Linux 系统的线程技术*

袁建红

(厦门市工业学校,福建 厦门 361006)

摘要:Linux 系统支持核心级和用户级线程,采用线程可以提高系统效率,减少系统开销.同样,用户利用线程技术也可编写出高效率的应用程序,实现多任务的并发执行.

关键词:线程技术;系统调用;同步;信号量

中图分类号:TP 361.89 **文献标识码**:A **文章编号**:1004-2911(2003)04-0389-03

传统的 UNIX 进程概念在开发分布式系统中的许多应用时已经显得力不从心,有时甚至连简单的窗口响应问题都很难做好.解决这些问题的最好方法就是线程,线程推广了进程的概念,使一个进程可以包含多个活动.在现代操作系统(如 Windows、UNIX、Linux)中已经普遍实现了线程机制.使用线程的优点在于:改进程序的实时响应能力;更有效的使用多处理器;改进程序结构,使多个线程共享同一地址空间;减少对系统资源的使用(线程的切换要比进程快几个数量级,尤其是用户态线程).

1 操作系统线程技术现状

目前,许多流行的多任务操作系统都提供线程机制,按照系统管理策略,线程可分为用户级线程和内核级线程两种基本类型.用户级线程不需要内核支持,可以在用户程序中实现,线程调度、同步与互斥都需要用户程序自己完成.内核级线程需要内核参与,由内核完成线程调度并提供相应的系统调用,用户程序可以通过这些接口函数对线程进行一定的控制和管理.相比之下,用户级线程具有更高的灵活性,它可以由用户程序创建并进行管理和控制,以完成特殊的应用要求.例如多媒体实时过程、互连网数据下载、数据采集等.

现在对线程库的定义规范有 3 种:WIN32、OS/2 和 POSIX .其中前两种都是专用的,只能用在它们各自的平台上,即 WIN32 线程仅能运行于 WINNT 和 WIN9X 平台上,OS/2 线程运行于 OS/2 平台上.POSIX 规范(IEEE1003.1c aka Pthreads)则是适用于各种平台,而且已经或正在所有主要的 UNIX 系统(包括 Linux)上实现.Solaris 线程是太阳微系统公司在 POSIX 委员会完成标准指定工作之前开发 Solaris2 所采用的线程库.尽管现在大部分程序员都将采用 POSIX 标准,但 POSIX 线程库和 Solaris 线程库仍是目前使用最广泛的线程库.这两个线程库的功能相似,并可以用在同样的应用中.但是只有采用 POSIX 标准的线程才能确保可以完全的移植到其他符合 POSIX 标准的环境中.

2 Linux 系统线程技术

由 Xavier Leroy,以及 Pavel Krauz,Richard Henderson 等人开发的,基于 Linux 下的一种线程库实现了一个叫"一对一"模型(One-to-One).Linux Threads 采用称为 One-to-One 的模型,每个线程实际上在核心是一个个单独的进程,核心的调度程序负责线程的调度,就象调度普通

* 收稿日期:2003-06-09

作者简介:袁建红(1964-),男,讲师,陕西宝鸡人,从事操作系统及计算机网络方向研究.

进程.线程是用系统调用 clone()创建的,clone()系统调用是 fork()的推广形式,它允许新进程共享父进程的存储空间、文件描述符和信号处理程序.除了"一对一"模型之外还有两种基本模型."多对一"模型是基于用户级的调度,线程切换完全由用户程序完成,从核心角度看,只有一个进程正在运行."多对多"模型结合了核心态和用户态的调度,数个核心态线程并发的执行,每个都作为一个用户态线程的调度者对用户态线程进行调度.

Linux 核心对线程的支持主要是通过其系统调用实现的,系统调用—clone()用于进程创建,实现于 Linux 2.0 或以上版本,并可以运行于 Intel,Alpha SPARC,m68k 以及 MIPS 等处理器的机器上.

系统调用 clone() 的代码如下:

```
asmlinkage int sys-clone(struct pt-regs regs)
{ unsigned long clone-flags;
unsigned long newsp;
clone-flags = regs.ebx;
newsp = regs.ecx;
if (! newsp) newsp = regs.esp;
return do-fork(clone-flags, newsp, ? s);
}
```

事实上,Linux 的 clone()系统调用与有 fork()系统调用在核心上都是通过 do-fork 函数来实现的.clone 与 fork 的区别在于传递了几个参数,而当中最重要的参数就是 conle-flags.在 Linux 系统中,线程共享资源是可以控制的,clone-flags 参数标识了线程可共享的父进程的那些资源.clone-flags 标志值的含义如下:

CLONE-VM 共享进程的地址空间;CLONE-FS 共享进程的文件系统信息;CLONE-FILES 共享进程打开的文件;CLONE-SIGHAND 共享进程信号处理程序;CLONE-PID 进程标识号.

Linux 系统支持内核级多线程.引入多线程的主要目的是减少系统开销,得到更大的系统吞吐量和更高的效率,使程序员可以编写并发程序.多线程在共享资源,尤其是内存空间时,需要解决多线程间的同步与互斥问题.从原理上讲,可以使用与进程同步、互斥同样的机制,如信号量机制.

下面是采用线程信号量实现两个线程同步的实例:

```
#include<pthread.h>
#include<stdio.h>
#include<semaphore.h>

void *thread-function(void *arg);
char message[10];
sem-t sem;

int main()
{
int tid;
pthread-t a-thread;
pthread-attr-t attr;
//初始化线程属性
pthread-attr-init(&attr);

tid=sem-init(&sem,0,0);
if(tid! =0)
{
perror("Semaphore initialization failed");
exit(1);
}

//创建另一线程
tid = pthread-create (&a-thread, &attr,
thread-function,message);
if(tid! =0)
```

```
{
perror("Thread create failed");
exit(1);
}

//输入串
printf("please input a string:\n");
scanf("%s",message);
sem_post(&sem);
sleep(1);
}

void *thread_function(void *arg)
{
printf("thread_function is waiting for input\n");
sem_wait(&sem);
printf("string of output is:\n");
printf(message);
printf("\n");
}
```

执行结果:

```
please input a string:
thread_function is waiting for input
[键盘输入] abcde
string of output is:
[打印输出]abcde
```

该例定义一个共享字符串,一个线程接收用户输入,然后通知另一个线程;另一个线程接到通知后,显示该字符串,否则显示正在等待接收.

在多线程编程中,除了要处理同步与互斥外,更重要的是如何使多线程相互合作.在采用多线程编程完成某一任务时,每个线程都只完成任务的一部分,只有密切地相互合作才能最终完成任务.目前,使用多线程的系统很多,例如:多线程下载程序、多线程数据采集系统、多线程数据查询等.

线程技术是现代操作系统最重要的功能之一,也是程序员开发高性能并发程序的得力工具.随着对线程技术研究的不断深入,随着 POSIX 标准的不断完善,线程技术以及多线程程序设计都将成为操作系统与编程人员最常用的方法和手段.

参考文献:

[1] 汤荷美,董　渊,李　莉,等.Linux 基础教程[M].北京:清华大学出版社,2001.36—38.

[2] 赵敏哲.64 位 Linux 操作系统与应用实例[M].北京:机械工业出版社,2001.33—36,66—69.

[3] 怀石工作室.Linux 上的 C 编程:2[M].北京:中国电力出版社,2001.47—74.

Thread technology of Linux system

YUAN　Jian—hong

(Xiamen Industry School, Xiamen Fujian　361006, China)

Key words:　thread technology; system call; synchrocyclotron; semaphores