ProActive SPMD and Fault Tolerance
Protocol and Benchmarks

Brian Amedro et al.
INRIA - CNRS

1st workshop INRIA-Illinois
June 10-12, 2009
Paris
Outline

• ASP Model Overview

• ProActive SPMD

• Fault Tolerance

• Benchmarks
Asynchronous Sequential Process

- Communication with message-passing
  - Request / Reply
  - No memory sharing

- Asynchronous request services
  - Request queue
  - Rendezvous
  - Future with wait-by-necessity

- Confluence and determinacy
  - Causal ordering
  - Activity state characterization

- Java implementation

Denis Caromel and Ludovic Henrio and Bernard Serpette
Asynchronous and Deterministic Objects, POPL’04
Asynchronous Sequential Process

- Communication with message-passing
  - Request / Reply
  - No memory sharing

- Asynchronous request services
  - Request queue
  - Rendezvous
  - Future with wait-by-necessity

- Confluence and determinacy
  - Causal ordering
  - Activity state characterization

- Java implementation

Denis Caromel and Ludovic Henrio and Bernard Serpette
Asynchronous and Deterministic Objects, POPL’04
Asynchronous Sequential Process

- Communication with message-passing
  - Request / Reply
  - No memory sharing

- Asynchronous request services
  - Request queue
  - Rendezvous
  - Future with wait-by-necessity

- Confluence and determinacy
  - Causal ordering
  - Activity state characterization

- Java implementation

Denis Caromel and Ludovic Henrio and Bernard Serpette
Asynchronous and Deterministic Objects, POPL’04
Asynchronous Sequential Process

- Communication with message-passing
  - Request / Reply
  - No memory sharing

- Asynchronous request services
  - Request queue
  - Rendezvous
  - Future with wait-by-necessity

- Confluence and determinacy
  - Causal ordering
  - Activity state characterization

- Java implementation

Futures and WBN

Denis Caromel and Ludovic Henrio and Bernard Serpette
Asynchronous and Deterministic Objects, POPL’04
Asynchronous Sequential Process

- Communication with message-passing
  - Request / Reply
  - No memory sharing

- Asynchronous request services
  - Request queue
  - Rendezvous
  - Future with wait-by-necessity

- Confluence and determinacy
  - Causal ordering
  - Activity state characterization

- Java implementation

Denis Caronel and Ludovic Henrio and Bernard Serpette
Asynchronous and Deterministic Objects, POPL’04

Thursday, June 11, 2009
Asynchronous Sequential Process

- Communication with message-passing
  - Request / Reply
  - No memory sharing

- Asynchronous request services
  - Request queue
  - Rendezvous
  - Future with wait-by-necessity

- Confluence and determinacy
  - Causal ordering
  - Activity state characterization

- Java implementation

Denis Caromel and Ludovic Henrio and Bernard Serpette
Asynchronous and Deterministic Objects, POPL’04

Thursday, June 11, 2009
Asynchronous Sequential Process

- Communication with message-passing
  - Request / Reply
  - No memory sharing

- Asynchronous request services
  - Request queue
  - Rendezvous
  - Future with wait-by-necessity

- Confluence and determinacy
  - Causal ordering
  - Activity state characterization

- Java implementation

http://proactive.inria.fr

Denis Caromel and Ludovic Henrio and Bernard Serpette
Asynchronous and Deterministic Objects, POPL’04
ProActive OO-SPMD

Objectives

• Provide an MPI-like programming model

• Ease the porting an MPI application to ProActive

• Give *Object Oriented* to SPMD model
ProActive OO-SPMD

Features

• Asynchronous collective operations
  Asynchronous barrier

• Asynchronous group communication
  Scattering, gathering

• Take into account topology
  Optimized algorithm
Fault Tolerance with ASP

Objectives

- **Transparency**
  No piece of code dedicated to Fault Tolerance in applications

- **Portability**
  No assumption about underlying hardware

- **Consistency**
Fault Tolerance with ASP

Propositions

• Rollback Recovery

• TTC + Communication Induced Checkpointing: Transparency
  No programmer intervention

• Constrained Checkpointability: Portability
  No Checkpoint during a service
  Un-consistency of recovery lines

Tolérance aux pannes pour objets actifs asynchrones - protocole, modèle et expérimentations
Fault Tolerance with ASP

Principles of the Protocol

\[ C_i^{n[Q_1, Q_2, Q_4]} \]
Fault Tolerance with ASP

Principles of the Protocol

• Orphan and In-transit Messages

• Promised Requests

• Request Reception History
Fault Tolerance with ASP

Orphan Messages

$Q_1$ is an *Orphan Request*
Fault Tolerance with ASP

Promised Request

synchronization with the actual request arrival
Fault Tolerance with ASP

In-transit Messages

Q₀ is an In-transit Request
Fault Tolerance with ASP

Request Reception History
Fault Tolerance with ASP

Synthesis

• Orphan Request
  Replace with a promised request with wait-by-necessity

• In-transit Requests
  Reception of such a request is journalized into a request reception history

• In-transit Reply
  Can’t happen: occur after the reception of an orphan request, so a checkpoint have been performed
Benchmarking Fault Tolerance

- NAS Parallel Benchmarks
- 5 kernels: EP, CG, FT, MG, IS
- About 10,000 LOC
Benchmarking Fault Tolerance

NAS Benchmark : CG.A

Checkpoint size (MB)

# nodes

Time (sec)

Checkpoint size (MB)
T standard
T without checkpoints
T with checkpoints
Benchmarking Fault Tolerance

NAS Benchmark : CG.C

- Checkpoint size (MB)
- T standard
- T without checkpoints
- T with checkpoints

Thursday, June 11, 2009
Thank you for your attention