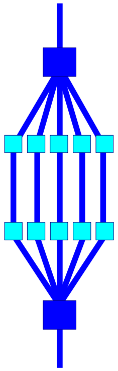


# Using OpenMP to Parallelize Interval Algorithms

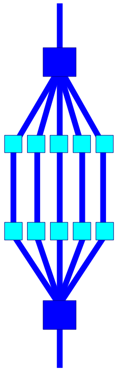
Ruud van der Pas

Senior Staff Engineer
Technical Developer Tools
Sun Microsystems, Menlo Park, CA, USA

## SCAN 2008

*The University of Texas at El Paso*
*El Paso, TX, USA*
*Sep 29-Oct 3, 2008*

# Goals Of This Talk



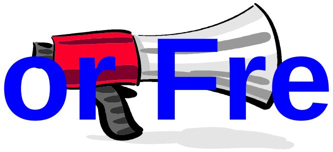
***Present the OpenMP Parallel Programming Model as a possible solution to speed up interval algorithms that require a significant time to compute***

***Demonstrate that Interval Algorithms are not exempt from Data Races***

# Outline

- ❑ *Interval Arithmetic in the Sun Studio Compilers*
- ❑ *The OpenMP Programming Model*
  - *Includes a short demo*
- ❑ *Data Races*
- ❑ *Extensive demo*
  - *An interval program*
    - ✔ *Written in Fortran*
    - ✔ *Parallelized with OpenMP*
  - *Thread Analyzer - Detects data races (and deadlock)*
- ❑ *Wrap Up*

# *The Sun Studio™ Compilers and Tools*



**For Free**

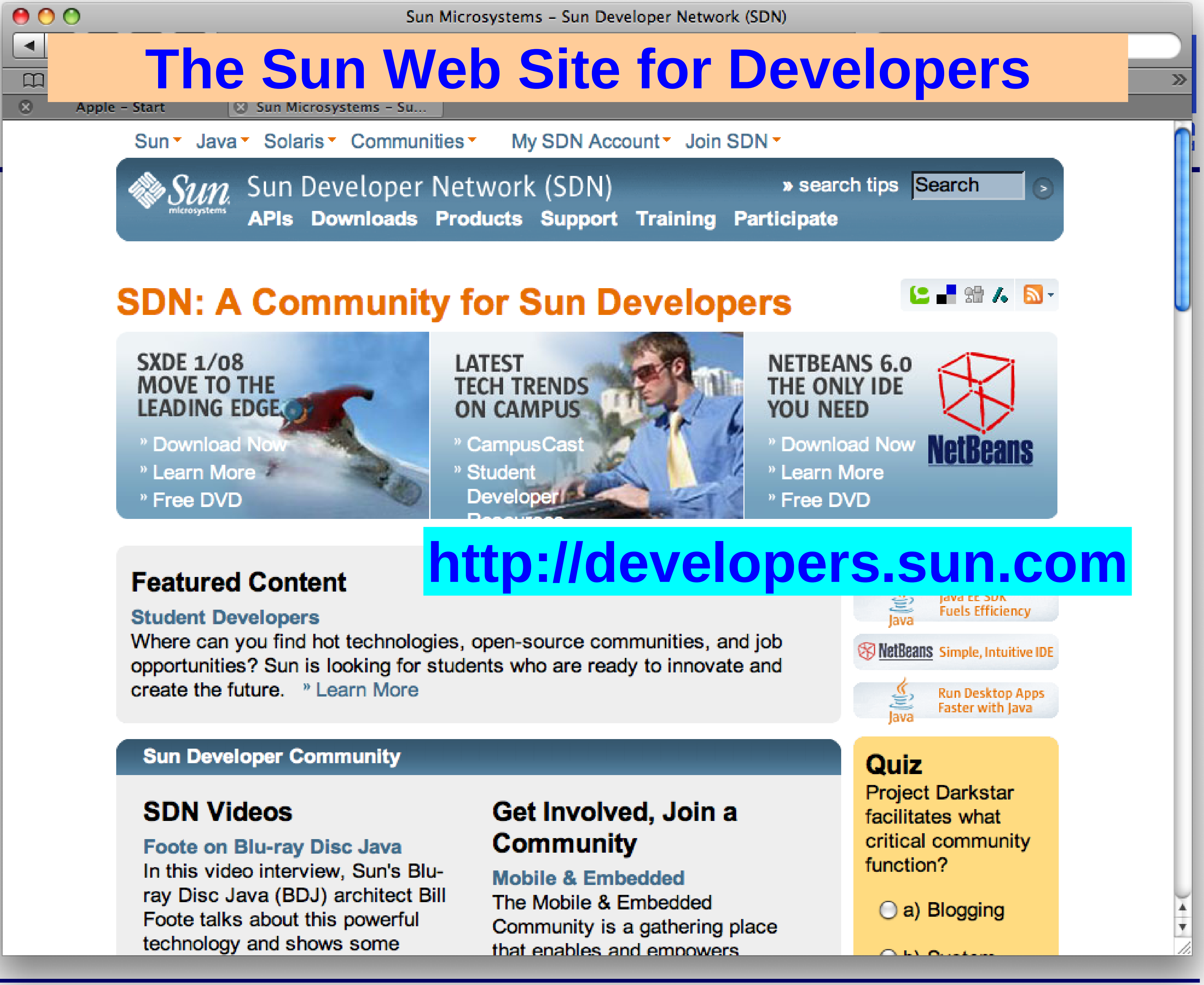
# Sun Studio Compilers and Tools

- ❑ *Fortran (f95), C (cc) and C++ (CC) compilers*
  - *Support sequential optimization, automatic parallelization and OpenMP*
- ❑ *The Sun Studio Performance Analyzer*
  - *Languages supported: Fortran, C, C++ and Java*
  - *Parallel: AutoPar, OpenMP, POSIX/Solaris Threads, MPI*
- ❑ *The Sun Studio Thread Analyzer*
  - *Languages supported: C, C++ and Fortran*
  - *Parallel: OpenMP, POSIX/Solaris Threads*
- ❑ *Sun Studio Integrated Development Environment*
- ❑ *Additional tools*

# Supported Platforms

- ❑ *The Sun Studio compilers and tools are supported on various AMD and Intel processors, as well as all SPARC processors*
  - *SPARC has the siam instruction to better support interval arithmetic*
- ❑ *Operating Systems supported*
  - *Solaris*
  - *Certain Linux implementations (RedHat, Suse)*
- ❑ *Regarding Interval Arithmetic*
  - *Fortran has the best and easiest support*
    - ✔ *Intervals are a built in, native, data type*
  - *C++ support is through a class library*

# The Sun Web Site for Developers



Sun Microsystems – Sun Developer Network (SDN)

Apple – Start | Sun Microsystems – Su...

Sun ▾ Java ▾ Solaris ▾ Communities ▾ My SDN Account ▾ Join SDN ▾

Sun Developer Network (SDN) » search tips Search

APIs Downloads Products Support Training Participate

## SDN: A Community for Sun Developers

SXDE 1/08
MOVE TO THE
LEADING EDGE
» Download Now
» Learn More
» Free DVD

LATEST
TECH TRENDS
ON CAMPUS
» CampusCast
» Student Developer Resources

NETBEANS 6.0
THE ONLY IDE
YOU NEED
» Download Now
» Learn More
» Free DVD

### Featured Content

**Student Developers**
Where can you find hot technologies, open-source communities, and job opportunities? Sun is looking for students who are ready to innovate and create the future. » Learn More

Java EE SDK Fuels Efficiency

NetBeans Simple, Intuitive IDE

Run Desktop Apps Faster with Java

### Sun Developer Community

### SDN Videos

**Foote on Blu-ray Disc Java**
In this video interview, Sun's Blu-ray Disc Java (BDJ) architect Bill Foote talks about this powerful technology and shows some

