IBM T.J. Watson Research Center

Potential Show-Stoppers for Transactional Synchronization

Christoph von Praun

PPoPP – Panel, March 15, 2007

1) Technical Challenges for TM

2) Environment, “Killer Apps“

Technical challenges for TM

- Semantics and simplicity of the programming interface:
  - handling of irreversible operations, compensation actions
  - modularity and nesting
  - conditional synchronization, communication with concurrent transactions
  - interaction of transactional and non-transactional code
  - large transactions, contention management

- Performance and implementation:
  - reduce overheads
  - ‘right’ combination of software and hardware mechanisms

  → tremendous progress over the past years

Multicore workloads (1/2)

Web-Services

- *The* growth field in commercial computing:
  - large investments that can drive technological advances
  - lots of web-service developers from emerging economies
- Programming model:
  - "containerized" application frameworks, e.g., J2EE (concurrency not exposed to programmer)
  - "shared nothing architectures", e.g., PHP, Ruby on Rails, ...

→ very high pressure to develop scalable middleware

Web-Services continued ...

- Middleware is tuned for scalable concurrency *now.*
- Alternative technologies to enable scalable concurrency are becoming common practice:
  - non-blocking algorithms, libraries for concurrency utilities
  - advanced locking schemes
  - speculative lock elision
  - read-copy-update, ...
- The bar for TM is rising: TM has to offer *very significant advantage* over alternative technologies to justify cost of change.
  - better programmability
  - higher performance

→ IT moves fast, timing matters

→ TM currently behind the train

Multicore workloads (2/2)

Scientific applications

- Focused usage context
  - programmers willing to rewrite some code
  - semantic limitations of TM are acceptable
- Users care about *performance*
- Parallel computing and algorithms are established in the community
  - several factors can limit scalability, TM may solve one of them

Game workloads [Tim Sweeney, POPL'06]

- Focused usage context
  - (S)TM seems right match for parallel game simulation
  - alternatives to transactional synchronization are unattractive
- Users care about *simplicity of the programming interface, programmability* (rapid development)

Summary

- TM is a great technology
  - technical challenges are not show-stoppers
- Success or failure of TM not only decided on technical merit
- Critical for widespread adoption of novel technology (TM) is economic context (need “killer-application”)
- Different domains have different challenges:
  - middleware for web-services: *timing*
  - scientific applications: *performance*
  - games: *simplicity of the programming interface, programmability*

praun@us.ibm.com