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 **S**imulation and **R**eal **T**ime Applications (**DS-RT**), 2016

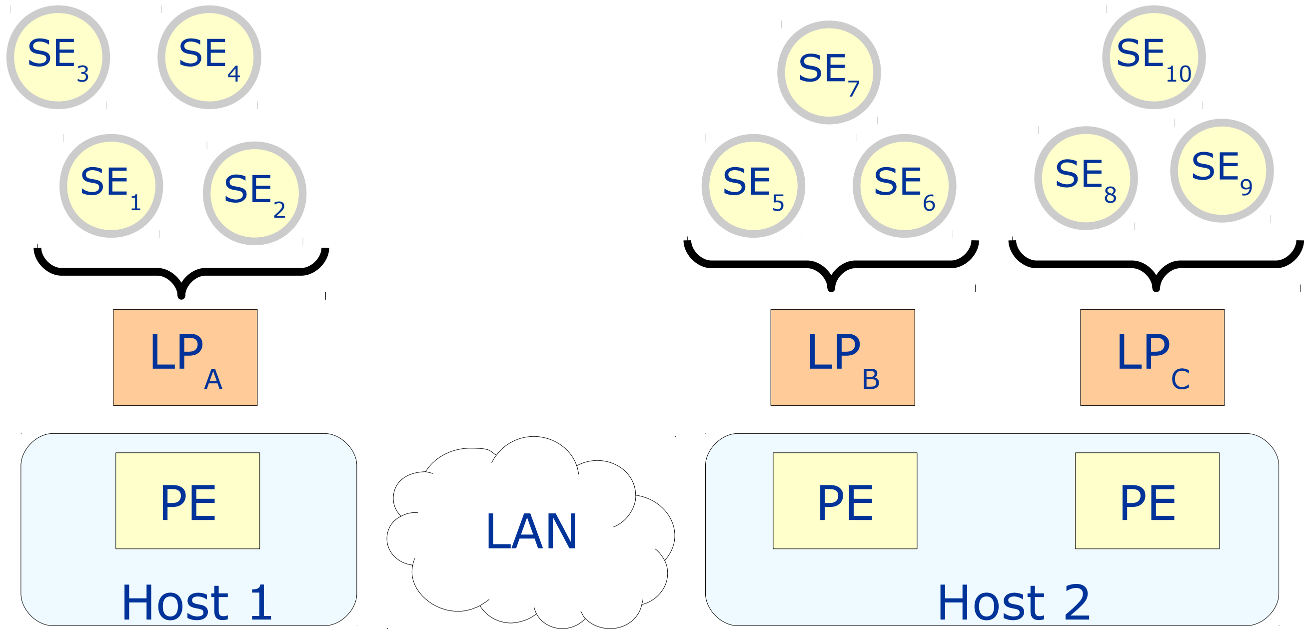
Presentation **outline**

- 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**

- **D**iscrete **E**vent **S**imulation (**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)

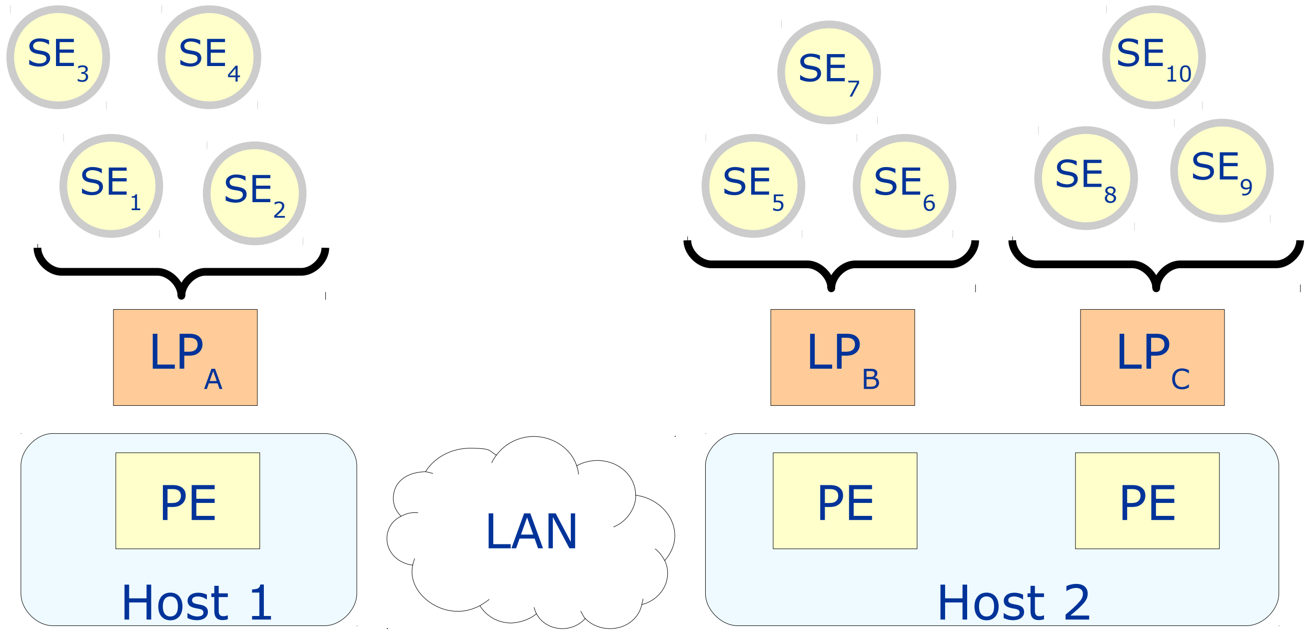


SE = Simulated Entity

LP = Logical Process

PE = Processing Element (*e.g. CPU core*)

PARTITIONING

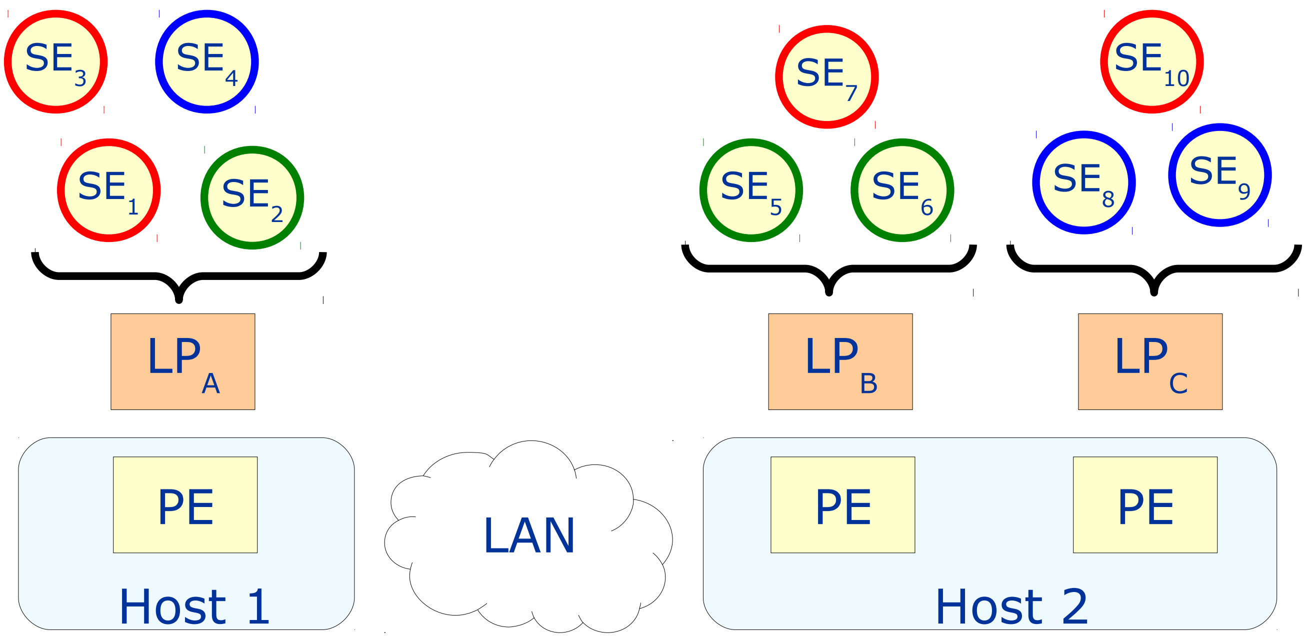


SE = Simulated Entity

LP = Logical Process

PE = Processing Element (*e.g. CPU core*)

Parallel And Distributed Simulation (PADS)

