

译 OpenMP 参考（指令详解）

2011年03月21日 17:53:00

阅读数：11962



1



目录



收藏



评论



微信



微博



QQ

共享工作（Work-Sharing）结构

- 共享工作结构将它作用的代码段在抵达此代码段的线程中分开执行
- Work-sharing 不会产生新线程
- 进入工作共享结构时没有关卡，但结束时会有

Work-Sharing 结构的种类：

注意: 关于Fortran 的workshare 结构稍后讨论

DO / for - 在一组线程中共享循环中的计数器 (iteration) . 表示一种 "数据并行处理".

SECTIONS - 将工作分成独立，不关联的片段。每个片段被一个线程执行。用来实现一种“函数并行处理”

SINGLE - 串行化代码段

规则：

- 为了能并行执行此指令，一个工作共享结构必须被并行区域动态封闭
- 一个线程组到达一个工作共享结构时，要么全部多线程被占用，要么不占用
- 线程组的所有成员必须以相同的顺序到达连续的工作共享结构

DO / for 指令

目的：

- DO/for指令表明接下来的循环将被并行执行。前提是一个并行区域已经被初始化，否则以串行的方式执行。

格式：

Fortran

```
!$OMP DO [clause ...]

SCHEDULE (type [,chunk])

ORDERED

PRIVATE (list)

FIRSTPRIVATE (list)

LASTPRIVATE (list)
```

```
REDUCTION (operator | intrinsic : list)
```

```
COLLAPSE (n)
```

```
do_loop
```

```
!$OMP END DO [ NOWAIT ]
```

```
#pragma omp for [clause ...] newline
```

```
schedule (type [,chunk])
```

```
ordered
```

```
private (list)
```

```
firstprivate (list)
```

```
lastprivate (list)
```

```
shared (list)
```

```
reduction (operator: list)
```

```
collapse (n)
```

```
nowait
```

```
for_loop
```

C/C++

子句:

- **SCHEDULE**: 描述线程组中的线程分配多少次循环。默认schedule依赖于具体实现

STATIC

循环按照chunk的值被等分并静态赋予线程。如果没有指定chunk, 那么将均分此循环。

DYNAMIC

循环按照chunk的值被等分并动态赋予线程。如果一个线程完成了一个分支, 那么它将继续运行下一个分支。默认的chunk值为1.

如果chunk为1，每个分支的值为剩余的循环次数除以线程数量，减少到1.如果chunk的值为K（比1大），每块分支也以同样的方式分配迭代次数，只是每块的迭代次数不小于K。默认chunk的值为1.

RUNTIME

调度的策略依赖于运行时的环境变量OMP_SCHEDULE.为这个子句指定chunk的值是非法的。

AUTO

调度策略依赖于编译器或者运行时系统

- **NO WAIT / nowait**: 如果指定了，线程在并行循环结束时不会同步.
- **ORDERED**: 指定该循环迭代必须以串行方式执行.
- **COLLAPSE**:指定在一个嵌套循环中，多少次循环被折叠到一个大的迭代空间并根据schedule子句来划分。所有相关循环中的迭代的执行顺序决定迭代空间中折叠迭代的顺序。
- 其他的子句将在[Data Scope Attribute Clauses](#) 章节介绍

规则:

- DO循环不能是DO WHILE循环，也不能没有循环控制。循环迭代变量必须是整型，并且对于所有的线程，循环控制参数都一样
- 程序的正确性不能取决于那个线程执行了特定的迭代
- 与DO/for 指令关联的循环要按照标准格式书写
- 由于不同的线程在赋值时并不同步，chunk的值必须指定为一个循环的不变整型表达式
- RDERED, COLLAPSE 和SCHEDULE 子句只可选其一.
- 查看OpenMP的相关文档获得额外的信息

范例: DO / for 指令

- 对vector执行加法操作的简单程序
 - 数组A, B, C, and 变量N 被所有线程共享.
 - 变量I对每个线程都是私有的；每个线程都会有一个它的唯一copy .
 - 循环的迭代将被动态地以CHUNK为单位分发.
 - 线程在完成各自的独立的工作时并不进行同步(NOWAIT).

