Journal of Image and Signal Processing 图像与信号处理, 2016, 5(4), 147-154
Published Online October 2016 in Hans. http://www.hanspub.org/journal/jisp
http://dx.doi.org/10.12677/jisp.2016.54017

Hans汉斯

Research and Realization of OpenMp Based on Muiticore DSP

Qi Zhang, Zhengyong Wang, Yanmei Yu

Image Information Institute, College of Electronics and Information Engineering, Sichuan University, Chengdu Sichuan
Email: 863354126@qq.com

Received: Sep. 18th, 2016; accepted: Oct. 4th, 2016; published: Oct. 7th, 2016

Abstract

Aiming at the complexity of multicore model in practical application, the three major models of data flow, master-slave and OpenMp are analyzed in this paper. Because the OpenMp model is easier to be implemented, it is taken as the major research object. Firstly, the implementation principle of OpenMp model is studied; the time consumed by the thread creation and task partitioning in the nested loop of the OpenMp model is analyzed in this paper. Then, the relationship between the core number and the number of threads in the OpenMp model is also studied. Finally, taking TMS320C6678 multi-core DSP of TI Company as the core processor, a simple image processing algorithm is used to verify the conclusions drawn from the above research. The results of test have proved OpenMp model when the number of threads is equal to the core number shortest execution time.

Keywords

TMS320C6678, OpenMp, Linear Assembly, Threads Scheduling

基于多核DSP的OpenMp研究与实现

张　琪，王正勇，余艳梅

四川大学电子信息学院图像信息研究所，四川　成都
Email: 863354126@qq.com

收稿日期：2016年9月18日；录用日期：2016年10月4日；发布日期：2016年10月7日

文章引用: 张琪, 王正勇, 余艳梅. 基于多核 DSP 的 OpenMp 研究与实现[J]. 图像与信号处理, 2016, 5(4): 147-154.
http://dx.doi.org/10.12677/jisp.2016.54017

摘　要

针对目前在实际应用场景中多核模型的复杂性，本文在分析了数据流、主从、OpenMp三大多核模型后，选取其中更易实现的OpenMp模型作为主要研究对象。本文研究了OpenMp模型的实现原理，对多核模型OpenMp在嵌套循环中的线程创建以及任务划分所消耗的时间进行了分析，然后研究了OpenMp模型的核数和线程数之间的关系，最终以TI公司的多核DSP TMS320C6678为核心处理器，使用典型的图像处理算法对在以上研究中得出的结论进行了验证。经测试，OpenMp模型在线程数等于核数时执行时间最短。

关键词

TMS320C6678，OpenMp，线性汇编，线程调度

1. 引言

随着社会的发展，摩尔定律预言的半导体发展在 2013 年年底开始逐渐放缓，于是单芯片多处理器的解决方案成为改善处理器性能的一个重要方向，使用多核处理器已经成为一种趋势，但对于多核 DSP 处理器而言，主从模型、数据流模型、OpenMp 模型是现在多核 DSP 处理器的三大主要并行程序处理模型。其中主从模型是以主核作为控制核心，从核在主核的调度下完成任务处理，从核之间工作相互独立，即如果其中一个核停止工作，不会影响其他核的执行；数据流模型是一个线性的处理模型，其工作方式按流水线的方式依次串行处理；OpenMp 则是由一个主线程开始，程序中的串行部分由主线程完成，并行的部分是由派生的其他线程来执行，但是如果并行部分没有结束，则不会执行后面的串行部分。

比较主从模型、数据流模型和 OpenMp 模型这三种方式，主从模型、数据流模型较为复杂，并且为了考虑负载的平衡，需要考虑算法处理时间的协调、任务的合理分割以及映射核的处理，而 OpenMp 使用就相对来说简单一些，大多数情况下只需要考虑该循环数据上的相关性，然后在其循环前面加上 OpenMp 的相关指令，就可以实现并行，这就降低了并行编程的难度和复杂度[1]。文献[2]利用 OpenMp 对 HEVC 进行解码，但是没有深入分析核数和线程数的关系，文献[3]分析了在不同处理平台下核数和线程数对处理速度的影响，但是没有针对具体算法进行验证。