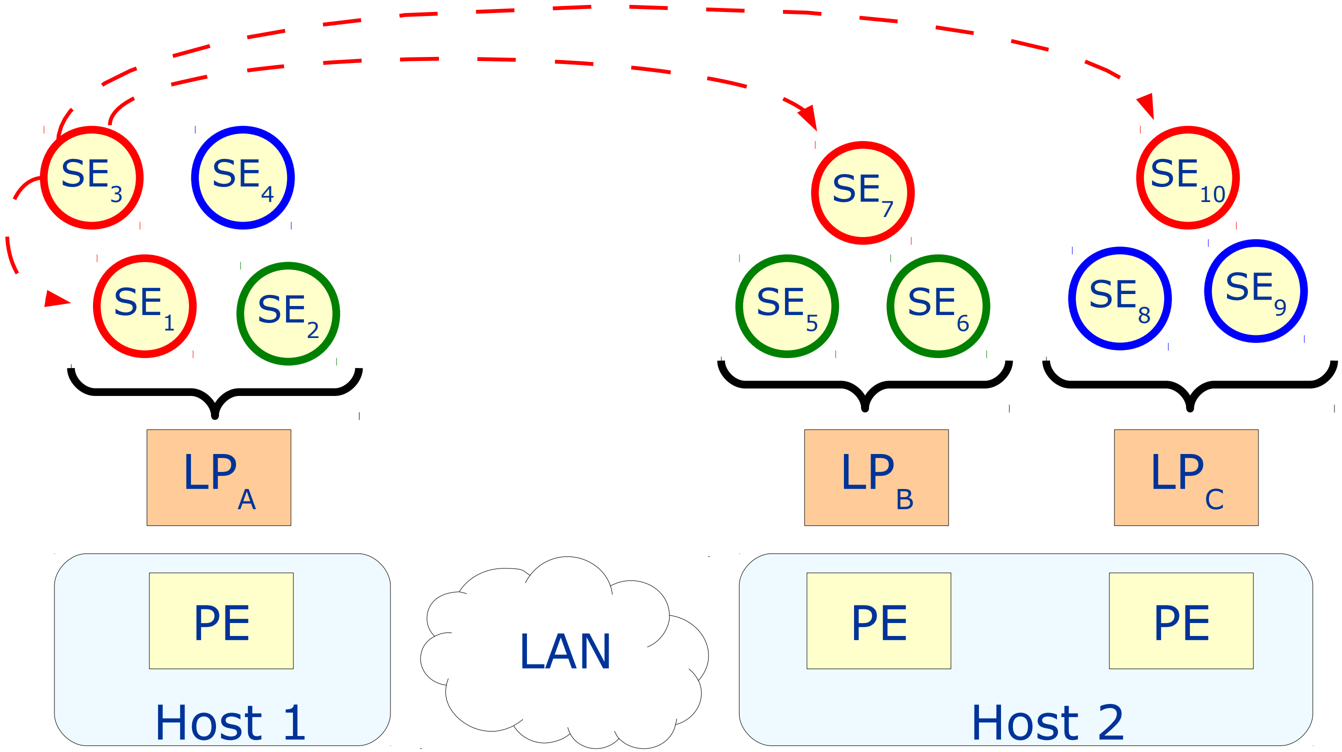


SE = Simulated Entity

LP = Logical Process

PE = Processing Element (*e.g. CPU core*)

Parallel **A**nd **D**istributed **S**imulation (**PADS**)

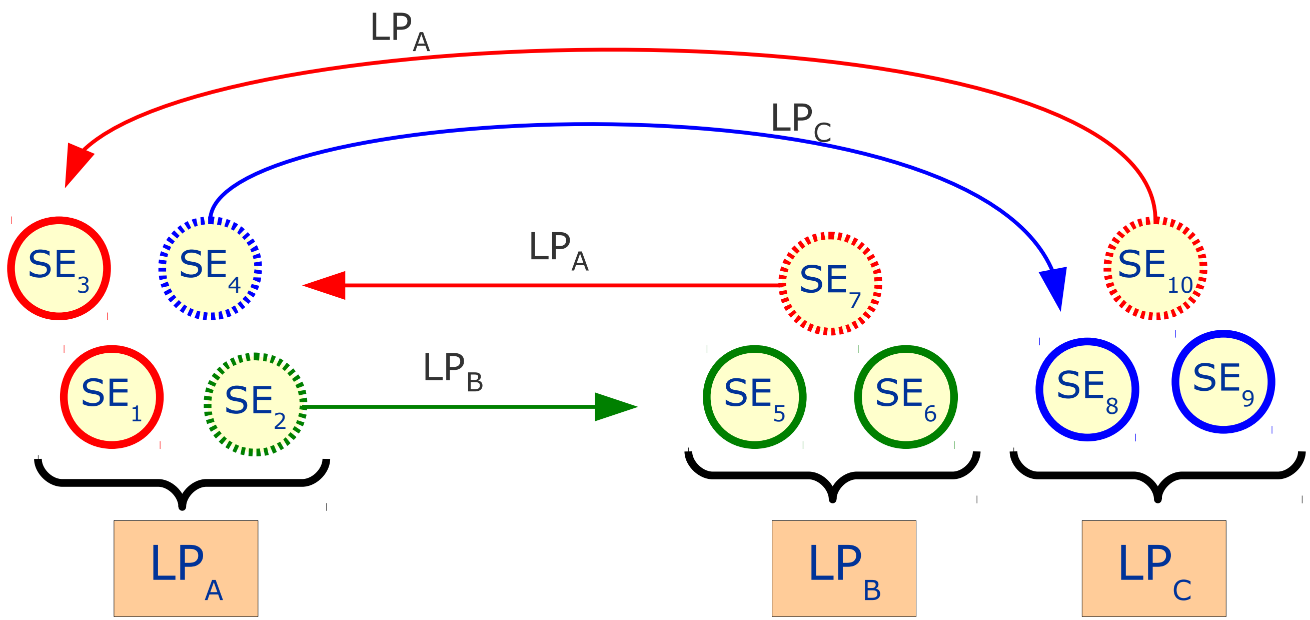


SE = Simulated Entity

LP = Logical Process

PE = Processing Element (*e.g. CPU core*)

Adaptive PADS: self-clustering of SEs



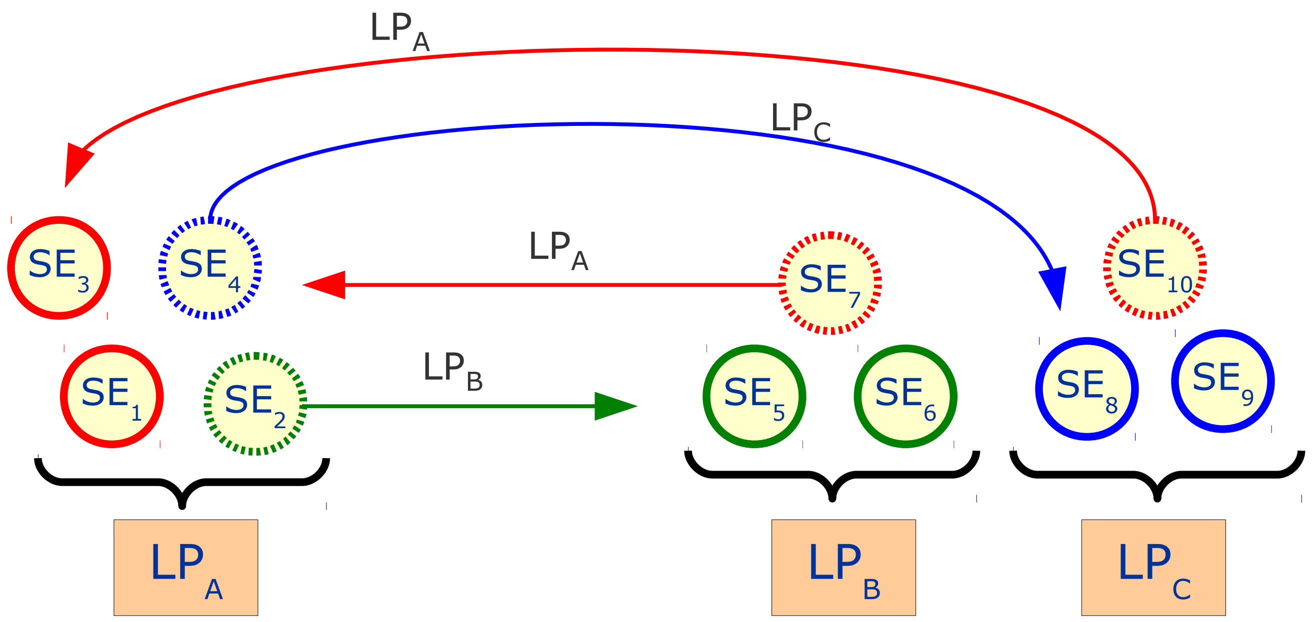
SE = Simulated Entity

LP = Logical Process

PE = Processing Element (*e.g. CPU core*)