Fortran - DO Directive Example

```
PROGRAM VEC_ADD_DO

INTEGER N, CHUNKSIZE, CHUNK, I

PARAMETER (N=1000)

PARAMETER (CHUNKSIZE=100)

REAL A(N), B(N), C(N)

!   Some initializations

DO I = 1, N
```

```
B(I) = A(I)
```

```
ENDDO
```

```
CHUNK = CHUNKSIZE
```

```
!$OMP PARALLEL SHARED(A,B,C,CHUNK) PRIVATE(I)
```

```
!$OMP DO SCHEDULE(DYNAMIC,CHUNK)
```

```
DO I = 1, N
```

```
C(I) = A(I) + B(I)
```

```
ENDDO
```

```
!$OMP END DO NOWAIT
```

```
!$OMP END PARALLEL
```

```
END
```

C / C++ - for Directive Example

```
#include <omp.h>
```

```
#define CHUNKSIZE 100
```

```
#define N 1000
```

```
main ()
```

```
{
```

```
int i, chunk;
```

```
float a[N], b[N], c[N];
```

```
/* Some initializations */

for (i=0; i < N; i++)

    a[i] = b[i] = i * 1.0;

chunk = CHUNKSIZE;

#pragma omp parallel shared(a,b,c,chunk) private(i)

{

    #pragma omp for schedule(dynamic,chunk) nowait

    for (i=0; i < N; i++)

        c[i] = a[i] + b[i];

} /* end of parallel section */

}
```

SECTIONS 指令

目的:

- SECTIONS指令是一种非迭代式工作共享结构。它指定段中的代码被分配到线程组的线程中执行。
- 独立的SECTION指令嵌套在SECTIONS指令中。每个SECTION仅被线程组中的一个线程执行。不同的段可能被不同的线程执行。在实现允许的情况下，如果一条线程足够快，它可能会执行好几个代码段。

格式:

Fortran

```
!$OMP SECTIONS [clause ...]

PRIVATE (list)

FIRSTPRIVATE (list)

LASTPRIVATE (list)
```

```
!$OMP SECTION
```

```
    block
```

```
!$OMP SECTION
```

```
    block
```

```
!$OMP END SECTIONS [ NOWAIT ]
```

C/C++

```
#pragma omp sections [clause ...] newline
```

```
    private (list)
```

```
    firstprivate (list)
```

```
    lastprivate (list)
```

```
    reduction (operator: list)
```

```
    nowait
```

```
{
```

```
    #pragma omp section newline
```

```
        structured_block
```

```
    #pragma omp section newline
```

```
structured_block
```

```
}
```

子句:

- 如果NOWAIT/nowait子句没有被使用, 在SECTIONS指令结束时会有一个隐藏的关卡 (barrier)
- 更多子句在[Data Scope Attribute Clauses](#) 章节详细讨论.

问答:

如果线程数量和SECTIONS的数量不一样会怎样? 比SECTIONS多呢? 比SECTIONS少呢?

哪条线程执行哪块SECTION?

规则:

- section块要按照格式书写 (It is illegal to branch into or out of section blocks.)
- SECTION指令只能出现在SECTIONS指令的静态范围内

范例: **SECTIONS** 指令

- 展示不同的工作块被不同的线程执行的简单程序

Fortran - SECTIONS Directive Example

```
PROGRAM VEC_ADD_SECTIONS

INTEGER N, I

PARAMETER (N=1000)

REAL A(N), B(N), C(N), D(N)

! Some initializations

DO I = 1, N

    A(I) = I * 1.5

    B(I) = I + 22.35

ENDDO

!$OMP PARALLEL SHARED(A,B,C,D), PRIVATE(I)
```

```
!$OMP SECTION

DO I = 1, N

    C(I) = A(I) + B(I)

ENDDO

!$OMP SECTION

DO I = 1, N

    D(I) = A(I) * B(I)

ENDDO

!$OMP END SECTIONS NOWAIT

!$OMP END PARALLEL

END
```