### Get Involved, Join a Community

**Mobile & Embedded**
The Mobile & Embedded Community is a gathering place that enables and empowers

### Quiz

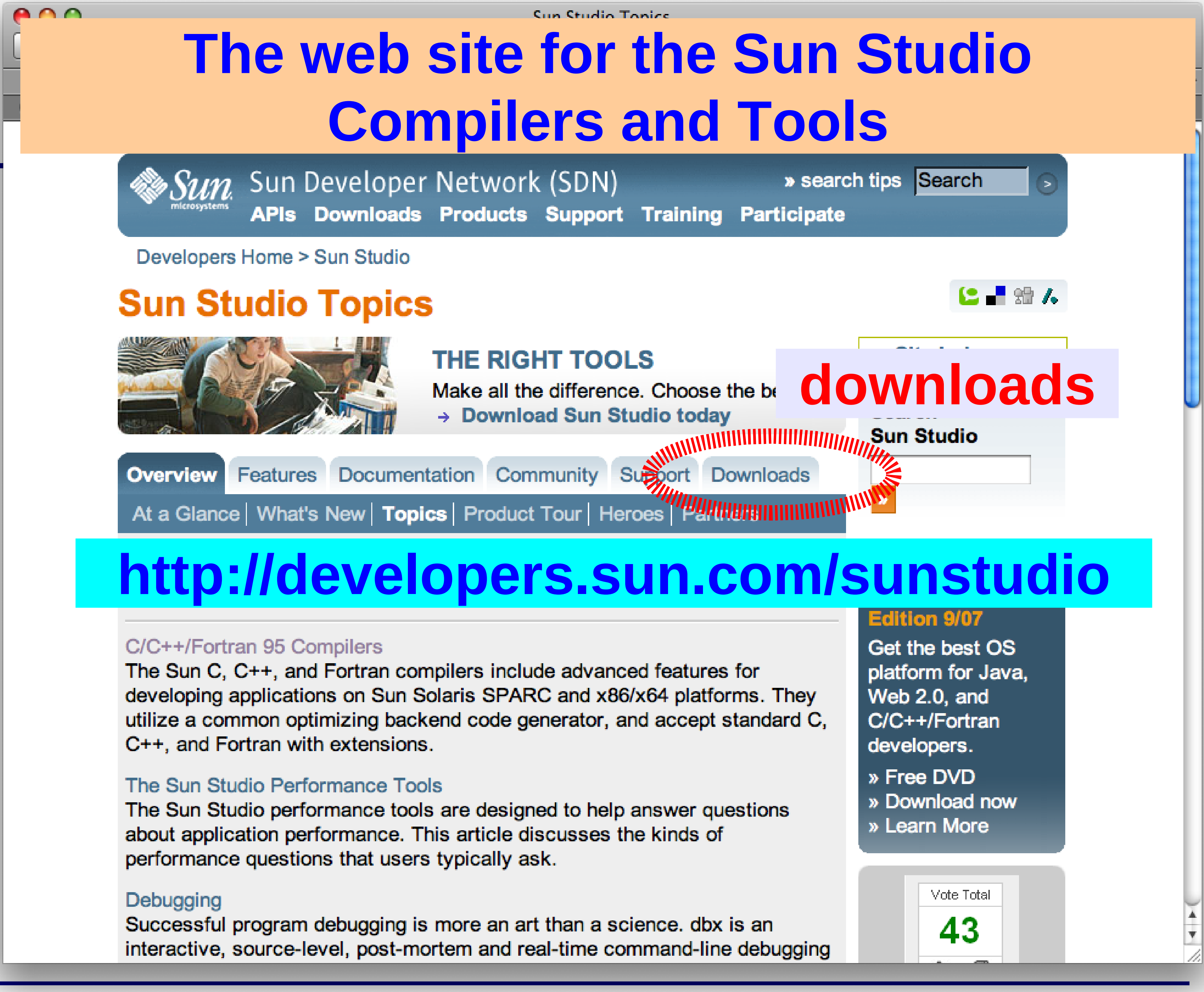
Project Darkstar facilitates what critical community function?

○ a) Blogging

○ b) Custom

**http://developers.sun.com**

# The web site for the Sun Studio Compilers and Tools



Sun Studio Topics

Sun Developer Network (SDN)

» search tips Search

APIs Downloads Products Support Training Participate

Developers Home > Sun Studio

Sun Studio Topics

THE RIGHT TOOLS

Make all the difference. Choose the be

→ Download Sun Studio today

downloads

Sun Studio

Overview | Features | Documentation | Community | Support | Downloads

At a Glance | What's New | Topics | Product Tour | Heroes | Partners

http://developers.sun.com/sunstudio

C/C++/Fortran 95 Compilers

The Sun C, C++, and Fortran compilers include advanced features for developing applications on Sun Solaris SPARC and x86/x64 platforms. They utilize a common optimizing backend code generator, and accept standard C, C++, and Fortran with extensions.

The Sun Studio Performance Tools

The Sun Studio performance tools are designed to help answer questions about application performance. This article discusses the kinds of performance questions that users typically ask.

Debugging

Successful program debugging is more an art than a science. dbx is an interactive, source-level, post-mortem and real-time command-line debugging

Edition 9/07

Get the best OS platform for Java, Web 2.0, and C/C++/Fortran developers.

» Free DVD
» Download now
» Learn More

Vote Total

43

# Sun Studio Support for Interval Arithmetic

# Intervals in Fortran - Key Features

- *Native Interval Data Type*
- *Fortran Intrinsic Functions (e.g. EXP, LOG, SIN, ...)*
- *Interval Specific Intrinsic (Set) Functions*
    - *width, midpoint, hull, union, subset, element of, ....*
- *Order Relations (e.g. “certainly less than”)*
- *Input/Output can be handled in different ways*
- *Integer Power understands Dependence*
- *Mixed mode interval expressions*
- *Context dependent literal interval constants*

# Support in C++

- □ ***Implemented as class library***
- □ ***SPARC only***
- □ ***Same functionality as Fortran***
  - ***No mixed mode support because of C++ language standard and not a native data type***

# Basic Arithmetic Operations

Assume that [a,b] and [c,d] are intervals

For a basic operator "op" in {+,-,*,/} we can then define:

$[a,b]\ \text{"op"}\ [c,d] \supseteq \{x\ \text{"op"}\ y \mid x \in [a,b] \text{ and } y \in [c,d]\}$

## Formulas for basic operations:

```
[a,b] + [c,d] = [a+c,b+d]
[a,b] - [c,d] = [a-d,b-c]
[a,b] * [c,d] = [min(a*c,a*d,b*c,b*d,
                 max(a*c,a*d,b*c,b*d)]
[a,b] / [c,d] = [min(a/c,a/d,b/c,b/d),
                 max(a/c,a/d,b/c,b/d)]
                      (if 0 is not included in [c,d])
```

# Support For Intrinsic Functions

## All Fortran intrinsic functions have an interval counterpart if they either return a REAL, or accept a REAL type argument

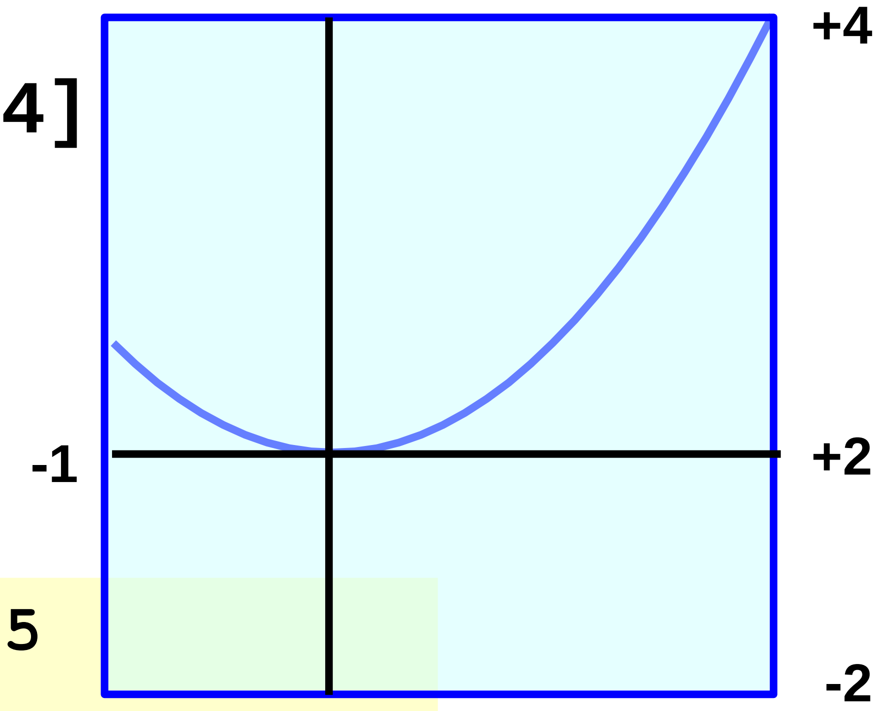
```
% cat -n cos.f95
  1     program demo
  2
  3 print *,'cos (-0.5)      = ',cos(-0.5D0)
  4 print *,'cos (+0.5)      = ',cos(+0.5D0)
  5 print *,'cos [-0.5,+0.5] = ',cos([-0.5,+0.5])
  6
  7 stop
  8 end
```

```
% f95 -o cos -xia cos.f95
% ./cos
 cos (-0.5)      =  0.8775825618903728
 cos (+0.5)      =  0.8775825618903728
 cos [-0.5,+0.5] =  [0.87758256189037264,1.0]
```

# Integer Powers

## The Dependence Problem:

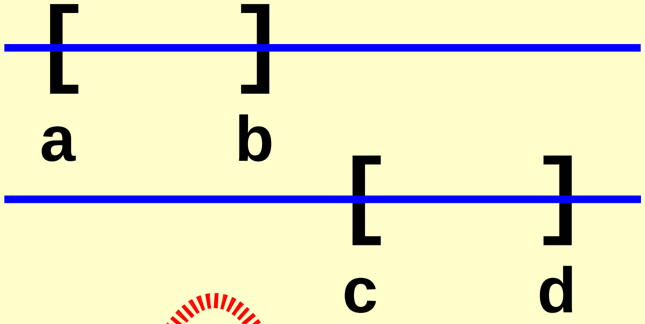
[-1,2]*[-1,2] = [-2,4]



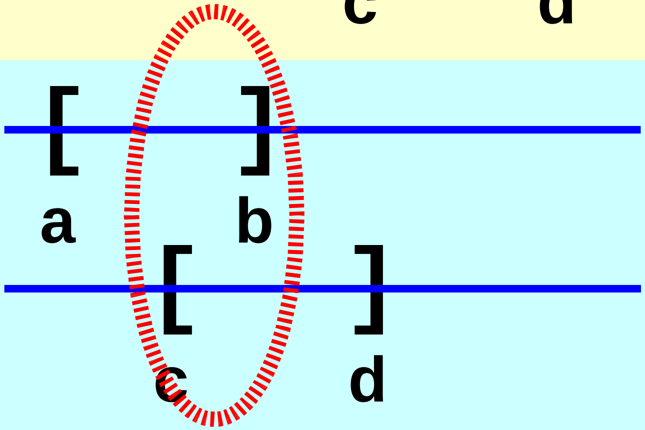
## The Sun Compiler will do the right thing:

```
% f95 -o pow -xia pow.f95
% ./pow
X     = [ -1.00000000,  2.00000000]
X*X   = [ -2.00000000,  4.00000000]
X**2  = [  0.00000000,  4.00000000]
```

