Using TSX For NVM Consistency
Pradeep Fernando, Irina Calciu, Jayneel Gandhi

1. Non-volatile memory (NVM) provides persistent load/stores at memory speeds
   - Crash-consistent/atomic NVM updates still need software transactions

2. Undo-logging needs frequent cache flushes and memory fences
   - Redo-logging needs read re-direction
   - Software transactions have high overheads!
   - Intel TSX supports atomic and isolated execution of code blocks
   - TSX does not guarantee durability nor crash-consistency

3. Keep speculative cache lines in L1
   - Use cache coherence protocol to detect conflicting updates
   - Optimistic concurrency semantic

4. No memory fencing and cache flushing after every write
   - No read re-direction
   - Logging is overlapped with transactional execution of the code
   - Optimistic concurrency enabled durable transactions

5. Vacation benchmark – travel reservation system
   - Application runs with single thread, on real hardware.

6. Graph showing time (seconds) vs. no persistence