C-DAC Four Days Technology Workshop

ON

Hybrid Computing – Coprocessors/Accelerators
Power-Aware Computing – Performance of Applications Kernels

hyPACK-2013
(Mode-1:Multi-Core)

Lecture Topic:

Multi-Core Processors : Shared Memory Prog: An Overview of Intel Thread Building Blocks (TBB)

Venue : CMSD, UoHYD ; Date : October 15-18, 2013

An Overview of Intel Thread Building Blocks

Lecture Outline

Following topics will be discussed

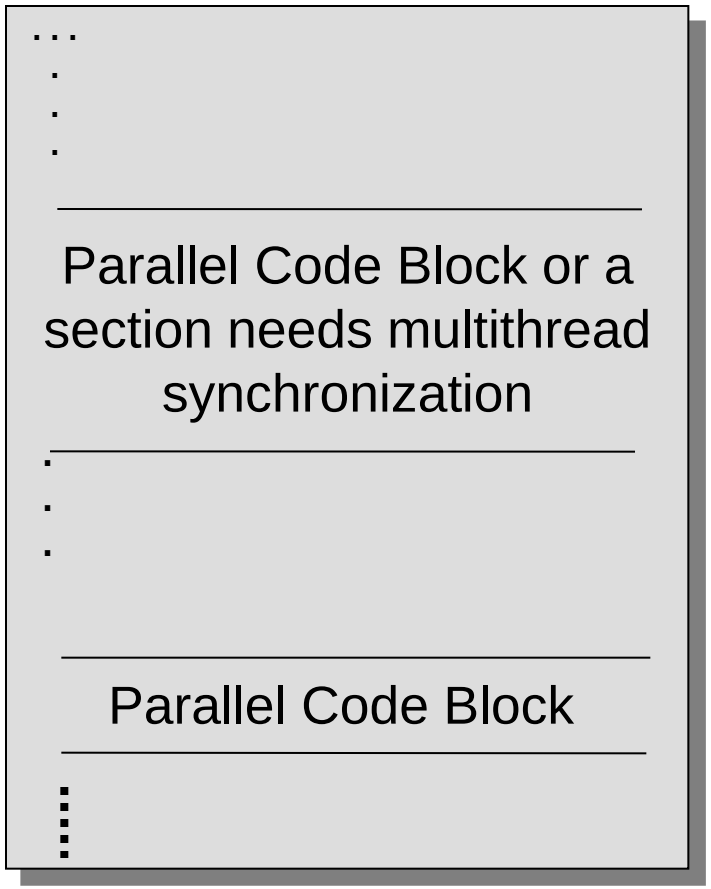
- Why Thread Building Blocks (TBB) ?
- Summary of high-level templates Loops - Loop Parallelization
- Algorithms templates – Intel TBB
- Memory Allocation – TBB – Performance Issues
- TBB Task Scheduler – Speed-Up issues
- TBB Application Perspective

