

# 操作系统

Operating system

徐子川

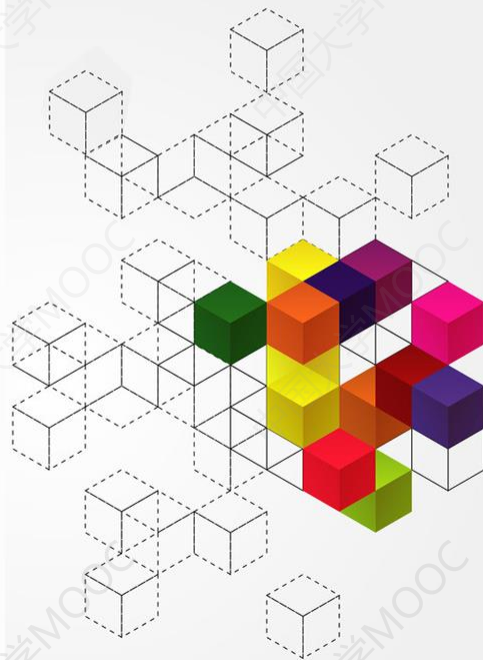
大连理工大学

一、线程分类

二、几种典型的线程模型


三、Linux线程模型

四、Solaris线程模型




# 一、线程分类

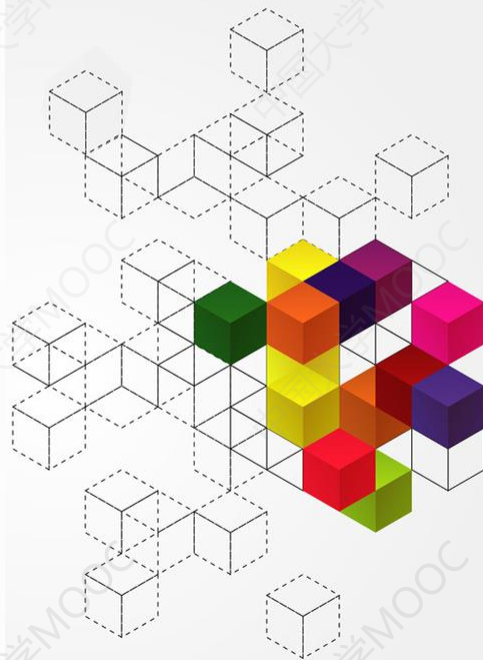
## 用户级线程

- 
- 在用户态以线程库的形式实现
  - 对用户级线程的操作通过调用用户态线程库API进行

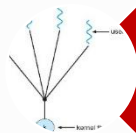
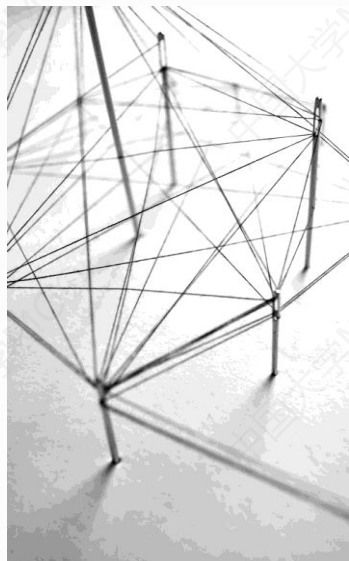
---