Adaptive PADS: self-clustering of SEs

MIGRATION

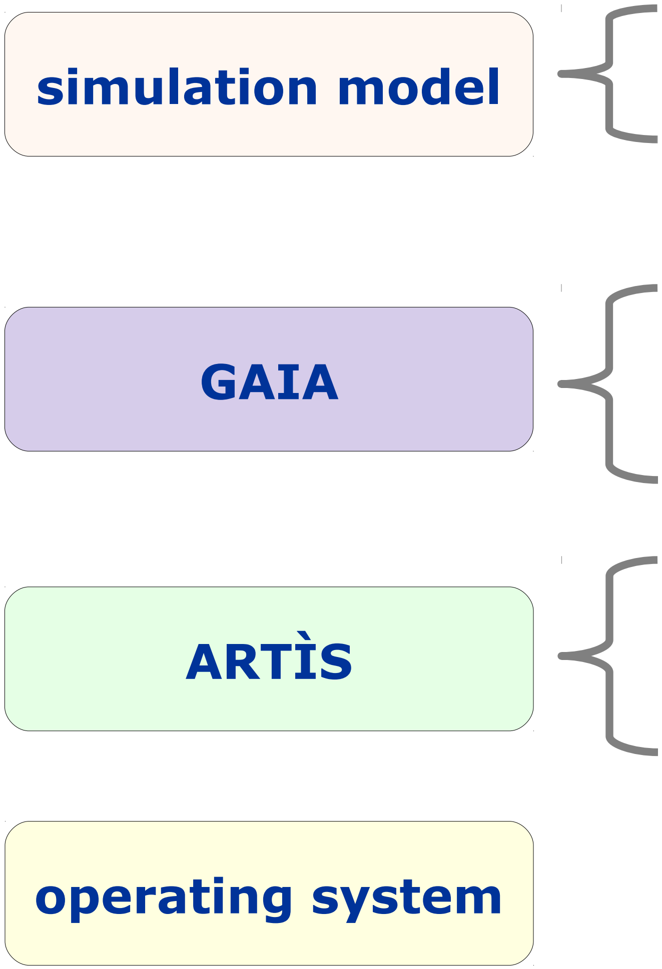


SE = Simulated Entity

LP = Logical Process

PE = Processing Element (*e.g. CPU core*)

Software Architecture



model behavior:

- state variables
- event handlers

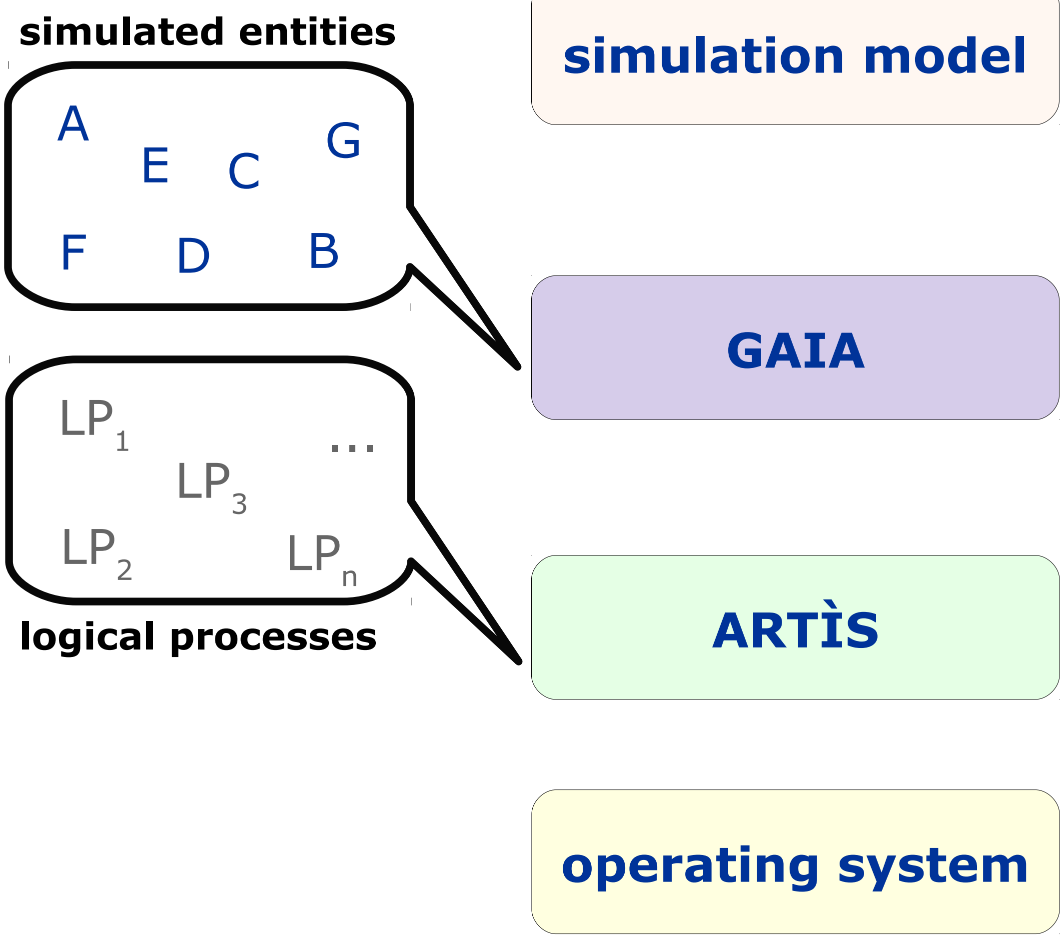
framework services:

- high level communication APIs
- migration support
- clustering heuristics

runtime services:

- synchronization
- communication
- simulation management

GAIA/ARTÌS Software Architecture



model behavior:

- state variables
- event handlers

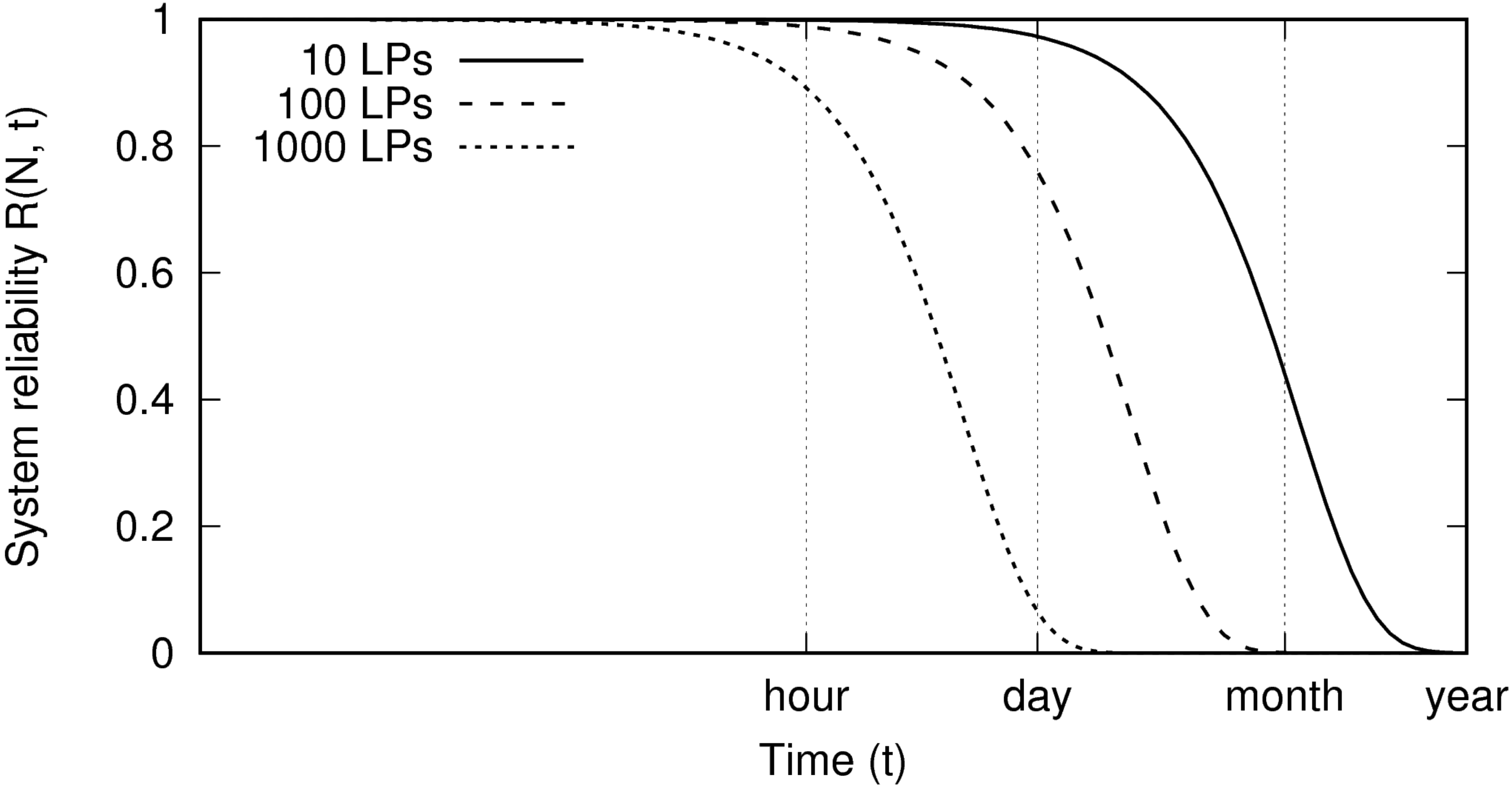
framework services:

- high level communication APIs
- migration support
- clustering heuristics

runtime services:

- synchronization
- communication
- simulation management

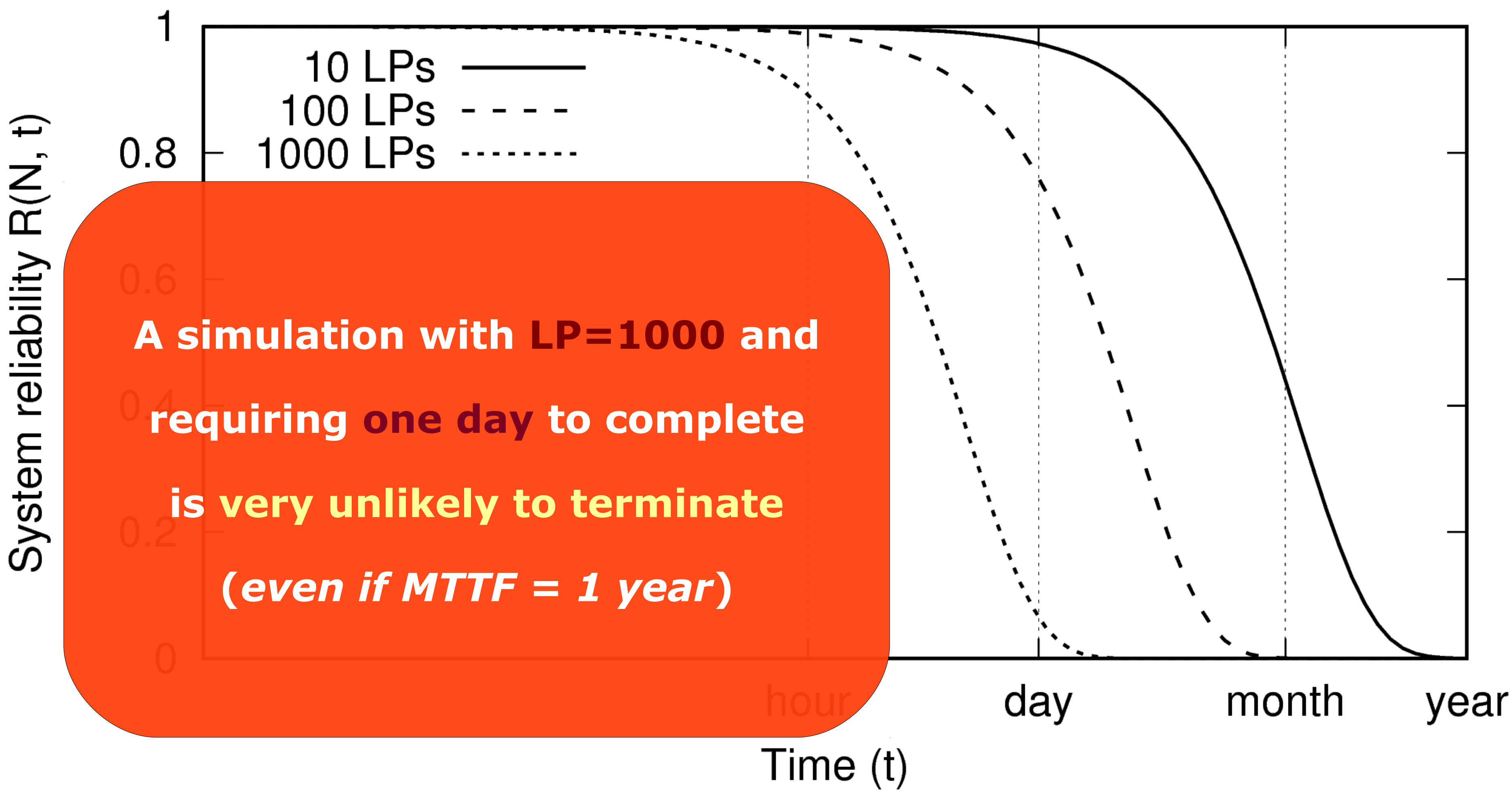
Problem: System Reliability



R(N,t) = joint probability that all **N** LPs operate without failures for ≥ **t** time units

ASSUMPTION: Mean Time To Failure (MTTF) = 1 year

Problem: System Reliability



R(N,t) = joint probability that all **N** LPs operate without failures for ≥ **t** 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*

FT-GAIA works by replicating Simulation Entities (SEs)

For example:

given N distinct SEs

FT-GAIA generates N x M entities

(that is M independent instances of each SE)

- **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 (**e.g. M**)*

Fault Tolerance in Distributed Systems

FT-GAIA works by replicating Simulation Entities (SEs)

For example:

given N distinct SEs

FT-GAIA generates N x M entities

(that is M independent instances of each SE)

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

simulation model

model behavior:

- state variables
- event handlers

FT-GAIA

fault-tolerance management:

- transparent functional replication
- handling of failures

GAIA

framework services:

- high level communication APIs
- migration support
- clustering heuristics

ARTÌS

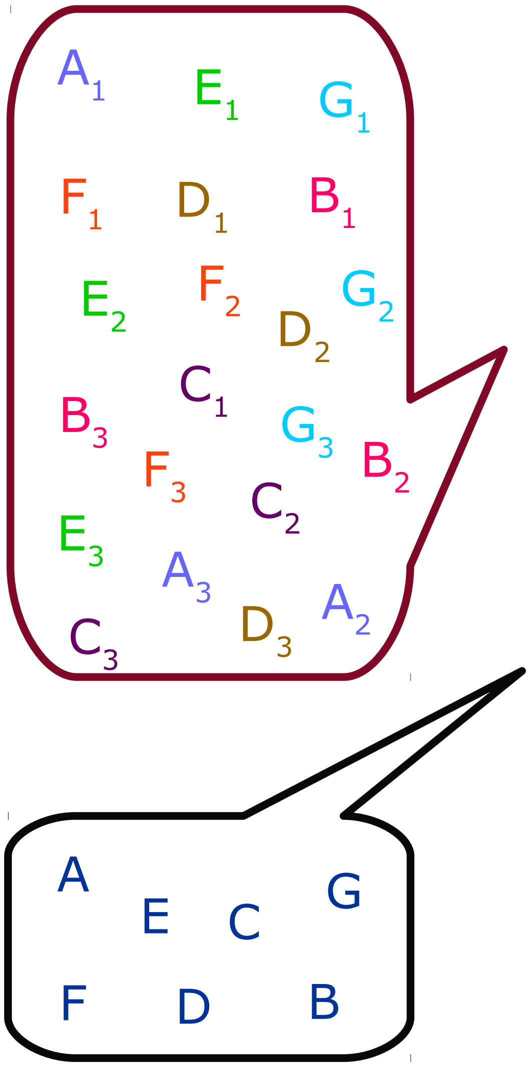
runtime services:

- synchronization
- communication
- simulation management

operating system

FT-GAIA Software Architecture

simulated entities



simulated entities

simulation model

model behavior:
- state variables
- event handlers

FT-GAIA

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

GAIA

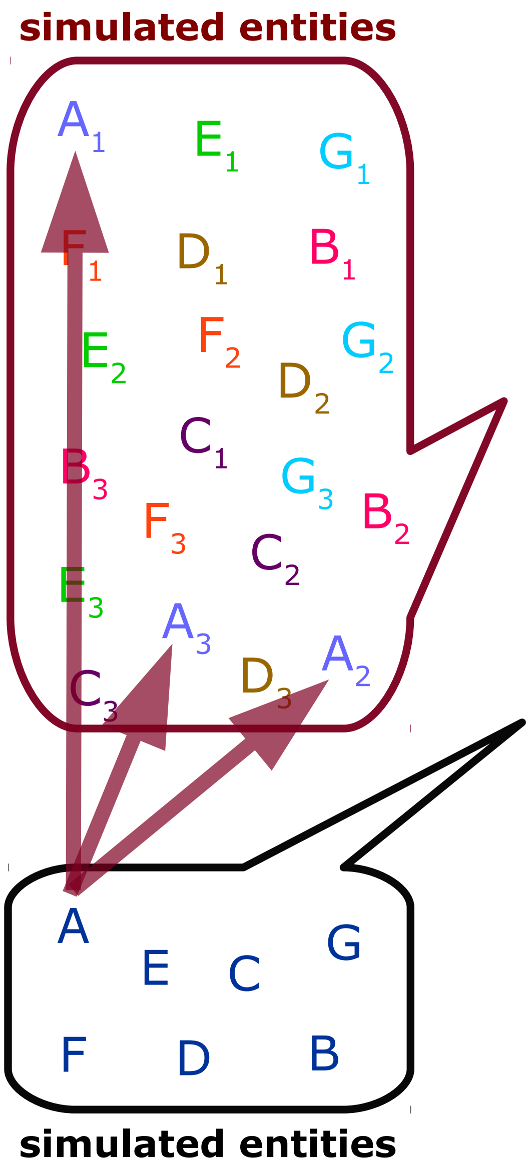
framework services:
- high level communication APIs
- migration support
- clustering heuristics

