e。

然而,每个数据包实际使用处理机的时间,只是总的执行时间中的一个很小的部分,事实上,N 个作业使用处理机的时间之和也不可能超过运行任一个作业所用时间的一小部分。也就是说,处理机的大量的运行时间是用在等待计算机准备好资源或者数据上了。而如果一个进程采用多线程的方式来处理数据包的话,处理机可以在线程 X 进行输入/输出操作时切换执行线程 Y。由于处理机处理时间远小于一个作业的

运行时间，通过几个线程的并行工作，总的处理时间将大大减少，大约为：maximum(a,b,c,d,e)。

对于网络游戏服务器，它对于处理的实时性有很高的要求，采用单线程的处理方式不但浪费了大量的CPU处理时间，而且对于游戏玩家而言，如此慢的响应时间也是不可忍受的。采用多线程的服务器结构是必然的事，但是数量繁多的线程操作也带来了程序的复杂和逻辑的混乱，频繁地建立与关闭线程，也将极大地减低系统的性能。所以人们提出了线程池的概念。

2 线程池基本原理

在传统服务器结构中，常是有一个总的监听线程监听有没有新的用户连接服务器，每当有一个新的用户进入，服务器就开启一个新的线程用户处理这个用户的数据包。这个线程只服务于这个用户，当用户与服务器端关闭连接以后，服务器端销毁这个线程。

然而频繁地开辟与销毁线程极大地占用了系统的资源。而且在大量用户的情况下，系统为了开辟和销毁线程将浪费大量的时间和资源。

线程池提供了一个解决外部大量用户与服务器有限资源的矛盾，线程池和传统的一个用户对应一个线程的处理方法不同，它的基本思想就是在程序开始时就在内存中开辟一些线程，线程的数目是固定的，他们独自形成一个类，屏蔽了对外的操作，而服务器只需要将数据包交给线程池就可以了。当有新的客户请求到达时，不是新创建一个线程为其服务，而是从“池子”中选择一个空闲的线程为新的客户请求服务，服务完毕后，线程进入空闲线程池中。如果没有线程空闲的话，就将数据包暂时积累，等待线程池内有线程空闲以后再进行处理。通过对多个任务重用已经存在的线程对象，降低了对线程对象创建和销毁的开销。当客户请求时，线程对象已经存在，可以提高请求的响应时间，从而整体地提高了系统服务的表现。

在这里，我们举个例子，假设一个服务器同时有1000个用户发送请求，并且每个用户请求需要一个单独的线程完成。比较利用线程池技术和不利于线程池技术的服务器处理这些请求时所产生的线程总数。在线程池中，线程数一般是固定的，所以产生线程总数不会超过线程池中线程的数目或者上限（以下简称线程池尺寸），而如果服务器不利用线程池来处理这些请求，则线程总数为1000。而一般线程池内的线程数目是远小于1000的。线程池在面对大量用户时优越性是相当明显的：

1）虽然外界可能有1000个用户，但不可能每个用户都有数据包需要处理，可能有许多用户地处理线程处于空闲状态，由于计算机采用的是时间片的调度方式，进程将处理机的允许占用时间平均分摊到每个线程之上的，就算有线程无事可做也要占用同样多的处理机时间。假设当前有200个用户有数据包发到服务器，而我们总共开了1000个服务于用户的线程。很明显，整整有80%的处理机时间被浪费了。而线程池的工作方式类似于时分复用，只要某个线程空闲就会有新的数据包处理，理论上不会有处理机时间被浪费的情况。

2）由于线程数目过多，频繁的线程间切换也将占用大量的执行时间，因为线程间的切换也是需要时间的，所以处理机真正在处理数据包的时间被缩短了。而因为线程池内的线程数目是远小于1000，相应的线程间的切换的次数和频率也将小的多。

3）过多的线程也将占用大量的内存和系统资源，导致系统资源匮乏，降低了服务器的处理能力。采用线程池结构开辟的线程较少，相应的占用的资源也较少。

一个简单线程池应该至少包含下列组成部分：

（1）管理线程，用于创建并管理线程池。

（2）工作线程，线程池中的线程。

（3）任务队列，用于存放需要被线程池处理的数据包，这是一个缓冲器，暂时保存未被处理的数据包。

3 线程池的实现

以下有一个线程池的例子，本文给出了管理线程的流程图：

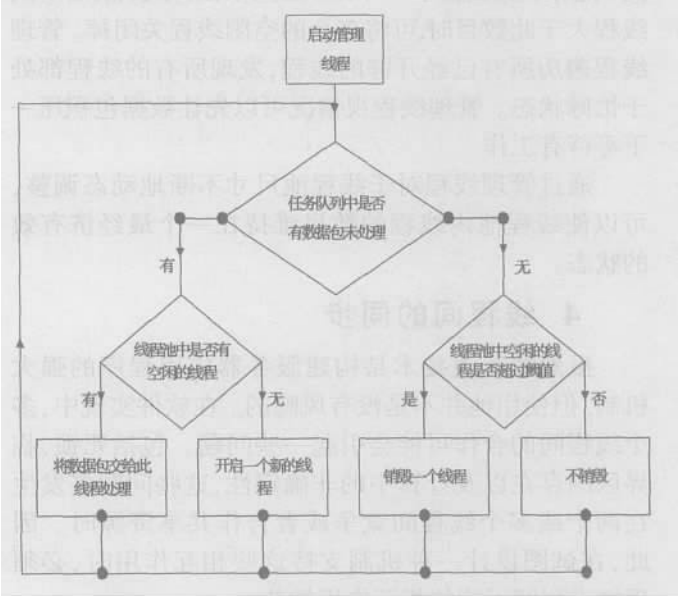

图2 管理线程工作流程

由服务器的主线程在初始化的时候开启两个线程：数据包接收线程和线程池管理线程，数据包接收线程负责接收网络上发来的数据包并放在任务队列中。线程池管理线程会首先执行初始化函数根据允许开启的工作线程总数开启相应数目的工作线程。

