CN 43-1258/TP
ISSN 1007-130X

计算机工程与科学
Computer Engineering & Science

第 39 卷第 5 期 2017 年 5 月
Vol. 39, No. 5, May. 2017

文章编号:1007-130X(2017)05-0829-05

面向 ARM64 架构多核微处理器的模板计算性能优化研究*

冯璐霞,李春江,黄亚斌
(国防科学技术大学计算机学院,湖南 长沙 410073)

摘　要:模板计算是一类重要的计算核心,广泛存在于图像和视频处理以及大规模科学和工程计算领域。但是,针对 ARM64 高性能处理器的模板计算性能的优化研究还很少。为了实现典型模板计算核心在 ARM64 架构多核微处理器上的并行化和性能优化,基于 AMCC X-GENE2 和飞腾 FT-1500A 多核微处理器特点,提出了基于两维度绑定的优化方法,该方法通过线程与 CPU 绑定以及线程与数据块绑定,减少了线程调度的并行开销,增加了 Cache 的命中率。实验结果表明,该方法提升了模板计算在 ARM64 架构多核微处理器上的性能,且在两种 ARM64 架构多核微处理器平台上都表现出较好的可扩展性。

关键词:模板计算;ARM64;AMCC X-GENE2;FT-1500A;并行化;线程绑定

中图分类号:TP303　　**文献标志码**:A

doi:10.3969/j.issn.1007-130X.2017.05.002

Performance optimization of stencil computation on ARM64 multi-core microprocessor

FENG Lu-xia, LI Chun-jiang, HUANG Ya-bin
(College of Computer, National University of Defense Technology, Changsha 410073, China)

Abstract: Stencil computation is a class of important calculation kernels widely used in the field ranging from image and video processing to large-scale scientific and engineering simulation and calculation. However, the evaluation of stencil computation on the ARM64 high-performance processor is rare. According to the features of AM-CC X-GENE2 and Phytium FT-1500A, we design an optimization method based on two-dimension bound, which reduces the parallelism overheads of thread scheduling, and increases the Cache hit rate by the thread-CPU bound and thread-data-block bound. Experimental results show that this method can improve the performance of the stencil calculation on ARM64 architecture, and the results of our kernel demonstrate the good scalability on the two ARM64 multi-core microprocessor platforms.

Key words: stencil computation; ARM64; AMCC X-GENE2; FT-1500A; parallelism; thread bound

1　引言

近年来,ARM 架构发展迅速。2011 年 10 月,ARM 公司正式发布了处理器架构 ARMv8,ARMv8 将 64 位架构支持引入 ARM 架构中(简称 ARM64)。ARM64 包括 64 位的通用寄存器和 64 位运算支持。目前,cortex-A53[2] 和 cortex-A57[3] 是两款主流的 64 位 ARM 高性能处理器核心。ARMv8[1] 体系结构的发布把 ARM 架构推向了大规模企业服务器领域。

模板计算是一类科学和工程计算领域的重要计算核心,它具有计算密集和访存密集的特点,且内存消耗与计算时间都会随数组元素增加呈线性增长,所以模板代码在各个平台上的并行实现及优化研究受到了广泛的关注。齐金[4] 基于 Intel ®

* **收稿日期**:2017-01-07;**修回日期**:2017-03-20
基金项目:国家自然科学基金(61170046);国家 863 计划(2012AA010903)
通信地址:410073 湖南省长沙市国防科学技术大学计算机学院
Address: College of Computer, National University of Defense Technology, Changsha 410073, Hunan, P. R. China

MIC 平台对 3D 梯度向量流场(3D GVF Field)中的模板计算核心实现了加速优化;Christen 等[5]主要针对生物热方程中的 Stencil Code 实现了局部性和并行性优化。