C / C++ - sections Directive Example

```
#include <omp.h>

#define N      1000

main ()

{

    int i;

    float a[N], b[N], c[N], d[N];

    /* Some initializations */

    for (i=0; i < N; i++) {
```



```
b[i] = i + 22.35;

}

#pragma omp parallel shared(a,b,c,d) private(i)

{

#pragma omp sections nowait

{

#pragma omp section

for (i=0; i < N; i++)

    c[i] = a[i] + b[i];

#pragma omp section

for (i=0; i < N; i++)

    d[i] = a[i] * b[i];

} /* end of sections */

} /* end of parallel section */

}
```

WORKSHARE 指令

目的:

- Fortran 适用
- WORKSHARE指令将密封的结构块分成一些独立的工作单元来执行，每个工作单元被执行一次。
- 结构块只能由如下组成：
 - array assignments
 - scalar assignments
 - FORALL statements
 - FORALL constructs
 - WHERE statements

- critical constructs
- parallel constructs

- 更多信息请查看OpenMP API 文档.

格式:

```
!$OMP WORKSHARE

Fortran      structured block

!$OMP END WORKSHARE [ NOWAIT ]
```

规则:

- 除了ELEMENTAL函数, 该结构不能包含任何用户函数调用.

范例: **WORKSHARE** 指令

- Simple array and scalar assignments shared by the team of threads. A unit of work would include:
 - Any scalar assignment
 - For array assignment statements, the assignment of each element is a unit of work

Fortran - WORKSHARE Directive Example

```
PROGRAM WORKSHARE

INTEGER N, I, J

PARAMETER (N=100)

REAL AA(N,N), BB(N,N), CC(N,N), DD(N,N), FIRST, LAST

!      Some initializations

DO I = 1, N
```

```

        AA(J,I) = I * 1.0

        BB(J,I) = J + 1.0

    ENDDO

ENDDO

!$OMP PARALLEL SHARED(AA,BB,CC,DD,FIRST, LAST)

!$OMP WORKSHARE

    CC = AA * BB

    DD = AA + BB

    FIRST = CC(1,1) + DD(1,1)

    LAST = CC(N,N) + DD(N,N)

!$OMP END WORKSHARE NOWAIT

!$OMP END PARALLEL

END

```

SINGLE Directive

目的:

- SINGLE 指令表示其作用的代码将单线程执行.
- 当执行非线性安全代码段（如I/O）时会派上用场

格式:

Fortran

```
!$OMP SINGLE [clause ...]
```

```
FIRSTPRIVATE (list)
```

```
block
```

```
!$OMP END SINGLE [ NOWAIT ]
```

```
#pragma omp single [clause ...] newline
```

```
private (list)
```

```
firstprivate (list)
```

```
nowait
```

C/C++

```
structured_block
```

子句:

- 除非指定了NOWAIT/nowait指令，线程组中没有执行SINGLE指令的线程将在密封代码块的结束处等待。
- 更多的子句描述见[Data Scope Attribute Clauses](#) 章节。

规则:

- SINGLE块要按照格式书写（It is illegal to branch into or out of a SINGLE block.）

合并的并行工作共享结构

OpenMP提供了三个方便指令：

- PARALLEL DO / parallel for
 - PARALLEL SECTIONS
 - PARALLEL WORKSHARE (fortran 适用)
- 大多数情况下，这些指令的行为和一个独立的PARALLEL指令后面紧跟一个单独的work-sharing指令相同
 - 大多数对指令的法则，子句和限制都有效。更多细节参考OpenMP API.
 - 下面是一个使用PARALLEL DO/PARALLEL FOR联合指令的范例。

Fortran - PARALLEL DO Directive Example

```

INTEGER N, I, CHUNKSIZE, CHUNK

PARAMETER (N=1000)

PARAMETER (CHUNKSIZE=100)

REAL A(N), B(N), C(N)

!   Some initializations

DO I = 1, N

    A(I) = I * 1.0

    B(I) = A(I)

ENDDO

CHUNK = CHUNKSIZE

!$OMP PARALLEL DO

!$OMP& SHARED(A,B,C,CHUNK) PRIVATE(I)

!$OMP& SCHEDULE(STATIC,CHUNK)

DO I = 1, N

    C(I) = A(I) + B(I)

ENDDO

!$OMP END PARALLEL DO

END