# Order Relations - What To Do ?



[a,b] certainly less than [c,d]



[a,b] possibly less than [c,d]

**Implementation in the Sun compiler:**

**One of {C, P, S}, followed by LT/LE/EQ/NE/GE/GT**

**Example: A .CLT. B**

# Set-Theoretic Interval Operators

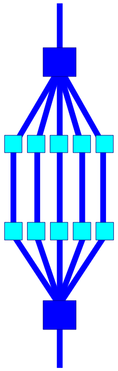
| Name | Math. Notation | Fortran | Result Type |
|---|---|---|---|
| Interval hull | $X \underline{\cup} Y$ | X .IH. Y | Interval |
| Intersection | $X \cap Y$ | X .IX. Y | Interval |
| Disjoint | $X \cap Y = \emptyset$ | X .DJ. Y | Logical |
| Element | $r \in Y$ | R .IN. Y | Logical |
| Interior | $\underline{X} < \underline{Y}$ and $\overline{X} < \overline{Y}$ | X .INT. Y | Logical |
| Proper subset | $X \subset Y$ | X .PSB. Y | Logical |
| Proper superset | $X \supset Y$ | X .PSP. Y | Logical |
| Subset | $X \subseteq Y$ | X .SB. Y | Logical |
| Superset | $X \supseteq Y$ | X. SP. Y | Logical |

# Interval Specific Intrinsics

| Name | Definition | Name | Result Type |
|---|---|---|---|
| Infimum | inf([a,b]) = a | INF | REAL |
| Supremum | sup([a,b]) = b | SUP | REAL |
| Width | w([a,b]) = b-a | WID | REAL |
| Midpoint | (a+b) / 2 | MID | REAL |
| Magnitude | max(\|a\|,\|b\|) | MAG | REAL |
| Mignitude | min(\|a\|,\|b\|)* | MIG | REAL |
| Empty Test | TRUE if empty | ISEMPTY | LOGICAL |
| Number Of Digits | Max. digits | NDIGITS | INTEGER |

***) Returns 0 if 0 $\in$ [a,b]***

# Additional Features



- ❑ *A closed interval system in which all expressions (including singularities and indeterminate forms) are defined*
  - *Examples: 1/0, $x^y$ with x=y=0, operations involving +∞ and/or -∞*
- ❑ *Domain constraints on intrinsic functions are gracefully handled*
  - *Example: SQRT( [ -1 , +1 ])  = [ 0 , 1 ]*
- ❑ *Input / Output can be handled in different ways*
- ❑ *Context dependent literal interval constants*
- ❑ *Mixed mode expressions*

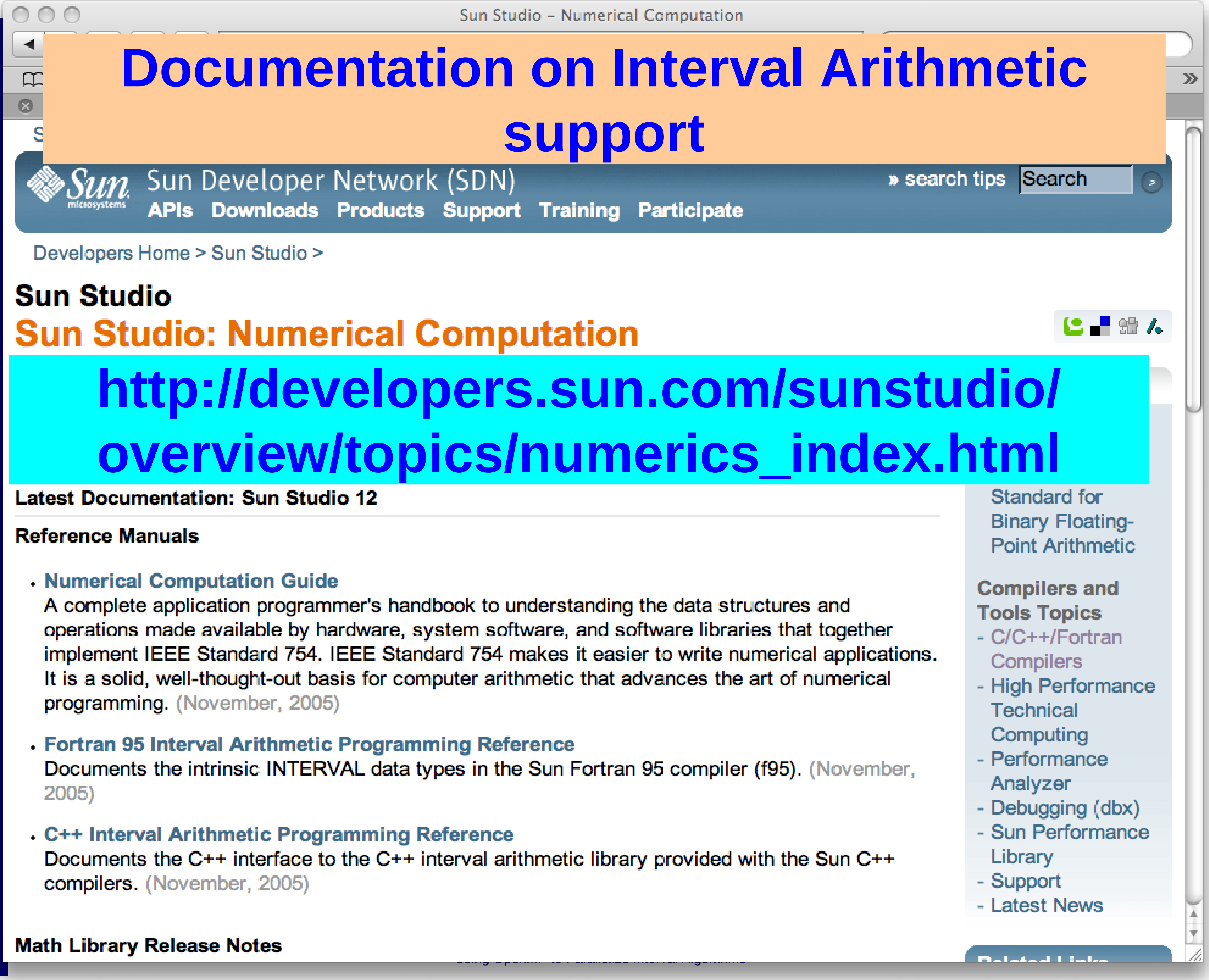
# Example Code

```
      Program Demo
      logical :: not_done = .true.
      interval(kind=8)    :: ai, bi
      write(*,*) 'Please give values for A and B'
      do while ( not_done )
         read(*,*,end=9000) ai, bi

         write(*,9010) '+',ai,'+',bi,ai+bi
         write(*,9010) '-',ai,'-',bi,ai-bi
         write(*,9010) '*',ai,'*',bi,ai*bi
         write(*,9010) '/',ai,'/',bi,ai/bi
         write(*,*)
      end do

9000  continue
      stop
9010  format(1X,'A',1X,(A),1X,'B =',VF17.4,1X,(A), &
      1X,VF17.4,' = ',VF17.4)
      end
```

# Example Closed Interval System

```
% f95 -xia math.f95
% ./a.out
Please give values for A and B
A + B =[-1.0000, 3.0000] + [ 1.0000, 2.0000] = [ 0.0000, 5.0000]
A - B =[-1.0000, 3.0000] - [ 1.0000, 2.0000] = [-3.0000, 2.0000]
A * B =[-1.0000, 3.0000] * [ 1.0000, 2.0000] = [-2.0000, 6.0000]
A / B =[-1.0000, 3.0000] / [ 1.0000, 2.0000] = [-1.0000, 3.0000]

A + B =[ 1.0000, 2.0000] + [-1.0000, 3.0000] = [ 0.0000, 5.0000]
A - B =[ 1.0000, 2.0000] - [-1.0000, 3.0000] = [-2.0000, 3.0000]
A * B =[ 1.0000, 2.0000] * [-1.0000, 3.0000] = [-2.0000, 6.0000]
A / B =[ 1.0000, 2.0000] / [-1.0000, 3.0000] = [   -Inf,     Inf]
```

# Documentation on Interval Arithmetic support



Sun Studio – Numerical Computation
Sun microsystems
Sun Developer Network (SDN)
APIs Downloads Products Support Training Participate
» search tips
Search

Developers Home > Sun Studio >

## Sun Studio

## Sun Studio: Numerical Computation

**http://developers.sun.com/sunstudio/overview/topics/numerics_index.html**

**Latest Documentation: Sun Studio 12**

**Reference Manuals**

- **Numerical Computation Guide**
A complete application programmer's handbook to understanding the data structures and operations made available by hardware, system software, and software libraries that together implement IEEE Standard 754. IEEE Standard 754 makes it easier to write numerical applications. It is a solid, well-thought-out basis for computer arithmetic that advances the art of numerical programming. (November, 2005)
- **Fortran 95 Interval Arithmetic Programming Reference**
Documents the intrinsic INTERVAL data types in the Sun Fortran 95 compiler (f95). (November, 2005)
- **C++ Interval Arithmetic Programming Reference**
Documents the C++ interface to the C++ interval arithmetic library provided with the Sun C++ compilers. (November, 2005)