因此本文基于 TMS320C6678 硬件平台、采用 OpenMp 模型、以图像处理中值滤波作为分析验证，分析了多核多线程的调度，然后以 512 × 512 的椒盐噪声图片为素材，采用 3 × 3 的滤波模板实现了图片的去噪。实验结果表明，在 TMS320C6678 (1 Ghz 8 核)上经过各种优化后得到处理的平均时间为 34.5 ms；比较多核 DSP 的单核处理时间与八核处理时间，得出在 TMS320C6678 上使用 OpenMp 实现中值滤波算法加速达到 7.88 倍。

2. TMS320C6678 介绍

多核定点浮点信号处理器 TMS320C6678 是基于 KeyStone 多核架构，整合了 8 个 C66x 核，每个核的频率可以运行在 1.0 GHz 到 1.25 GHz，提供每秒高达 40 GB MAC 定点运算和 20 GB FLOP 浮点运算能力，可在 10 W 功耗下实现 160GFLOP (Giga-Floating Point Operation per Second)浮点计算性能[4]。因此 TMS320C6678 非常适用于高速信号处理领域，如医学成像，虚拟现实以及人工智能等。此外，C66x 核 DSP 也是 TI 第一款支持 OpenMp API 的多核 DSP [5]。

3. OpenMp 介绍

OpenMp 是由 OpenMP Architecture Review Board 开始提出的，是用共享内存并行系统的多线程程序设计的一套指导性的编译处理方案(Compiler Directive)，并且已经被广泛采用。OpenMP 支持的编程语言包括 C、C++和 Fortran [6]，支持 OpenMp 的编译器包括 Sun Compiler、GNU Compiler 和 Intel Compiler 等。OpenMp 提供了对并行算法的高层抽象描述，程序员通过在源代码中加入专用的 pragma 来指明自己的意图，由此编译器可以自动将程序进行并行化，并在必要之处加入同步互斥以及通信信号。当选择忽略这些 pragma，或者编译器不支持 OpenMp 时，程序又可退化为通常的程序(一般为串行)，代码仍然可以正常运作，只是不能利用多线程来加速程序执行。

OpenMp 的实现是基于 fork-join 模型的，主线程顺序执行，当主线程遭遇并行区域时，分配的子线程按照任务各自执行，直到并行区域结束[7]，主线程处理流程如图 1 所示。

在 C/C++中，OpenMP 指令使用的格式为：

#pragma　omp　指令　[子句[子句]…]

其中常用指令如表 1 所示。

其中常用字句如表 2 所示。

其中常用调度类型如表 3 所示。

4. OpenMp 的并行分析

4.1. OpenMp 在嵌套循环中的分析

在图像处理领域中，for 循环扮演着不可或缺的角色，而本文 OpenMp 也是主要针对 for 循环进行展开的，实际处理中 OpenMp 指令在 for 循环中位置不同，程序执行的方式也会随之不同，图 2 为 OpenMp 针对 for 循环的两种表现形式。

图 2(a)这种放置位置表示把最外层的 for 循环进行并行处理，即对循环变量 i 进行线程创建以及线程分配，假设其创建线程花费时间为 a，线程分配花费时间为 b，则其花费的总时间为 $a + b$，图 2(b)表示把嵌套里面的 for 循环进行并行处理，即对循环变量 j 进行线程创建以及线程分配，那么其花费的总时间为 $i \times (a + b)$，显然 $i \times (a + b) > a + b$；在某些处理的时候还需要对特定变量进行写保护，即有且只能让一个线程去操作该变量，但是由于每个核派生的任务不一样，造成执行完任务的时间也会不相同，所以当操作该变量时可能需要等待，那么这也导致程序执行效率降低。

