Fault-Tolerant Adaptive Parallel and Distributed Simulation

Gabriele D’Angelo
<g.dangelo@unibo.it>
http://www.cs.unibo.it/gdangelo/

joint work with:

Stefano Ferretti, Moreno Marzolla and Lorenzo Armaroli

London, England
Distributed Simulation and Real Time Applications (DS-RT), 2016
Presentation **outline**

- Assumptions and Motivations
- Parallel And Distributed Simulation (PADS)
- Adaptive PADS (self-clustering of Simulated Entities)
- GAIA/ARTÌS Software Architecture
- Problem: System Reliability
- Fault Tolerance in Distributed Systems
- FT-GAIA Software Architecture
- Fault-tolerance Type of Failures
- Experimental Evaluation
- Conclusions
Assumptions and Motivations

- **Discrete Event Simulation (DES)**
  - a set of *interacting entities* (can be seen as *agents*)
  - simulation is updated by *events*
  - the events happen at *discrete points in time*

- **Sequential** DES techniques are *not suitable* for the simulation of complex systems

- Parallel DES → **Parallel And Distributed Simulation (PADS)**
  - complex execution architecture (*multi-core, multi-processors, clusters, cloud*)
  - (aiming for) better scalability
Parallel And Distributed Simulation (PADS)

SE = Simulated Entity
LP = Logical Process
PE = Processing Element (e.g. CPU core)
Parallel And Distributed Simulation (PADS)

PARTITIONING

SE = Simulated Entity
LP = Logical Process
PE = Processing Element (e.g. CPU core)
Parallel And Distributed Simulation (PADS)

**SE** = Simulated Entity

**LP** = Logical Process

**PE** = Processing Element (*e.g.* CPU core)
Parallel And Distributed Simulation (PADS)

SE = Simulated Entity
LP = Logical Process
PE = Processing Element (e.g. CPU core)
Adaptive PADS: self-clustering of SEs

SE = Simulated Entity
LP = Logical Process
PE = Processing Element (e.g. CPU core)
Adaptive PADS: self-clustering of SEs

SE = Simulated Entity
LP = Logical Process
PE = Processing Element (e.g. CPU core)
Software Architecture

Simulation model

model behavior:
- state variables
- event handlers

Framework services:
- high level communication APIs
- migration support
- clustering heuristics

Runtime services:
- synchronization
- communication
- simulation management

Operating system

GAIA

ARTÌS
GAIA/ARTÌS Software Architecture

- **Simulation model**
  - **Model behavior:**
    - State variables
    - Event handlers

- **GAIA**
  - **Framework services:**
    - High level communication APIs
    - Migration support
    - Clustering heuristics

- **ARTÌS**
  - **Runtime services:**
    - Synchronization
    - Communication
    - Simulation management

- **Operating system**

- **Simulated entities**
  - A, E, C, G

- **Logical processes**
  - LP<sub>1</sub>, LP<sub>2</sub>, ..., LP<sub>n</sub>
**Problem: System Reliability**

\[ R(N, t) = \text{joint probability that all } N \text{ LPs operate without failures for } \geq t \text{ time units} \]

**ASSUMPTION:** Mean Time To Failure (MTTF) = 1 year
Problem: System Reliability

A simulation with LP=1000 and requiring one day to complete is very unlikely to terminate (even if MTTF = 1 year)

\[ R(N,t) = \text{joint probability that all } N \text{ LPs operate without failures for } \geq t \text{ time units} \]

**ASSUMPTION:** Mean Time To Failure (MTTF) = 1 year
Fault Tolerance in Distributed Systems

Many approaches are possible, two main categories:

- **Checkpointing**: rollback based recovery scheme (e.g. the checkpoints are periodically saved on stable storage)
  - *the interval between checkpoints is a parameter*

- **Functional Replication**: some (or all) parts of the PADS are replicated
  - *the degree of replication is a parameter*
Fault Tolerance in Distributed Systems

Many approaches are possible, two main categories:

- **Checkpointing**: rollback based recovery scheme (e.g. the checkpoints are periodically saved on stable storage)
  - *the interval between checkpoints is a parameter*

- **Functional Replication**: some (or all) parts of the PADS are replicated
  - *the degree of replication is a parameter*
Fault Tolerance in Distributed Systems

FT-GAIA works by replicating Simulation Entities (SEs)

For example:

- Checkpointing: rollback based recovery scheme (e.g. the checkpoints are periodically saved on stable storage)

  *given N distinct SEs*

  *FT-GAIA generates N x M entities*

- Functional Replication: some (or all) parts of the PADS are replicated

  *the degree of replication is a parameter (e.g. M)*
Fault Tolerance in Distributed Systems

FT-GAIA works by replicating Simulation Entities (SEs)

For example:

- Checkpointing: rollback based recovery scheme (e.g. the checkpoints are periodically saved on stable storage)
  
  **given N distinct SEs**
  
  **FT-GAIA generates N x M entities**
  
  *(that is M independent instances of each SE)*

- Functional Replication: some (or all) parts of the PADS are replicated
  
  **The cost of Functional Replication:**
  
  a) **additional processing power** (e.g. CPU load)
  
  b) **number of messages**
  
  *(i.e. M redundant messages for each “original” message)*
FT-GAIA Software Architecture

- **model behavior:**
  - state variables
  - event handlers

- **fault-tolerance management:**
  - transparent functional replication
  - handling of failures

- **framework services:**
  - high level communication APIs
  - migration support
  - clustering heuristics

- **runtime services:**
  - synchronization
  - communication
  - simulation management

- **simulation model**

- **FT-GAIA**

- **GAIA**

- **ARTÌS**

- **operating system**
FT-GAIA Software Architecture

**simulated entities**

A₁ E₁ G₁
F₁ D₁ B₁
E₂ F₂ D₂ G₂
B₃ C₁ G₃ B₂
E₃ F₃ C₂ B₃
C₃ D₃ A₂

**simulation model**

**FT-GAIA**

**GAIA**

**ARTÌS**

**operating system**

**model behavior:**
- state variables
- event handlers

**fault-tolerance management:**
- transparent functional replication
- handling of failures

**framework services:**
- high level communication APIs
- migration support
- clustering heuristics

**runtime services:**
- synchronization
- communication
- simulation management
FT-GAIA Software Architecture

**simulated entities**

- A1
- E1
- G1
- D1
- F1
- C1
- B1
- F2
- D2
- G2
- E2
- B2
- C2
- A2
- A3
- E3
- G3
- D3
- B3
- C3
- F3

**simulation model**

- state variables
- event handlers

**FT-GAIA**

- transparent functional replication
- handling of failures

**GAIA**

- high level communication APIs
- migration support
- clustering heuristics

**ARTÌS**

- synchronization
- communication
- simulation management

**operating system**

- simulated entities

- A
- E
- C
- G
- F
- D
- B
FT-GAIA Software Architecture

**simulated entities**

A
B
C
D
E
F
G

**operating system**

**simulation model**

**FT-GAIA**

**model behavior:**
- state variables
- event handlers

**fault-tolerance management:**
- transparent functional replication
- handling of failures

**framework services:**
- high level communication APIs
- migration support
- clustering heuristics

**runtime services:**
- synchronization
- communication
- simulation management

**constraint**

(implementation issue)

A given PE can not allocate more than one replica of each SE (that is, "no replications in the same LP")

**note:** this needs to be integrated in the adaptive migration scheme described before

**constraint**

A given PE can not allocate more than one replica of each SE (that is, "no replications in the same LP")
Fault Tolerance: Type of Failures

- **Crash failures:** a PE (e.g. CPU core) **halts but operated correctly** until it halted. The local state of all SEs on such PE is lost.

  - *to tolerate* $f$ *faults: $\geq f+1$ instances of each SE* ($M=f+1$)

- **Byzantine Failures:** all types of **abnormal behavior** of a PE (e.g. transmission of erroneous/corrupted data, computation errors)

  - *to tolerate* $f$ *faults: $M=2f+1$ replicas are needed (that is: “majority rule”)
Experimental Evaluation

- **Simulation model**: simple P2P communication protocol (PING-PONG) over randomly generated directed overlay graphs

- **Execution platform**: cluster of Intel® Core® i5-4590 3.30 GHz CPU, 8 GB RAM, Debian Jessie, Fast Ethernet LAN

- **Methodology**: 15 independent replications of each simulation run. Reported mean values with 99.5% confidence intervals

- **Warning**: prototype implementation, not dedicated cluster (i.e. background load)

All the source code and scripts used for this performance evaluation are available with a Free Software license from: http://pads.cs.unibo.it
Impact of the number of LPs and SEs

WCT with different num. of SEs

<table>
<thead>
<tr>
<th>LPs Configuration</th>
<th>Wall Clock Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 LPs, No FT</td>
<td></td>
</tr>
<tr>
<td>3 LPs, Crash FT</td>
<td></td>
</tr>
<tr>
<td>3 LPs, Byzantine FT</td>
<td></td>
</tr>
<tr>
<td>4 LPs, No FT</td>
<td></td>
</tr>
<tr>
<td>4 LPs, Crash FT</td>
<td></td>
</tr>
<tr>
<td>4 LPs, Byzantine FT</td>
<td></td>
</tr>
<tr>
<td>5 LPs, No FT</td>
<td></td>
</tr>
<tr>
<td>5 LPs, Crash FT</td>
<td></td>
</tr>
<tr>
<td>5 LPs, Byzantine FT</td>
<td></td>
</tr>
</tbody>
</table>