Part II

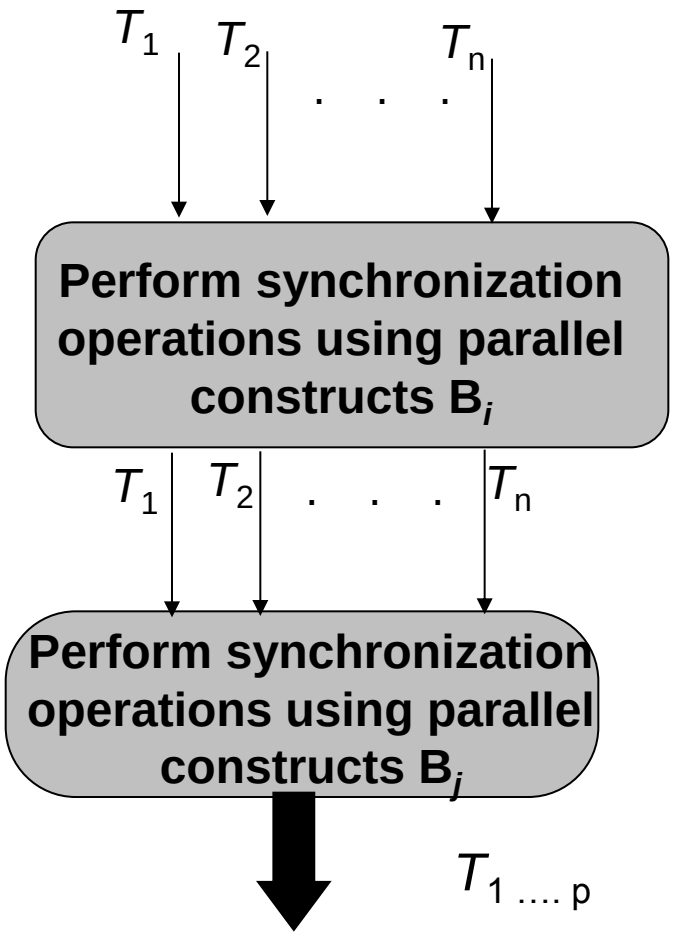
Intel TBB

Background : Operational Flow of Threads for an Application

Implementation Source Code

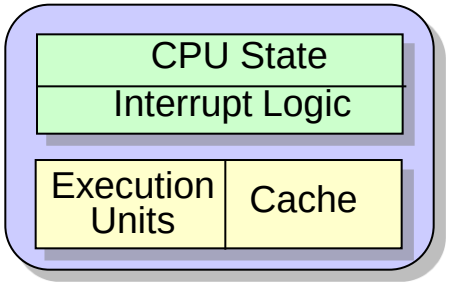


Operational Flow of Threads

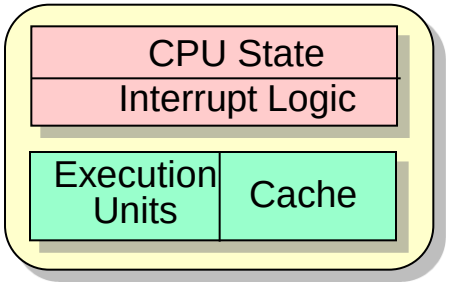


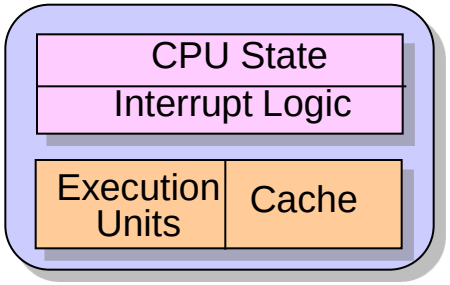
Source : http://www.intel.com ; Reference : [6]

