SAND2008-1459C

A Programming Model for Hybrid Parallelism with Consistent Numerical Results

H. Carter Edwards
Sandia National Laboratory

SIAM Conference on
Parallel Processing for Scientific Computing
March 12-14, 2008

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Two Coupled Topics
(Two Inconvenient Truths)

- **Future parallelism includes multicore/manycore**
  - Many parallel processing cores within a single CPU socket
  - Cores contending for shared memory resources
  - Hybrid parallelism (shared/distributed) may be necessary
  - Start with homogeneous cores, worry about heterogeneous later

- **Floating point addition is NOT associative**
  - Most of us learned this, ignored it, and suffer non-determinism
  - We tell users to expect results to vary with processor count
  - At least your parallel results don't vary from run-to-run (yes?)
  - Hybrid parallelism can exacerbate this issue

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Future: Networked Manycore Nodes

- **Continue to have distributed memory parallelism**
  - **Network of processing nodes**
  - **Well understood programming model**
  - **E.g. domain decomposition and MPI**

- **Multiple processing *cores* per node**
  - **Cores-per-node = cores-per-socket * sockets-per-node**
  - **Cores contend for socket's cache memory**
  - **Cores contend for *access* to memory hierarchy**
  - **Scale node's shared main memory by #cores or #sockets?**
  - **Concern: per-core or per-socket memory overhead**

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Re-opened Dialogue on Parallel Programming Model

- **Scalability with respect to cores-per-socket**
  - **Will unmanaged sharing of the socket-to-memory resource limit scalability? (e.g., just doing MPI on the cores)**
  - **Is intentional algorithmic management of this shared resource possible? Will it make a difference?**

- **Per-core consumption of main memory**
  - **If just doing MPI on the cores:**
  - **Overhead of handing each core its own executable image**
  - **Overhead of inter-core shared data**
  - **Overhead of inter-core communication**

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Conclusion: We Need to *Investigate* Hybrid Parallel Programming Model(s)

- **Two level programming model**
  - **Outer: distributed memory model (a.k.a. the MPI model)**
  - **Inner: shared memory / parallel threads model**

- **Start investigating with …?**
  - **Pthreads: library-based standard, does not define a model**
  - **Intel TBB: C++ STL-like hiding of Pthreads, defines a model**
  - **OpenMP: compiler-based standard, defines a model**
  - **Other non-standard language?**

- **Evaluate performance**
  - **Time and space**
  - **Usability and robustness**

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

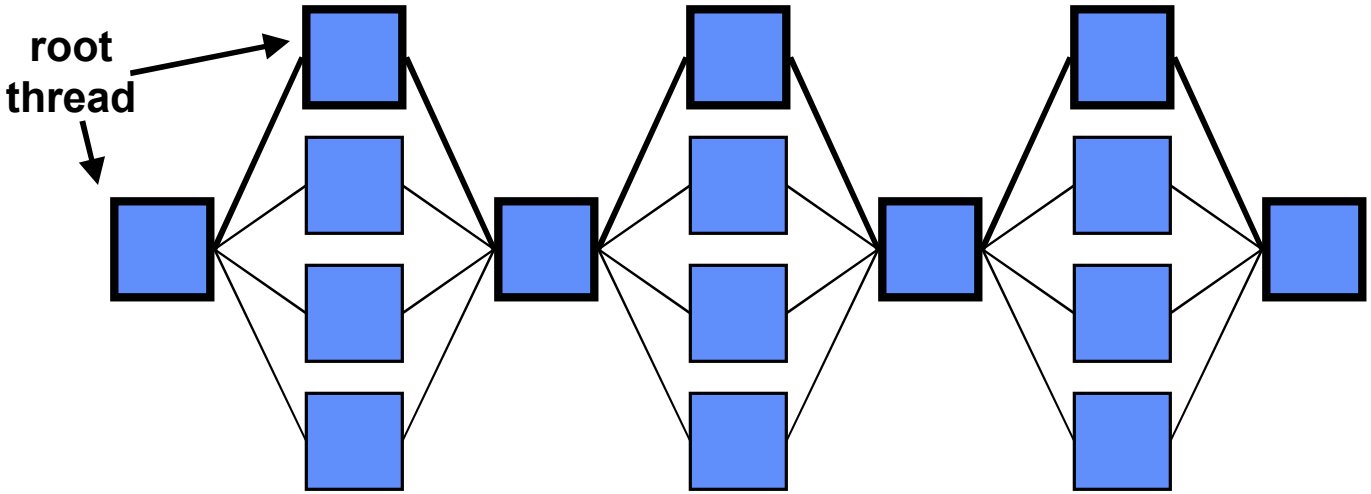
A Hybrid Parallel Programming Model: Inner Level Parallelism (for cores or sockets)