```

C / C++ - parallel for Directive Example

```
#include <omp.h>

#define N      1000

#define CHUNKSIZE  100

main ()  {

    int i, chunk;

    float a[N], b[N], c[N];

    /* Some initializations */

    for (i=0; i < N; i++)

        a[i] = b[i] = i * 1.0;

    chunk = CHUNKSIZE;

    #pragma omp parallel for /

        shared(a,b,c,chunk) private(i) /

        schedule(static,chunk)

    for (i=0; i < n; i++)

        c[i] = a[i] + b[i];

}
```

TASK 构造

目的:

- OpenMP 3.0的新构造
- TASK构造定义一个显式任务，该任务被遇到的线程执行，或者被线程组中的其他线程推迟执行
- 数据共享属性子句决定它的数据环境
- Task的执行遵从任务调度-更多信息请查看OpenMP3.0参考文档

Format:

```
!$OMP TASK [clause ...]  
  
    IF (scalar expression)  
  
        UNTIED  
  
        DEFAULT (PRIVATE | FIRSTPRIVATE | SHARED | NONE)  
  
        PRIVATE (list)  
  
        FIRSTPRIVATE (list)  
  
        SHARED (list)  
  
        block  
  
!$OMP END TASK
```

C/C++

```
#pragma omp task [clause ...] newline  
  
    if (scalar expression)  
  
        untied  
  
        default (shared | none)  
  
        private (list)  
  
        firstprivate (list)
```

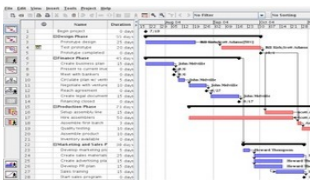
`structured_block`

子句和限制:

- 请参考 [OpenMP 3.0 规范文档](#)

文章标签: [parallel](#) [fortran](#) [newline](#) [list](#) [integer](#) [工作](#)

个人分类: [并行](#)



项目管理软件

17款免费且开源的项目管理工具

想对作者说点什么? [我来说一句](#)

OpenMP学习笔记

原文: http://blog.sina.com.cn/s/blog_66474b160100z15b.html 1. **OpenMP**是一种API, 用于编写可移植的多线程应用程序, 无需程序员进...

 zsc09_leaf 2012-07-17 10:07:38 阅读数: 32830

OpenMP Single

```
#include void work1() { std::cout
```

 liangjisheng 2017-05-28 17:25:32 阅读数: 140

Fortran&OpenMP - 1.环境配置 - CSDN博客

openmp常用指令(**fortran**版)(2) 1.**!OMP DO**!**OMP END DO**指令对使最近的**do**循环并行执行,将**do**循环分散到不同的线程,每个线程仅仅计算部分迭代,所有的线程...

2018-6-11


OpenMP 参考(指令详解) - CSDN博客

!\$OMP END PARALLEL END SINGLE Directive 目的: **SINGLE** 指令表示其**作用**的代码将...**PARALLEL DO** / parallel for **PARALLEL SECTIONS PARALLEL WORKSHARE** (**fortran** 适用...

2018-5-30

一起来学OpenMP (9) ——线程同步之事件同步机制

一、引言 前边已经提到过, 线程的同步机制包括互斥锁同步和事件同步。互斥锁同步包括atomic、critical、mutex函数, 其机制与普通多线程同步的机制类似。而事件同步则通过**nowait**、**sec...**

 mydear_33000 2011-10-31 13:51:51 阅读数: 4677

OpenMP常用指令释义 - CSDN博客

openmp常用指令(**fortran**版)(2) 1.**!OMP DO**!**OMP END DO**指令对使最近的**do**循环并行执行,将**do**循环分散到不同的线程,每个线程仅仅计算部分迭代,所

OpenMP循环结构的并行 - CSDN博客

只是在**fortran**中还可以添加!**\$omp** end parallel **do**指令来表示循环并行结束。OpenMP还提供了用于控制并行执行的一些循环选项条件(clause),根据这些条件的类型可以分为...