Simple Comparison of Single-core, Multi-processor, and Multi-core Architectures

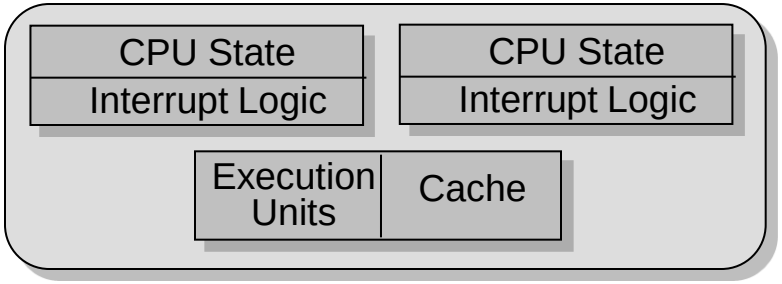


A) Single Core

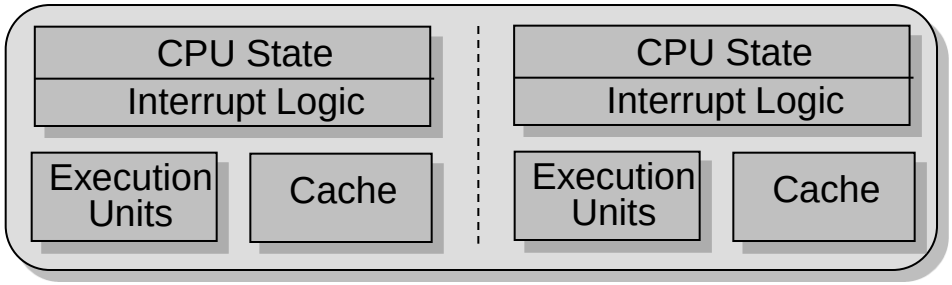


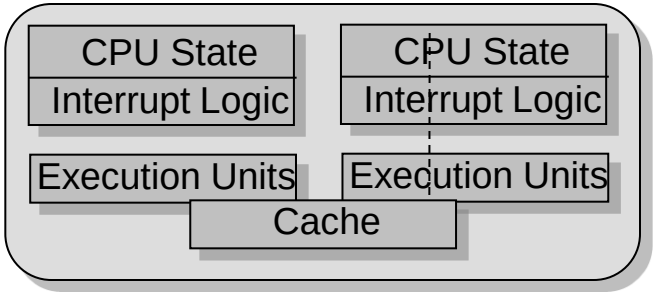


B) Multi Processor

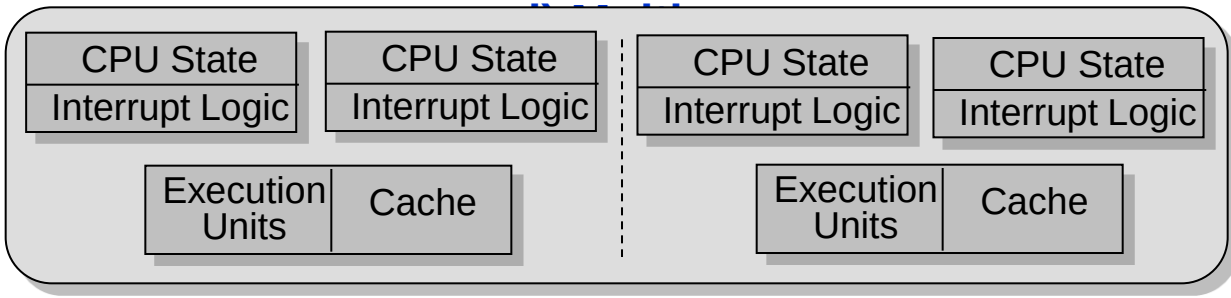


C) Hyper-Threading Technology





E) Multi-core with Shared Cache



F) Multi-core with Hyper-threading Technology

Source : http://www.intel.com ; Reference : [6], [29], [31]

Multi Core : Programming Issues

- Out of Order Execution
- Pre-emptive and Co-operative Multitasking
- SMP to the rescue
- Super threading with Multi threaded Processor
- Hyper threading the next step (Implementation)
- Multitasking
- Caching and SMT

