在Java应用程序中动态分配CPU资源 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/252/2021_2022__E5_9C_A8Java_E5_BA_94_c104_252335.htm Java的线程调度操作在运行时是与平台无关的。一个多任务系统需要在任务之间实现QoS(Quality of Service)管理时，如果CPU资源的分配基于Java线程的优先级，那么它在不同平台上运行时的效果是很难预测的。本文利用协调式多任务模型，提出一个与平台无关、并且能在任务间动态分配CPU资源的方案。 现在，由于计算机系统已经从人机交互逐步向机机交互转化，计算机和计算机之间的业务对于时间的要求非常高。软件系统对于业务的支持已经不仅表现为对不同业务的逻辑和数据(算法 数据结构)支持，而且还表现为对同时处理不同任务的时效性(任务响应速度)支持。一般，任务响应的速度可以通过算法优化及并行运算分担负载等手段来提高。但是，用户业务逻辑的复杂度决定了算法优化的发挥空间，硬件规模决定了所能够承担负载的大小。我们利用Java平台的特点，借鉴协调式多任务思想，使CPU资源能够在任务间动态分配，从而为时间要求强的任务分配更多的CPU运行资源。这也可以充分利用现有硬件，为用户业务提供最大的保障。 用Java解决问题 本着软件系统结构和现实系统结构一致的思想，开发复杂业务服务的程序一般按照计算机任务和现实业务对应的思路，最终形成一个大规模的多任务系统。由于其跨平台性，Java系统可以随着业务的扩大，平滑地升级到各种硬件平台上。由于Java自身的发展及其应用场合的不断扩大，用它实现多任务系统已经成为当前的应用方向。在J2EE(Java2 Enterprise Edition)推

出以后，Sun公司已经将Java的重心放在了服务器端(Server Side)系统的构造上。由于客户/服务器模型固有的多对一的关系，服务器端程序也必然是一个多任务系统。 在Java多任务应用中，动态地将CPU资源在任务间分配有很重要的意义。比如一个Internet服务商的系统往往有多种任务同时运行，有HTTP、FTP、MAIL等协议的支持，也有商务、娱乐、生活、咨询等业务的服务。在白天，网站希望系统的CPU资源尽量保障网上用户的服务质量，提高电子商务等任务的响应速度.晚上则希望让自己的娱乐服务和资料下载尽可能满足下班后人们的需要。另外，在新兴的网管(比如TMN, Telecommunication Management Network)等应用领域中，服务程序往往需要支持成千上万个并发响应事件的被管理对象(MO，Managed Object)。对于被管理对象执行的操作，不同用户在不同时刻往往有不同的时间要求。 方案选择 在考虑动态分配CPU资源的实施方案时，往往有以下两点要求： 1.须充分利用现有硬件资源，在系统空闲时，让低优先级任务也能够得到系统所能给予的最快响应。 2.当硬件资源超负荷运行时，虽然系统中有大规模、多数量的任务不能处理，但它不应受影响，而能够顺利处理那些能够被处理的、最重要的高优先级任务。 多任务系统要用多线程实现的最简单方法就是将线程和任务一一对应，动态调整线程的优先级，利用线程调度来完成CPU资源在不同任务间动态分配。这种思路在以前使用本地化代码(Native Code)，充分利用特定硬件和操作系统技巧的基础上是基本可行的。但在跨平台的Java环境中，这个思路对仅有小规模任务数的简单系统才可行，原因有以下两点： 1. Java的线程虽然在编程角度(API)是与平台无

关的，但它的运行效果却和不同操作系统平台密切相关。为了利用更多的CPU资源，Java中的一个线程(Thread)就对应着不同操作系统下的一个真实线程。因为Java虚拟机没有实现线程的调度，所以这些Java的线程在不同操作系统调度下运行的差异性也就比较明显。例如在Windows系统中，不仅线程的优先级少于Java API参数规定的十个优先级，而且微软明确反对程序员动态调整线程优先级。即使在操作系统中有足够的优先权，让线程优先级的参数和真实线程的优先级对应，不同操作系统的调度方式也会有许多不同。这最终会造成代码在不同平台上的行为变得不可预测。这就很难满足复杂的、大规模并发任务的众多优先级需求，从而很难达到用户业务需要达到的效果。 2. 由于在Java系统中，线程被包装在一个Java语言的对象类Thread中，所以为了完成Java语言对象和操作系统线程的对应，Java线程的系统开销还是比较大的(在NT 4.0中，平均每个线程大致占用30KB内存)。因此如果让Thread对象个数和成千上万的任务数同比例增长，就显然是不合理的。 综上所述，根据并发多任务的大规模需求和Java平台固有的特点，想要利用Java Thread对象的优先级调整CPU资源的分配是非常困难的，所以应该尽量避免让线程和任务直接对应，也尽量避免使用操作系统线程优先级的调度机制。 解决方案 根据以上分析，问题的症结在于：多任务系统中的任务在Java语言中的对应以及任务间的相互调度。从本质上看，一个任务就是一系列对象方法的调用序列，与Java的Thread对象或者别的类的对象没有必然联系。在避免使用不同操作系统线程调度且同时Java虚拟机又没有线程调度能力的情况下，要想构造一个协调式多任务系统，让各个任