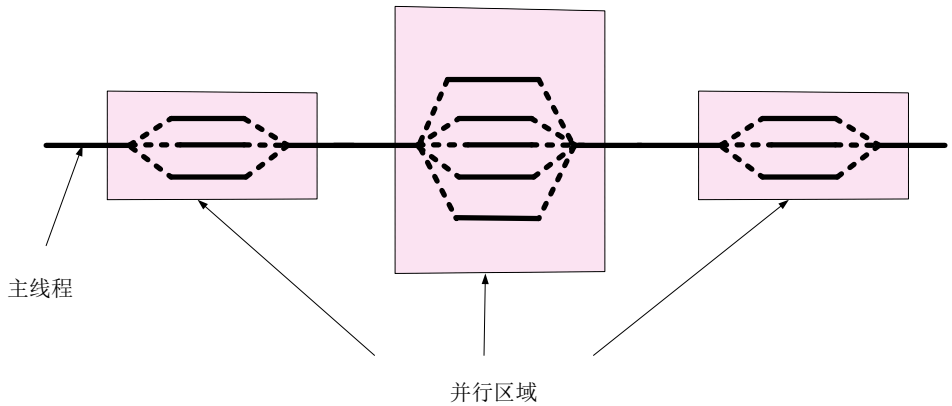

Figure 1. OpenMp main thread processing
图 1. OpenMp 主线程处理流程

Table 1. Commonly used instructions
表 1. 常用指令

Parallel	表示这段代码将被多个线程并行执行
For	一般用在 for 循环之前，表示将循环分配到多个线程中并行执行；也可以与 parallel for 相结合，表示 for 循环的代码将被多个线程并行执行
Sections	一般用在可能会被并行执行的代码段之前
Critical	一般用在代码的临界区之前

Table 2. The commonly used words
表 2. 常用字句

Private	表示每个线程都拥有自己的变量私有副本
Schedule	表示如何调度 for 循环迭代
Shared	表示一个或多个变量将成为多个线程间的共享变量

Table 3. Commonly used scheduling type
表 3. 常用调度类型

Dynamic	循环变量区域分为 *n* 等份，某个线程执行完 1 份之后再执行其他需要执行的那一份任务
Guided	循环变量区域由大到小分为不等的 *n* 份，运行方法类似于 dynamic
Runtime	在运行时使用上述三种调度策略中的一种，默认是使用 static
Static	循环变量区域分为 *n* 等份，每个线程平分 *n* 份任务

```
#pragma omp parallel for
    for (i=0; i<h ; i++)
     {
      for(j=0; j< w; j++)
        {
            循环体；
        }
     }
```

(a)

```
for (i=0; i<h; i++)
 {
  #pragma omp parallel for
      for(j=0; j< w; j++)
        {
          循环体；
        }
 }
```

(b)

Figure 2. Two kinds of forms of OpenMp in view of the for loop
图 2. OpenMp 针对 for 循环的两种表现形式

4.2. OpenMp 中关于线程数与核数的分析

当使用 OpenMp 模型时，还应该对其核数与线程数加以考虑，假设在一个任务中，其核数为 m，线程数为 n，当 $m > n$ 时，即核数大于线程数，那么操作系统在分配线程时，将存在一些核分配不到线程，而造成核出现饥饿状态；当 $m < n$ 时，即核数小于线程数，那么操作系统在分配线程时，将存在一些线程由于核数不够导致其出现等待状态；综上，我们可以得出当线程数等于核数时，即没有线程等待，也没有核出现饥饿，即一个核执行一个线程。

5. OpenMp 的实现

5.1. OpenMp 实现中值滤波

中值滤波是图像处理领域中比较基础的处理之一，是基于排序统计理论的一种有效抑制噪声处理技

术，原理是把数字图像中一点的值用该点的一个邻域中各点值的中值代替，让周围的像素值更加接近真实值，从而消除孤立噪点。本文在分析了 OpenMp 的一些原理及调度方式后，使用中值滤波处理算法作为验证，之所以选择中值滤波是由于其处理时可以通过使用其他共享变量来存储执行结果而不会相互干扰；在本文中，采取 static、dynamic、guided、runtime 调度方式衡量程序的运行时间。在中值滤波的循环前面采用关键字#pragma omp parallel for 来实现程序处理的并行，采用 private 实现变量的私有化，即对 for 循环中的循环变量进行保护，使每一个线程都能独立地拥有循环变量的私有拷贝。

TMS320C6678 一共拥有 8 个核，每个核都能实现最高 1.25 Ghz 的主频，但在本文中，使用 1 Ghz 作为每个核的主频，然后线程设置为 8 线程，即每个核分配一个线程，然后对其执行的时间进行比对，部分程序展示如图 3 所示，private (*i*, *j*, arry)表示循环变量 *i*，*j*，arry 在每个线程里都拥有变量私有副本，其他线程不能访问，shared (dataout)表示 8 个线程共同使用 dataout 数组，schedule (guided)表示调度方式为 guided，即在起始阶段每个线程会被分配到较大的迭代块，后面分配到的迭代块将依次递减，直到减为 1，sort (arry)表示对 arry 数组里的像素进行排序，dataout[] = arry[5]表示把排序后的中间像素值赋给当前位置。