ARTÌS

runtime services:
- synchronization
- communication
- simulation management

operating system

FT-GAIA Software Architecture



simulation model

model behavior:

- state variables
- event handlers

FT-GAIA

fault-tolerance management:

- transparent functional replication
- handling of failures

GAIA

framework services:

- high level communication APIs
- migration support
- clustering heuristics

ARTÌS

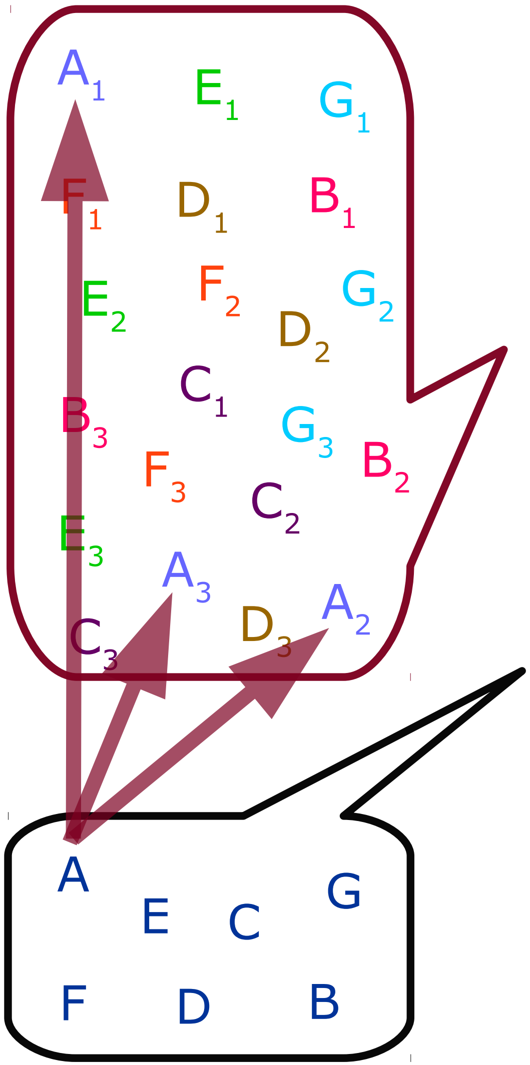
runtime services:

- synchronization
- communication
- simulation management

operating system

FT-GAIA Software Architecture

simulated entities



simulated entities

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

Fault Tolerance: Type of Failures

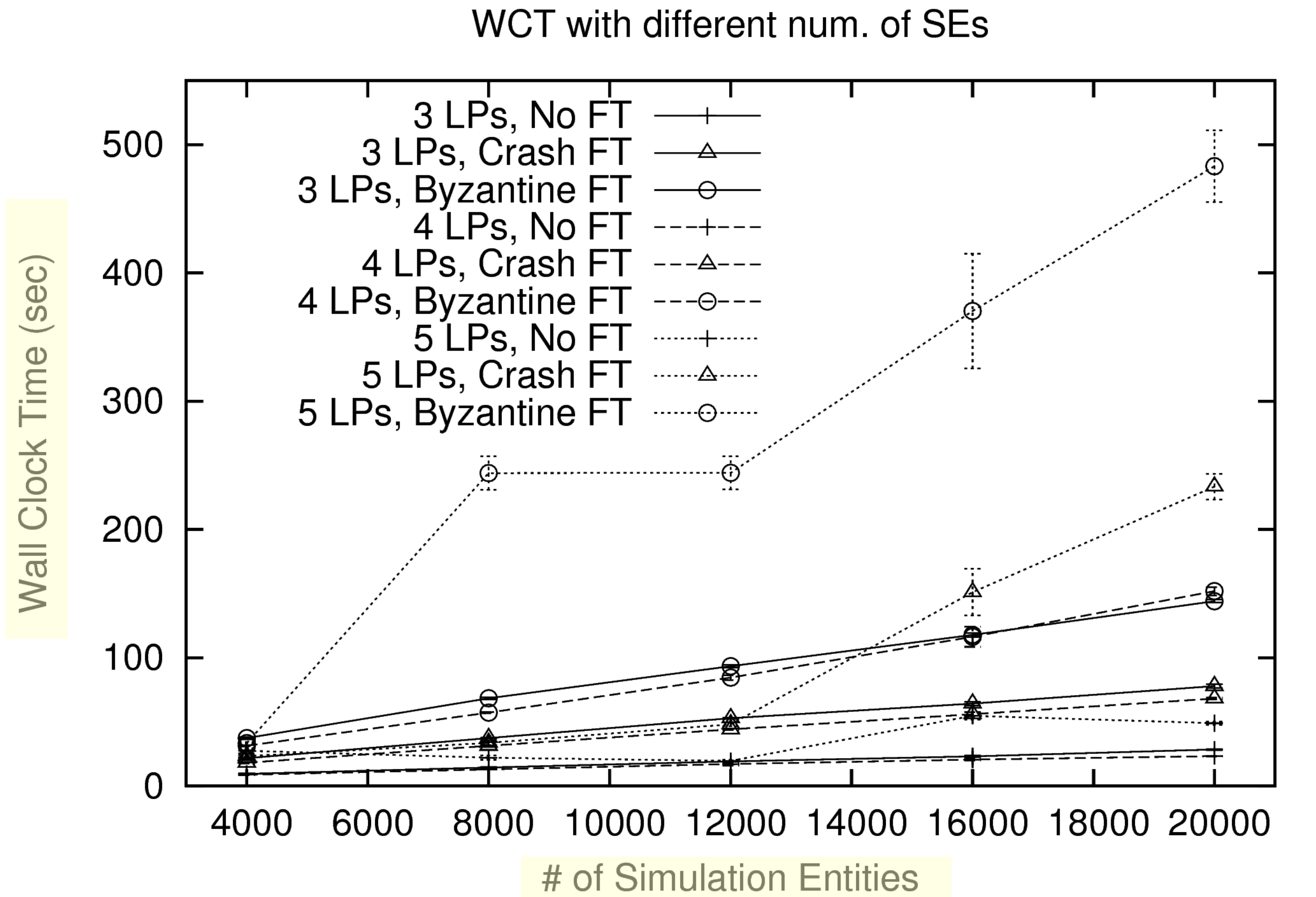
- **Crash failures:** a PE (e.g. CPU core) **halts but operated correctly** until it halted. The local state of all SEs on such PE is lost
  - *to tolerate **f** faults: **≥ f+1** instances of each SE (**M=f+1**)*
- **Byzantine Failures:** all types of **abnormal behavior** of a PE (*e.g transmission of erroneous/corrupted data, computation errors*)
  - *to tolerate **f** faults: **M=2f+1** replicas are needed (that is: "majority rule")*