- **Goals: highly portable, simple, minimize overhead, applicable to nontrivial / complicated data structures**
- **Personal preference: standard C and C++, Unix-like OS**
  - **Regrets to developers of new languages, language extensions**
- **Use Pthreads, but how?**
  - **Oversubscribe cores or not?**
    - **Answer: at most one thread per core**
    - **Rational: avoid thread context switching overhead**
    - **Concern: thread affinity to cores**
  - **Persistent or transient threads?**
    - **Answer: create once and re-use (thread pool)**
    - **Rational: avoid thread creation / destruction overhead**

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Model for Inner Level Parallelism

- **Simplicity: only parallel operation are parallel**
  - **Sequential operations performed by a single thread**
  - **Inner level parallel operations performed by all threads**
  - **Inner level parallel operations have a local and temporary scope**
  - **Conceptually compatible with OpenMP and TBB model**



Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Prototype Thread Pool Interface (for C)

```
typedef
void (*TPI_parallel_subprogram)( void * shared_data,
                                 TPI_ThreadPool pool);

int TPI_Run( TPI_ThreadPool pool ,
             TPI_parallel_subprogram routine ,
             void * shared_data );
```

- 'routine' is called thread-parallel with:
  - The thread pool environment in which the routine is run
  - Shared 'routine_data'; never use shared global data!
  - Use 'routine_data' to pass in work, e.g. a 'struct' or 'class'
  - Include work partitioning or work stealing parameters

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Prototype Use in 'C': A simple 'dot'

```
struct TaskXY { /* routine data */
   double        sum ;
   const double * x ;
   const double * y ;
   unsigned       n ;
};

double tpi_ddot( TPI_ThreadPool pool, unsigned n,
                 const double * x, const double * y )
{
  struct TaskXY data = { 0.0 , x , y , n };
  TPI_Set_lock_size( pool , 1 );
  TPI_Run( pool, & tpi_ddot_work , & data );
  return data.sum ;
}
```

Prototype Use in 'C': A simple 'dot'

```
void tpi_ddot_work(void * arg , TPI_ThreadPool pool )
{
  struct TaskXY * const t = (struct TaskXY *) arg ;

  /* … partition the work among threads … */

  unsigned local_n = … ;
  const double * local_x = … ;
  const double * local_y = … ;

  double local_sum = ddot( local_n, local_x, local_y);

  TPI_Lock(pool,0);
  t->sum += local_sum ;
  TPI_Unlock(pool,0);
}
```

- **Non-determinism: partitioning and race to t->sum += local_sum**

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Change the Parallel Programming Model? An Opportunity to Fix Non-determinism!