Part II

Application Perspective

Multi Cores - Unique Challenges

TBB can be targeted for ease of development

Issues to be Addressed

Bigger *OR* Smaller Cores

Performance OR Scalability

Compute *OR* I/O Intensive

Cache Friendly OR Memory Intensive

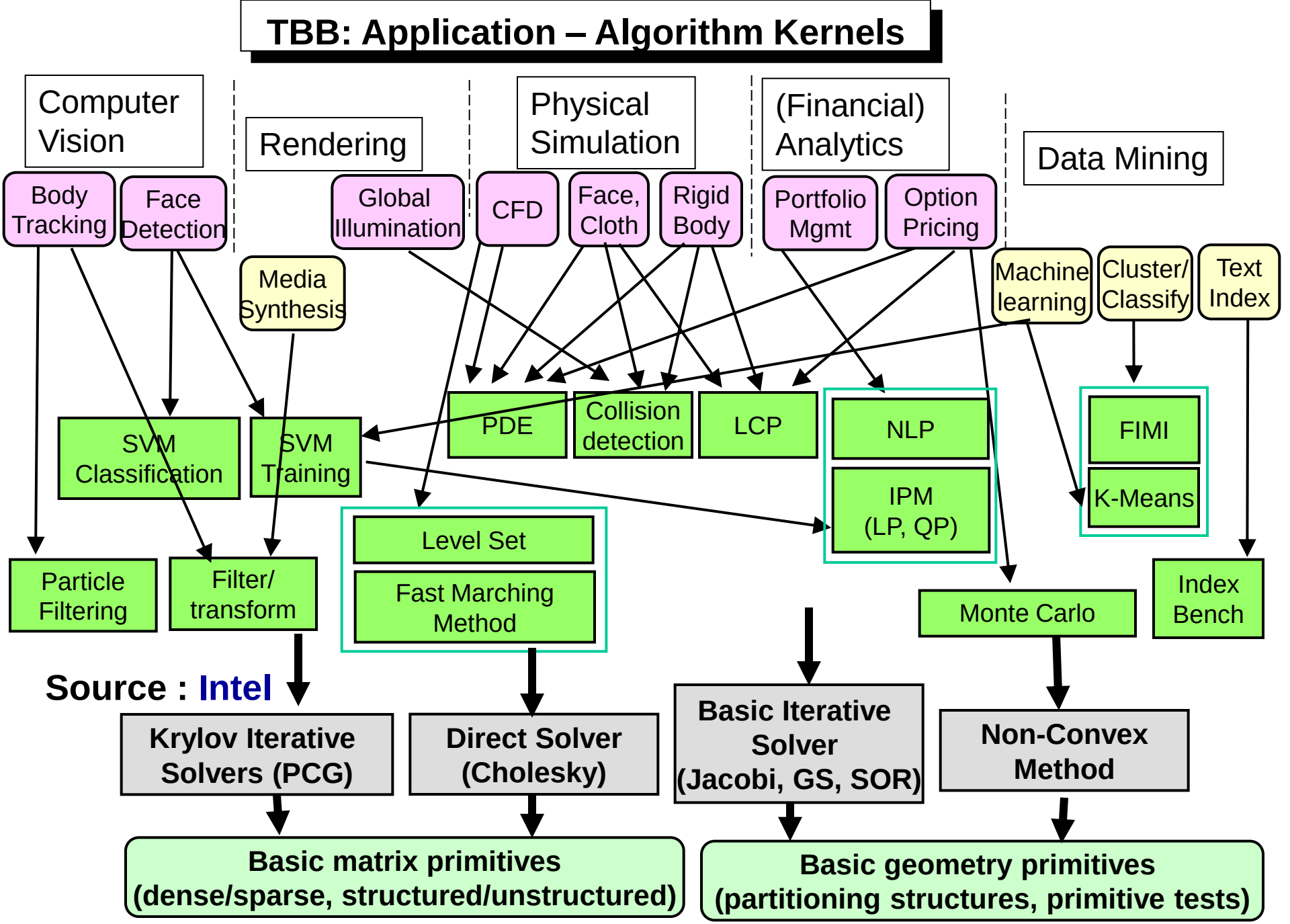
Source : http://www.intel.com ; Reference : [6]

Emerging "Killer Apps of Tomorrow" Parallel Algorithms Design

Use of TBB : Several Class of Applications

- Video : Body Tracking and Ray-Tracing
- Search for Video based imaging & Creation of Videos with animation
- Games / Interactive 3D simulation
- Real-time realistic 3D Visualization of body systems
- Wall Street Financial Analysis; Business: Data Mining – Security: Real Time Video surveillance Astrophysics,
- BioInformatics – Pattern Search Algorithms

TBB: Application – Algorithm Kernels



Programming Aspects Examples

TBB : Other Issues to be addressed

Example : Implementation of Streaming Media Player on Multi-Core

- One decomposition of work using Multi-threads
- It consists of
  - A thread Monitoring a network port for arriving data,
  - A decompressor thread for decompressing packets
  - Generating frames in a video sequence
  - A rendering thread that displays frame at programmed intervals

Source : Reference : [4]

