(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 106648546 A
(43)申请公布日 2017.05.10

(21)申请号 201610805632.4

(22)申请日 2016.09.07

(71)申请人 北京大学
地址 100871 北京市海淀区颐和园路5号

(72)发明人 梁云 谢小龙

(74)专利代理机构 北京万象新悦知识产权代理事务所(普通合伙) 11360
代理人 黄凤茹

(51)Int.Cl.
G06F 9/30(2006.01)
G06F 9/38(2006.01)
G06F 9/50(2006.01)

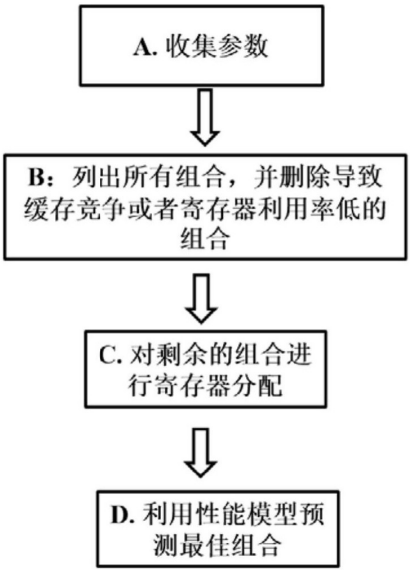
权利要求书1页 说明书4页 附图1页

(54)发明名称

用于GPU寄存器分配和并行度管理的协同优化编译方法

(57)摘要

本发明公布了一种协同优化编译方法,所述方法用于GPU寄存器分配和并行度管理协同优化的编译,使得寄存器的分配和并行度管理的优化同时进行,且不引起缓存的竞争;包括:收集寄存器分配参数、线程级并行参数和缓存性能参数;列出所有可能的线程块数量TLP和寄存器数量Reg的组合;删除导致缓存竞争的组合和导致寄存器浪费的组合;对剩下的组合进行寄存器分配;使用性能模型进行预测并选择性能最佳的组合,作为寄存器分配编译的结果。本发明技术方案可在不引起缓存竞争的前提下,最大化寄存器的使用率,最终提升整个程序的性能。



CN 106648546 A

1.一种协同优化编译方法,所述方法用于GPU寄存器分配和并行度管理协同优化的编译,使得寄存器的分配和并行度管理的优化同时进行,且不引起缓存的竞争;包括如下步骤:

A)收集寄存器分配参数、并行度管理参数和缓存性能参数;

B)通过步骤A中所获得的参数,列出所有可能的线程块数量TLP和寄存器数量Reg的组合;删除导致缓存竞争的组合和导致寄存器浪费的组合;

C)对剩下的线程块数量TLP和寄存器数量Reg的组合进行寄存器分配,在寄存器的分配过程中,用染色算法将该程序的所有变量放入数量为Reg的寄存器中;当出现部分变量无法放入到Reg个寄存器时,将所述部分变量放在共享内存中;当共享内存空间不足时,再把所述部分变量放置到片下存储中;

D)使用性能模型预测得到各个组合的性能,并选择性能最佳的组合,作为寄存器分配编译的结果。

2.如权利要求1所述协同优化编译方法,其特征是,A)所述寄存器分配参数包括每个线程至少获得的寄存器数量MinReg和每个线程最多需要的寄存器数量MaxReg;所述并行度参数包括不考虑寄存器的情况下每个GPU核上最多的的线程块数量MaxTLP和不引起缓存竞争的最多线程块数量OptTLP;缓存性能参数包括每个线程块在不引起线程数下降的情况下可获得的最多共享内存数量ShmSize。

3.如权利要求1所述协同优化编译方法,其特征是,B)所述导致缓存竞争的组合为每个GPU核心上的并发线程块数量TLP大于不引起缓存竞争的最多线程块数量OptTLP的组合;所述导致寄存器浪费的组合为寄存器Reg没有达到对应的TLP最大的寄存器数量的组合。

4.如权利要求1所述协同优化编译方法,其特征是,D)所述性能模型为:

(1) $TLP_{gain} = 1 - \frac{TLP \times BlockSize}{TLP \times BlockSize + MaxThread}$

(2) $Spill_{cost} = Num_{local} \times Cost_{local} + Num_{shm} \times Cost_{shm} + Num_{others}$

(3) $TPSC = TLP_{gain} \times Spill_{cost}$

其中:公式(1)中,TLPgain表示由于每个GPU核心上的并发线程块数量(TLP)的提高,GPU程序的执行时间的变化;TLP表示该组合的线程并行度;BlockSize值该GPU程序中每个线程块(Thread block)的大小;MaxThread表示该GPU中每个核心所允许的最大并发线程数;公式(2)中,参数Spillcost表示寄存器分配对于GPU应用执行时间的影响;Num_{local}、Num_{shm}、Num_{others}分别指本地存储指令(local memory)、共享存储指令和其他指令的数量;$Cost_{local}$,$Cost_{shm}$分别指本地存储指令、共享存储指令的周期数;公式(3)中,参数TPSC是最终的执行时间。

