# ScalaMemAnalysis: A Compositional Approach to Cache Analysis of Compressed Memory Traces

Saransh Gupta
under the guidance of
Dr. Frank Mueller

Computer Science
NC STATE UNIVERSITY

# Outline

- ScalaMemTrace Background
- ScalaMemTrace Re-design
- ScalaMemAnalysis Design (Under Development)
- LDC Code Comparison(Serial & OpenMP)
  - SMT/SMA vs Dinero IV
    - Cache Simulation Time Comparison
    - Cache Hit Comparison
- Future Work

# Background - ScalaMemTrace

**ScalaMemTrace (prior work at NCSU):**

- Creates compressed traces of memory accesses
  - Uses the PIN tool (a binary instrumentation tool)
  - instruments all load and stores made by an application to memory
  - memory access has a unique signature
  - computed using a stack-walk
    - criteria for recognizing patterns
- PIN Compressor compresses trace file.
- Compressed trace file contains:
  - Regular Section Descriptors (RSD):
    - Address Accessed
    - Signature
    - Stride
    - Type of access
  - Power RSD (PRSD):
    - Length of loop
    - Loop iteration count

# ScalaMemTrace Redesign (Multi-Threaded Support)

- Data structures with varied stride at different loop levels recorded for uni-processor.
  - Multi-threaded program's compression ignore different loop levels
- Support for more than one level of stride for uni-processor
  - nested loops are recorded inaccurately as single loops for multi-threaded programs while compressing.
- Currently, working on addition of data structures on inter-thread and inter-node level
  - support for stride information per loop level.
  - Criteria for match includes
    - signature match
    - stride match
- Now would be possible to accurately record nested loops
  - with stride information per loop level

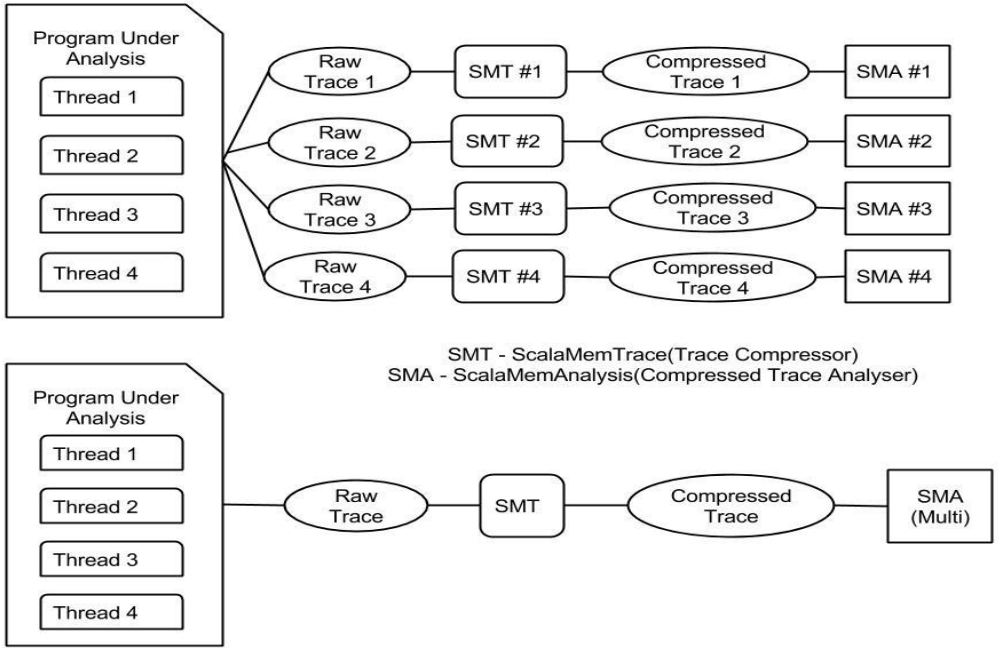
# Workflow for MultiThreaded Version

**Per-Thread Approach:**

- Collect individual traces
- Run multiple instances of SMT and pipe traces threadwise.
- Pipe threadwise traces into respective SMA.