- **The same application solving the same problem *should* yield the same answer – regardless of parallelism**
  - **Typically violated when:**
  - **using a different number of processors**
  - **using a different decomposition on the same processors**
  - **hopefully not for same decomposition & same processors!**
  - **But could be for thread-parallel race condition**

- **Non-deterministic behavior is user-hostile**
  - **Which is the "right" answer? A verification issue**
  - **How to deal with a bug occurring on 1000s of processors that cannot be repeated when debugging on fewer processors?**

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

A Common Source of Non-determinism in Parallel Programming

- **Floating point summation: $\sum a[i]$**
  - **Intrinsic error = $n*\varepsilon$, *for non-negative values***
  - **Repartitioning yields a different answer (number nodes)**
  - **Reordering yields a different answer (domain decomposition)**
  - **Small n $\Rightarrow$ enforce a consistent ordering**
  - **Large n $\Rightarrow$ reduce ε**

- **Algorithmic**
  - **E.g., domain decomposition (DD) dependent algorithms**
  - **Decouple algorithm's DD from number of nodes or threads**

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Prototype Use in 'C': Deterministic 'dot'

- **Non-determinism from lack of associative addition**
  - Given floating point precision of ε (~1e-16 for double)
  - Error in $\sum a[i]$ is $O(n*\varepsilon)$, given best case of $0 \leq a[i]$
- **Two summations, two sources of non-determinism**
  - Algorithm sums within threads, and then sums among threads

    among threads: $\sum$ (within each thread: $\sum$ a[i])
  - Different number of threads yields different partial sums
  - Race condition to contribute each thread's partial sum

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

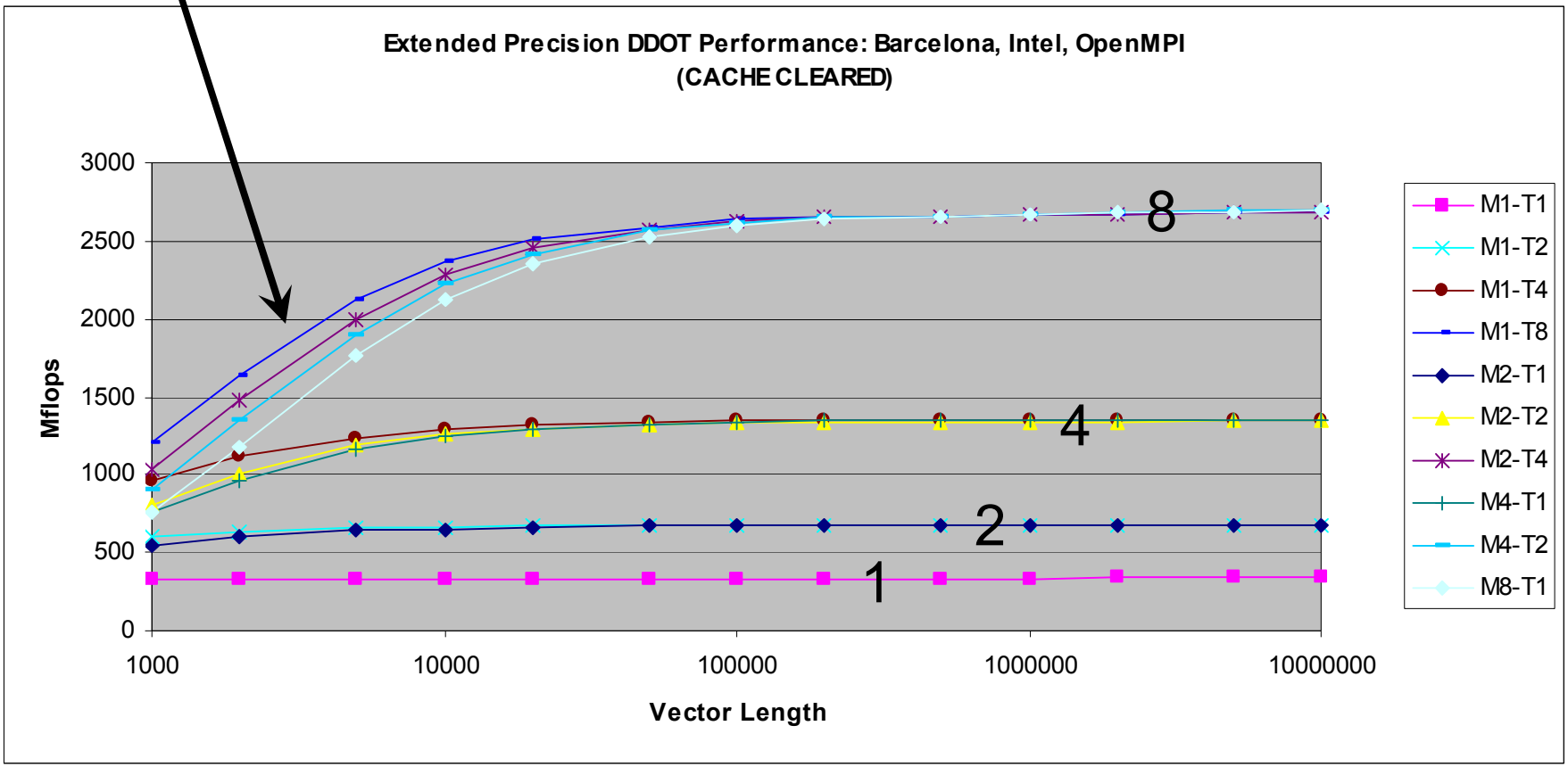
Prototype Use in 'C': Deterministic 'dot'