本文基于 ARM64 架构多核微处理器的特点,采用通用的共享存储并行编程方法实现了模板代码的并行化,并提出了基于两维度绑定的优化方法。该方法通过线程与 CPU 绑定和线程与数据块绑定,分别将第一级 Cache 和第二级 Cache 对应的核数个线程都绑定在共享此级 Cache 的每个 CPU 上,将数据连续平均地绑定到每一个线程,从而减少了线程调度的开销,增加了 Cache 的命中率。通过在 AMCC X-Gene2 和飞腾 FT-1500A 上的大量实验表明,与典型的 OpenMP 并行相比较,本文提出的优化方法很好地提升了并行性能。线程与 CPU 绑定使 2D-5P 和 3D-7P 模板计算的并行性都能获得很好的优化效果,在 AMCC X-Gene2 服务器上,2D-5P 模板计算性能提升了 15.41%,3D-7P 模板计算的性能提升了 0.09%;在 FT-1500A 上,2D-5P 模板计算的性能提升了 1.07%,3D-7P 模板计算的性能显著提升了 20.67%;线程与数据块绑定对 2D-5P 模板计算有加速效果,在 AMCC X-Gene2 服务器上,在线程与 CPU 绑定的两种方式(1-1 模式和 1-2 模式)上进行线程与数据块绑定的加速比分别为 1.03 和 1.05;在 FT-1500A 上,在线程与 CPU 绑定的两种方式(1-1 模式和 1-4 模式)上进行线程与数据块绑定的加速比分别为 1.07 和 1.03。最后针对 2D-5P 模板计算的线程与 CPU 绑定的一种方式(1-1 模式)以及在此方式上再进行线程与数据块绑定的方法评估了两种 ARM64 多核处理器平台的可扩展性,且在两种平台上都表现出较好的可扩展性。

2 模板计算

模板计算(Stencil Computation)[6],也称模板代码(Stencil Code),是根据模板更新数组元素的一类迭代核心。用于更新数组元素的某种固定模式称为模板,在模板操作中,每一个数组元素的值由其自身及相邻点的值更新。Jacobi[7]迭代方法是求解拉普拉斯方程[8]的常用算法。拉普拉斯方程求解是电磁学、天文学和流体力学等领域经常遇到的一类重要的数学问题。Jacobi 方法常用的模板是冯·诺依曼型邻居(Von Neumann Neighborhood)[9],本文选择双精度浮点数据的 2D-5P 和 3D-7P 模板计算核心作为研究对象,Jacobi 伪代码如图 1 所示。

由此可知,每一次时间步迭代内,各点的更新相互独立,不存在数据相关,为实现并行化奠定了基础。

```
1   ......
2  for (k=0; k<100; k++){              //时间步迭代;
3    for( i = 1; i < N-1; i++)         //2D模板更新
4      for( j = 1; j < N-1; j++)
5        a[i][j] = c * (b[i][j]+ b[i-1][j] + b[i+1][j]
6            +b[i][j-1] + b[i][j+1]);
7    for (i=0; i<N;i++)                //交换
8      for (j=0; j<N;j++)
9        b[i][j] = a[i][j];
10       }
11  ......
```

a 2D

```
1   ......
2  for (k=0; k<10; k++){               //时间步迭代次数
3    for( i = 1; i < N-1; i++)         //3D模板更新
4      for( j = 1; j < N-1; j++)
5        for (z = 1;z < N-1;z++ )
6          {
7              a[i][j][z] =  c1 * b[i][j][z] +  c2 * (  b[i-1][j][z] +
b[i+1][j][z]+ b[i][j-1][z]
8                + b[i][j+1][z]+ b[i][j][z-1] + b[i][j][z+1]) ;
9          }
10  for( i = 0; i < N; i++)            //交换
11    for( j = 0; j < N; j++)
12      for (z = 0;z < N;z++ )
13        b[i][j][z] = a[i][j][z];
14      }
15  ......
```

b 3D

Figure 1 Jacobi pseudocodes

图 1 Jacobi 伪代码

3 X-GENE2 和 FT-1500A 体系结构

在 ARMv8 架构下,处理器可以运行全新的 AArch64 执行状态,并向后兼容以往的 32 位程序指令。ARM64 指令集是一种支持 AArch64 执行状态的全新 32 位固定长度指令集,它提供了 31 个通用 64 位寄存器、PC(程序计数器)、SP(栈指针)和可用于大多数指令的专用零寄存器,另外还提供 128 位寄存器用于 SIMD 向量操作。