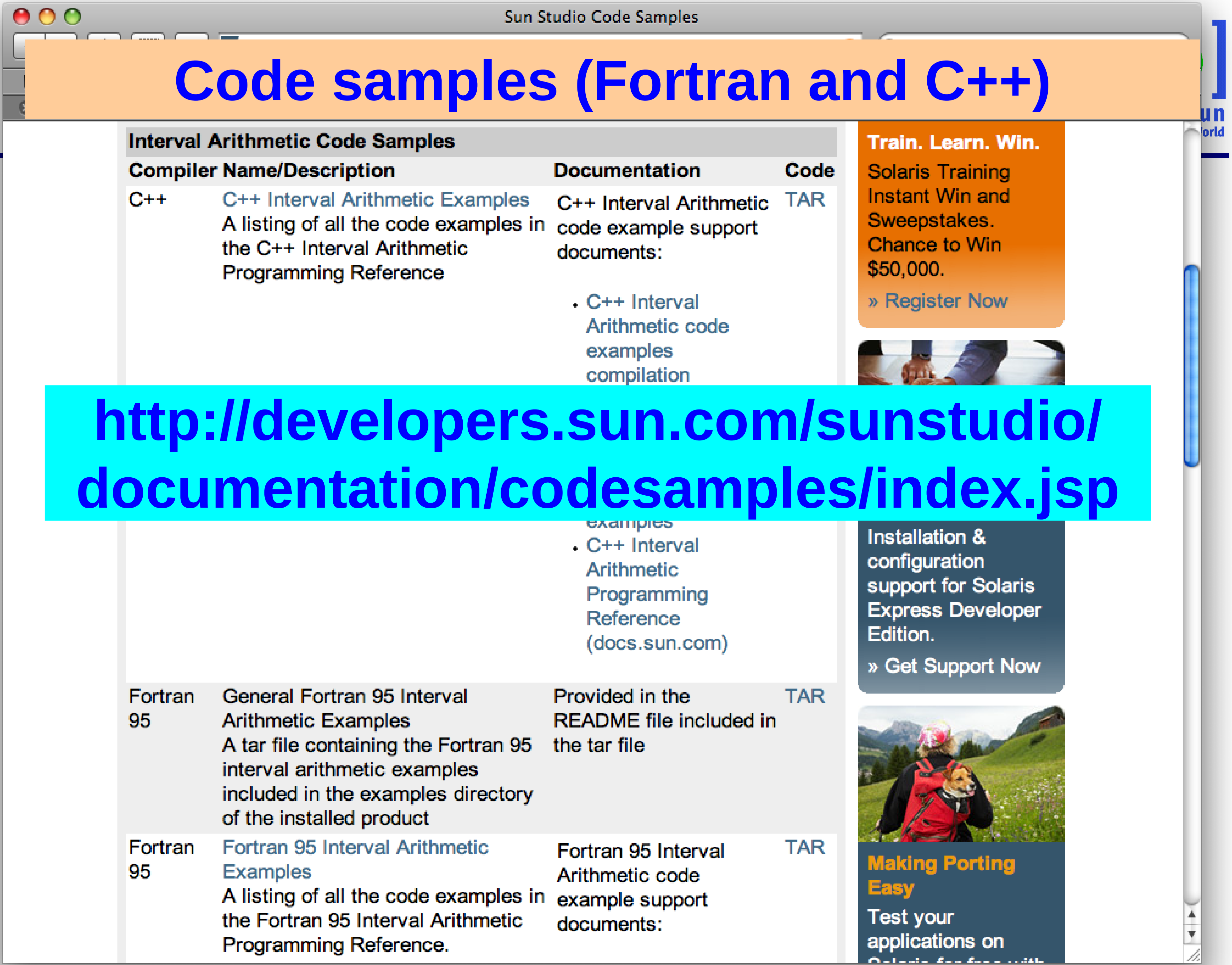
**Math Library Release Notes**

Standard for Binary Floating-Point Arithmetic

**Compilers and Tools Topics**

- C/C++/Fortran Compilers
- High Performance Technical Computing
- Performance Analyzer
- Debugging (dbx)
- Sun Performance Library
- Support
- Latest News

# Code samples (Fortran and C++)



Sun Studio Code Samples

**Interval Arithmetic Code Samples**

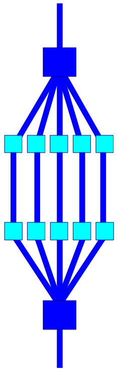
| Compiler | Name/Description | Documentation | Code |
|---|---|---|---|
| C++ | C++ Interval Arithmetic Examples<br>A listing of all the code examples in the C++ Interval Arithmetic Programming Reference | C++ Interval Arithmetic code example support documents:<br>• C++ Interval Arithmetic code examples compilation<br>examples<br>• C++ Interval Arithmetic Programming Reference (docs.sun.com) | TAR |
| Fortran 95 | General Fortran 95 Interval Arithmetic Examples<br>A tar file containing the Fortran 95 interval arithmetic examples included in the examples directory of the installed product | Provided in the README file included in the tar file | TAR |
| Fortran 95 | Fortran 95 Interval Arithmetic Examples<br>A listing of all the code examples in the Fortran 95 Interval Arithmetic Programming Reference. | Fortran 95 Interval Arithmetic code example support documents: | TAR |

## http://developers.sun.com/sunstudio/documentation/codesamples/index.jsp

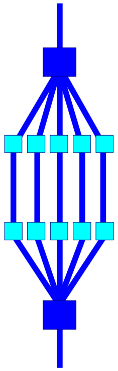
# Pointers To More Information

- □ *Documentation*
  - ● *Fortran Interval Arithmetic Programming Reference*
    - ✓ *http://docs.sun.com/app/docs/doc/819-5271*
  - ● *C++ Interval Arithmetic Programming Reference*
    - ✓ *http://docs.sun.com/app/docs/doc/819-5272*
- □ *More information, plus code examples, can be downloaded from http://developers.sun.com/sunstudio*
- □ *Another useful web site (on numerical computations):*
  - ● *http://developers.sun.com/sunstudio/overview/topics/numerics_index.html*

# Summary Interval Support



- *The Sun Fortran and C++ compilers support Interval Arithmetic*
- *The regular Basic Arithmetic Operations, intrinsic functions and logical operations have been extended to intervals*
- *In addition to this, several quality of implementation features are supported:*
  - *Closed interval system, domain constraints on intrinsic functions, input/output, ontext dependent literal interval constants, etc.*
- *We believe that this provides for a production quality interval compiler*

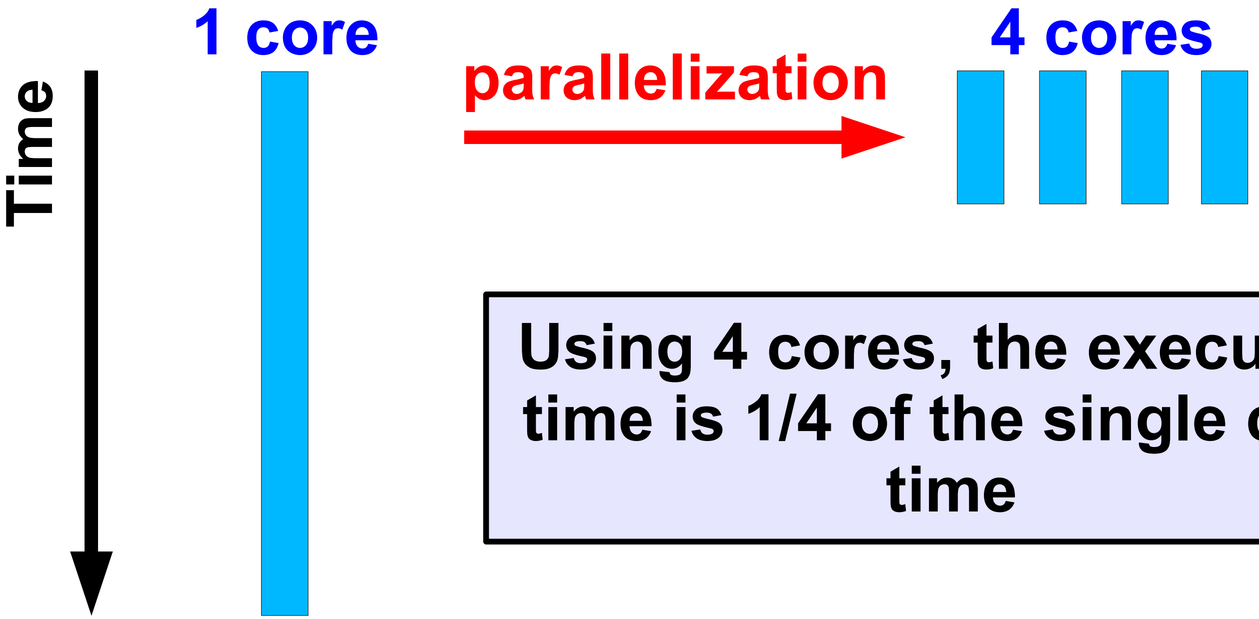


# About Parallelization

# Why Parallelization ?

*Parallelization is another optimization technique.*
*The goal is to reduce the execution time.*

*To this end, multiple processors, or cores, are used*



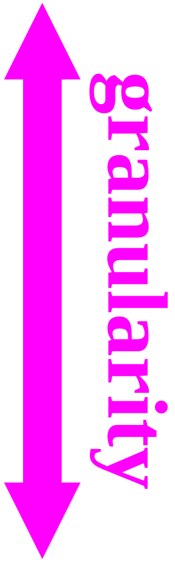
Using 4 cores, the execution time is 1/4 of the single core time