2018-2-10

并行编程OpenMP基础及简单示例

OpenMP基本概念 **OpenMP**是一种用于共享内存并行系统的多线程程序设计方案,支持的编程语言包括C、C++和Fortran。**OpenMP**提供了对并行算法的高层抽象描述,特别适合在多核CPU机器...

 dcrmg 2016-12-24 22:30:16 阅读数: 11425

OpenMP命令与子句

1、为了在**OpenMP**中创建线程,需要指定一些代码块并行地运行,C/C++中可以通过指定**#pragma omp parallel**完成。 2、**OpenMP**要求I/O库是线程级安全的,但它并...

 zaishaoyi 2015-06-16 15:34:25 阅读数: 1326

OpenMP & Fortran - CSDN博客

在**Fortran**中应用OpenMP,有时会遇到一行中openmp的编译指令太长,需要换行的情形。今天在网上看到一段例程,明白了该如何处理这种情形。!**\$omp** parallel &!**\$omp** ...

2018-5-22

fortran 语言使用 - 周健华的专栏 - 博客频道 - CSDN.NET

\$OMP END **DO** [NOWAIT] **fortran** 特有的方式WORKSHARE,在不能多线程计算的程序代码块,可以用这种方式声明只能有一个thread进行计算 **\$OMP** WORKS HARE FORALL ...

2016-7-26

OpenMP 参考 (同步构造)

原文: <https://computing.llnl.gov/tutorials/openMP/>

 saga1979 2011-03-29 09:42:00 阅读数: 5110

OpenMP学习笔记 - CSDN博客

#pragma omp critical{ ...} 2.2 atomic 作用同critical... tmp =**do**_lots_of...程序设计方案,支持的编程语言包括C、C++和**Fortran**...

2018-6-5

新手写openmp程序,结果不能运行,求教高手指点 - CSDN博客

我用visual studio2013 +intel **fortran**2013来编写... do k=row,2 !**\$OMP** DO SCHEDULE...流介绍 CUDA流在加速应用程序方面起着重要的作用。...

2018-5-23

OpenMP中的同步和互斥

在多线程编程中必须考虑到不同的线程对同一个变量进行读写访问引起的数据竞争问题。如果线程间没有互斥机制,则不同线程对同一变量的访问顺序是不确定的,有可能导致错误的执行结果。**OpenMP**中有两种不同类型...

 dcrmg 2016-12-26 22:41:21 阅读数: 3857

在C++中使用openmp进行多线程编程

声明: 本文是基于Joel Yliuoma写的Guid into **OpenMP**:Easy multithreading programming for C++而写的,基本是按照自己的理解,用自己语言组...

 acaiwlj 2015-11-13 15:53:40 阅读数: 8559

OpenMP 参考 (同步构造) - CSDN博客

#pragma omp atomic newline statement_expression 限制: 只**作用于**紧随其后的一...**Fortran** C / C++ BARRIER END PARALLEL CRITICAL and END CRITICAL END **DO** EN...

2018-6-13


OpenMP Tutorial学习笔记(5)OpenMP指令之共享工作构造(Work-Sharing)

OpenMP Tutorial: <https://computing.llnl.gov/tutorials/openMP/#WorkSharing> 共享工作构造: Work-Sharing Constr...

 gengshenghong 2011-11-10 22:15:47 阅读数: 2663

omp的一个不错的文章

实验平台: win7, VS2010 1. 介绍 并行计算机可以简单分为共享内存和分布式内存, 共享内存就是多个核心共享一个内存, 目前的PC就是这类 (不管是只有一个多核CPU还是可以插多...

 xuyiqiang87 2016-10-09 09:49:38 阅读数: 533

OpenMP3.0的新特性Task指令基础

从OpenMP3.0开始, OpenMP增加了task指令, 这是OpenMP3.0中最激动人心的一个新特性。本文的"术语"大多数是根据个人理解用词, 不保证用词准确性。

(1) task基础OpenMP Tu...