X-GENE2 是 AMCC(Applied Micro Circuits Corporation)推出的基于 64 位 ARMv8 处理器的第二代芯片,AMCC X-GENE 是世界上第一款 64 位 ARMv8 服务器芯片。X-GENE2 的处理器(主频 2.40 GHz)使用双核模块化设计,并使用单芯片四模块结构。每个处理器核拥有独立的 32 KB L1 级指令 Cache 和 32 KB L1 级数据 Cache,而每个模块有两个处理器核,并共享 256 KB 的 L2 级 Cache;模块间共享一块容量可扩展的 L3 级

Cache。

另外一款兼容 ARM64 指令集的处理器 FT-1500A,其处理器(主频 2.0 GHz)使用四核模块化设计,并使用单芯片四模块结构。每个处理器核拥有独立的 32 KB L1 级指令 Cache 和 32 KB L1 级数据 Cache,而每个模块有四个处理器核,并共享 2 MB的 L2 级 Cache;模块间共享一块 8 MB 的多体 L3 级 Cache。

4 模板计算的并行化

本文在多核多级 Cache 的高性能微处理器上,用典型的 OpenMP 并行方法来实现模板计算的并行化,并针对其并行开销较大和重用数据在 Cache 中容易缺失等特点,提出了基于两维度绑定的优化方法。该方法减少了 OpenMP 执行过程中线程调度的开销,增加了 Cache 的命中率。

4.1 典型的 OpenMP 并行

OpenMP[10] 以线程为基础,通过编译指导指令来显示地指导并行化。其中的 # pragma omp parallel 指明并行区域,且由于时间步之间的数据相关性,所以每个时间步结束后或开始前需要进行显示的线程同步,由 # pragma omp barrier 实现。循环中每次迭代的计算量相等,可以采用显示静态调度 *schedule*(static,1)子句来进行任务调度,因此每个线程将大约分配到$(N-2)/num_threads$次迭代的任务。但是,典型的模板代码 OpenMP 并行开销较大,且未考虑重用数据。

4.2 线程与 CPU 绑定

线程绑定 CPU,即建立线程和某个或某些 CPU 的关联,使得线程只能运行在指定的 CPU 上,主要优点在于以下两个方面:

(1) 增加 Cache 的命中率。线程频繁地在各个 CPU 间进行切换,会不断地使原来运行线程的 CPU 的 Cache 失效,因此将线程绑定在共享同一级 Cache 的 CPU,能充分利用重用数据,增加 Cache 的命中率。

(2) 减少线程调度的开销。将线程绑定到某个或某些 CPU,一定程度上减少了线程竞争 CPU 的开销。

由于 AMCC X-Gene2 和 FT-1500A 都具有多级 Cache,本文采用了两种绑定 CPU 的方式,分别将第一级 Cache 和第二级 Cache 对应的核数 m(L1 Cache 的 $m=1$,L2 Cache 的 $m=2$ 或 4)个线程都绑定在共享同一级 Cache 的每个 CPU 上,本文将其称之为 1-1 模式、1-2 模式(AMCC)或 1-4 模式(FT-1500A),如图 2 所示。将线程与 CPU 绑定,可以通过 *pthread_setaffinity_np* 和 *pthread_getaffinity_np* 两个函数设置线程之间的亲和性(affinity)来达到目的。

4.3 线程与数据块绑定

为了充分利用 Cache 中的重用数据,需要将迭代连续的数据块绑定到每一个线程,并且使共享同一级 Cache 的线程执行的数据块的边界相邻。使用 *omp_get_thread_num*( )函数获得各线程的线程号,将所有的数据平均分成 n 份,按线程号顺序分别绑定到 n 个线程上。如纯种与数据块绑定代码所示,首先划分数据块的大小,将每一份划分好的数据块绑定到每一个线程,且由于第一个线程的线程号为 0,因此需要注意循环索引 i 必须从 1 开始。通过线程与数据块绑定,重用数据有更大的可能性在各级 Cache 中被命中,从而减少访存开销。