# What Is Parallelization ?

*"Something" is parallel if there is a <u>certain level of independence</u> in the order of operations*

*In other words, it doesn't matter in what order those operations are performed*

- ***A sequence of machine instructions***
- ***A collection of program statements***
- ***An algorithm***
- ***The problem you're trying to solve***



# How To Program A Parallel Computer?

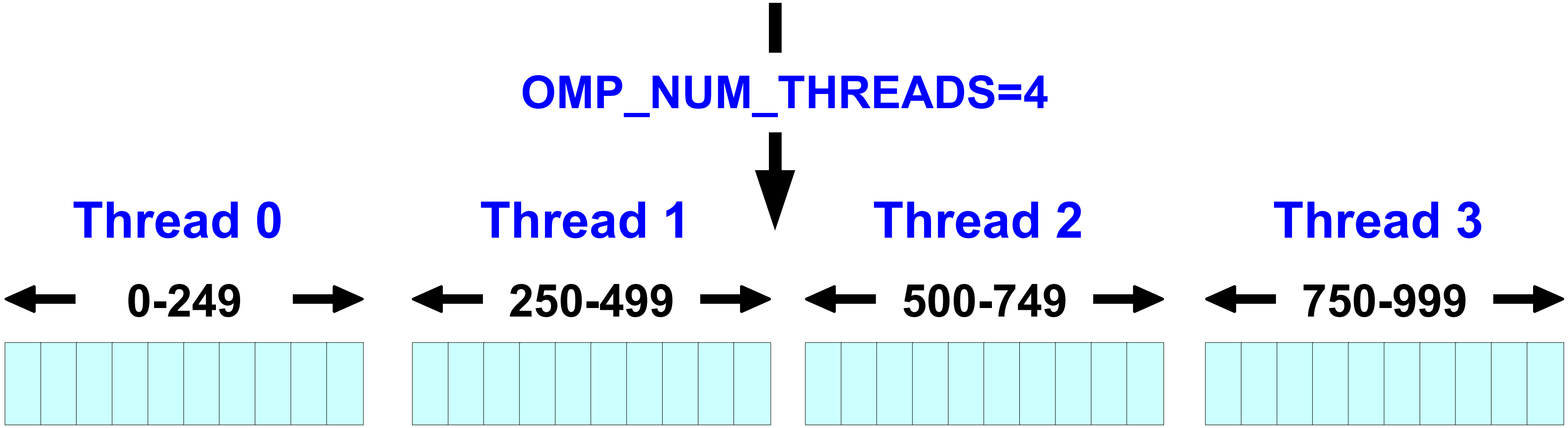
❑ ***The more well-known parallel programming models:***

- ***A Single System ("Shared Memory")***
  - ✔ *POSIX Threads (standardized, low level)*
  - ✔ *OpenMP (de-facto standard)* ← **today's focus**
  - ✔ *Automatic Parallelization (compiler does it for you)*
- ***A Cluster Of Systems ("Distributed Memory")***
  - ✔ *Sockets (standardized, low level)*
  - ✔ *MPI - Message Passing Interface (de-facto standard)*
- ***A Cluster of Shared Memory/Multicore Systems***
  - ✔ *The best and worse of both worlds*

# Automatic Parallelization

- □ *Compiler analyzes loop for parallelism to exploit*
- □ *Different iterations of the loop executed in parallel*
- □ *Same binary used for any number of threads*

```
for (i=0; i<1000; i++)
   a[i] = b[i] + c[i];
```



OMP_NUM_THREADS=4

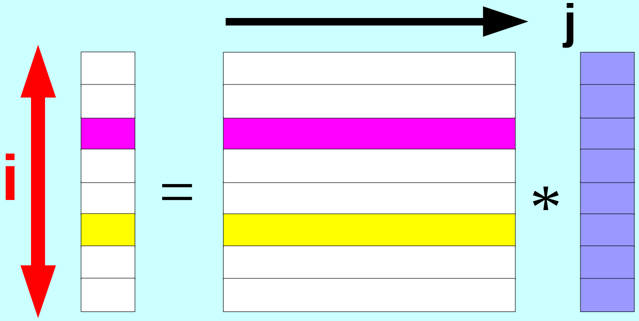
| Thread 0 | Thread 1 | Thread 2 | Thread 3 |
|---|---|---|---|
| 0-249 | 250-499 | 500-749 | 750-999 |

# Automatic Parallelization Example

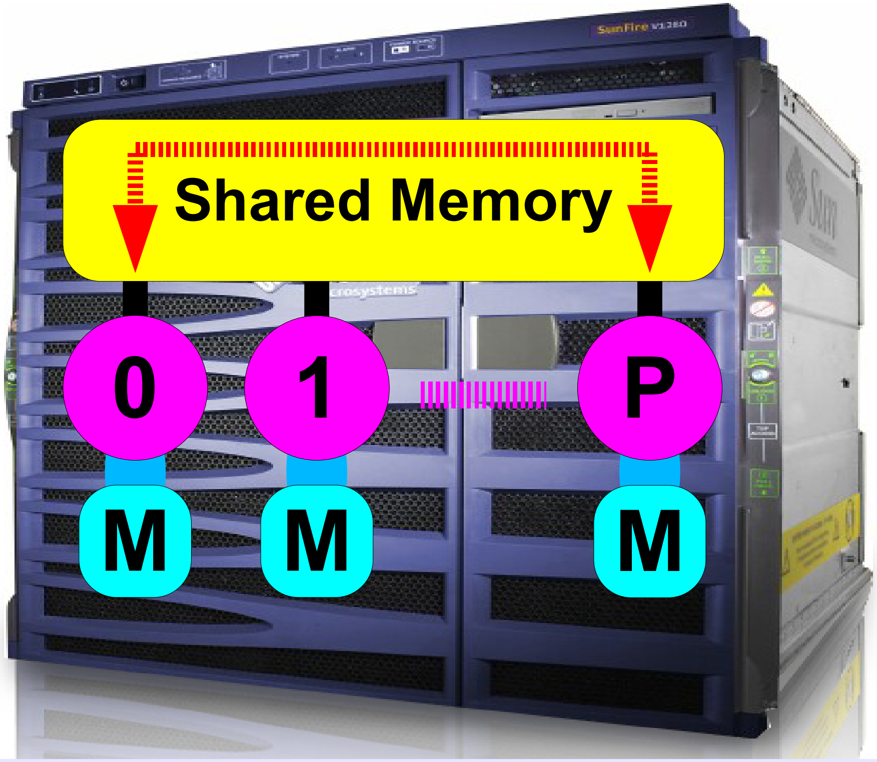
```
% cc -c -fast -xrestrict -xautopar -xloopinfo mxv.c
"mxv.c", line 6: PARALLELIZED, and serial
         version generated
"mxv.c", line 9: not parallelized, unsafe
         dependence (sum)
```

```
 6   for (int i=0; i<m; i++)
 7   {
 8     sum = 0.0;
 9     for (int j=0; j<n; j++)
10       sum += b[i][j]*c[j];
11      a[i] = sum;
12   }
13  }
```