 Augustdi 2013-04-16 10:52:54 阅读数: 4823

OpenMP Tutorial学习笔记(7)OpenMP指令之任务构造 (Task Constructs)

OpenMP Tutorial: <https://computing.llnl.gov/tutorials/openMP/#Combined> 任务构造: Task Constructs ...

 gengshenghong 2011-11-14 19:08:11 阅读数: 2338

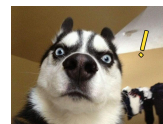
vs上C/C++并行计算#pragma omp

在一个vs内的工程进行并行计算, 首先先修改属性内的C/C++ 语言—OpenMP:是。头文件不一定需要#include #pragma omp parallel sections//告诉编译器有几...

 qq_19764963 2016-05-17 15:47:12 阅读数: 5057

程序猿不会英语怎么行? 英语文档都看不懂!

不背单词和语法, 一个公式教你读懂天下英文→



OpenMP中数据属性相关子句详解(2):shared/default/copyin/copyprivate子句的使用

(1) shared shared子句可以用于声明一个或多个变量为共享变量。所谓的共享变量, 是值在一个并行区域的team内的所有线程只拥有变量的一个内存地址, 所有线程访问同一地址。所以, 对于并行区域...

 gengshenghong 2011-11-22 14:29:39 阅读数: 4641

#pragma omp parallel for schedule(dynamic) private(i)

```
#include #include #include int main() { int i; #pragma omp parallel for sch...
```

 adream307 2011-06-21 12:35:00 阅读数: 2921

OpenMP与C++: 事半功倍地获得多线程的好处(下)

声明:本文并未获得翻译授权,本人翻译这篇文章仅用于学习和研究之用,任何人不得在未经授权之前将原文和译文用以商业用途.因版权原因,暂不建议转载本文.本文发表于<http://blog.csdn.net/>...

 lanphaday 2007-02-11 20:09:00 阅读数: 18305

OpenMP共享内存并行编程详解

实验平台: win7, VS2010 1. 介绍 并行计算机可以简单分为共享内存和分布式内存, 共享内存就是多个核心共享一个内存, 目前的PC就是这类 (不管是只有一个多核CPU还是可以插多...

 B10090411 2017-04-02 11:36:39 阅读数: 1194

一起来学OpenMP (4) ——数据的共享与私有化

一、引言 在并行区域中, 若多个线程共同访问同一存储单元, 并且至少会有一个线程更新数据单元中的内容时, 会发送数据今生。本节的数据共享与私有化对数据竞争做一个初步的探讨, 后续会在同步、互斥相关文章中进行...

 mydear_33000 2011-10-31 13:49:46 阅读数: 5057

域名查询域名

IP/服务器地址查询



并行计算—OpenMP—共享与私有

// OpenMP1.cpp : 定义控制台应用程序的入口点。 // 共享变量和私有变量 #include "stdafx.h" #include #include #include int...

LY_624 2016-10-25 13:54:11 阅读数: 371

OpenMP 参考（指令格式）

原文: <https://computing.llnl.gov/tutorials/openMP/>

saga1979 2011-03-21 14:34:00 阅读数: 3863

OpenMP

Tutorials | Exercises | Abstracts | LC Workshops | Comments | Search | Privacy & Legal Notice O...

Real_Myth 2016-09-18 15:08:04 阅读数: 1977

OpenMP中的数据处理子句

OpenMP中的数据处理子句相关文档连接: 多核编程中的任务随机竞争模式的概率分析 多核编程中的任务分组竞争模式 多核编程中的负载均衡难题 多核编程中的锁竞争难题 多核编程的几个难题及其应对策略 (难题...

drzhouweiming 2008-01-10 11:02:00 阅读数: 32062

OpenMP中的任务调度----schedule()

OpenMP中的任务调度 OpenMP中, 任务调度主要用于并行的for循环中, 当循环中每次迭代的计算量不相等时, 如果简单地给各个线程分配相同次数的迭代的话, 会造成各个线程计算负载不均...

freeboy1015 2012-03-19 16:05:25 阅读数: 3884

程序员不会英语怎么行?