在工作中,管理线程会不断查询是否有工作线程处于空闲状态,当找到有空闲的工作线程,管理线程即发出信号通知相应的工作线程读取数据包并进行处理。

线程池有两种工作方式,一种是让空闲的线程自己去任务队列中读取数据包进行处理,管理线程只需要负责创建和销毁线程池就可以了。另一种是由管理线程负责将数据包分配给空闲的线程。

前者的好处在于管理线程的任务比较轻,整个系统的代码也相对比较简单。但是线程池缺乏管理,所有的线程自行其是,有可能导致管理线程无法准确地知道工作线程的状态, 无法估计当前线程池的负荷量,从而无法动态地调整线程池的尺寸。而后者虽然将导致程序的复杂化,但是由于管理线程的功能比较强大,可以有效的对于整个线程池进行管理,实现线程池尺寸的动态调整,当有大量的数据包还未被处理时,管理线程将再开辟几个新的工作线程进行数据包处理,而当未被处理的数据包很少时,管理线程将销毁几个线程以节约系统资源。

为了动态调整线程池的尺寸,达到对于资源更有效的利用。本文采用的是第二种方法。工作线程每处理完一个数据包就陷入等待状态,直到管理线程发信号通知它重新进入工作。

当需要处理的数据包很少,大批的线程都处于空闲状态,本文设置了一个最大的空闲线程数目,当空闲线程大于此数目时,可将多余的空闲线程关闭掉。管理线程遍历所有已经开辟的线程,发现所有的线程都处于忙碌状态。管理线程视情况可以先让数据包积压一下等待有工作

通过管理线程对于线程池尺寸不断地动态调整,可以使线程池内线程的数目维持在一个最经济有效的状态。

4 线程间的同步

虽然线程池技术是构建服务器应用程序的强大机制,但使用他并不是没有风险的。在软件实现中,多个线程间的合作可能会引起一些问题, 包括死锁、临界区的存在以及计算中的非确定性,这些问题多发生在两个或多个线程间竞争或者合作共享资源时。因此,在试图设计一种机制支持这些相互作用时,必须理解进程间可能的相互作用细节。

对于网络游戏服务器而言,可能会出现这么一种情况,本来某用户的数据包 x 被分派给线程 a,但当线程 a 在读取 x 时产生了定时器中断, 处理机跳入了线程 b, 结果线程 b 读取了本来分派给线程 a 的数据包 x,而管理线程已经认为数据包 x 给了线程 a,于是就会产生混乱。

为了避免产生这种情况,我们引入“锁”,使两个线程明确协同它们的活动。程序如下所示:

```
DWORD WINAPI WorkThread(LPVOID
lpParameter)
{
CCriticalSection section;
section.Lock();
……//读取共享变量
section.Unlock();
}
```

假设线程 a 进入临界区, 将锁设置为 TRUE 以后(section.Lock();)被中断,跳入线程 b,而当线程 b 想要读取共享变量时将被锁所阻塞。直到处理机重新跳入线程 a,并完成了读取工作后将锁设置为 FALSE(section.Unlock();),线程 b 才可以去读取数据。

如下图所示:

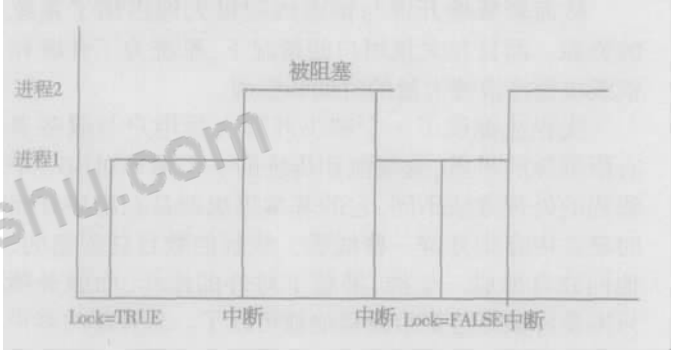

图 3 执行图示

5 结语

本文针对网络游戏的特点针对性地提出了一个线程池模型。它比较好地解决了网络游戏服务器具有的海量用户、数据包高度并发性和实时性高等要求。线程池技术可以显著提高服务器的性能,在网络游戏服务器方面具有巨大的应用价值。

参考文献:

[1]胡德斌.基于 cOS II 操作系统的嵌入式网络服务器的设计与实现[J].微计算机信息,2005,8- 2:24- 25

[2]Gary Nutt.操作系统现代观点[M] .孟祥由译. 北京:机械工业出版社,2004.

[3]JONESA,OHLUND J. Windows 网络编程技术[M]. 京京工作室, 译. 北京:机械工业出版社, 2000.

作者简介: 张渊(1982~)男,上海人, 硕士生, 主要从事网络服务器方面的研究。

(200072 上海市延长路 149 号 上海大学通信与信息工程学院)张渊 余小清 万旺根

(710071 陕西西安市太白南路 2 号 西安电子科技大学电子工程学院)崔滨

(投稿日期:2005.12.14) (修稿日期:2006.1.26)

相关推荐：

海峡那边的指挥官

ATA技术在网络游戏服务器中的应用

基于线程池技术DHCP服务器的设计与实现

线程池技术在并发服务器中的应用

线程池技术在网络游戏服务器中的应用

线程池技术在银行核心系统中的应用

线程池技术的应用

线程池技术研究与应用

基于线程池技术DHCP服务器的设计与实现

线程池在网络服务器程序中的应用