## 内核级线程

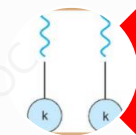
- 
- 在内核态实现，OS内核直接管理
  - 线程的创建由系统调用完成



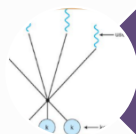
## 二、几种典型的线程模型



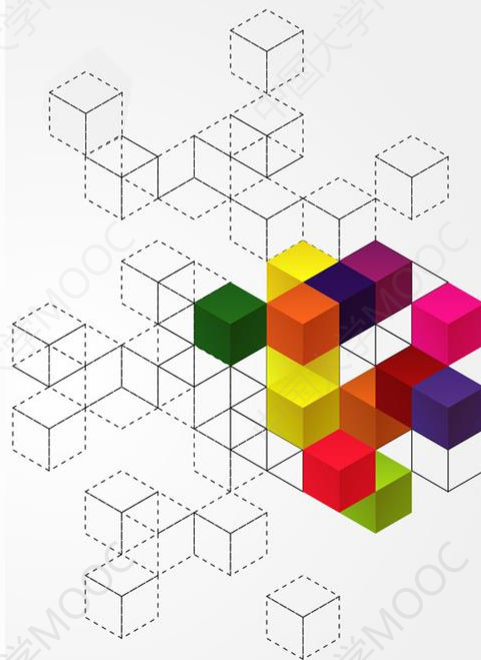
M:1



1:1

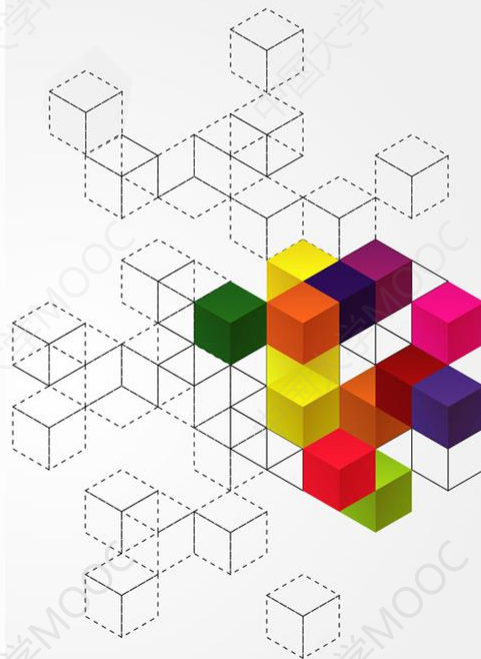
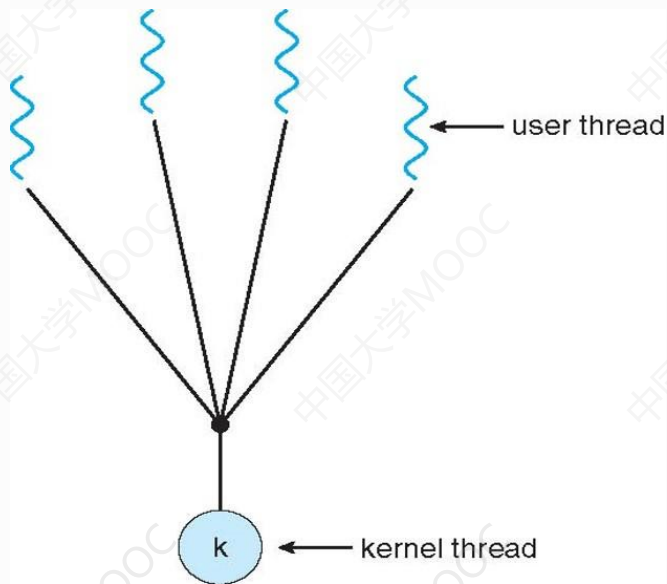