5.如权利要求4所述协同优化编译方法,其特征是,所述性能最佳为所述最终的执行时间TPSC值最小[TLP,Reg]组合,作为寄存器分配和并行度的优化结果。

用于GPU寄存器分配和并行度管理的协同优化编译方法

技术领域

[0001] 本发明涉及寄存器分配编译技术，尤其涉及一种用于GPU寄存器分配和并行度管理的协同优化编译方法。

背景技术

[0002] 寄存器的分配是计算机领域一种常见的编译问题。对于每一个程序，寄存器的数量往往是有限的，而程序所要使用的变量的数量可能会远远超过寄存器的数量。因此如果尽可能的程序的变量放到寄存器中、从而获得最大的性能，一直是计算机领域一个重要的基本问题。

[0003] 寄存器的分配被认为是一个K染色问题。假设我们有N个变量和K个寄存器。通常，编译器首先通过数据流和控制流分析，获得所有N各变量的活跃时间。然后构建一个大小为N的图，每个顶点代表一个变量。如果两个变量的活跃时间有重叠，那么就在对应的两个顶点之间连一条线。最后，使用K种颜色对该图进行染色，并规定两个连接的顶点不能使用同一种颜色。如果染色成功，那么K个寄存器足以放入所有变量。如果染色失败，则需要删除一些顶点。被删除的顶点将会被放入到内存中。因此，寄存器的分配问题就被转化成了K染色问题。

[0004] 但是，现有传统的寄存器分配算法仅适用于单线程程序。GPU(graphics processing unit)，也就是图形处理器是中核架构。为了支撑大量线程的并发执行，GPU配备了大容量的寄存器堆。因此，GPU的寄存器分配不仅要考虑K个寄存器是否能够容纳N个变量，还要考虑给每个线程分配具体多少寄存器，也就是决定K的大小。而这是现有传统的寄存器分配算法没有考虑的，因此，现有寄存器分配算法无法支撑大量线程的并发执行。

发明内容

[0005] 为了克服上述现有技术的不足，本发明提供一种GPU上寄存器分配的方法，是一种用于GPU的编译方法，涉及到寄存器分配和并行度管理两个方面的协同优化；该方法能够使得寄存器分配和并发度管理的优化同时进行，并且不引起缓存的竞争。在编译时，本发明同时考虑线程数、寄存器分配、缓存性能多个指标。

[0006] 本发明提供的技术方案是：

[0007] 一种寄存器分配编译方法，所述方法用于GPU寄存器分配和并行度管理协同优化的编译，使得寄存器的分配和并行度管理的优化同时进行，且不引起缓存的竞争；包括如下步骤：

[0008] A)收集寄存器分配参数、并行度参数和缓存性能参数；

[0009] B)通过步骤A中所获得的参数，列出所有可能的线程块数量TLP和寄存器数量Reg的组合；删除导致缓存竞争的组合和导致寄存器浪费的组合；

[0010] C)对剩下的组合进行寄存器分配，在寄存器的分配过程中，当出现部分变量无法放入寄存器时，尽量将这些寄存器放在共享内存中；当共享内存空间不足时，再把他们放置

到片下存储中；

[0011] D)使用性能模型预测得到各个组合的性能，并选择性能最佳的组合，作为寄存器分配编译的结果。

[0012] 针对上述寄存器分配编译方法，进一步地，A)所述寄存器分配参数包括每个线程至少获得的寄存器数量MinReg和每个线程最多需要的寄存器数量MaxReg；所述并行度参数包括不考虑寄存器的情况下每个GPU核上最多的的线程块数量MaxTLP和不引起缓存竞争的最多线程块数量OptTLP；缓存性能参数包括每个线程块在不引起线程数下降的情况下可获得的最多共享内存数量ShmSize。

[0013] 针对上述寄存器分配编译方法，进一步地，B)所述导致缓存竞争的组合为每个GPU核心上的并发线程块数量TLP大于不引起缓存竞争的最多线程块数量OptTLP的组合；所述导致寄存器浪费的组合为寄存器Reg没有达到对应的TLP最大的寄存器数量的组合。

[0014] 针对上述寄存器分配编译方法，进一步地，D)所述性能模型为：

[0015] $TPSC = TLP_{gain} \times Spill_{cost}$

[0016] $$TLP_{gain} = 1 - \frac{TLP \times BlockSize}{TLP \times BlockSize + MaxThread}$$

