

# Programming Models for Transactional Memory

Osman S. Unsal, BSC

3. June. 2008

# People

## BSC – Computer Sciences – Computer Architectures for Parallel Paradigms

<b>Name</b>	<b>Status</b>	<b>Nationality</b>	<b>Time</b>
Mateo Valero	Professor	Spain	Part-time
Eduard Ayguade	Professor	Spain	Part-time
Adrian Cristal	Doctor	Argentina	Full-time
Osman Unsal	Doctor	Turkey	Full-time
Enrique Vallejo	PhD Student	Spain	Part-time
Srdjan Stipic	PhD Student	Serbia	Full-time
Ferad Zyulkyarov	PhD Student	Bulgaria	Full-time
Sasa Tomic	PhD Student	Serbia	Full-time
Nehir Sonmez	PhD Student	Turkey	Full-time
Cristian Perfumo	PhD Student	Argentina	Full-time
Sutirtha Sanyal	PhD Student	India	Full-time
Roger Ferrer	PhD Student	Spain	Part-time
Vladimir Gajinov	PhD Student	Serbia	Full-time
Gokcen Kestor	PhD Student	Turkey	Full-Time

- Combining the Tm and OpenMP programming models
- TM Application development – the Haskell STM benchmark

# Architecture

## Transactional Memory

STM

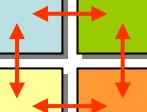
HTM

Applications  
↓  
Programming model

Functional  
Imperative

MSRC

BSC



# OpenMP TM Extensions

- Transaction clause

```
#pragma omp transaction
```

```
    [ shared(varlist) ]
```

```
    [ shared_write(varlist) ]
```

```
    [ local(varlist) ]
```

```
    [ unmanaged(varlist) ]
```

```
    [ default(shared|local|unmanaged) ]
```

```
    [ trigger_set (set) [ ON_ALL|ON_ANY ] ]
```

```
    [ when (condition) ]
```

} Data  
classification  
(hints to  
the compiler)

- Retry clause

```
#pragma omp retry
```

```
    [ when (condition) ]
```

# Classification of Transactional Data

- ***Shared*** – real transactional data
- ***Local*** – thread private data (not used for conflict detection but versioning required)
- ***Unmanaged*** – thread private data and initialized inside the transaction

```
#pragma omp parallel for private(i)  
  for(i = 0; i < N; ++i) ...  
    #pragma omp transaction unmanaged(i,d)
```

# Retry and Trigger Set

- TRIGGER SET:
  - Set of variables to watch for a transaction to restart.
  - Subset of a *read set*.
  - ON\_ANY | ON\_ALL switch.
- Example: Bounded buffer put operation

```
#pragma omp transaction
        trigger_set(size) {
    if (size == cap)
        #pragma omp retry
    size++;
    a[tail] = val;
    tail = (tail+1) % N;
}
```

# Haskell STM Benchmark

- **BlockWorld (CCHR):** 2 autonomous agents moving 100 distinct blocks
- **GCD (CCHR):** Greatest common divisor
- **Prime (CCHR):** Finds the first 4000 prime numbers
- **Sudoku (CCHR):** A sudoku solver
- **UnionFind (CCHR):** Efficiently maintain disjoint sets under union
- **SingleInt:** updating a single integer 200000 times
- **LinkedList10-100-1000:** Linked List (regular)
- **LinkedListUnr10-100-1000:** Linked list with unreadTVar
- **BinaryTree100-1000-10000 :** Binary Tree (regular)
- **BinaryTreeUnr100-1000-10000:** Binary Tree with unreadTVar