M:N



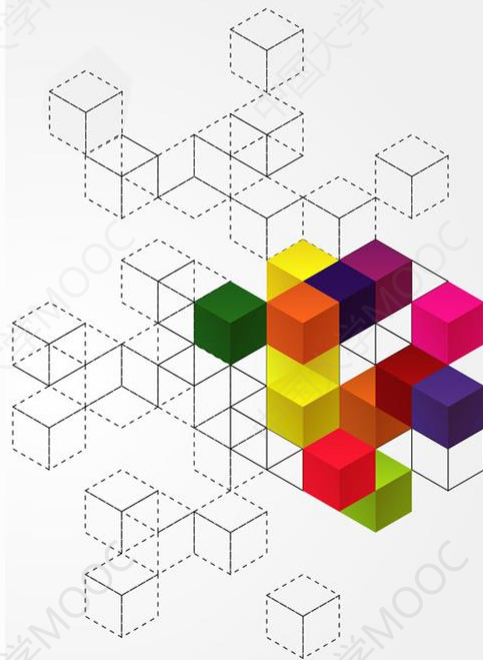
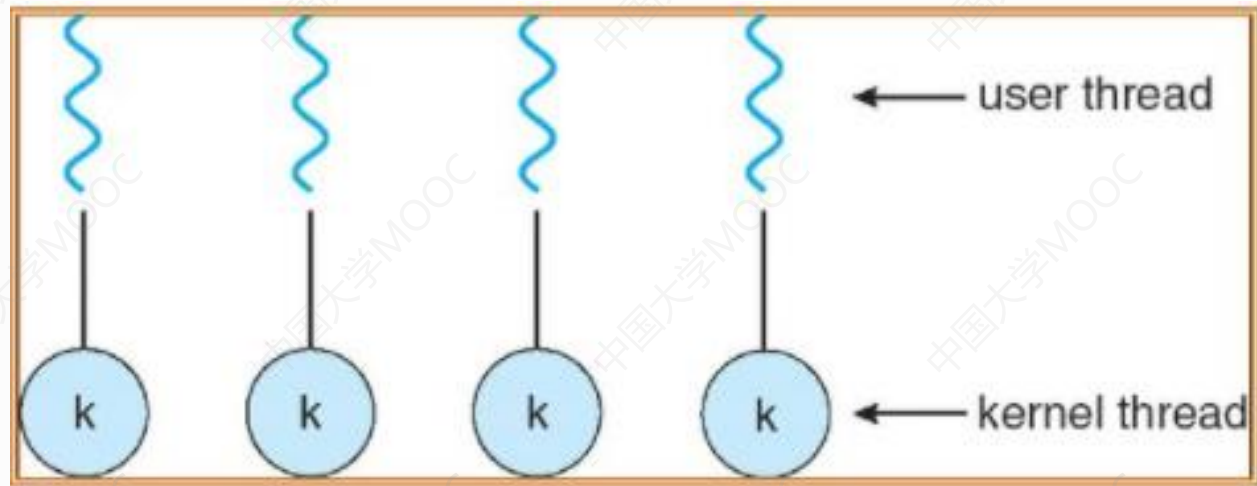
## 二、几种典型的线程模型 – M:1模型

- 多个用户级线程绑定到一个内核级线程(M:1)



## 二、几种典型的线程模型 -1:1模型

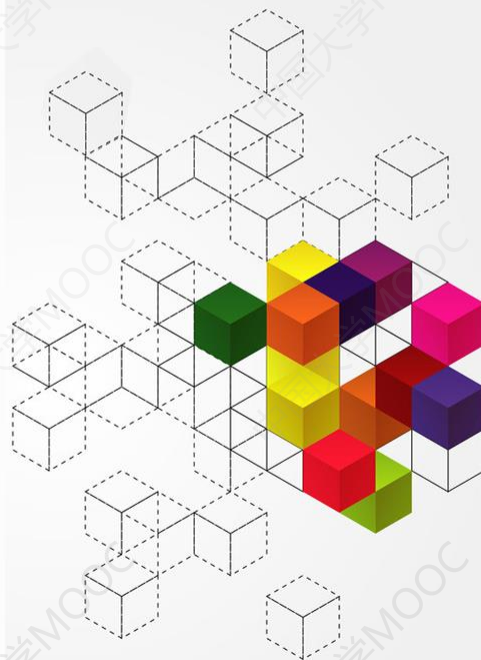
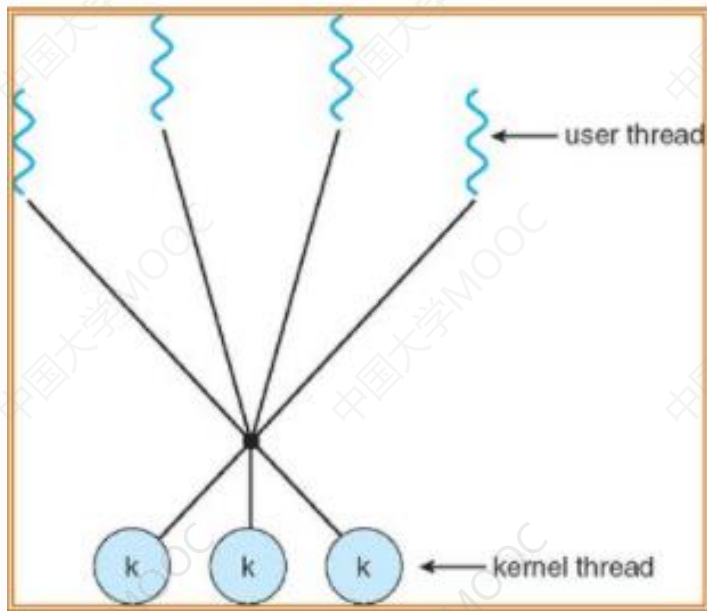
- 将1个用户级线程绑定到1个内核级线程(1:1)