[0017] $Spill_{cost} = Num_{local} \times Cost_{local} + Num_{shm} \times Cost_{shm} + Num_{others}$

[0018] 其中，TPSC指的是最终的执行时间，越小越好。BlockSize指每个thread block的大小。Num_{local}，Num_{shm}，Num_{others}分别指本地存储指令(local memory)、共享存储指令和其他指令的数量。$Cost_{local}$，$Cost_{shm}$分别指本地存储指令、共享存储指令的周期数。性能最佳为所述最终的执行时间TPSC值最小。

[0019] 与现有技术相比，本发明的有益效果是：

[0020] 现有的寄存器分配技术只考虑单线程性能，本发明技术方案同时考虑单线程性能、并发线程数和缓存性能以达到最佳性能。因此，本发明方法可以在不引起缓存竞争的前提下，最大化寄存器的使用率，并获得一个最佳的寄存器和并行度的这种方案，最终提升整个程序的性能。

附图说明

[0021] 图1是本发明提供的用于GPU寄存器分配和并行度管理的协同优化编译方法的流程框图。

[0022] 图2是本发明方法的寄存器分配部分的算法流程框图。

具体实施方式

[0023] 下面结合附图，通过实施例进一步描述本发明，但不以任何方式限制本发明的范围。

[0024] 本发明提供一种GPU上寄存器分配的方法，该方法能够使得寄存器的分配和并行度管理的优化同时进行，并且不引起缓存的竞争。在编译时，本发明同时考虑线程数、寄存器分配、缓存性能多个指标。

[0025] 图1是本发明提供的用于GPU寄存器分配和并行度管理协同优化的编译方法的流程框图，包括如下步骤：

[0026] A)首先收集寄存器分配、并行度和缓存性能的参数。

[0027] A1.MinReg,MaxReg.MinReg指每个线程至少可以获得多少寄存器,这个参数是由硬件制定的,一般是寄存器的总数量除以最大线程数。MaxReg指此线程最多需要多少寄存器,MaxReg个寄存器足以将整个图进行染色。

[0028] A2.MaxTLP,OptTLP。MaxTLP指的是在不考虑寄存器的情况下每个GPU核上最多的的线程块数量。OptTLP指的是不引起缓存竞争的最多线程块数量。OptTLP小于等于MaxTLP,如果在每个核上放置多于OptTLP个线程块,会因为缓存竞争导致性能下降。本专利中,TLP指每个GPU核心上的并发线程块数量。

[0029] A3.ShmSize,每个线程块在不引起线程数下降的情况下,可获得的最多的共享内存数量。

[0030] B)首先通过步骤A中获得的参数,我们列出所有可能的线程块数量(TLP)和寄存器数量(Reg)的组合。然后两类组合会被删除。

[0031] B1.TLP大于OptTLP的组合。这类组合会导致缓存竞争。

[0032] B2.Reg没有达到对应的TLP最大的寄存器数量的组合。这类组合会导致寄存器浪费。

[0033] C)然后我们会对剩下的线程块数量(TLP)和寄存器数量(Reg)的组合进行寄存器分配。在寄存器的分配过程中,我们使用染色算法将该程序的所有变量放入数量为Reg的寄存器中。如果出现部分变量无法放入到Reg个寄存器中的情况,我们会将这些变量放在共享内存中。如果出现共享内存空间不足的情况,再把这些变量放置到片下存储中。

[0034] D)最后,使用性能模型预测各个组合的性能,并选择性能最佳的组合。

[0035] 本发明使用的性能模型如下:

[0036] (1) $$TLP_{gain} = 1 - \frac{TLP \times BlockSize}{TLP \times BlockSize + MaxThread}$$

[0037] (2) $Spill_{cost} = Num_{local} \times Cost_{local} + Num_{shm} \times Cost_{shm} + Num_{others}$

[0038] (3) $TPSC = TLP_{gain} \times Spill_{cost}$

[0039] 该模型共包含三个公式。首先该模型根据公式(1)计算出参数TLPgain。TLPgain表示由于每个GPU核心上的并发线程块数量(TLP)的提高,GPU程序的执行时间的变化。它可以通过(1)右侧的公式计算得到。其中,TLP表示该组合的线程并行度,BlockSize值该GPU程序中每个线程块(Thread block)的大小。MaxThread表示该GPU中每个核心所允许的最大并发线程数。

[0040] 然后根据公式(2)计算出参数Spillcost,该参数表示寄存器分配对于GPU应用执行时间的影响。公式(2)右侧是计算方法,计算方法是把不同种指令的数量和执行时间相乘后进行累加。其中Num_{local},Num_{shm},Num_{others}分别指本地存储指令(local memory)、共享存储指令和其他指令的数量。$Cost_{local}$,$Cost_{shm}$分别指本地存储指令、共享存储指令的周期数。

