广西科学院学报
Journal of Guangxi Academy of Sciences
2012,28(1):28~30
Vol. 28, No. 1 February 2012

一种用 IOCP 模型提高网络设备管理系统性能的方法

A Method Using IOCP Model to Improve the Performence of Network Equipment Management Systems

王新宇
WANG Xin-yu

(广西瀚特信息产业股份有限公司,广西桂林 541004)
(Guangxi Hunter Information Industry Co., Ltd., Guilin, Guangxi, 541004, China)

摘要:在简要介绍 IOCP 模型实现方法的基础上,阐述一种在网络设备管理系统中使用 IOCP 模型的思路和方法,使得网络设备管理系统在各方面都能达到一个较好的性能水平。

关键词:管理系统 网络设备 IOCP 模型
中图法分类号:TP315 **文献标识码**:A **文章编号**:1002-7378(2012)01-0028-03

Abstract: Based on brief introduction of the methods for realizing the IOCP model, an idea and method using IOCP model in Network Devices Management System is stated. By introduce IOCP model, the performances of system is improved.

Key words: management systems, network devices, IOCP model

随着计算机网络通讯技术的迅猛发展,越来越多的行业应用或者方案都涉及或者使用到分布于各地的电子设备,而对于这些设备的集中管理则是必须解决的一个关键问题,特别是对于设备数量庞大或者设备之间相距很远的系统和应用,人工管理维护几乎成为不可能,网络设备管理系统[1]则应运而生。随着网络规模不断扩大,网络设备不断增加,网设备管理系统的性能也逐渐成为一个不能忽视的重要因素[2]。

IOCP 模型(I/O 完成端口)是 Windows 平台上性能最好的一种的 I/O 模型。IOCP 模型就是不断投递重叠 I/O 操作,将耗时的操作交给系统后台进行,应用程序便获得了解放,可以自由去做其他事情,当操作完成后,系统会通知等待操作结果的线程进行进一步的处理。IOCP 模型将最为耗时的 IO 操作交给了系统底层,大大提高了应用程序的 CPU 利用率。相对于其他网络通信模型来说,IOCP 具有系统资源占用少、不受 SOCKET 连接数限制、数据吞吐率高等优势。虽然 IOCP 模型也可以用在文件 I/O 等任何与 I/O 相关的场合,但是就目前来看,IOCP 更多的还是应用于网络通信等具有大量并发用户和大数据量吞吐的场合。本文在简要介绍 IOCP 模型的基础上,阐明在网络设备管理系统中使用 IOCP 模型的一种思路和方法,使得网设备管理系统在各方面都能达到一个较好的性能水平。

1 IOCP 模型的实现方法

实现 IOCP 模型首先要求用 CreateIoCompletionPort()函数创建一个 Win32 完成端口对象[3]。CreateIoCompletionPort()函数的原型为:

```
HANDLE CreateIoCompletionPort(
    HANDLE FileHandle,
    HANDLE ExistingCompletionPort,
    ULONG_PTR CompletionKey,
    DWORD NumberOfConcurrentThreads
);
```

创建完成端口对象的时候,该函数的前 3 个参数应该都传入 NULL,第 4 个参数传入工作者线程的数量。如果函数调用成功,将返回已创建的完成端口的句柄。

收稿日期:2011-12-19
作者简介:王新宇(1982-),男,助理工程师,主要从事物联网应用系统的研究。

要使创建的完成端口真正起作用,还需要将一个或多个设备与它进行关联绑定,完成端口只会向进行了绑定的设备发送操作完成的通知,此处的设备在这里就是指用于网络通信的套接字句柄,设备绑定依然是调用 CreateIoCompletionPort()函数,只不过这次前面 3 个参数必须传入有效的值。代码为:CreateIoCompletionPort((HANDLE) socket, completionPort,(ULONG_PTR) perHandleData, 0);其中 socket 就是用于网络通信的套接字句柄,completionPort 则是前面创建的完成端口句柄,而第 3 个参数 perHandleData 称作完成键。完成键实际上是一个指针,可以指向任何数据。使用 IOCP 模型一般都会自定义一个完成键数据结构,叫做"每句柄数据",系统会将它放到每个操作完成的通知中,这样应用程序可以通过它获得套接字相关的信息。在网络设备管理系统中作者使用的数据结构为:

```
typedef struct PerHandleData
{
    SOCKET socket;  // 套接字句柄
    String ip;  // 网络设备或客户端通信 IP
    uint16 port;  // 网络设备或客户端通信端口
};
```

创建完成端口并且将套接字句柄与其绑定之后,还需要创建线程来等待接收和处理完成通知,这样的线程叫做工作者线程。一般来说,工作者线程的数量等于 CPU 核心处理器数量的两倍再加上二,代码为:SYSTEM_INFO sysInfo;::GetSystemInfo (&sysInfo); threadCount = sysInfo.dwNumberOfProcessors * 2 + 2。工作者线程的任务就是不断调用 GetQueuedCompletionStatus()函数来获取完成通知,其函数原型:

```
BOOL GetQueuedCompletionPort(
    HANDLE CompletionPort,// 完成端口句柄
    LPDWORD lpNumberOfBytes,// 用于接收 I/O 操作成功传输的字节数
    PULONG_PTR lpCompletionKey,// 完成键,指向与套接字相关的每句柄数据
    LPOVERLAPPED * lpOverlapped,// 重叠结构指针,指向每 I/O 数据的指针
    DWORD dwMilliseconds// 等待超时
};
```