# of Simulation Entities
Impact of the number of LPs and SEs

WCT with different num. of SEs

- 3 LPs, No FT
- 3 LPs, Crash FT
- 3 LPs, Byzantine FT
- 4 LPs, No FT
- 4 LPs, Crash FT
- 4 LPs, Byzantine FT
- 5 LPs, No FT
- 5 LPs, Crash FT
- 5 LPs, Byzantine FT

Wall Clock Time (sec)

# of Simulation Entities
Impact of the number of LPs and SEs

WCT with different num. of SEs

- 3 LPs, No FT
- 3 LPs, Crash FT
- 3 LPs, Byzantine FT
- 4 LPs, No FT
- 4 LPs, Crash FT
- 4 LPs, Byzantine FT
- 5 LPs, No FT
- 5 LPs, Crash FT
- 5 LPs, Byzantine FT

Wall Clock Time (sec)

# of Simulation Entities
Impact of the number of LPs and SEs

WCT with different num. of SEs

WCT:
- 4 LPs < 3 LPs
- NO FT < Crash FT < Byzantine FT

Wall Clock Time (sec)

# of Simulation Entities
Impact of the number of LPs and SEs

WCT with different num. of SEs

Wall Clock Time (sec)

# of Simulation Entities
Impact of the number of LPs and SEs

5 LPs:
- higher communication overhead
- high cost of fault tolerance
Impact of the number of failures (5 LPs)

WCT with different numbers of faults (5 LPs)

- Crash FT, 2000 SEs
- Byzantine FT, 2000 SEs
- Crash FT, 6000 SEs
- Byzantine FT, 6000 SEs

Wall Clock Time (sec)

# of Faults
Impact of the number of failures (8 LPs)

WCT with different numbers of faults (8 LPs)

- Crash FT, 2000 SEs
- Byzantine FT, 2000 SEs
- Crash FT, 6000 SEs
- Byzantine FT, 6000 SEs

Wall Clock Time (sec) vs. # of Faults
Impact of the number of failures (8 LPs)

WCT with different numbers of faults (8 LPs)

Wrong partitioning!
the computational load is too low for 8 LPs

Crash FT, 2000 SEs
Byzantine FT, 2000 SEs
Crash FT, 6000 SEs
Byzantine FT, 6000 SEs

# of Faults
Impact of SEs migration

WCT for varying num. of Simulation Entities, Migration on/off

- No migr., No FT
- Migration, No FT
- No migr., Crash FT
- Migration, Crash FT
- No migr., Byzantine FT
- Migration, Byzantine FT

Wall Clock Time (sec)

# of Simulation Entities
Impact of SEs migration

WCT for varying num. of Simulation Entities, Migration on/off

- No migr., No FT
- Migration, No FT
- No migr., Crash FT
- Migration, Crash FT
- No migr., Byzantine FT
- Migration, Byzantine FT

Wall Clock Time (sec)

# of Simulation Entities
Impact of SEs migration

WCT for varying num. of Simulation Entities, Migration on/off

- No migr., No FT
- Migration, No FT
- No migr., Crash FT
- Migration, Crash FT
- No migr., Byzantine FT
- Migration, Byzantine FT

Wall Clock Time (sec)

# of Simulation Entities
Impact of SEs migration

WCT for varying num. of Simulation Entities, Migration on/off

- No migr., No FT
- Migration, No FT
- No migr., Crash FT
- Migration, Crash FT
- No migr., Byzantine FT
- Migration, Byzantine FT

Wall Clock Time (sec)

# of Simulation Entities
Conclusions

- A high degree of Fault-tolerance can be achieved in very large scale parallel/distributed simulation.

- The cost of Fault-tolerance is a moderate increase in the computational and communication load.

- This permits the usage of “low reliability” computational resources or “cheap” interruptible spot cloud instances.

- The efficiency of the GAIA self-clustering mechanism needs to be improved when used with FT-GAIA.
Further Information

Gabriele D'Angelo, Stefano Ferretti, Moreno Marzolla, Lorenzo Armaroli

Fault-Tolerant Adaptive Parallel and Distributed Simulation


A draft version of this paper is available on the open e-print archive (arxiv)

All the source code used in this paper will be soon available at http://pads.cs.unibo.it

Gabriele D'Angelo

- E-mail: <g.dangelo@unibo.it>
- http://www.cs.unibo.it/gdangelo/
Fault-Tolerant Adaptive Parallel and Distributed Simulation

Gabriele D’Angelo
<g.dangelo@unibo.it>
http://www.cs.unibo.it/gdangelo/

joint work with:

Stefano Ferretti, Moreno Marzolla and Lorenzo Armaroli

London, England
Distributed Simulation and Real Time Applications (DS-RT), 2016