Slave Memories and Dynamic Storage Allocation

--读书报告

这篇文章是由M.V.Wilks所写的，他是英国计算机协会的第一任主席，是著名的计算机科学家，这次研读的这一篇文献是他对于从属存储器和动态内存分配问题的阐释。

开篇的summary中，作者就提出了使用快速内存作为核心内存比慢速内存的效率显然是要大得多的。在当时使用的分层存储系统中，核心存储器是由磁盘备份的，但是当一个快速的核心内存由一个大的慢速的核心内存进行备份的时候，因为两个内存的访问都是真正的随机访问，这就造成了有一部分字节在后面的运行中没有被使用，从而被浪费掉。所以之后作者提出使用快速内存作为从属内存，由于从属内存是主内存大小的一部分，所以里面的信息无法无限期的保留在其中，这就需要将信息不断地更新，这样就可以达到提升信息读取速度的效果。并且使用从属内存的方法是被证明对减少访问时间有着很明显的提升效果的。在设计从属内存的时候，设有长度不定的标记位，每当需要指令的时候，首先检查从属存储器中看看是否包含该指令，如果包含，则取走指令；若没有，就从主内存和从属内存中保留的副本中获取指令。在写入的时候，也需要进行检查，查看从属存储器是否包含即将更新的字节，如果是，就必须在从属内存和主内存中更新该字节。

从属原理在几十年前就已经应用于与计算机控制相关的非常小的超高速存储器。但是，使用正常大小的核心内存作为大型核心内存的从属服务器也是有可能的。作者在大型从属内存这一节中就详细介绍了各种方法。其中的简单描述方案中，程序被拆分成32K字块，每个用户使用一个或多个块执行其程序。大型的核心内存有一个基本的寄存器，其中包含当前处于活动状态的32K块的起始地址。其中为了避免浪费，作者避免每次激活时将整个块转移到快速核心内存。然后设计两个标记位，从而实现读和写操作。之后当基本寄存器中的编号发生修改时，使新的程序在当前活动程序的位置变成活动状态，就会启动快速的内存扫描。每个寄存器有测试位，如果第一个标记位为0，就不对该寄存器执行任何操作，；如果第一个标记位为1，第二个标记位为0，也不执行操作，但是，如果两个标记位都是1，则就检查寄存器中的字节复制在大型内存中的适当位置。此外，作者还对这种最简单的方案做了一种变体，这里就不再做赘述。

除了第一种简单的方案之外，作者还提出了另一种选择。那就是保留32K作为块长，但安排快速内存作为从属内存。其中作者假设有七个基本的寄存器，每个寄存器都包含程序块启动的主内存中的寄存器的地址。需要四个标记位，前三个标记位代表是哪个寄存器，第四个标记位指示在从属存储器中是否更改了字节。在给定的任何时间里，七个程序块之一处于活动状态时。每当需要访问内存中的字节时，硬件就会查看是否能够在从属寄存器中找到该单词，这就是比对前三个标记位的作用。其中前三个标记位的不同情况代表了寄存器的不同状态和操作。

以上两种方案，可以说为之后的几十年至今一直使用的两级（甚至多级）核心存储系统指明了道路，可以说M.V.Wilks的贡献是卓越的，他通过这一篇论文详细的阐述了之前遇到的问题和解决的方案，每一个方案用一个例子让人能够易于理解。从文献中，我能够体会到计算机的先贤们严谨的逻辑思维和惊人的才学。这也督促我要打好基础。我从这篇论文中也是受益良多。