**All Threads Integrated Analysis:**

- *Collect one integrated raw trace.*
- *Compress using SMT that compresses the raw trace into one compressed trace.*
- *SMA analyses patterns considering the thread-id's as well.*




Computer Science
NC STATE UNIVERSITY

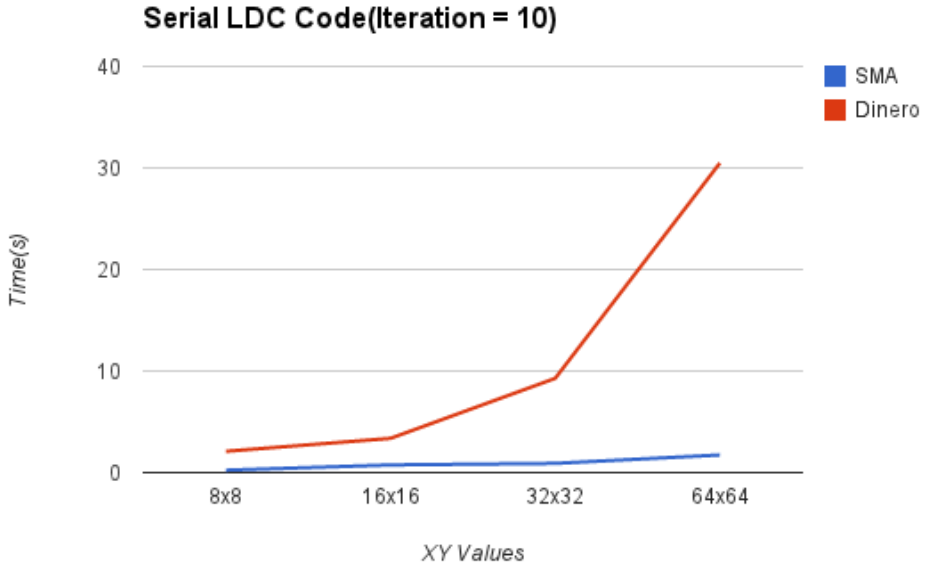
# ScalaMemAnalysis: Multi-threaded Extension (Under Development)

**MultiThreaded Extension : All Threads Integrated Analysis**

- Reuse Distance Algorithm - Distance\Iteration(Thread-wise) between two similar PRSD's.
- Analyse the final merged trace from SMT
  - SMT compresses all the inter-thread traces on a node into one trace file.
    - EPRSD's contain additional loops.
    - No additional data structures required.
    - Additional conditional checks for multi level strides added.
  - SMT compresses the inter-node traces into one final trace file.
    - EPRSD's contain another level of abstraction.
    - Radix tree like structure used while compression.
- Identify patterns using context based reuse distance, node-id and thread-id.
- Extrapolation can be done with ease.
  - Weak Scaling: Number of threads as well as problem size increases.
  - Strong Scaling: Problem size constant but increasing thread count.

# Cache Simulation Time Comparison of LDC Code: Serial (SMT/SMA vs Dinero IV)

| Input XY | Raw Trace (MB) | Dinero IV Trace(MB) | SMT PRSD Trace(MB) |
|---|---|---|---|
| 8x8 | 14 | 1.2 | 1.2 |
| 16x16 | 31 | 2.8 | 1.5 |
| 32x32 | 93 | 8.5 | 1.9 |
| 64x64 | 329 | 31 | 1.9 |




Cache Simulation Time Graph

- PGI Compiler
- Varying x-y input from 8x8 to 64x64
- SMA simulation takes less time as compared to Dinero.

Computer Science
NC STATE UNIVERSITY

# Cache Hit Comparison of LDC Code: Serial SMT/SMA vs Dinero IV

- The number of hits reported by Dinero are considerable to SMA.

| Matrix Size | 8x8 | 16x16 |
|---|---|---|
| Dinero | 4172 | 4341 |
| SMA | 4071 | 4565 |

# Threadwise Cache Simulation Time Comparison of LDC Code: OpenMP (SMT/SMA vs Dinero IV)

| Input XY (8x8) | # of Raw Trace References | Removing OMP Waiting Thread References | # of Raw Trace References |
|---|---|---|---|
| **Thread 0** | 3184 | 3184 - 0 | 3184 |
| **Thread 1** | 35844 | 35844 -(11868+(2967 *2))*2 | 240 |
| **Thread 2** | 16228 | 16228 -(5332+(1331)*2)*2 | 240 |
| **Thread 3** | 25896 | 25896 -(8552+(2138)*2)*2 | 240 |

- The behavior of raw traces takes into account the memory references made by threads between two parallel sections while waiting. We do not need them as they are not a part of the memory references that the program makes.
- Working on a way to remove the recurring trace pattern of the threads while waiting to get the sanitized raw traces.

# Future Work

- Nishanth's work on ScalaMemAnalysis Compositional Cache Analysis of Compressed Memory Traces is under submission to ISPASS'15
- Implement multi-threaded SMT & SMA extensions capable of recording multi-stride programs.
- Implement SMA Extrapolation Extension.
- Report large anomalies to users.
- Based on extrapolation, possible to provide theoretical performance predictions for large number of threads, such as on a GPU.
- Improve accuracy of SMA. Better handling of holes.

# THANK YOU

Computer Science
NC STATE UNIVERSITY