多线程编程与Actor模式

笔记本:	无聊的学习		
创建时间:	2019/8/11 18:34	**更新时间:**	2019/8/11 19:49
作者:	travmymail@gmail.com		
标签:	多线程, 设计模式		

多核时代的并行编程

免费的性能午餐已经结束了

"The Free Lunch Is Over"
-By Herb Sutter, 2005

摩尔定律：微处理器的性能每隔18个月提高一倍，或价格下降一半。

十几年前，我们习惯看到500MHz的CPU被1GHz的取代，1GHz被2GHz取代等等，但是这种增长并不会如摩尔定律所说一般永远持续下去，在今天，我们主流电脑的CPU主频徘徊在3GHz~4GHz之间。

最新的i9-9900K

这篇05的文章早就预测10年后不会有10GHz的CPU存在。

摩尔定律的终结?

我们获得了更多的核心、更多的线程、更大的缓存、更大的带宽。

第九代智能英特尔® 酷睿™ 处理器

处理器编号	基本主频（GHz）	英特尔® 睿频加速技术 2.0 可达单核睿频频率（GHz）	内核/线程数	TDP	英特尔® 智能高速缓存	未锁频	平台PCIE通道数量	内存支持	英特尔® 傲腾™ 内存支持
第九代智能英特尔® 酷睿™ i9-9900K 处理器	3.6	5.0	8/16	95W	16MB	是	多达40条	双通道 DDR4-2666	是
第九代智能英特尔® 酷睿™ i7-9700K 处理器	3.6	4.9	8/8	95W	12MB	是	多达40条	双通道 DDR4-2666	是
第九代智能英特尔® 酷睿™ i5-9600K 处理器	3.7	4.6	6/6	95W	9MB	是	多达40条	双通道 DDR4-2666	是

我们得不到更高的主频。

Actor模式

单线程编程

单核单机时代一般都是单线程编程，如果把程序比作一个工厂，那么只有一个工人，这个工人负责所有的事情。
所有的原料、工具产品等都放到一个地方。
因为只有一个人，因此使用一套工具就行，取原料也不用排队等。

多线程编程 - 共享内存

到了多核时代，有多个工人，这些工人共同使用一个仓库和车间，干什么都要排队。比如我要从一块钢料切出一块来用，我得等别人先用完。有个扳手，另一个人在用，我得等他用完。 - 资源分配
"哲学家就餐"问题 - 死锁

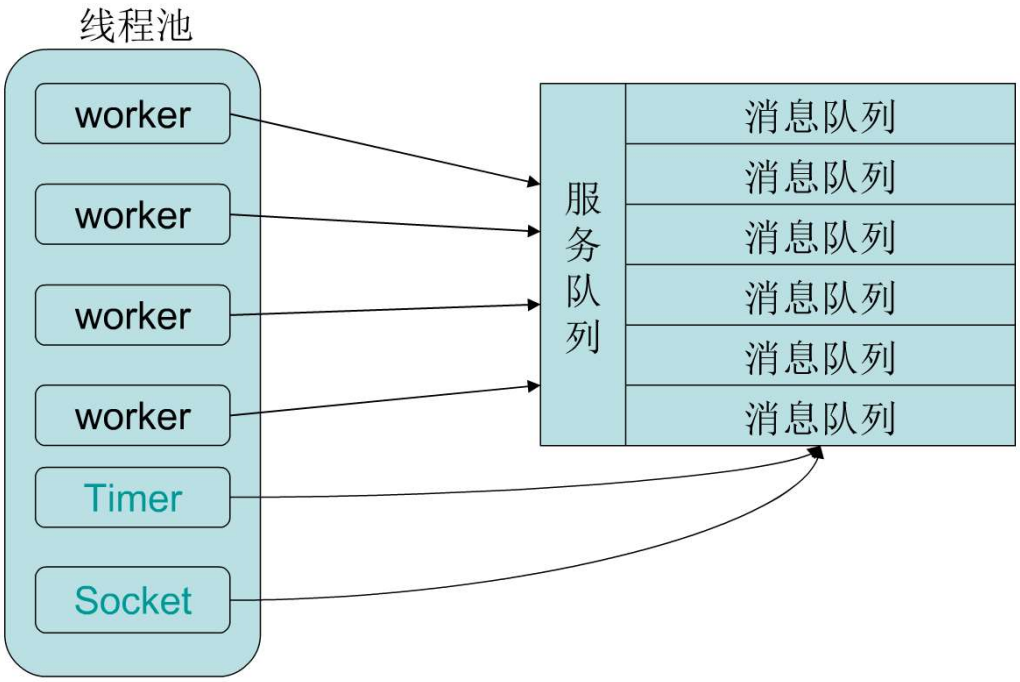
多线程编程 - Actor模式

工厂已经用流水线了，每个人都有明确分工，这就是Actor模式。每个线程都是一个Actor，这些Actor不共享任何内存，所有的数据都是通过消息传递的方式进行的。

一切都是Actor

- 一切都是Object?
- 并行的面向对象模式!
- 创建 Actor / 处理消息 / 发送消息

- 发送的消息和发送者解耦、异步通信



简单来说，面向对象里面就是所有东西都看作是对象，所有的通讯都看作是对象间用消息做通讯，你跟一个对象发一个消息他就做出反应。
Actor模式也是类似，Actor你也可以看作是对象，但是他与传统的面向对象不同的是他默认是异步的消息而且是并行的。
传统的面向对象中，比如A发一个消息，B收到了处理完了才返回过来，A才继续跑。
C++里面消息就是调用对象里的一个方法 / 函数。
Actor可以理解为是并行的面向对象，他是真的有一个消息投递过去，而且投递之后消息和Actor本身就解耦了。
当我投递一个消息给你的时候，我还在不在以及我在干什么已经不重要了，你处理完之后你要干什么都随你。

这种特性就比较适合做并行，注意是并行而不是并发哦。？

材料

Skynet 开源的游戏服务器框架 云风

- https://github.com/cloudwu/skynet
- 中文文档：https://github.com/cloudwu/skynet/wiki
- 邮件列表：skynet-users@googlegroups.com
- QQ 群：340504014

Erlang 98年的编程语言 函数式编程
阿里云在用