线程与数据块绑定代码:

```
1  int block;//划分数据块
2  if(N-2)%num_threads! =0)
```

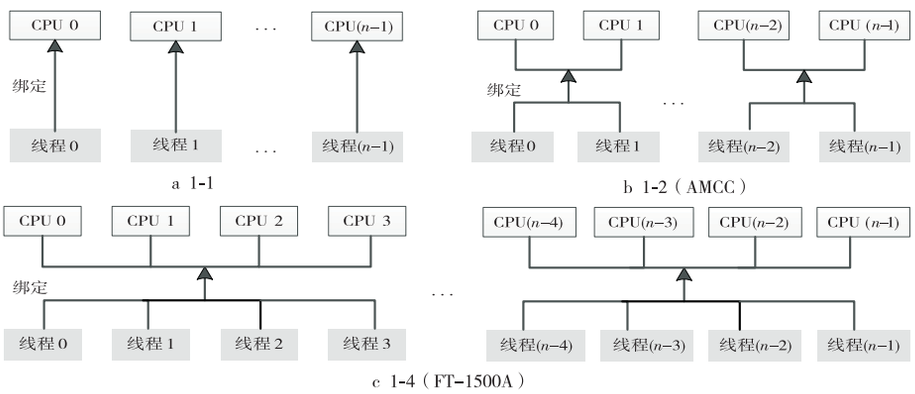

Figure 2 Three ways that threads bind the CPU of same level shared cache

图 2 线程绑定在共享同一级 Cache 上的 CPU 的三种方式

```
3     block=(N-2)/num_threads+1;
4   else
5     block=(N-2)/num_threads;
6   for…{//时间步迭代次数
7       # pragma omp parallel num_threads(num_threads)
8     {//并行区
9       myfun(omp_get_thread_num());/*线程与CPU绑定函数*/
10        for(i=omp_get_thread_num()*block;i<N-1&&i<(omp_get_thread_num()*block+block);i++);
11          //线程与数据块绑定
12          {
13            if(i==0)continue;/*线程号为0,1,2,…,(n-1),i必须从1开始*/
14            for(j=1;j<N-1;j++)
15              ……//模板更新
16          }
17        }
18  #pragma omp barrier//并行同步
19    …//交换
20  }
```

5 验证与分析

5.1 实验环境

(1)软件环境:AMCC X-Gene2 处理器上的操作系统为 Applied Micro Linux,内核 Linux version 3.12.0-mustang_sw_1.12.09-beta;FT-1500A 处理器上的操作系统 Neokylin 4.0-la,内核 Linux version 3.14.0-rcl-201510-phytium+;编译器为 Linaro 组织针对 ARM 架构对 GCC 编译器进行了修正和优化移植的 Linaro GCC 5.3-201602 版(基于 GCC5.3.1),支持 OpenMP 4.0 规范。

(2)硬件环境:AMCC X-Gene2、FT-1500A。

5.2 实验内容与结果分析

本文以 2D-5P 和 3D-7P 模板计算核心作为性能评估对象。线程数 *num_threads* 为核数 n(8 或 16),2D 模板测试数据 N 为 1 024,时间步长 k 为 100;3D 模板测试数据 N 为 512,时间步长 k 为 10;采用 OpenMP 指导指令进行并行扩展;每个测试程序测试 5 次,取其平均结果作为最终性能。