Programming Aspects Examples

Example : Implementation of Streaming Media Player on Multi-Core

- The thread must communicate via shared buffers –
  - an in-buffer between the network and decompressor,
  - an out-buffer between the decompressor and renderer
- It consists of
  - Listen to port ……..Gather data from the network
  - Thread generates frames with random bytes (Random string of specific bytes)
  - Render threads pick-up frames & from the out-buffer and calls the display function
  - Implement using the Thread Condition Variables

Killer Apps of Tomorrow

TBB: Application-Driven Architectures:Analyzing Workloads

- Workload convergence
  - The basic algorithms shared by these high-end workloads
- Platform implications
  - How workload analysis guides future architectures
- Programmer productivity
  - Optimized architectures will ease the development of software
- Call to Action
  - Benchmark suites in critical need for redress

Why Threading Building Blocks ?

Benefits

- Easy way to express parallelism in a C++ Program
  - Library that helps to you leverage Multi-core processor performance
  - No need of having threading expert for parallelisztion
  - High Level, tasks-based paralleism - for Performance and Scalability

Part III

TBB : Algorithms - Templates

Threading Building Blocks – Overview

Benefits

- Provides rich set of templates
  - Templates and C++ concept of generic programming (C++ Standard Template Library (STL)
- Doe not require special language or compilers
- Ability to use Threading Building Blocks – any processor with any C++ Complier
- Promote Scalable Data Parallelism

Threading Building Blocks – Overview

Benefits

- Scalability
  - Data Parallel Programming – Applications
  - Take advantage of all cores on Multi core Processor
- Specify Tasks instead of Threads
  - Runtime library
  - Automatically schedules tasks onto threads
  - Makes use of efficient processor resources
  - Load balancing many tasks

Threading Building Blocks - Overview

Benefits

- Task Scheduling
  - To use TBB library, you specify tasks, not threads
- Library maps tasks onto threads in an efficient manner
  - Writing Parallel_for loop – tedious using threading packages
  - Scalable program – harder ; No benefits in Performance
- Templates can give a creditable idea of performance with respect to problem size as the number of core increases

Thread Building Blocks – Overview

Benefits

- Better portability
- Easier programming,
- More understandable source code,
- Scalability and Performance – can be predicted
- Avoid programming in a raw native thread model

Remarks :

- Programming in assembly language of parallel programming -
  - An alternative use of raw threads.
  - Maximum flexibility – Costs are very high

Threading Building Blocks – Overview

Summary

- TBB enables developer to specify tasks instead of threads
- TBB targets threading for performance on Multi-cores
- TBB Compatible with other threading packages
- TBB emphasizes scalable, data-parallel programming
- Link libraries such as Intel's Math Kernel Library (MKL) and Integrated Performance Primitives (IPP) library are implemented internally using OpenMP.
  - You can freely link a program using Threading Building Blocks with the Intel MKL or Intel IPP library.

Threading Building Blocks – Overview

Issues to be Addressed

- TBB enables developer to specify tasks instead of threads
- TBB targets threading for performance on Multi-cores
- TBB Compatible with other threading packages
- TBB emphasizes scalable, data-parallel programming
- Link libraries such as Intel's Math Kernel Library (MKL) and Integrated Performance Primitives (IPP) library are implemented internally using OpenMP.
  - You can freely link a program using Threading Building Blocks with the Intel MKL or Intel IPP library.

Application Perspective - Parallel Algorithms Design -TBB

How TBB can hide several task Scheduling events ?

- Static Load Balancing
  - Mapping for load balancing
  - Minimizing Interaction
- Data Sharing Overheads
- Dynamic Load Balancing
  - Overheads in parallel algorithms design

Application Perspective - Parallel Algorithms Design -TBB

How TBB can hide several task Scheduling events ?

- Maximize data locality
- Minimize volume of data
- Minimize frequency of Interactions
- Overlapping computations with interactions.
  - Data replication
  - Minimize construction and Hot spots.
  - Use highly optimized collective interaction operations.
    - Collective data transfers and computations
  - Maximize Concurrency.