## 二、几种典型的线程模型 – M:N模型

- 将多个用户级线程绑定到多个内核级线程(M:N)



### 三、Linux线程模型

LinuxThreads

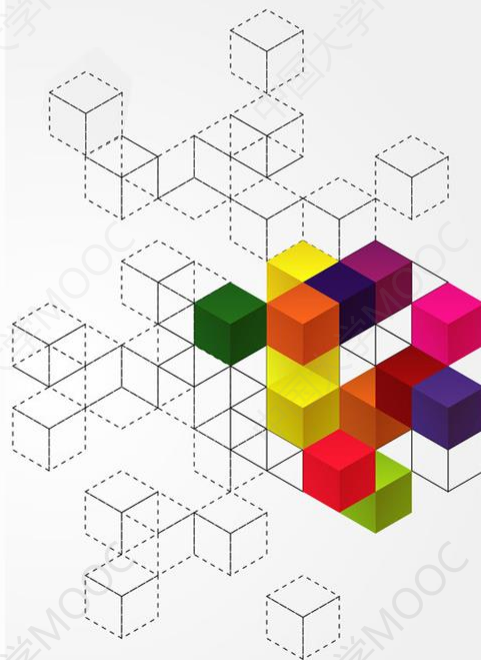
最早的模型，部分  
实现Posix  
Threads

NGPT

Next Generation  
Posix Threads(已  
终止)

NPTL

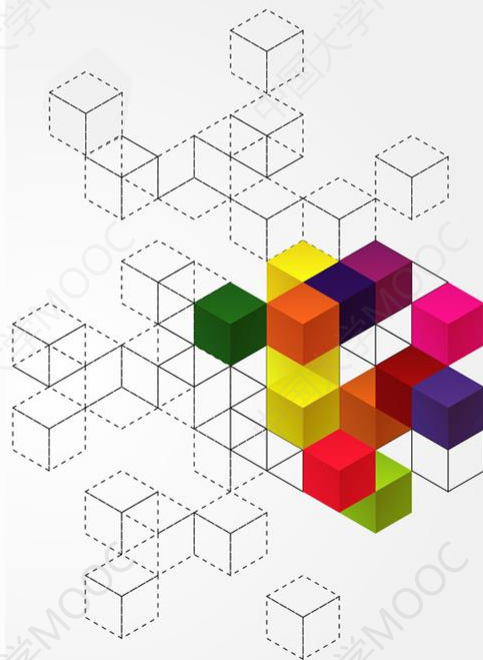
Native Posix  
Thread Library,  
自2.6内核以来使  
用的线程模型