5.2.1 线程与 CPU 绑定的效果

本文主要测试了线程与 CPU 绑定的三种方式(1-1 模式、1-2 模式(AMCC)或 1-4 模式(FT-1500A))的性能,并与典型的 OpenMP 并行方法作对比,大部分优化方式提升了模板计算的性能,少数方式无效果或有轻微性能损失,测试结果如图 3 所示。由图 3 可知,在 AMCC X-Gene2 上,两种方式都使 2D-5P 模板计算有了很大的性能提升,分别提升了 15.41%和 7.86%,1-2 模式提升了 3D-7P 模板计算 0.90%的性能;在 FT-1500A 上,1-4 模式提升了 2D-5P 模板计算 1.07%的性能,两种方式都使 3D-7P 模板计算有了显著的性能提升,分别提升了 20.67%和 20.50%。总体而言,通过线程与 CPU 绑定,减少了线程调度的开销,提高了 Cache 的命中率,能使 2D-5P 和 3D-7P 模板计算获得很好的并行优化效果。

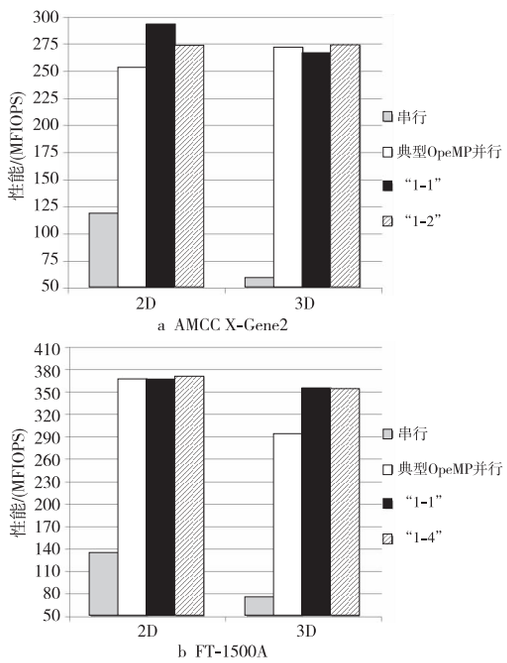

Figure 3 Parallelism results that stencil computation threads bind CPU

图 3 模板计算线程与 CPU 绑定的并行化效果

5.2.2 线程与数据块绑定的效果

本文同线程与 CPU 绑定的两种方式进行了测试对比,线程与数据块绑定使 2D-5P 模板计算的性能提升了。线程与数据块绑定对 2D-5P 模板计算的效果如图 4 所示,在 AMCC X-Gene2 上,在 1-1 模式和 1-2 模式上进行线程与数据块绑定的加速比分别为 1.03 和 1.05;在 FT-1500A 上,在 1-1 模式和 1-4 模式上进行线程与数据块绑定的加速比分别为 1.07 和 1.03。

5.2.3 两种 ARM64 多核微处理器平台的模板计算可扩展性评估

可扩展性[11]是指在确定的应用背景下,当现有架构的配置规模(处理机数)增大时,计算机系统

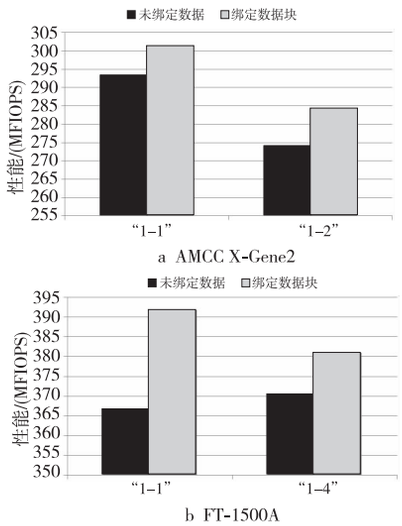

Figure 4 Result that 2D-5P stencil computation threads bind data block

图 4 2D-5P 模板计算线程与数据块绑定的效果

性能会线性上升。本文针对 2D-5P 模板计算的 1-1 模式以及在此模式上将线程与数据块绑定的方法,分别在 AMCC X-Gene2 和 FT-1500A 平台进行了可扩展性测试,结果如图 5 所示,在最大允许线程范围内,两个平台上的并行性能都随线程数增多而线性提升。因此,基于两维度绑定的优化方法在 ARM 64 位架构多核微处理器上表现出较好的可扩展性。