parallel loop



# The Shared Memory Model

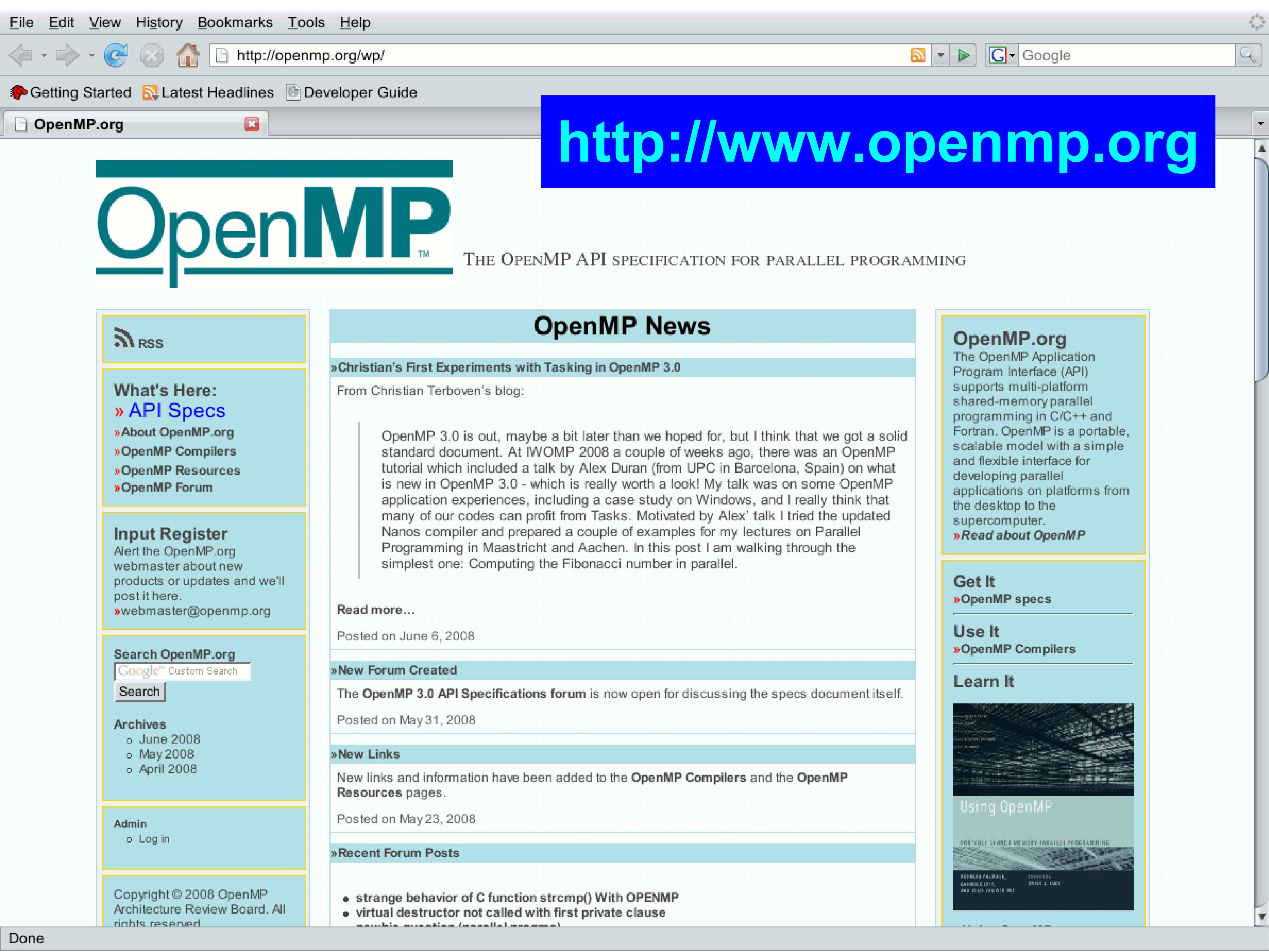


A Single System

**http://www.openmp.org**

**http://www.compunity.org**

# http://www.openmp.org



OpenMP™

The OpenMP API specification for parallel programming

RSS

**What's Here:**

- » API Specs
- »About OpenMP.org
- »OpenMP Compilers
- »OpenMP Resources
- »OpenMP Forum

**Input Register**

Alert the OpenMP.org webmaster about new products or updates and we'll post it here.

»webmaster@openmp.org

**Search OpenMP.org**

Google™ Custom Search

Search

**Archives**

- June 2008
- May 2008
- April 2008

**Admin**

- Log in

Copyright © 2008 OpenMP Architecture Review Board. All rights reserved.

## OpenMP News

**»Christian's First Experiments with Tasking in OpenMP 3.0**

From Christian Terboven's blog:

> OpenMP 3.0 is out, maybe a bit later than we hoped for, but I think that we got a solid standard document. At IWOMP 2008 a couple of weeks ago, there was an OpenMP tutorial which included a talk by Alex Duran (from UPC in Barcelona, Spain) on what is new in OpenMP 3.0 - which is really worth a look! My talk was on some OpenMP application experiences, including a case study on Windows, and I really think that many of our codes can profit from Tasks. Motivated by Alex' talk I tried the updated Nanos compiler and prepared a couple of examples for my lectures on Parallel Programming in Maastricht and Aachen. In this post I am walking through the simplest one: Computing the Fibonacci number in parallel.

**Read more…**

Posted on June 6, 2008

**»New Forum Created**

The **OpenMP 3.0 API Specifications forum** is now open for discussing the specs document itself.

Posted on May 31, 2008

**»New Links**

New links and information have been added to the **OpenMP Compilers** and the **OpenMP Resources** pages.

Posted on May 23, 2008

**»Recent Forum Posts**

- strange behavior of C function strcmp() With OPENMP
- virtual destructor not called with first private clause

**OpenMP.org**

The OpenMP Application Program Interface (API) supports multi-platform shared-memory parallel programming in C/C++ and Fortran. OpenMP is a portable, scalable model with a simple and flexible interface for developing parallel applications on platforms from the desktop to the supercomputer.

*»Read about OpenMP*

**Get It**

»OpenMP specs

**Use It**

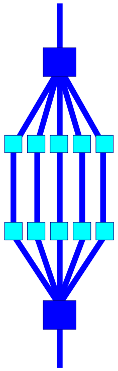
»OpenMP Compilers

**Learn It**

Using OpenMP

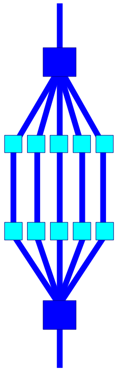
PORTABLE SHARED MEMORY PARALLEL PROGRAMMING

# What is OpenMP™ ?



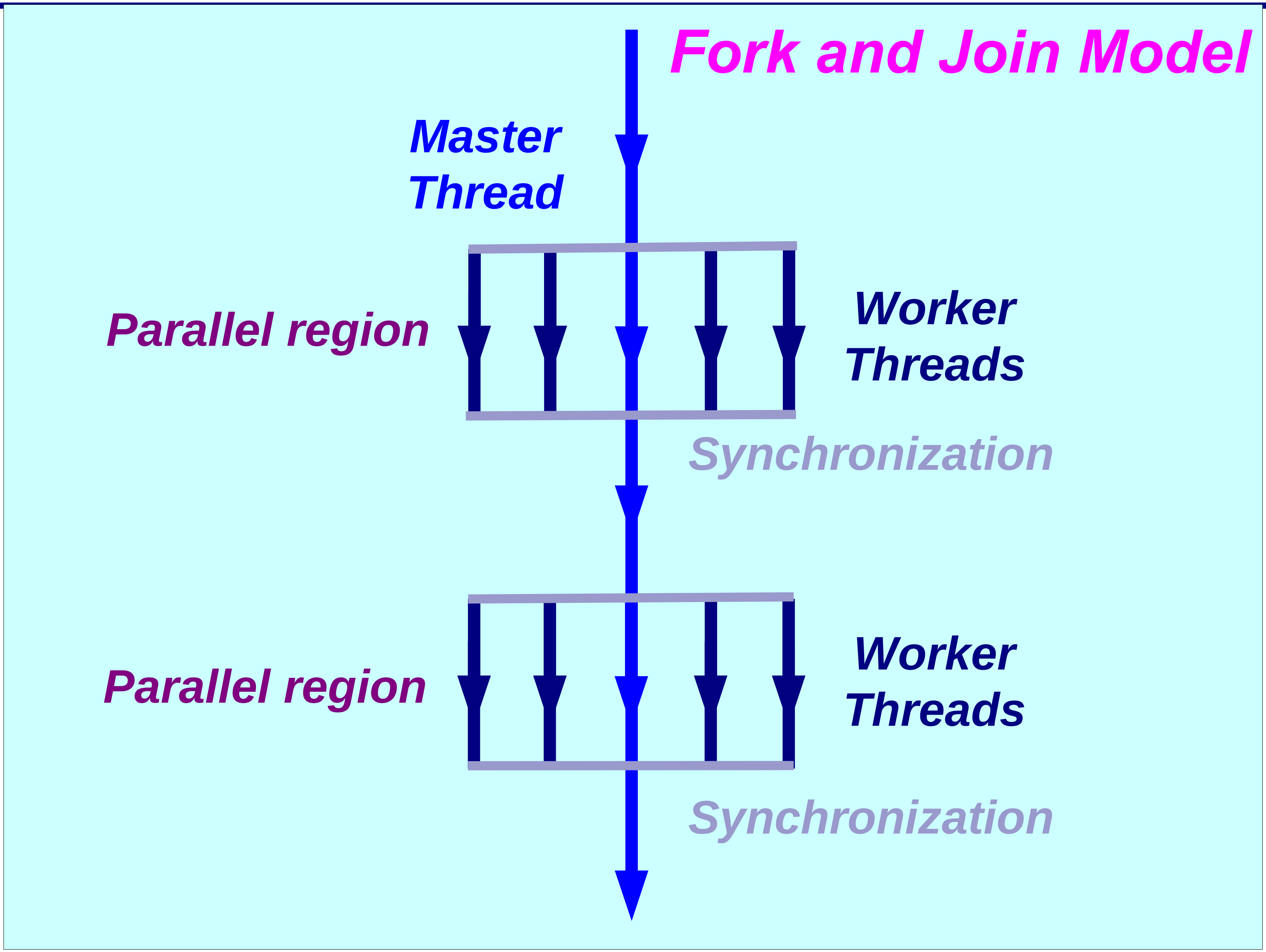
- *De-facto standard API for writing shared memory parallel applications in C, C++, and Fortran*
- *Consists of:*
  - *Compiler directives*
  - *Run time routines*
  - *Environment variables*
- *Specification maintained by the OpenMP Architecture Review Board (ARB)*
- *Version 3.0 was released May 2008*
  - *First compiler support now appearing*

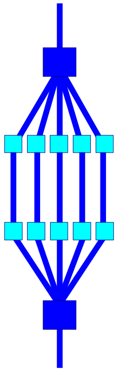
# Advantages of OpenMP



- *Good performance and scalability*
    - *If you do it right of course*
- *De-facto and mature standard*
    - *Supported by a large number of compilers*
- *Requires little programming effort*
- *Preserves sequential version of application*
- *Supports incremental parallelization*
- *Maps naturally onto a multicore architecture:*
    - *Lightweight*
    - *Each thread efficiently executed by a hardware thread*

# The OpenMP Execution Model





# Demo

# Basic Parallelization with OpenMP

# Components of OpenMP 2.5

## Directives

- Parallel regions
- Work sharing
- Synchronization
- Data-sharing attributes
  - private
  - firstprivate
  - lastprivate
  - shared
  - reduction
- Orphaning

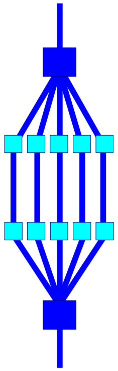
## Environment variables

- Number of threads
- Scheduling type
- Dynamic thread adjustment
- Nested parallelism