上面参数中出现的重叠结构是应用程序用来跟踪重叠操作的基础。当应用程序调用一个重叠操作(如发送或接收)时,要传入一个指向对应重叠结构的指针,当操作完成后,应用程序可以通过 GetQueuedCompletionStatus()函数来拿回这个指针。不过,系统定义的 OVERLAPPED 结构主要用于后台的操作,在应用层没有包含任何有意义的数据,因此,应用程序要实现对重叠操作的跟踪,必须自定义一个数据结构,即每 I/O 数据,除了必须包含一个 OVERLAPPED 数据外,其他数据信息均可以自定义,工作者线程在获取重叠结构时只要做一个强制转换就可以使用自定义的每 I/O 数据。在网络设备管理系统中作者定义的每 I/O 数据为:

```
typedef struct PerIoData
{
    OVERLAPPED ol;//重叠结构
    int op;//操作码,指示本次操作是主动连接、被动连接、发送数据还是接收数据
    WSABUF wsaBuf;//WSA 缓冲区结构
    uint8 buf[BUFFER_SIZE];//真正的数据缓冲区
    size_t len;//传输的字节数
};
```

每 I/O 数据与每句柄数据是有区别的,每 I/O 数据与每一次 I/O 操作相关联,而每句柄数据与套接字相关联。在一个套接字上进行的所有操作都会返回同一个每句柄数据,而每个操作返回的每 I/O 数据都不同。

2 IOCP 模型在网络设备管理系统中的使用方法

一个网络设备管理系统,一方面要在底层实现对网络设备的管理和监控,另一方面要在应用层让用户能够直观地看到网络设备的运行状态和对网络设备进行控制和设置。在这个网络飞速发展的时代,网络设备管理系统的主流架构也一般是采用 C/S 或者 B/S 架构,而无论选择哪种架构,都必须有个 S 端,也就是服务端,而 IOCP 模型则可以作为服务端的底层网络通信模型。

为了使用的方便以及代码的重用,作者在网络设备管理系统中将 IOCP 封装成一个服务器类 IocpServer,提供连接、监听、发送、接收等基本网络通信功能。同时以该类为父类派生出一个 ManagerServer 子类,用于实现服务端系统的管理功能。在本系统中,网络设备的管理与客户的管理都在 ManagerServer 子类中进行,同时因为网络设备与

客户端都可以使用IP和端口来进行唯一识别，因此每句柄数据和每I/O数据使用上文定义的数据结构就可以满足需要。

IOCP模型实现的一个难点在于资源的释放，因为套接字可以同时进行发送和接收操作，也就是系统后台在同一个套接字上可能同时存在两个重叠操作，而应用程序无法预知这些重叠操作什么时候能够完成，因此资源释放的时间点往往难以把握。针对这个问题，作者使用资源池加引用计数来解决。首先看资源池，为每句柄数据和每I/O数据各创建一个资源池，并且预分配一定数量的资源，当应用程序请求每句柄数据或者每I/O数据的时候，就从对应的资源池中获取一个空闲的资源，同时该资源状态变为非空闲，如果资源池中已经没有任何空闲的资源，则重新分配预定义数量的资源来供给；当应用程序释放每句柄数据或者每I/O数据的时候，并不真正地释放其所占用的内存和系统资源，而是将它放回对应的资源池中并重新标记为空闲状态，以便下一次重新分配使用。资源池的使用使得对每句柄数据和每I/O数据的管理和跟踪更加便利，同时因为使用了预分配内存的方式，大大减少了内存分配和释放所带来的运行开销。资源池起到了管理的作用，但是真正的难题——资源释放的时机还是没能解决，这就需要用到引用计数机制。对于任何一个每句柄数据和每I/O数据，都有一个引用计数值，默认为0，在该资源被分配和每一次使用之前，该引用计数值都必须加1，当应用程序请求释放资源的时候，不是立即进行释放动作，而是首先将引用计数值减1，然后判断该值是否为0，如果不为0，说明该资源仍然在程序的其他地方被使用，不能进行回收释放，如果为0，则说明该资源已不再被使用，可以安全地回收释放。这就解决了资源释放时机的问题，只需要在应用程序认为应该释放的所有地方都进行释放请求，剩下的问题引用计数将帮我们全部解决。

因为在本系统中设备与客户都在ManagerServer中进行管理，所以需要2个列表来进行保存，在这里使用的是STL中的map数据结构，其内部是以红黑树来实现的，其根据关键字搜索的时间复杂度为O(lgN)，当设备或者客户数量比较大的时候，得到的效率提升是相当显著的。本系统中的关键字就是设备与客户的IP地址，当接收到一个完成通知时，首先根据发出完成通知的对象的IP地址，分别在2个列表中进行搜索，以确定是设备还是客户的操作，进而进行相应的处理。

3 结束语

将IOCP模型应用在网络设备管理系统中，提高了应用程序与设备之间数据交换的效率，提高了监控的实时性，同时使得网络通信底层在整个网络设备管理系统中只占用很少的系统资源，为上层系统提供了稳定高效的网络通信支持，使得整个网络设备管理系统在各方面性能上都提高了一个台阶。

网络的高速发展使得各种网络设备管理系统也越发庞大和复杂，对底层网络通信模型也提出了更高的要求，IOCP模型以其独有的特点和优势，在这方面逐渐成为系统设计人员的首选。

参考文献：

[1] Anthony Jones, Jim Ohlund. Networking programming for microsoft windows second edition[M]. 2nd ed. USA: Microsoft Press, 2002: 145-169.

[2] 朱庆弦，张杰，张骏温. 网络管理技术的发展趋势[J]. 电视技术，2005(12)：54-56.

[3] 吴永强. 网络管理技术的应用与发展[J]. 辽宁农业职业技术学院学报，2004(1)：52-53.

（责任编辑：邓大玉）