- **Restore determinism via improved accuracy summation**
  - **Provide "error free" result, i.e. error < ε**
    - **Accumulate positive & negative contributions separately**
    - **Accumulate in double-double precision**
    - **Error is now O(n*ε*ε) < O(ε) for n < 1,000,000,000,000,000 < 1/ε**
  - **Additional cost? 6 extra adds and 2 branches per term**
    - **Total 10 flops / 2 doubles**
    - **'dot' is a bandwidth-limited operation**
    - **"Free" in-register flops?**
- **Use double-double reduce op in the MPI_Allreduce**

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Scaling of High-Accuracy 'dot(x,y)'

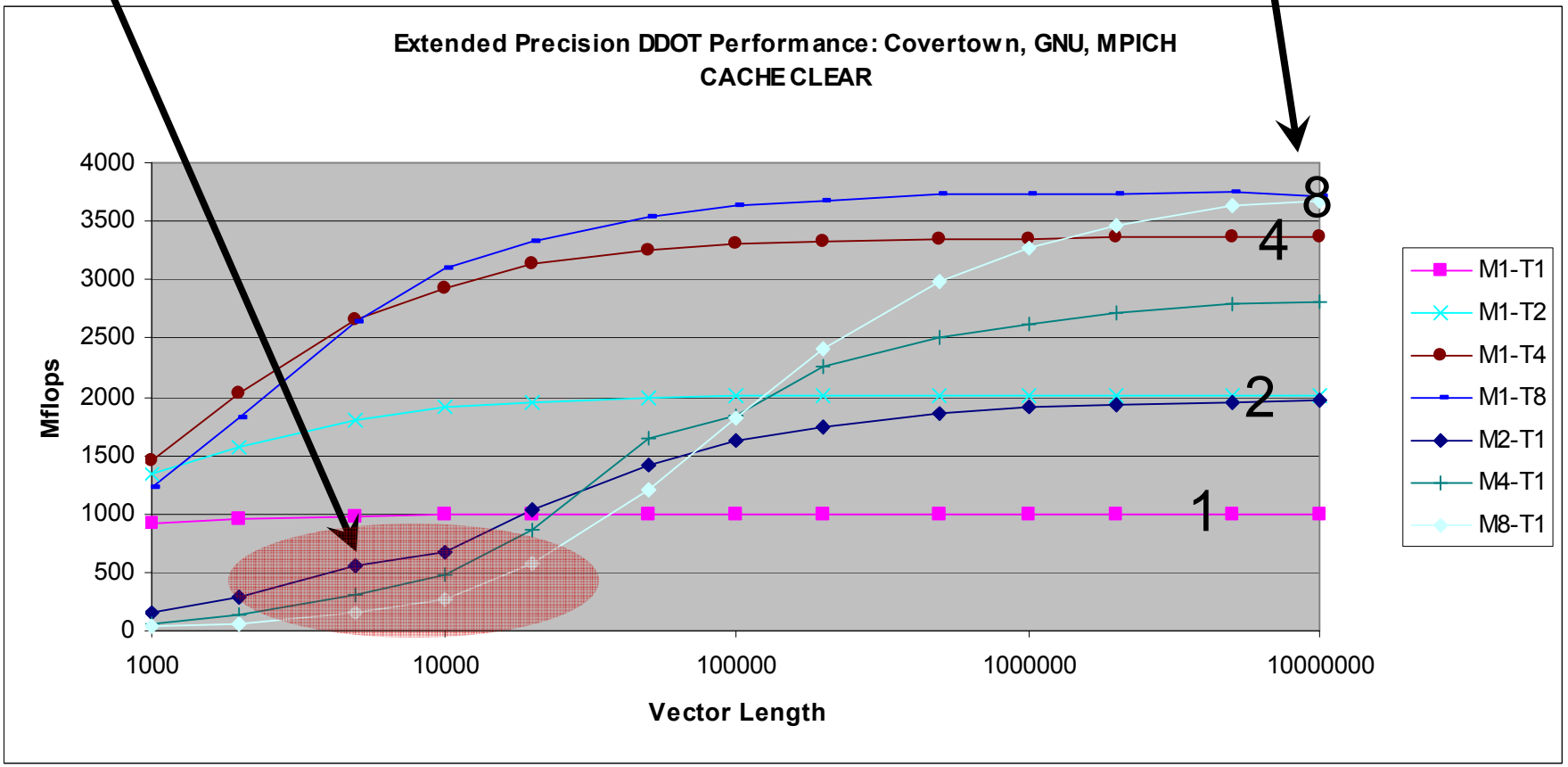
- Barcelona (AMD 2x4core) with OpenMPI
- Hybrid parallel: #Processes = MPI*Pthreads
- MPI_Allreduce overhead is expected; Scaling is great



Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Scaling of High-Accuracy 'dot(x,y)'

- Clovertown (Intel 2x4core) with MPICH
- Hybrid parallel: #Processes = MPI*Pthreads
- MPI_Allreduce overhead! Memory bandwidth saturates!



Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Sandia National Laboratories

Work Partitioning (a.k.a. Load Balancing)

- **Dot product example – trivial load balancing**
  - Partition vector into #threads ~equal length subvectors
  - sum(dot(x_p , y_p))

- **Work partitioning not always so easy**
  - Apply F(a[i] , b[i] , c[i] , …); work is non-uniform (nonlinear)
  - Data structure for 'N' arrays?
  - Time to pre-partition vs. time lost to imbalance?
  - "Just in time" load balancing via "work queue" approach

- **Work queue (a.k.a. task pool)**
  - Arbitrarily overpartition problem into pool of M chunks of work
  - E.g., F(•) applied to disjoint subsets of a[i], b[i], c[i], …
  - Could be heterogeneous: different functions and data

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Work Queue Parallelism (a.k.a. task pool)

- **Threads share a queue of work**
  - Simple linear queue, or graph of dependent units of work
  - Each thread:
    1. Lock work queue iterator
    2. Claim chunk of work / advance work queue iterator
    3. Unlock work queue iterator (release lock ASAP)
    4. Perform work on chunk
    5. Repeat until work queue is empty

- **Performance**
  - Load balancing is approximate – last thread to finish
  - Overhead – work queue iterator locking & unlocking
  - Tuning – chunks of work large enough to amortize overhead but not so large as to cause severe imbalance

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Summary

- Networks of manycore nodes are coming, ready?
  - Scalability with increasing cores per socket
  - If pure MPI, critical to have multicore leveraging implementation
- Pure MPI or hybrid MPI / thread programming model?
  - Hybrid may be necessary to address memory access contention
  - Hybrid can help with inter-core communication
  - Hybrid provides new opportunities for load balancing
- Time (past time) to address non-determinism
  - Same application and same data ⇒ give the same answer
  - Independent of #nodes, #sockets, #cores, domain decomposition
  - Will require greater discipline in algorithms and data structures

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.