## Runtime environment

- Number of threads
- Thread ID
- Dynamic thread adjustment
- Nested parallelism
- Timers
- API for locking

# Learning Curve - Data Scoping

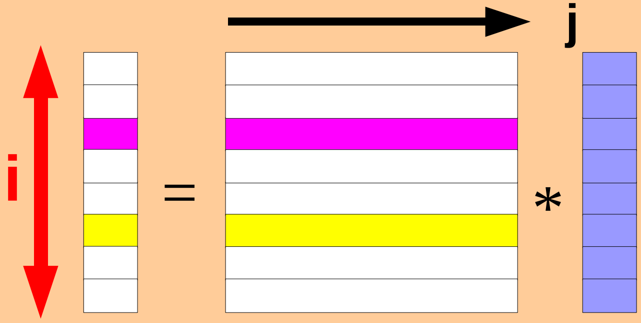


- ❑ *In the Shared Memory Programming Model one has to think about the use of the variables (“scoping”)*
- ❑ *There are two main types to distinguish*
  - *Private*
    - ✔ *Each thread has a local copy of the variable(s)*
    - ✔ *Variable is “owned” by a thread*
    - ✔ *Other threads will not see changes made*
  - *Shared*
    - ✔ *There is only one instance of the variable(s)*
    - ✔ *Correct updates to such a variable is under control of the developer*

# OpenMP Example - Matrix Times Vector

```
#pragma omp parallel for default(none) \
        private(i,j,sum) shared(m,n,a,b,c)
for (i=0; i<m; i++){
   sum = 0.0;
   for (j=0; j<n; j++)
     sum += b[i][j]*c[j];
   a[i] = sum;
 }
```

parallel loop



**Thread 0**

```
for (i=0,1,2,3,4)
  i = 0
  sum = Σ b[i=0][j]*c[j]
  a[0] = sum
  i = 1
```

**Thread 1**

```
for (i=5,6,7,8,9)
  i = 5
  sum = Σ b[i=5][j]*c[j]
  a[5] = sum
  i = 6
```

... etc ...

# Shameless Plug - “Using OpenMP”

## *“Using OpenMP”*

***Portable Shared Memory Parallel Programming***

***Chapman, Jost, van der Pas***

**MIT Press, October 2007**

**ISBN-10: 0-262-53302-2**
**ISBN-13: 978-0-262-53302-7**

**List price: 35 $US**

SCIENTIFIC AND ENGINEERING COMPUTATION SERIES

Using OpenMP

PORTABLE SHARED MEMORY PARALLEL PROGRAMMING

BARBARA CHAPMAN,
GABRIELE JOST,

foreword by
DAVID J. KUCK

*All examples available soon!*

*(also plan to start a forum on www.openmp.org)*

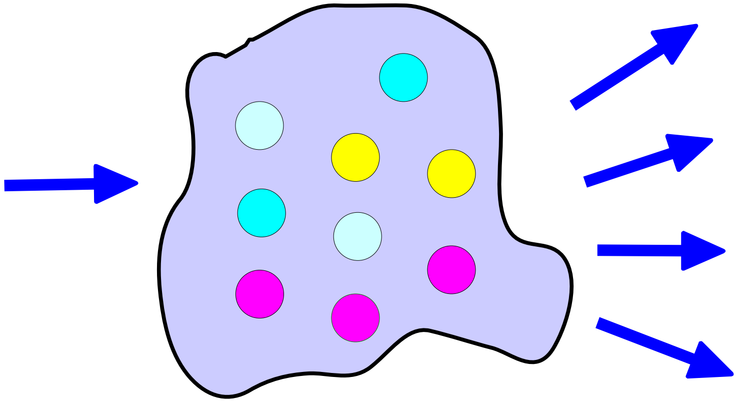
# *Why The Excitement About OpenMP 3.0 ?*

***Support for TASKS !***

***With this new feature, a wide range of applications can now be parallelized***

# Tasking Concept in OpenMP 3.0

Encountering thread adds task to pool



Threads execute tasks in the pool

***Developer specifies tasks in application***
***Run-time system executes tasks***

# Example - A Linked List

```
my_pointer = listhead;

while(my_pointer) {

   (void) do_independent_work(my_pointer);

   my_pointer = my_pointer->next ;
}
```

***Hard to do before OpenMP 3.0:***
***First count number of iterations, then convert while loop to for loop***

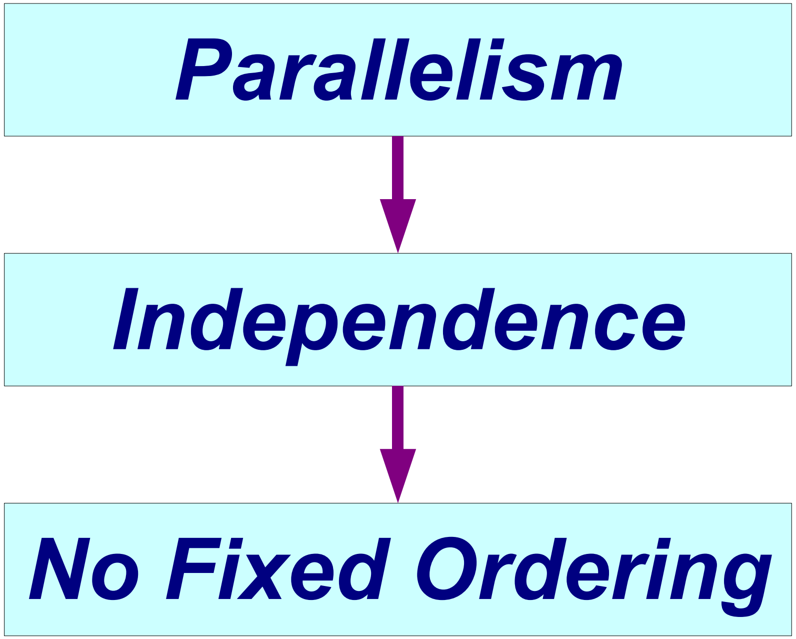
# Example - A Linked List With Tasking

```
my_pointer = listhead;
#pragma omp parallel
{
   #pragma omp single
   {
      while(my_pointer) {
        #pragma omp task firstprivate(my_pointer)
        {
            (void) do_independent_work(my_pointer);
        }
        my_pointer = my_pointer->next ;
      }
   } // End of single - implied barrier
} // End of parallel region - implied barrier
```