务相互配合就成了最直接的思路。协调式多任务系统一般有以下特点： 1. 任务由消息驱动，消息的响应代码完成任务逻辑的处理. 2. 消息队列完成消息的存储和管理，从而利用消息处理的次序体现任务优先级的不同. 3. 任务中耗时的消息响应逻辑能够主动放弃CPU资源，让别的任务执行(像Windows 3.1中的Yield函数、Visual Basic中的DoEvents语句)。 可能出于巧合，Java语言具有构造协调式多任务系统天然的条件。Java对象的方法不仅是一个函数调用，它还是一个Java.lang.reflect.Method类的对象。而所有对象的方法都可以通过Method类的invoke方法调用。如果能使每个任务所对应的一系列方法全部以对象形式包装成消息，放到消息队列中，然后再按照自己的优先级算法将队列中的消息取出，执行其Method对象的invoke调用，那么一个基本的协调式多任务系统就形成了。其中，任务的优先级和线程的优先级没有绑定关系。该系统的主体调度函数可以设置成一个“ 死循环 ”，按照需要的优先级算法处理消息队列。对于有多重循环、外设等待等耗时操作的消息响应函数，可以在响应函数内部递归调用主体调度函数,这一次调用把原来的“ 死循环 ”改成在消息队列长度减少到一定程度(或者为空)后退出。退出后，函数返回，执行刚才没有完成的消息响应逻辑，这样就非常自然地实现了协调式系统中任务主动放弃CPU资源的要求。 如果仅仅做到这一步，完成一个像Windows 3.1中的多任务系统，实际只用了一个线程，没有利用Java多线程的特点。应该注意到，虽然Java系统中线程调度与平台相关，但是相同优先级的线程之间分时运行的特点基本上是不受特定平台影响的。各个相同优先级的线程共享CPU资源，而线程又被映射

成了Java语言中的Thread对象。这些对象就可以被认为是CPU资源的代表。Thread与线程执行代码主体的接口Runnable之间是多对一的关系。一个Runnable可以被多个Thread执行。只要将Runnable的执行代码设置成上述的消息调度函数，并和消息队列对应上，那么就可以通过控制为它服务的Thread个数来决定消息队列执行的快慢，并且在运行时可以动态地新增(new)和退出Thread对象。这样就能任意调整不同消息队列在执行时所占用CPU资源的多少。至此，任何一个Java调用都可以在Thread个数不同的消息队列中选择，并可以调整这些消息队列服务的Thread个数，从而实现在运行时调整任务所占用的CPU资源。 纵观整个方案，由于仅仅基于Java语言固有的Method对象，不同任务间动态分配CPU资源并没有对任务的性质及其处理流程有任何限制，那么在消息队列中没有高优先级消息时，低优先级消息的处理函数自然会全部占用CPU资源。在不同消息队列处理速度任意设置时，并没有将特定的消息限制在快的或者慢的消息队列上。如果系统的负荷超出(比如消息队列长度超过一定限制)，只要将队列中低优先级消息换出或者拒绝不能处理的消息进入，那么系统的运行就可以基本上不受负荷压力的影响，从而最大保障用户的关键业务需求。 当然，协调式多任务的思想也有其局限性，主要就是它的调度粒度比较大。系统能够保证的粒度是一次消息处理过程。如果消息处理逻辑非常费时，那么编程人员就必须再处理函数内部，让系统主动让出CPU资源。这虽然需要在处理消息响应逻辑时增加一个考虑因素，但是，在Windows系统盛行的今天，这是一个已经被普遍接受的思路。由于方案中并没有局限为消息队列服务的线程数目，所

以一个长时间的消息响应只会影响一个线程，而不会对整个系统产生致命的影响。除了调度粒度的问题以外，还有访问消息队列操作在各个线程间互斥的问题。取出消息的过程是串行化的，因此对于这一瓶颈的解决方案就是：假设取出一条消息的操作相对于处理消息的消耗可以忽略不计，那么对于多次调用且仅有两三行响应逻辑的消息，编程人员通过函数调用就可以直接执行。 前面比较详细地阐述了多任务系统中任务的划分以及执行等内容。虽然这些是一个系统的核心，但是在一个实用的系统中，还需要任务间的同步、互斥等机制。在上述框架内，互斥可以简单地用Java的Synchronized机制实现。由于任务可以主动让出执行权限，要实现等待(Wait任务中止)和通知(Notify任务继续)，从而实现任务同步也就比较容易了。 100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com