Experimental Evaluation

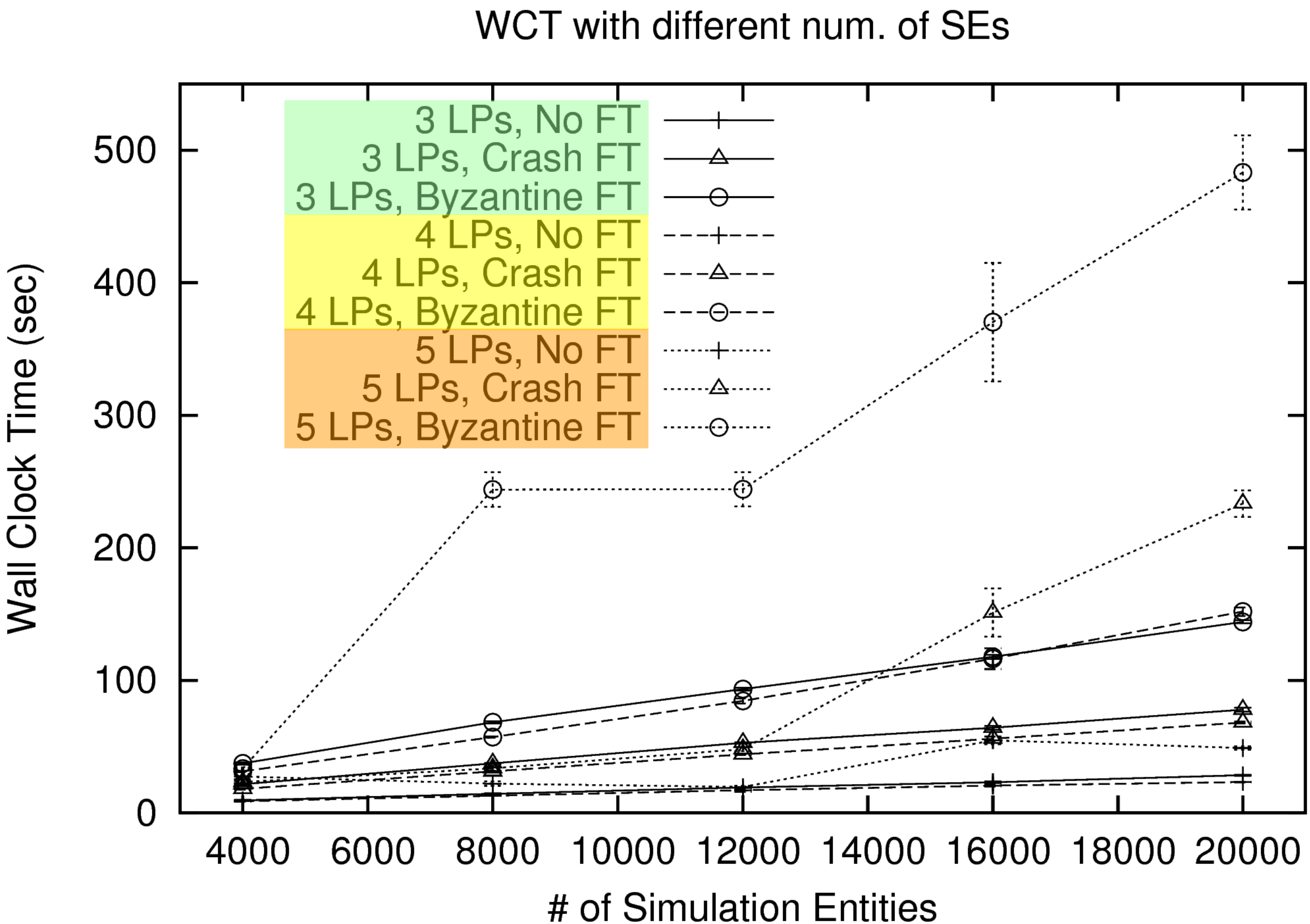
- **Simulation model**: simple P2P communication protocol (PING-PONG) over randomly generated directed overlay graphs
- **Execution platform**: cluster of Intel® Core© i5-4590 3.30 GHz CPU, 8 GB RAM, Debian Jessie, Fast Ethernet LAN
- **Methodology**: 15 independent replications of each simulation run. 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