5.2. 线性汇编优化

线性汇编语言是介于 C/C++和汇编语言之间的一种语言，它相对于汇编语言编写相对简单，TI 官方有提供线性汇编的编程格式与标准[8] [9]。

中值滤波函数的原理是用其周围邻域的中值代替当前像素值，方法是对其模板内的像素进行排序，生成单调上升(或下降) [10]，由于该排序函数在遍历该图像时需要对图像中的每一个像素点进行排序，故对其排序函数进行线性汇编优化，优化条件是在 8 核 8 线程 DSP 下，对 512 × 512 图片采用 3 × 3 中值滤波模板，实验结果如表 4 所示。

5.3. 其他优化

1) 字节对齐：对于程序中经常访问的存储空间，采用 32 字节对齐，以空间换取执行时间。

2) 缓存优化：由于 CPU 执行速度快于存取速度，所以在执行程序时采用缓存，本文缓存设置为：64K L2，32K L1，32K P1。

3) -O3 优化：文件级优化，会大量地消除未使用的函数及变量，本文在最后采用此优化，效果明显。

6. 实验结果及数据分析

图 4 为 OpenMP 不同调度方式下，循环迭代次数与执行时间的关系。由图中可看出，使用 runtime 类型时，即运行时选择前面三种调度类型中的一种时，迭代次数对执行时间没有影响；而 guided、static、dynamic 在自动调度和循环迭代次数相对较少时表现更好，如果给其分配 size 过大，反而执行时间变长，这是由于分配给线程的迭代次数过多，即花费了大量的时间去创建线程以及分配线程浪费了时间；但是当循环迭代次数达到一定值时，即大于 4000 次时，其执行时间几乎没有变化，这是由于分配给线程的迭代次数过大后，即把循环中所有的线程都分配给一个核去处理了，其他核则分配不到线程。

图 5 为 OpenMP 线程数、核数与执行时间的关系图。首先把核数等于线程数作为分界线，从图中看出，当核数大于线程数时，程序的执行时间随着线程数的增加而减少，当核数小于线程数时，程序的执行时间随着线程数的增加而几乎不变，原因是由于当核数大于线程数时，某些核分配不到线程，造成核处于饥饿状态，因此，当线程数增加时，饥饿的核由于分配到线程，处理时间减少；当核数小于线程数时，是因为无论线程怎么增加，执行程序的核是一定的，在没执行完上一个线程之前，是不会执行下一个线程的，所以执行时间几乎没有改变；综上，当核数接近线程数时程序执行时间最短。

测试背景是基于四幅 512 × 512 的带椒盐噪声的彩色图片，原图和中值滤波后的测试结果图如图 6 所示。在经过线性汇编、-O3 等优化后得到单核下平均处理时间为 272.1 ms，在 8 核 8 线程下平均处理时间为 34.5 ms。

Table 4. Linear assembly optimization
表 4. 线性汇编优化

	不优化	线性汇编优化
耗时时间	288 ms	137 ms

```
#pragma omp parallel for private(i,j,arry,…)shared(dataout,…) schedule(guided)
    for( i=2;i<h-2;i++)
    {
        for( j=2;j<w-2;j++)
        {
            ……
            sort(arry);
            dataout[]=arry[5];
            ……
        }
    }
```