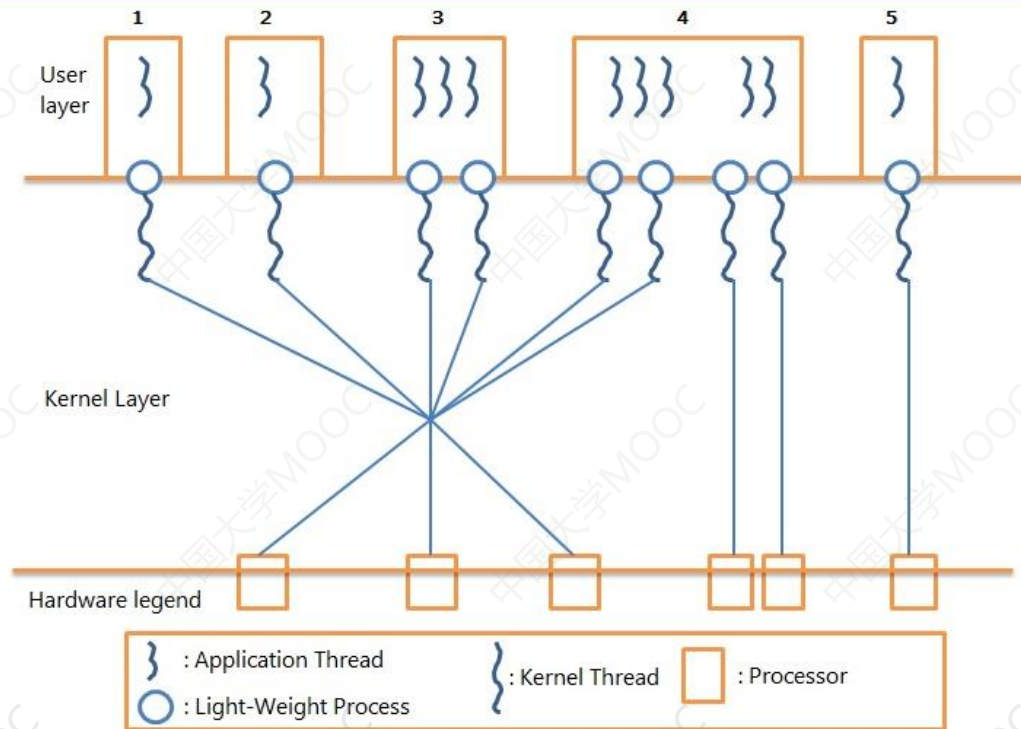


### 三、Linux线程模型

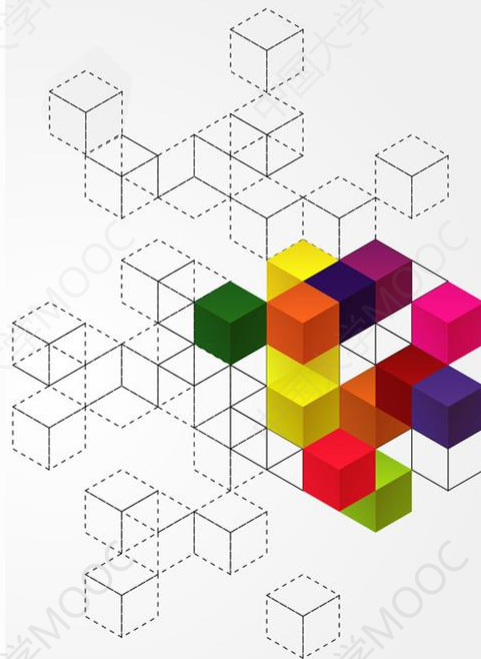
- 在NPTL中，采用的是1：1线程模型
  - 当调用pthread库创建一个线程时，pthread\_create会实际调用clone系统调用，最终会创建一个LWP（轻量级进程）和一个内核线程
  - 内核线程具有对应的task\_struct结构，是个独立的内核调度单元



## 四、Solaris线程模型



- Solaris 2-8: M:N混合模型
- Solaris9-10: 默认线程模型改为1:1模型



# 本讲小结

- 线程分类
- 几种典型的线程模型
- Linux线程模型
- Solaris线程模型