老司机教你一个数学公式秒懂天下英语



openmp 任务调度 for schedule static dynamic guided runtime

在OpenMP中, 对for循环并行化的任务调度使用schedule子句来实现, 下面介绍schedule的用法。 schedule的使用格式为: schedule(type[,size]) schedu...

billbliss 2015-03-08 11:48:47 阅读数: 1344

OpenMP并行构造的schedule子句详解

schedule子句是专门为循环并行构造的时候使用的子句, 只能用于循环并行构造 (parallel for) 中。 根据OpenMP Spec (<http://openmp.org/mp-document...>

gengshenghong 2011-11-22 23:22:51 阅读数: 13129

OpenMP Sections

const int N = 10000000; int i = 0; float *a = new float[N]; float *b = new float[N]; float *c = new ...

liangjisheng 2017-05-28 17:07:17 阅读数: 130

openmp在多重循环内的简单使用及其详解

由于项目需求, 在三重循环内加入了并行计算, 但由于只能在内层循环加入, 而内层循环只有32维度, 因此速度提高的也就那么几毫秒。 在此 不再将代码贴出! 以下是转载的别人博客中的详细讲解, 很不错! ...

Allyli0022 2016-09-29 15:44:40 阅读数: 6045

OpenMP中数据属性相关子句详解（1）：private/firstprivate/lastprivate/threadprivate之间的比较

private/firstprivate/lastprivate/threadprivate, 首先要知道的是, 它们分为两大类, 一类是private/firstprivate/lastprivate子句...

gengshenghong 2011-11-22 23:22:51 阅读数: 13129

直播系统源码

方维直播系统源码



[并行计算] 2. OpenMP简介

OpenMP简介（这篇翻译只涉及与C/C++相关的代码和示例，忽略了与Fortran相关的代码和示例，感兴趣的读者可以[参考原文](#)）1 摘要OpenMP是由一组计算机硬件和软件供应商联合定义的应用程序接口...

 magicbean2 2017-07-27 10:39:34 阅读数：11237

OpenMP知识点汇总

OpenMP知识点汇总

 fengbingchun 2013-12-11 21:23:27 阅读数：8913

（一）初步了解并行计算、OpenMP

(1) 什么是HPC? (2) 什么是并行计算? (3) 什么是OpenMp? (4) OpenMP支持的语言? ...

 lemon4869 2016-11-26 21:52:10 阅读数：1135

OpenMP并行编程应用—加速OpenCV图像拼接算法

OpenMP是一种应用于多处理器程序设计的并行编程处理方案，它提供了对于并行编程的高层抽象，只需要在程序中添加简单的指令，就可以编写搞笑的并行程序，而不用关心具体的并行实现细节，降低了并行编程的难度和...

 dcrmg 2017-01-11 07:17:33 阅读数：5493

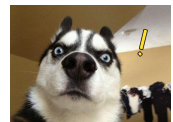
OpenMP编程指南

OpenMP编程指南进入多核时代后，必须使用多线程编写程序才能让各个CPU核得到利用。在单核时代，通常使用操作系统提供的API来创建线程，然而，在多核系统中，情况发生了很大的变化，如果仍然使用操作系...

 drzhouweiming 2009-04-20 10:13:00 阅读数：56107

程序猿不会英语怎么行？英语文档都看不懂！

不背单词和语法，一个公式教你读懂天下英文→



OpenMp 程序优化，怎么让并行达到并行的效果！

参考链接：<http://blog.csdn.net/donhao/article/details/5651156> 常用的库函数 函数原型 ...

 fulva 2012-10-25 20:19:44 阅读数：5421

最简单的并行计算——OpenMP的使用

简介 OpenMP的英文全称是Open Multiprocessing，一种应用程序界面（API，即Application Program Interface），是一种单进程多线程并行的实现和方法，...

 xiangxianghehe 2018-01-20 00:25:14 阅读数：1267

OpenMP简介

OpenMP是一种用于共享内存并行系统的多线程程序设计的库(Compiler Directive),特别适合于多核CPU上的并行程序开发设计。它支持的语言包括：C语言、C++、Fortran；不过，用...