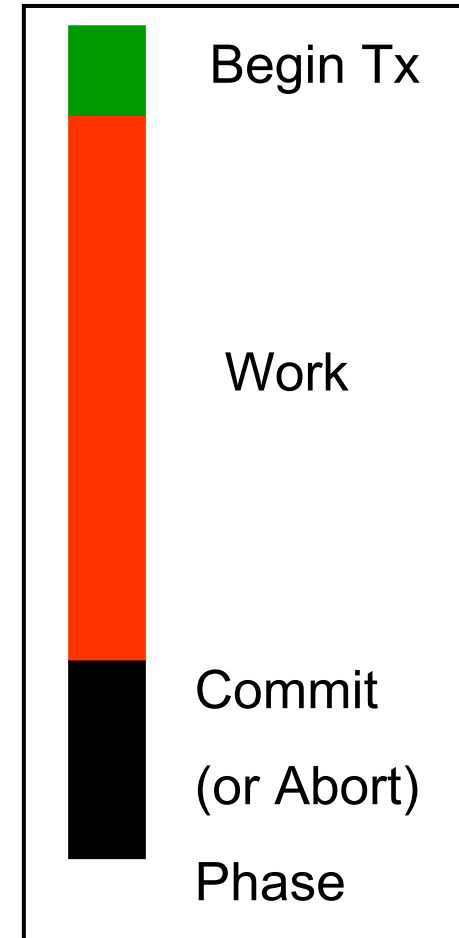
# How did we do this?

- We developed a multithreaded version of the Haskell runtime system (mostly in C) for the IA64 (Itanium) processor\*.
- Updating some counters at certain transactional events (e.g. readTVar).
- Hardware counters: Mostly generic and some native (IA64) PAPI events.
- Haskell STM Benchmarks run on SGI Altix 4700 with 128 cores.

8 \* Thanks to the Haskell community and gelato.org.

# Gathered statistics

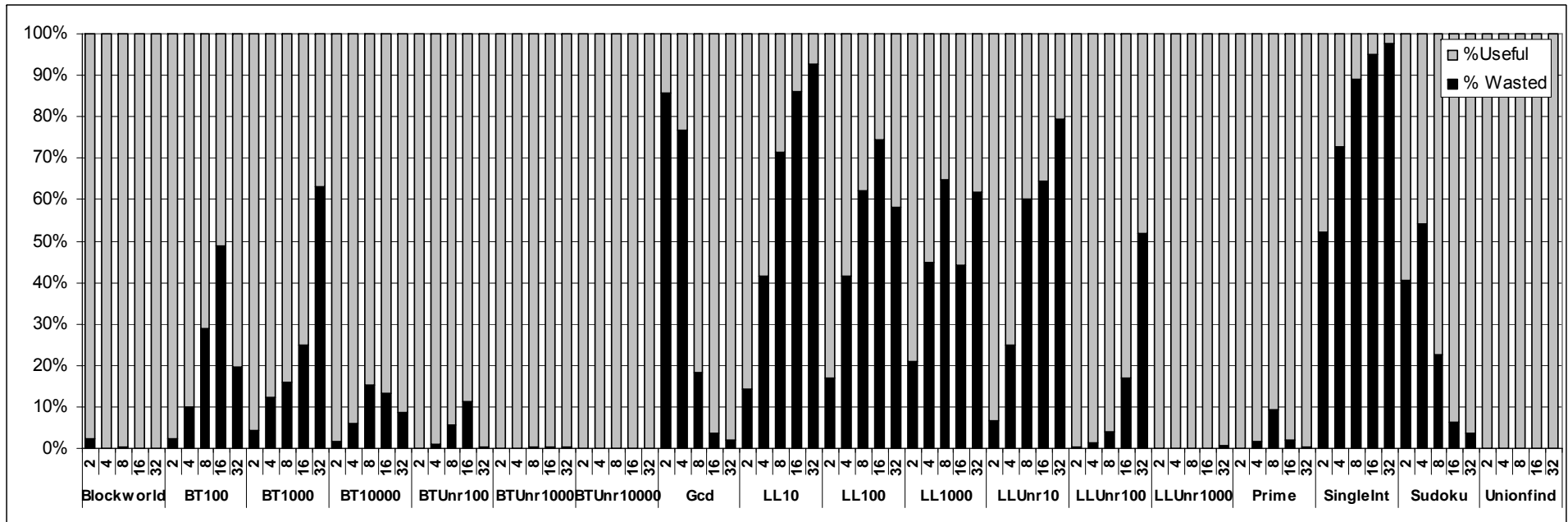
- For committed and aborted transactions:
  - Number of transactions.
  - Commit Phase Overhead
    - $\text{Commit phase time} / \text{Commit Phase} + \text{Work}$
  - Number of transactional reads and writes.
  - ReadSet and WriteSet lengths (in objects/words).
  - Wasted Work
    - $(\text{Aborted Tx time} / \text{Total Tx time})$
- Histogram of rollbacks
- Hardware counters (PAPI Library)
  - Cache:
    - Miss ratio
    - Invalidations & snoop requests
  - Branch mispredictions
  - Issued/Completed ratio