Types of Parallelism

TBB : Other Issues to be addressed

Types of Parallelism :

- Data parallelism and Task parallelism
- Combination of Data & Task parallelism
- Stream parallelism

TBB templates

- Initializing and Terminating the Library
- Loop Paralleization
  - Parallel_for, Parallel_reduce, Parallel_scan (Grain size, Interval, Workload for iteration, Time Taken)
  - Automatic Grain Size (Not easy) – Performance Issues
  - Recursive Range Specifications (block_range), Partitioning

TBB templates- Advanced Algorithms

- parallel_while
  - Use of an unstructured stream or pile of work. Offers the ability to add additional work to the file running
- pipeline (Throughput of pipeline)
- Parallel_sort
  - Algorithm complexity
- Parallel algorithms for Streams

TBB templates- Advanced Algorithms

- Containers
  - TBB provides highly concurrent containers that permit multiple threads to invoke a method simultaneously on the same container.
  - Concurrent queue, vector, and hash map are provided.
  - These can be used with the library, OpenMP, or raw threads
- Remark : Highly concurrent containers are very important because STL containers generally are not concurrent friendly.
- TBB provides Fine-grain locking and Lock free algorithm
  - Algorithm complexity
- Parallel algorithms for Streams

TBB Scalable Memory Allocation

- Problems in Memory Allocation
  - Each Competes for global lock for each allocation and deallocation from a single global heap
  - False Sharing
- TBB offers two memory allocators
  - Scalable_allocator
  - Cache_aligned_allocator
- Memory Consistency and Fence
- TBB provides atomic Templates

TBB Mutual Exclusion / Time

- TBB - Mutex
  - Mutual exclusion will be in terms of tasks. Mutual exclusion of tasks will lead to mutual exclusion of the corresponding threads upon which TBB maps defined tasks.
  - When to Use Mutual Exclusion (To prevent race conditions and other non-deterministic and undesirable behavior of tasks)
  - Mutex behavior
- TBB provides a thread-safe and portable method to compute elapsed time
  - tick_count Class

TBB Task Scheduler

- TBB - Task-based programming can improve the performance
  - Task Scheduler manages a thread pool and hides complexity which is much better than Raw Native Threads
  - OverSubscription – Getting the number of threads right is difficult
  - Fair Scheduling – OS uses; round-robin fashion
- Load imbalance can be handled easily comparison to thread-based programming
- Portability – TBB interfaces

Intel TBB conclusions

- An Overview of TBB on Multi Cores is discussed
- TBB algorithm templates can be used
- TBB memory allocator may help to reduce the memory overheads
- TBB can improve the performance for certain class of applications on 2/4/8/24 cores
- TBB Codes can be plugged with different libraries (Use only thread-safe libraries in your TBB application)
- Intel Tools can be used for debugging the TBB application.
- TBB can be used for several Data Parallel Applications.

References

1. Andrews, Grogory R. **(2000),** Foundations of Multithreaded, Parallel, and Distributed Programming, Boston, MA : Addison-Wesley

2. Butenhof, David R **(1997),** Programming with POSIX Threads , Boston, MA : Addison Wesley Professional

3. Culler, David E., Jaswinder Pal Singh **(1999),** Parallel Computer Architecture - A Hardware/Software Approach , San Francsico, CA : Morgan Kaufmann

4. Grama Ananth, Anshul Gupts, George Karypis and Vipin Kumar **(2003),** Introduction to Parallel computing, Boston, MA : Addison-Wesley

5. Intel Corporation, **(2003),** Intel Hyper-Threading Technology, Technical User's Guide, Santa Clara CA : Intel Corporation Available at : http://www.intel.com

6. Shameem Akhter, Jason Roberts **(April 2006),** Multi-Core Programming - Increasing Performance through Software Multi-threading , Intel PRESS, Intel Corporation,

7. Bradford Nichols, Dick Buttlar and Jacqueline Proulx Farrell **(1996),** Pthread Programming O'Reilly and Associates, Newton, MA 02164,

8. James Reinders, Intel Threading Building Blocks – (**2007**) , O'REILLY series

9. Laurence T Yang & Minyi Guo (Editors), (**2006**) *High Performance Computing - Paradigm and Infrastructure* Wiley Series on Parallel and Distributed computing, Albert Y. Zomaya, Series Editor

10. Intel Threading Methodology ; Principles and Practices Version 2.0 copy right **(March 2003),** Intel Corporation

References

11. William Gropp, Ewing Lusk, Rajeev Thakur **(1999),** Using MPI-2, Advanced Features of the Message-Passing Interface, The MIT Press..