 carson2005 2012-05-26 09:58:06 阅读数：10800

OpenMP并行程序设计（二）

OpenMP并行程序设计（二） ... 11、fork/join并行执行模式的概念... 12、OpenMP指令和库函数介绍... 13、parallel 指令的用法... 34、for指令的使用方法...

 drzhouweiming 2006-09-04 15:31:00 阅读数：69307

intel openmp

介绍在串行应用中找到何处能够有效实施并行。作者 Clay P. Breshears显式线程化方法（如、Windows* 线程或 POSIX* 线程）使用库调用创建、管理并同步

免费云主机试用一年

有哪些可以免费试用一年左右的云服务器



OpenMP 参考（简介）

openmp参考, 原文: <https://computing.llnl.gov/tutorials/openMP/>

 saga1979 2011-03-14 21:05:00 阅读数: 3329

openmp openmp

2011年06月13日 375KB [下载](#)



OpenMP并行编程计算 π 值及PSRS排序

OpenMP简介OpenMP是一个共享存储并行系统上的应用程序接口。它规范了一系列的编译制导、运行库例程和环境变量。它提供了C/C++和FORTRAN等的应用编程接口, 已经应用到UNIX、Window...

 rectsuly 2017-04-08 23:11:06 阅读数: 2306

OpenMP中的任务调度

OpenMP中的任务调度OpenMP中, 任务调度主要用于并行的for循环中, 当循环中每次迭代的计算量不相等时, 如果简单地给各个线程分配相同次数的迭代的话, 会造成各个线程计算负载不均衡, 这会使得有些线...

 drzhouweiming 2007-10-26 12:31:00 阅读数: 27198

OpenMP并行程序设计（一）

OpenMP并行程序设计（一） OpenMP是一个支持共享存储并行设计的库, 特别适宜多核CPU上的并行程序设计。今天在双核CPU机器上试了一下OpenMP并程序程序设计, 发现效率方面超出想象, 因此写出...

 drzhouweiming 2006-08-28 11:17:00 阅读数: 73949

50万码农评论：英语对于程序员有多重要？

不背单词和语法, 一个公式学好英语



OpenMP 环境变量

OpenMP 规范定义了四个用于控制 OpenMP 程序执行的环境变量。下表对它们进行了概括。表 2-1 OpenMP 环境变量 环境变量 功能 ...

 zhuxianjianqi 2012-12-12 19:30:24 阅读数: 3074

使用OpenMP实现并行归并排序（Report）

归并排序算法：归并排序算法是一种经典的分治算法。分治分治算法分为由三部分组成： 分解：将原问题分解为一系列子问题； 解决：递归的解决各个子问题。若子问题足够小, 那么直接求解。 合并：将子问题的结...

 ACM_Fish 2017-06-05 16:41:19 阅读数: 1285

openMP的一点使用经验【非原创】

按照百科上说的, 针对于openmp的编程, 最简单的就是在开头加个#include, 然后在后面的for上加一行#pragma omp parallel for即可, 下面的是较为详细的介绍了openmp的...

 shouhuxianjian 2015-06-22 12:32:16 阅读数: 1493

一起来学OpenMP（3）——for循环并行化基本用法

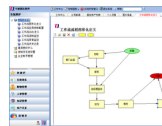
一、引言在“一起来学OpenMP（1）——初体验”中给出了一个for循环并行化的例子, 这里做进一步的分析, 但本节仅描述for循环并行化的基本用法。二、for循环并行化的几种声明形式#include ...

 donhao 2010-06-06 23:08:00 阅读数: 18261

OpenMP学习日记1

工作流系统

一个工作流管理系统的设计和实现



Linux环境下的OpenMP多线程编程

Linux环境下的OpenMP多线程编程 在正式开始前，我先说一下，我们的OpenMP编程在linux下写完以后编译工具为GCC，编译命令如下：`gcc -fopenmp filename.c -o ...`

qq_20198487 2016-06-10 11:08:58 阅读数：5931