[0041] 最后,本模型通过公式(3)计算出参数TPSC,TPSC指的是最终的执行时间,越小越好,通过将公式(1)的TLPgain和公式(2)的Spillcost相乘获得。最终,TPSC最小的[TLP,Reg]组合成为本技术选取的寄存器分配和并行度的优化方案。

[0042] 下面通过实例对本发明做进一步说明。

[0043] 实施例一:

[0044] 假定一个GPU的核函数不存在缓存竞争问题(OptTLP为最大值),经过步骤A收集参数之后,存在[MinReg,MaxReg]个寄存器分配可能,[MinTLP,MaxTLP]个并发度分配可能。共存在(MaxReg-MinReg)*[MaxTLP-MinTLP]种寄存器和并发度分配组合(每种组合用(Reg,TLP)表示,其中Reg表示每个线程的寄存器数量,TLP表示每个GPU核心上的线程块数量)。

[0045] 列出所有可能的线程块数量(TLP)和寄存器数量(Reg)的组合,通过步骤B将如上各种组合中,寄存器利用率不是最大的组合删除,经过这一步骤,一般会剩余3～5种不同组合。

[0046] 通过步骤C对剩下的组合进行寄存器分配,在寄存器分配的过程中,根据需要,可能会将被分配到本地存储中的变量,重新分配到共享存储中,以减少对本地存储的访问。

[0047] 通过步骤D,根据性能模型,并根据几种不同(Reg,TLP)所产生的代码,预测不同配置的性能,并选择TPSC值最小的组合。

[0048] 实施例二:

[0049] 假定一个GPU的核函数存在缓存竞争问题,经过步骤A收集参数之后,存在[MinReg,MaxReg]个寄存器分配可能,[MinTLP,MaxTLP]个并发度分配可能。共存在(MaxReg-MinReg)*[MaxTLP-MinTLP]种寄存器和并发度分配组合(每种组合用(Reg,TLP)表示,其中Reg表示每个线程的寄存器数量,TLP表示每个GPU核心上的线程块数量)。

[0050] 步骤B首先将如上几种组合中,寄存器利用率不是最大的、而且TLP大于OptTLP的组合删除,经过这一步骤,一般会剩余1～3种不同组合。如果最终组合只剩一种,直接结束算法,并输出这一组合。如果剩余组合多于一种,仿照实施例一,继续步骤C和步骤D,最终选择TPSC值最小的组合。

[0051] 需要注意的是,公布实施例的目的在于帮助进一步理解本发明,但是本领域的技术人员可以理解:在不脱离本发明及所附权利要求的精神和范围内,各种替换和修改都是可能的。因此,本发明不应局限于实施例所公开的内容,本发明要求保护的范围以权利要求书界定的范围为准。

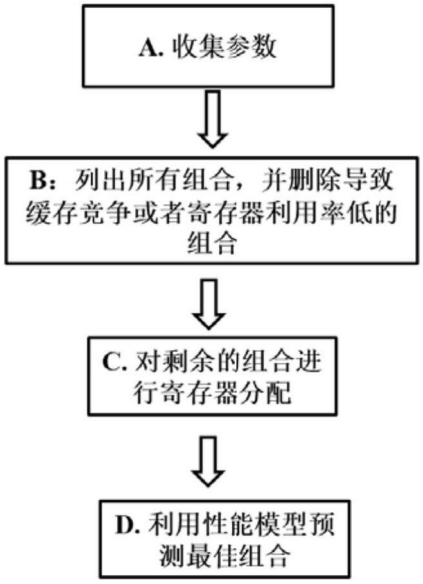

A. 收集参数
B：列出所有组合，并删除导致缓存竞争或者寄存器利用率低的组合
C. 对剩余的组合进行寄存器分配
D. 利用性能模型预测最佳组合

图1

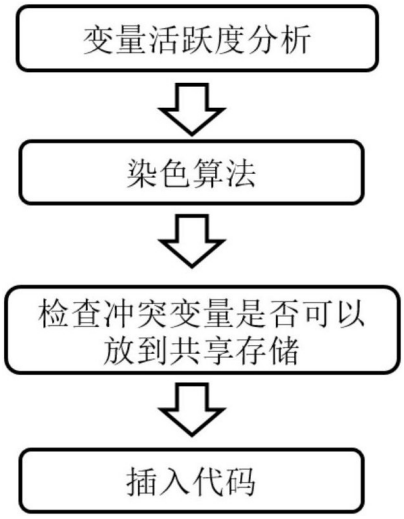

变量活跃度分析
染色算法
检查冲突变量是否可以放到共享存储
插入代码

图2