# Example: Wasted work

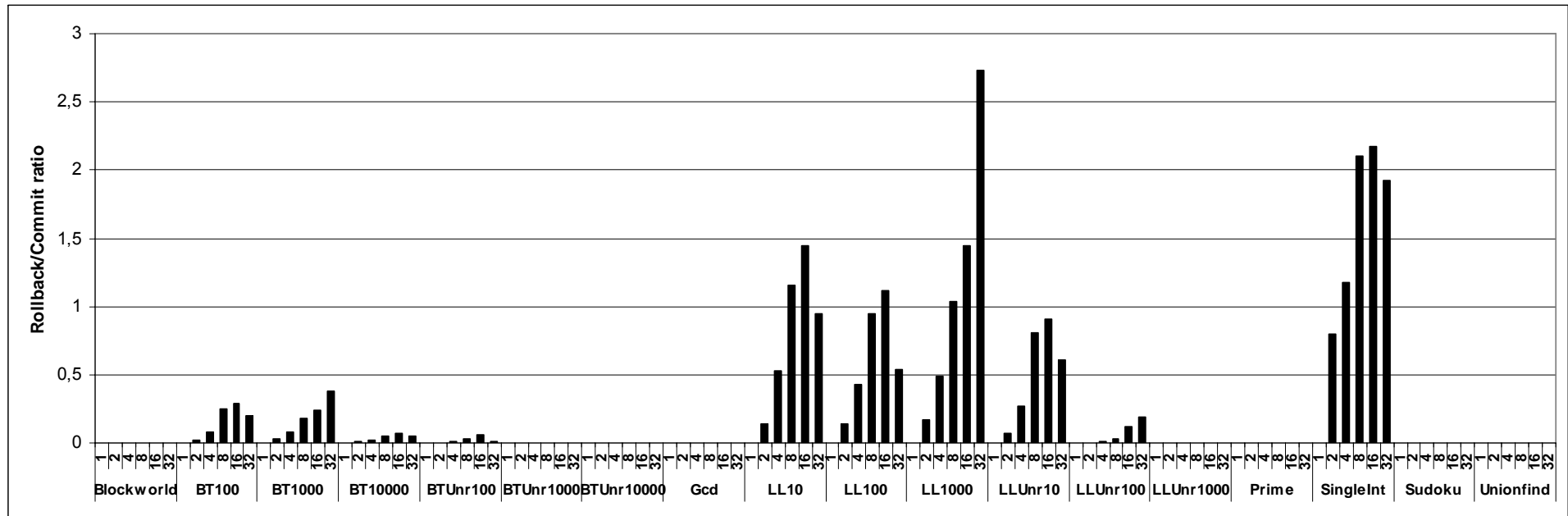
- Wasted work:

$$\frac{T(\textit{aborted})}{T(\textit{aborted}) + T(\textit{committed})}$$



# Example: Rollback rate

- Allows classifying applications in different groups.
- Accordingly to the group they belong to, the STM runtime can implement different optimizations.