12. Pacheco S. Peter, **(1992),** Parallel Programming with MPI, , University of Sanfrancisco, Morgan Kaufman Publishers, Inc., Sanfrancisco, California

13. Kai Hwang, Zhiwei Xu, (**1998**), Scalable Parallel Computing (Technology Architecture Programming), McGraw Hill New York.

14. Michael J. Quinn (**2004**), Parallel Programming in C with MPI and OpenMP McGraw-Hill International Editions, Computer Science Series, McGraw-Hill, Inc. Newyork

15. Andrews, Grogory R. **(2000),** Foundations of Multithreaded, Parallel, and Distributed Progrmaming, Boston, MA : Addison-Wesley

16. SunSoft. Solaris multithreaded programming guide. SunSoft Press, Mountainview, CA, **(1996),** Zomaya, editor. Parallel and Distributed Computing Handbook. McGraw-Hill,

17. Chandra, Rohit, Leonardo Dagum, Dave Kohr, Dror Maydan, Jeff McDonald, and Ramesh Menon, **(2001)**,Parallel Programming in OpenMP San Fracncisco Moraan Kaufmann

18. S.Kieriman, D.Shah, and B.Smaalders **(1995),** Programming with Threads, SunSoft Press, Mountainview, CA. 1995

19. Mattson Tim, **(2002),** Nuts and Bolts of multi-threaded Programming Santa Clara, CA : Intel Corporation, Available at : http://www.intel.com

20. I. Foster **(1995,** Designing and Building Parallel Programs ; Concepts and tools for Parallel Software Engineering, Addison-Wesley (1995)

21. J.Dongarra, I.S. Duff, D. Sorensen, and H.V.Vorst **(1999),** Numerical Linear Algebra for High Performance Computers (Software, Environments, Tools) SIAM, 1999

References

22. OpenMP C and C++ Application Program Interface, Version 1.0". **(1998),** OpenMP Architecture Review Board. October 1998

23. D. A. Lewine. *Posix Programmer's Guide:* **(1991),** Writing Portable Unix Programs with the Posix. 1 Standard. O'Reilly & Associates, 1991

24. Emery D. Berger, Kathryn S McKinley, Robert D Blumofe, Paul R.Wilson, *Hoard : A Scalable Memory Allocator for Multi-threaded Applications* ; The Ninth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS-IX). Cambridge, MA, November (**2000)**. Web site URL : http://www.hoard.org/

25. Marc Snir, Steve Otto, Steyen Huss-Lederman, David Walker and Jack Dongarra, (**1998**) *MPI-The Complete Reference: Volume 1, The MPI Core, second edition* [MCMPI-07].

26. William Gropp, Steven Huss-Lederman, Andrew Lumsdaine, Ewing Lusk, Bill Nitzberg, William Saphir, and Marc Snir (**1998**) *MPI-The Complete Reference: Volume 2, The MPI-2 Extensions*

27. A. Zomaya, editor. Parallel and Distributed Computing Handbook. McGraw-Hill, **(1996)**

28. OpenMP C and C++ Application Program Interface, Version 2.5 (**May 2005**)", From the OpenMP web site, URL **:** **http://www.openmp.org/**

29. Stokes, Jon 2002 Introduction to Multithreading, Super-threading and Hyper threading *Ars Technica*, October **(2002)**

30. Andrews Gregory R. 2000, Foundations of Multi-threaded, Parallel and Distributed Programming, Boston MA : Addison – Wesley (**2000)**

31. Deborah T. Marr , Frank Binns, David L. Hill, Glenn Hinton, David A Koufaty, J . Alan Miller, Michael Upton, "Hyperthreading, Technology Architecture and Microarchitecture", Intel (**2000-01)**

Thank You

Any questions ?