Task Defined Here

# Data Races

# About Parallelism



*Parallelism*

*Independence*

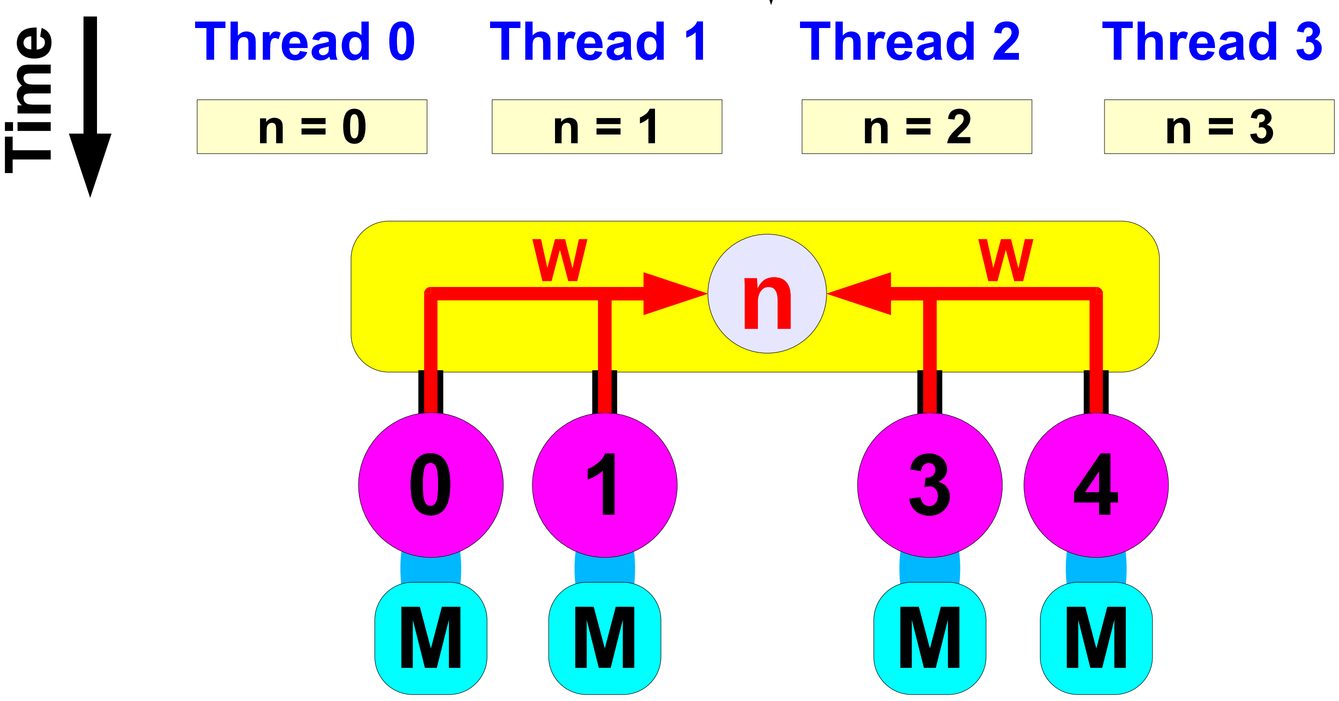
*No Fixed Ordering*

***"Something" that does not obey this rule, is not parallel (at that level)***

# “Hello Data Race World”

```
#pragma omp parallel shared(n)
{n = omp_get_thread_num();}
```

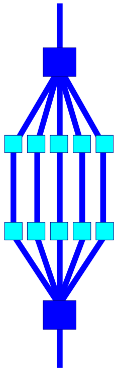
OMP_NUM_THREADS=4



# What is a Data Race?

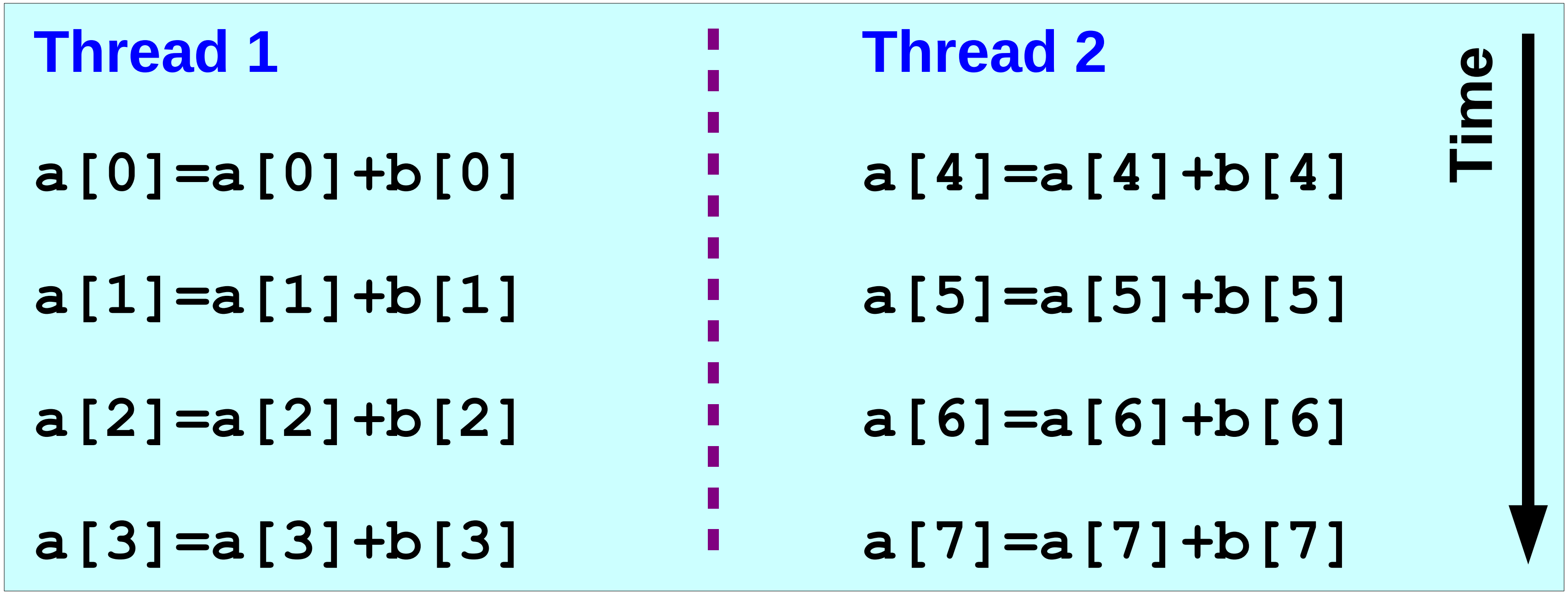
- □ *Two different threads in a multi-threaded shared memory program*
- □ *Access the same (=shared) memory location*
  - *Concurrently* *and*
  - *Without holding any common exclusive locks* *and*
  - *At least one of the accesses is a write/store*

# A Parallel Loop



```
for (i=0; i<8; i++)
   a[i] = a[i] + b[i];
```

*The result does not depend on the order of execution*

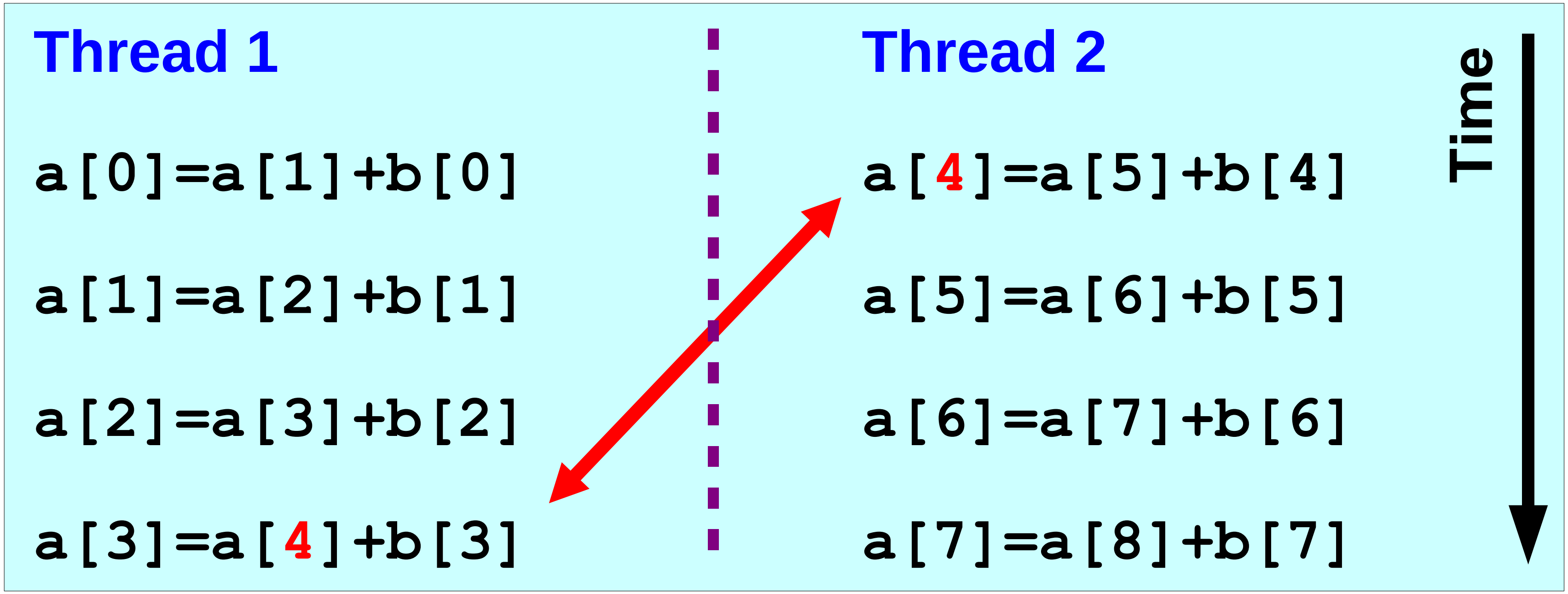


| Thread 1 | Thread 2 |
|---|---|
| `a[0]=a[0]+b[0]` | `a[4]=a[4]+b[4]` |
| `a[1]=a[1]+b[1]` | `a[5]=a[5]+b[5]` |
| `a[2]=a[2]+b[2]` | `a[6]=a[6]+b[6]` |
| `a[3]=a[3]+b[3]` | `a[7]=a[7]+b[7]` |

# Not A Parallel Loop

```
for (i=0; i<8; i++)
   a[i] = a[i+1] + b[i];
```

*The result is not deterministic if executed in parallel !*



**Thread 1**

a[0]=a[1]+b[0]

a[1]=a[2]+b[1]

a[2]=a[3]+b[2]

a[3]=a[4]+b[3]

**Thread 2**

a[4]=a[5]+b[4]

a[5]=a[6]+b[5]

a[6]=a[7]+b[6]

a[7]=a[8]+b[7]

# Numerical Results

```
threads:   1 checksum   1953 correct value   1953
threads:   1 checksum   1953 correct value   1953
threads:   1 checksum   1953 correct value   1953
threads:   1 checksum   1953 correct value   1953

threads:   2 checksum   1953 correct value   1953
threads:   2 checksum   1953 correct value   1953
threads:   2 checksum   1953 correct value   1953
threads:   2 checksum   1953 correct value   1953

threads:   4 checksum   1905 correct value   1953
threads:   4 checksum   1905 correct value   1953
threads:   4 checksum   1953 correct value   1953
threads:   4 checksum   1937 correct value   1953

threads:  32 checksum   1525 correct value   1953
threads:  32 checksum   1473 correct value   1953
threads:  32 checksum   1489 correct value   1953
threads:  32 checksum   1513 correct value   1953
```

**Data Race In Action !**

# *Demo*

# *Parallelizing An Interval Algorithm Using OpenMP*

# Bottom Line About Data Races

***Data Races Are Easy To Put In***
***But***
***Very Hard To Find***

***"Finding errors in software is particularly important in computer programs that claim to be mathematically rigorous."***

***R. Baker Kearfott - SCAN 2008, El Paso, TX***

***That is why a special tool to find data races is highly recommended to use***

# Wrap Up

- *The Sun Studio Fortran and C++ compilers support Interval Arithmetic*
  - *Fortran implementation most elegant and powerful*
- *OpenMP provides for an easy to use, but yet very powerful, portable parallel programming model*
  - *Also very suitable for multicore architectures*
- *Despite this, parallel programming can still be tricky*
- *As always, good tools can make all the difference when it comes to productivity and correctness*

# That's It

# Thank You and ..... Stay Sharp !

***Ruud van der Pas***
***ruud.vanderpas@sun.com***