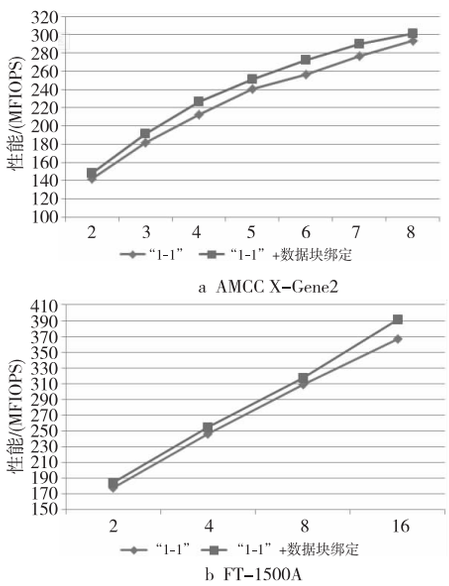

Figure 5 Scalability test for modified OpenMP parallelism

图 5 改进的 OpenMP 并行的可扩展性测试

6 结束语

本文提出了基于两维度绑定的优化方法,通过线程与 CPU 绑定,减少了 OpenMP 执行过程中线程调度的开销,增加了 Cache 的命中率,再通过线程与数据块绑定,增加了重用数据在 Cache 的命中率,并进行了实验验证。从实验结果可知,此方法提升了模板计算在 ARM64 架构多核微处理器上的并行性能,且并行性能随线程数增多而不断提升,表现出较好的可扩展性。

参考文献:

[1] ARM. The ARMv8-A architecture [EB/OL]. [2016-08-11]. http://www.arm.com/products/processors/armv8-architecture.php.

[2] ARM. cortex-A53 processor[EB/OL]. [2016-08-11]. http://www.arm.com/products/processors/cortex-a/cortex-a53-processor.php.

[3] ARM. cortex-A57 processor[EB/OL]. [2016-08-11]. http://www.arm.com/products/processors/cortex-a/cortex-a57-processor.php.

[4] Qi Jin, Li Kuang, Yang Can-qun, et al. Accelerating 3D GVF field conputation on Xeon Phi using stencil optimization [J]. Computer Engineering & Science, 2014, 36(8): 1435-1440. (in Chinese)

[5] Christen M, Schenk O, Neufeld E, et al. Parallel data-locality aware stencil computations on modern micro-architectures [C]// Proc of IEEE International Symposium on Parallel & Distributed Processing, 2009: 1-10.

[6] Wikipedia. Stencil code[EB/OL]. [2016-08-11]. https://en.wikipedia.org/wiki/Stencil_code.

[7] Lester B P. The art of parallel programming[M]. NJ: Prentice-Hall Inc, 1993.

[8] Wikipedia. Laplace's equation [EB/OL]. [2016-08-11]. https://en.wikipedia.org/wiki/Laplace%27s_equation.

[9] Wikipedia. Von_Neumann_neighborhood[EB/OL]. [2016-08-11]. https://en.wikipedia.org/wiki/Von_Neumann_neighborhood.

[10] Chapman B, Jost G, van der Pas R. Using OpenMP: Portable shared memory par-allel programming[J]. Journal of Computer Science & Technology, 2008, 10(3): 157-158.

[11] Sivasubramaniam A. An approach to scalability study of shared memory parallel systems[J]. ACM SIGMETRICS Performance Evaluation Review, 1993, 22(1): 171-180.

附中文参考文献:

[4] 齐金,李宽,杨灿群,等. Xeon Phi 平台上基于模板优化的 3D GVF 场计算加速[J]. 计算机工程与科学, 2014, 36(8): 1435-1440.

作者简介:

冯璐霞(1992 -),女,湖南湘潭人,硕士生,研究方向为计算机体系结构和编译优化。**E-mail**:932744732@qq.com

FENG Lu-xia, born in 1992, MS candidate, her research interests include computer architecture, and compilation optimization.