# Publications

- C. Perfumo, N. Sonmez, S. Stipic, O. Unsal, A. Cristal, T. Harris, M. Valero, "The Limits of Software Transactional Memory (STM): Dissecting Haskell STM Applications on a Many-Core Environment", ACM International Conference on Computing Frontiers, May 2008
- E. Vallejo, S. Sanyal, T. Harris, F. Vallejo, R. Beivide, O. Unsal, A. Cristal, M. Valero, "Towards fair, scalable, locking", Workshop on Exploiting Parallelism with Transactional Memory and other Hardware Assisted Methods (EPHAM 2008) in conjunction with GCO, April 2008
- E. Vallejo, T. Harris, A. Cristal, O. Unsal, M. Valero, "Hybrid Transactional Memory to Accelerate Safe Lock-based Transactions", Third ACM SIGPLAN Workshop on Transactional Computing TRANSACT, February 2008
- M. Milovanović, R. Ferrer, O. Unsal, A. Cristal, X. Martorell, E. Ayguadé, J. Labarta and M. Valero, "Transactional Memory and OpenMP", International Journal of Parallel Programming, To Appear.
- M. Milovanovic, R. Ferrer, V. Gajinov, O. Unsal, A. Cristal, E. Ayguadé, M. Valero, "Multithreaded Software Transactional Memory and OpenMP", Workshop on Memory performance: Dealing with Applications, systems and architecture (Medea) in conjunction with PACT, September 2007
- F. Zyulkyarov, O. Unsal, A. Cristal, M. Milovanovic, E. Ayguade, M. Valero, T. Harris, "Memory Management for Transaction Processing Core in Heterogeneous Chip Multiprocessors", Workshop on Operating System support for Heterogeneous Multicore Architectures (OSHMA) in conjunction with PACT, September 2007
- M. Milovanovic, O. Unsal, A. Cristal, F. Zyulkyarov, S. Stipic, M. Valero, "Extending the C/C++ Language with Atomic Construct", XVI Jornadas de Paralelismo Conference, September 2007
- N. Sonmez, C. Perfumo, S. Stipic, A. Cristal, O. Unsal, M. Valero, "Increasing the Performance of Haskell Software Transactional Memory", XVI Jornadas de Paralelismo Conference, September 2007
- S. Tomic, A. Cristal, O. Unsal, M. Valero, "Hardware Transactional Memory with Operating System Support, HTMOS", Workshop on Highly Parallel Processing on a Chip in conjunction with Euro-Par, August 2007
- N. Sonmez, C. Perfumo, S. Stipic, A. Cristal, O. Unsal, M. Valero, "Increasing the Performance of Haskell Software Transactional Memory", Trends in Functional Programming, Volume 8
- C. Perfumo, N. Sonmez, O. Unsal, A. Cristal, M. Valero, Tim Harris, "Dissecting Transactional Executions in Haskell", Second ACM Workshop on Transactional Computing TRANSACT, August 2007
- T. Harris, S. Stipic, "Abstract Nested Transactions", Second ACM Workshop on Transactional Computing, August 2007
- M. Milovanovic, R. Ferrer, O. Unsal, A. Cristal, E. Ayguade, J. Labarta, M. Valero, "Transactional Memory and OpenMP," International Workshop on OpenMP, June 2007
- T. Harris, A. Cristal, O. Unsal, E. Ayguade, F. Gagliardi, B. Smith, M. Valero, "Transactional Memory: An Overview", IEEE Micro, May-June 2007
- C. Perfumo, N. Sonmez, S. Stipic, O. Unsal, A. Cristal, M. Valero, "UnreadTVar: Extending Haskell Software Transactional Memory for Performance", Symposium on Trends in Functional Programming, April 2007
- M. Milovanovic, O. Unsal, A. Cristal, S. Stipic, F. Zyulkyarov, M. Valero, "Compile Time Support for Using Transactional Memory in C/C++ Applications," Workshop on Interaction between Compilers and Computer Architecture, February 2007