Figure 3. Median filtering openmp part program
图 3. 中值滤波 openmp 部分程序

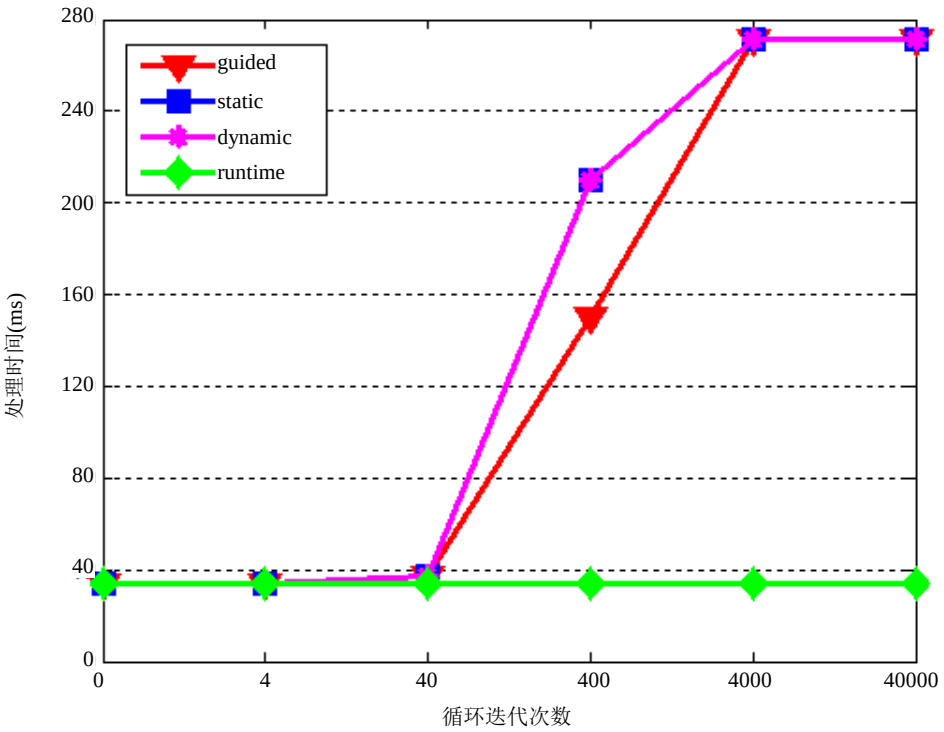

Figure 4. OpenMP scheduling and loop iterations execution time
图 4. OpenMP 调度方式以及循环迭代次数执行时间

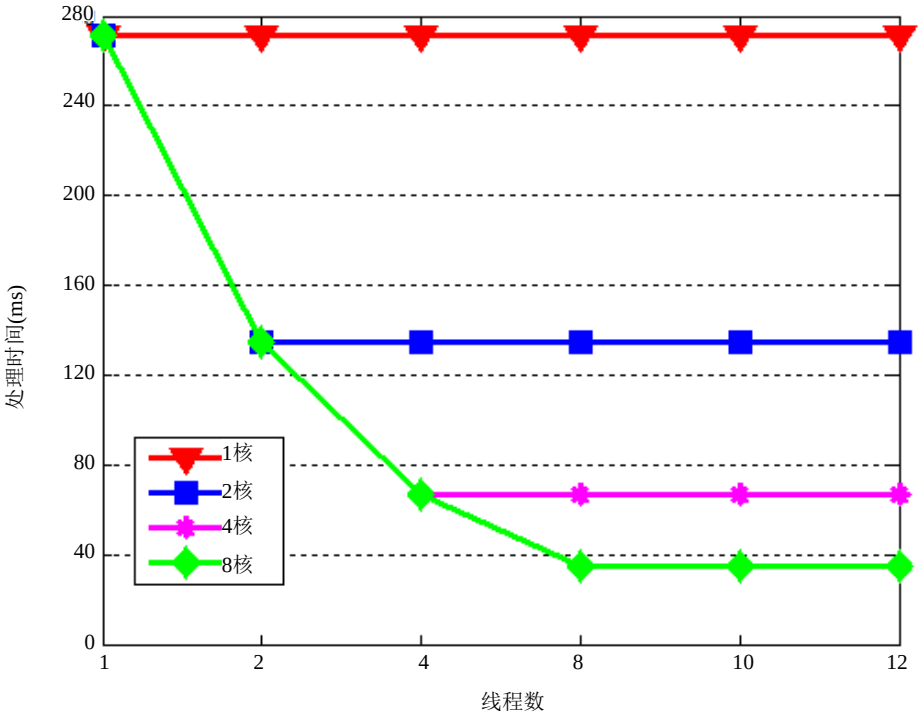

Figure 5. OpenMP the number of threads and nuclear execution time
图 5. OpenMP 线程数与核数的执行时间对比

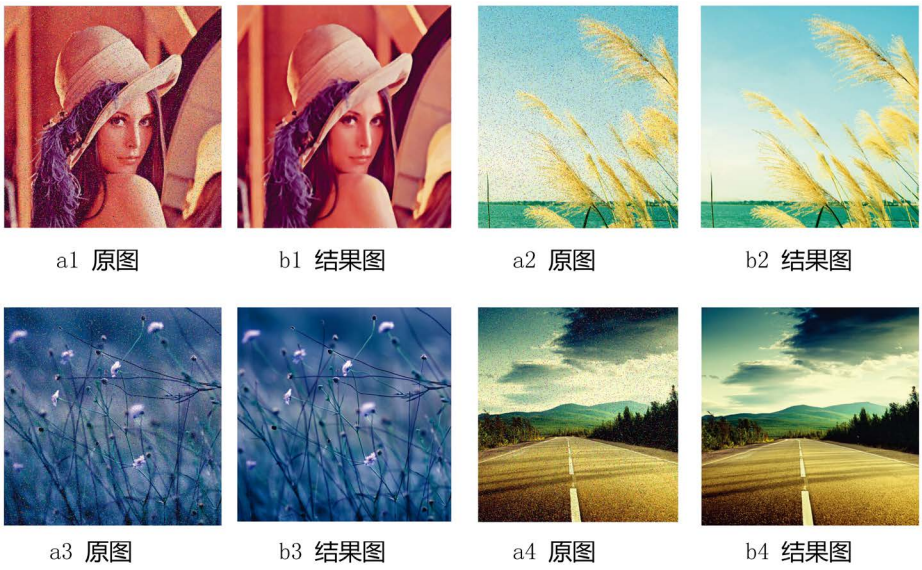

Figure 6. Test the original image and the result
图 6. 测试原图与结果图

7. 结论

本文对多核 DSP 的三种模型 OpenMp、主从模型、数据流模型做了一些简要介绍，由于 OpenMp 模型在使用上相对于其他两种模型简单，编程者只需要在程序中加入适当的编译器指令来表其们的意图，程序将自动并行化。本文以 512 × 512 的图片为例进行图像中值滤波处理，并对 OpenMp 模型的调度方式

和线程数量进行测试，最终图像在经过线性汇编等一系列优化后得到单核下的平均处理时间为 272.1 ms，8 核 8 线程下的平均处理时间为 34.5 ms，测试结果表明了基于多核 DSP 的 OpenMp 模型在处理类似像素点或则像素块问题上具有良好的性能，并且根据实验数据可得出，OpenMp 模型在线程数等于核数时执行时间最短。

参考文献 (References)

[1] Mou, X.G., Wei, G.H. and Zhou, X. (2014) Parallel Programming and Optimization Based on TMS320C6678. *Applied Mechanics & Materials*, **615**, 259-264. http://dx.doi.org/10.4028/www.scientific.net/AMM.615.259

[2] Chavarrias, M., Pescador, F., Garrido, M.J., *et al*. (2015) A Multicore DSP HEVC Decoder Using an Actorbased Dataflow Model and OpenMP. *IEEE Transactions on Consumer Electronics*, **61**, 236-244. http://dx.doi.org/10.1109/TCE.2015.7150599

[3] Zheng, Z., Chen, X., Wang, Z., *et al.* (2011) Performance Model for OpenMP Parallelized Loops. *IEEE International Conference on Transportation, Mechanical, and Electrical Engineering* (*TMEE*), 16-18 December 2011, 383-387. http://dx.doi.org/10.1109/tmee.2011.6199223

[4] Texas Instrument. DataSheet Tms320c6678 Multicore Fixed and Floating-Point Digital Signal Processor. SPRS691D April 2013.

[5] Texas Instrument (2012) OpenMP Programming for Key Stone Multicore Processors.

[6] Mego, R. and Fryza, T. (2013) Performance of Parallel Algorithms Using OpenMP. *Radioelektronika*, 16-17 April 2013, 236-239. http://dx.doi.org/10.1109/radioelek.2013.6530923

[7] Texas Instrument (2011) OpenMP Programming for TMS320C66x Multicore DSPs.

[8] Texas Instrument (2012) TMS320C6000 Optimizing Compiler v7.4 User's Guide. SPRU187U.

[9] Texas Instrument (2012) TMS320C6000 Assembly Language Tools v7.4 User's Guide. SPRU186W.

[10] 张丽, 陈志强, 高文焕, 等. 均值加速的快速中值滤波算法[J]. 清华大学学报: 自然科学版, 2004, 44(9): 1157-1159.