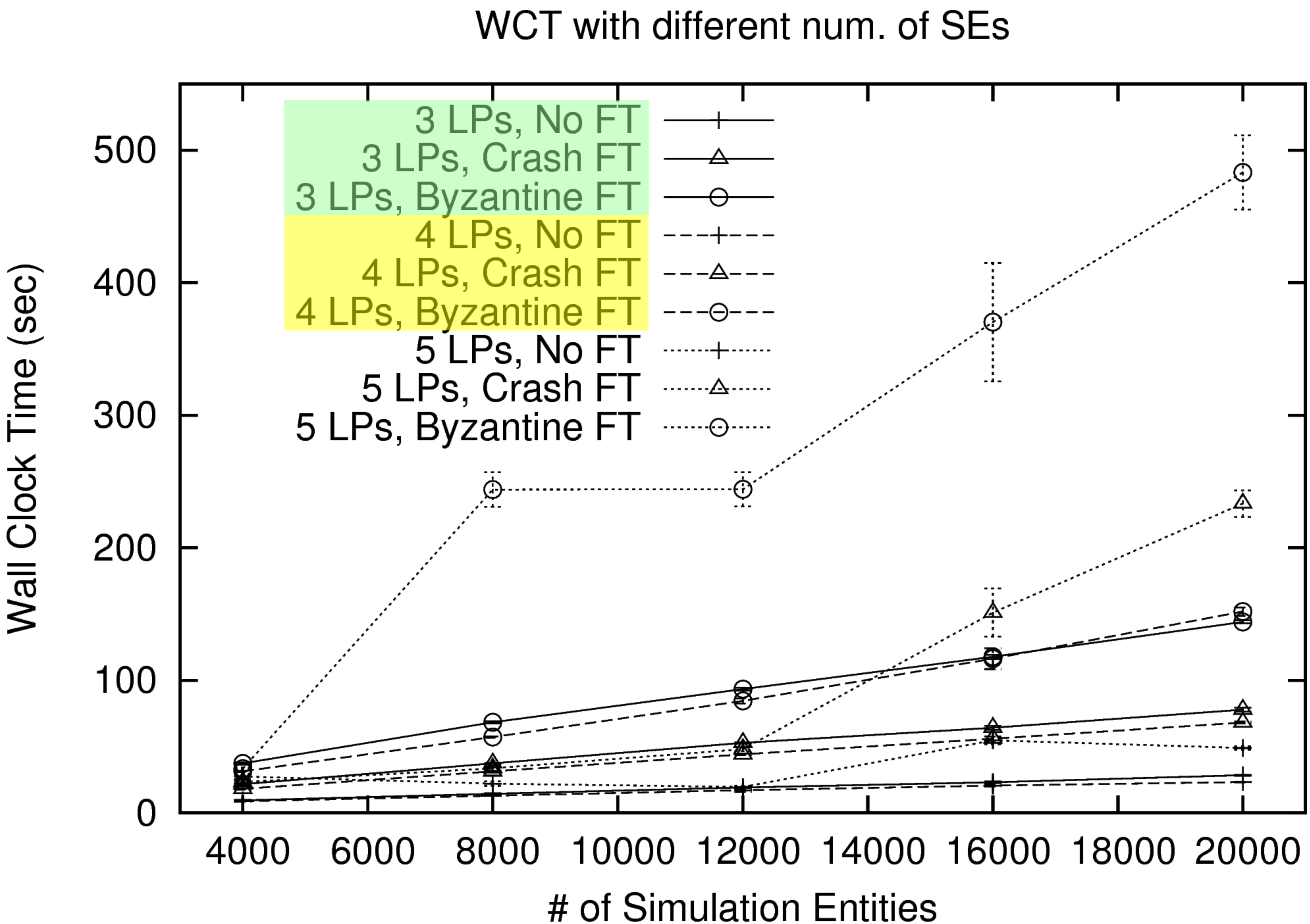
Impact of the number of LPs and SEs



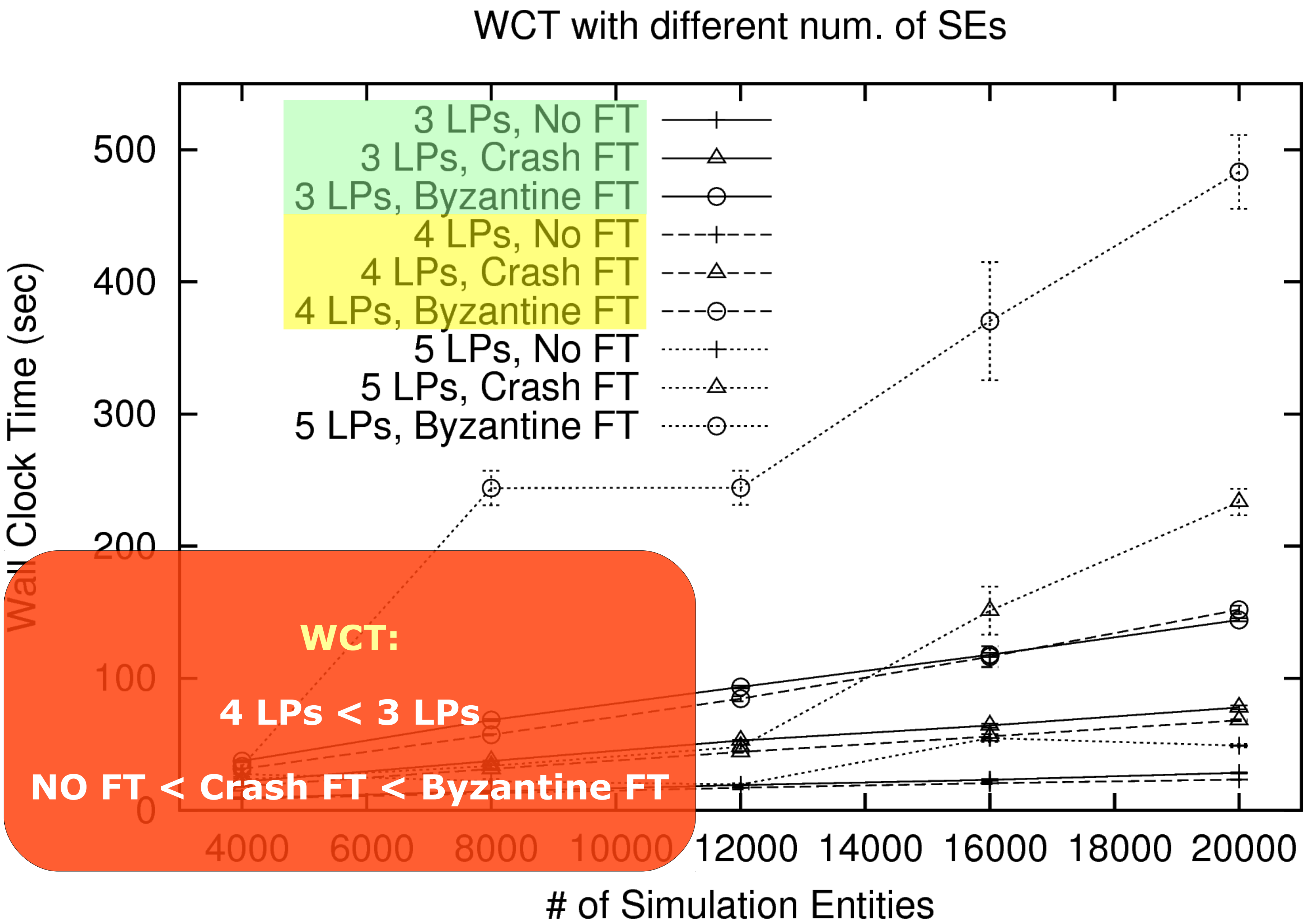
Impact of the number of LPs and SEs



Impact of the number of LPs and SEs



Impact of the number of LPs and SEs

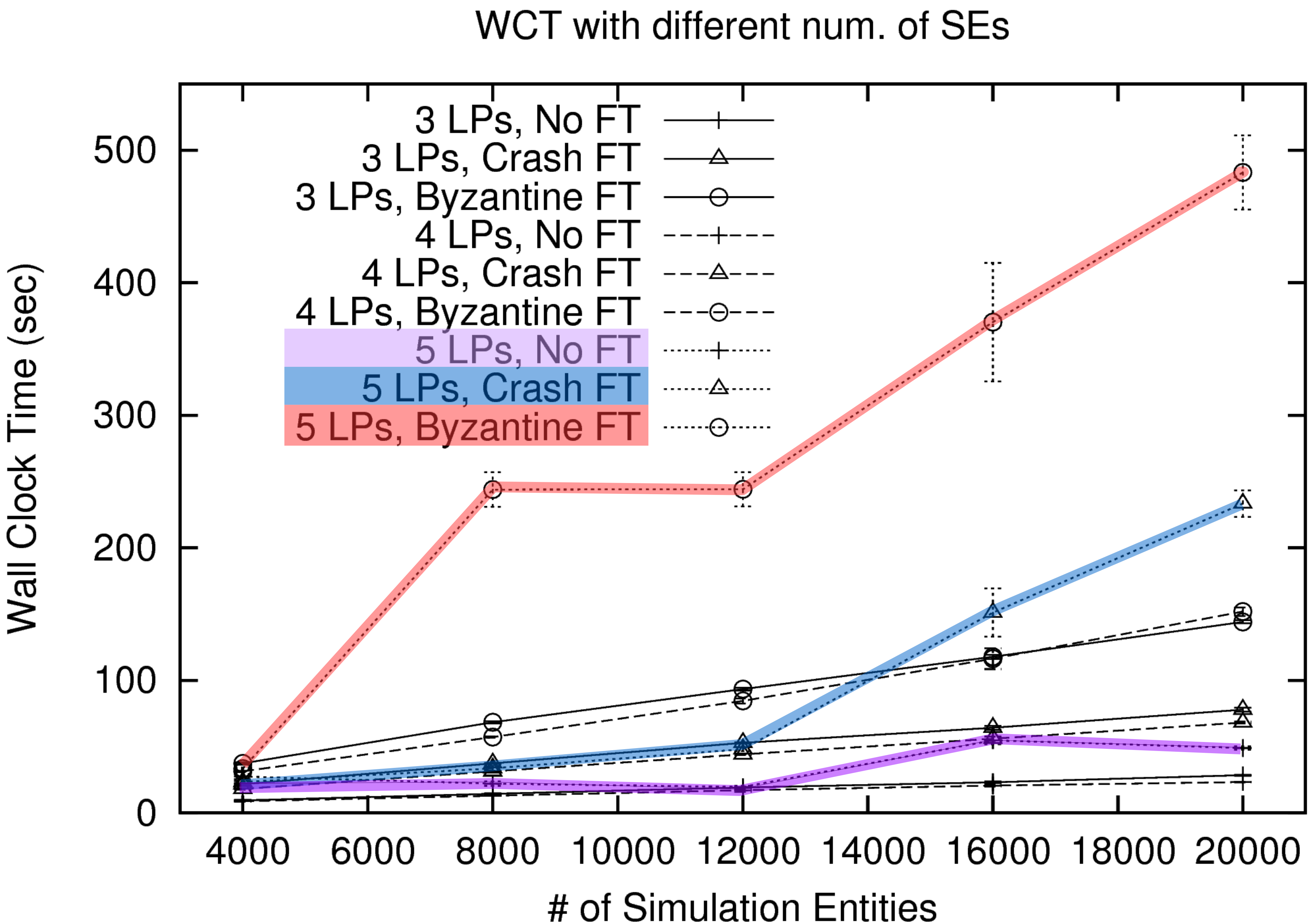


WCT:

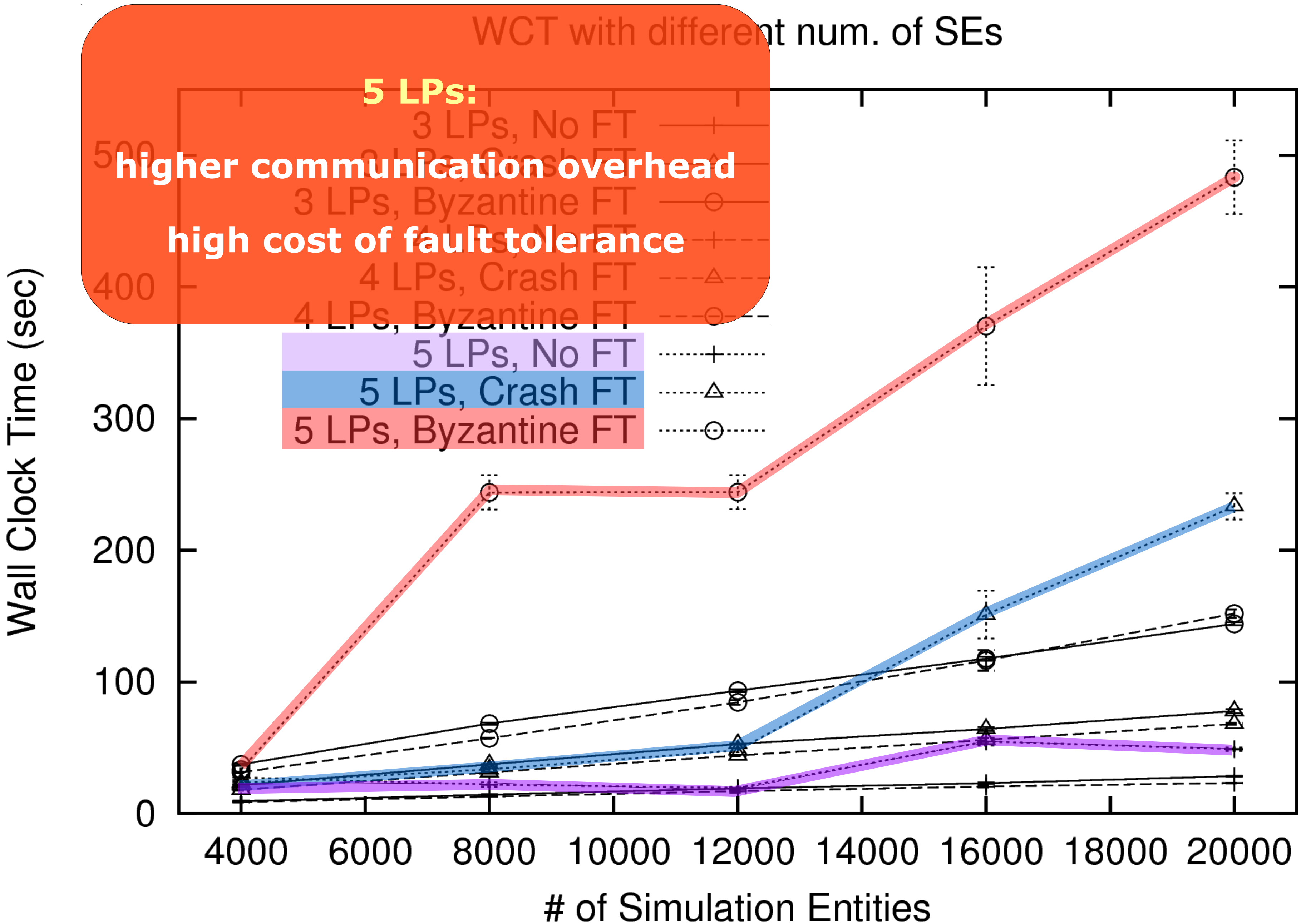
4 LPs < 3 LPs

NO FT < Crash FT < Byzantine FT

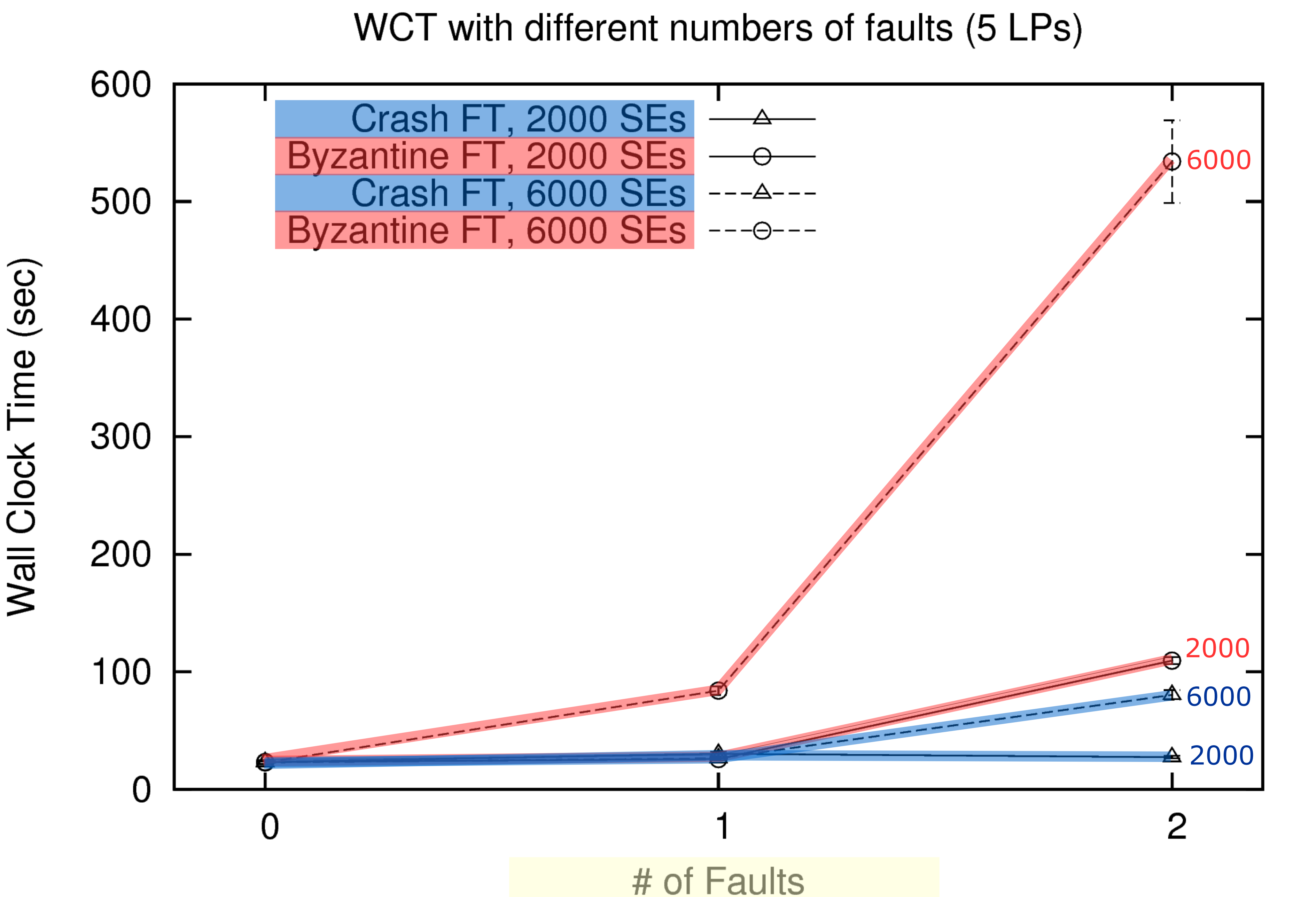
Impact of the number of LPs and SEs



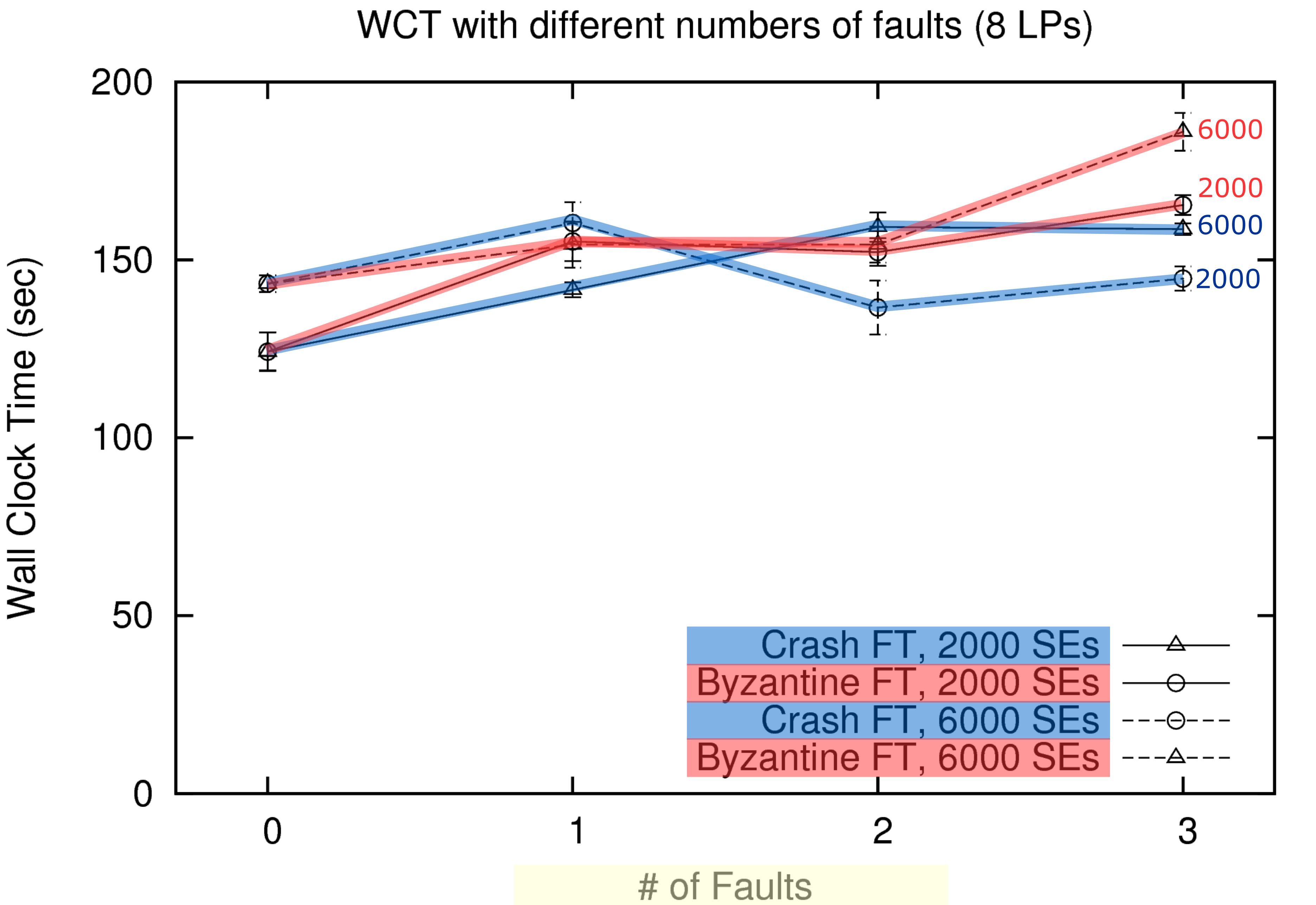
Impact of the number of LPs and SEs



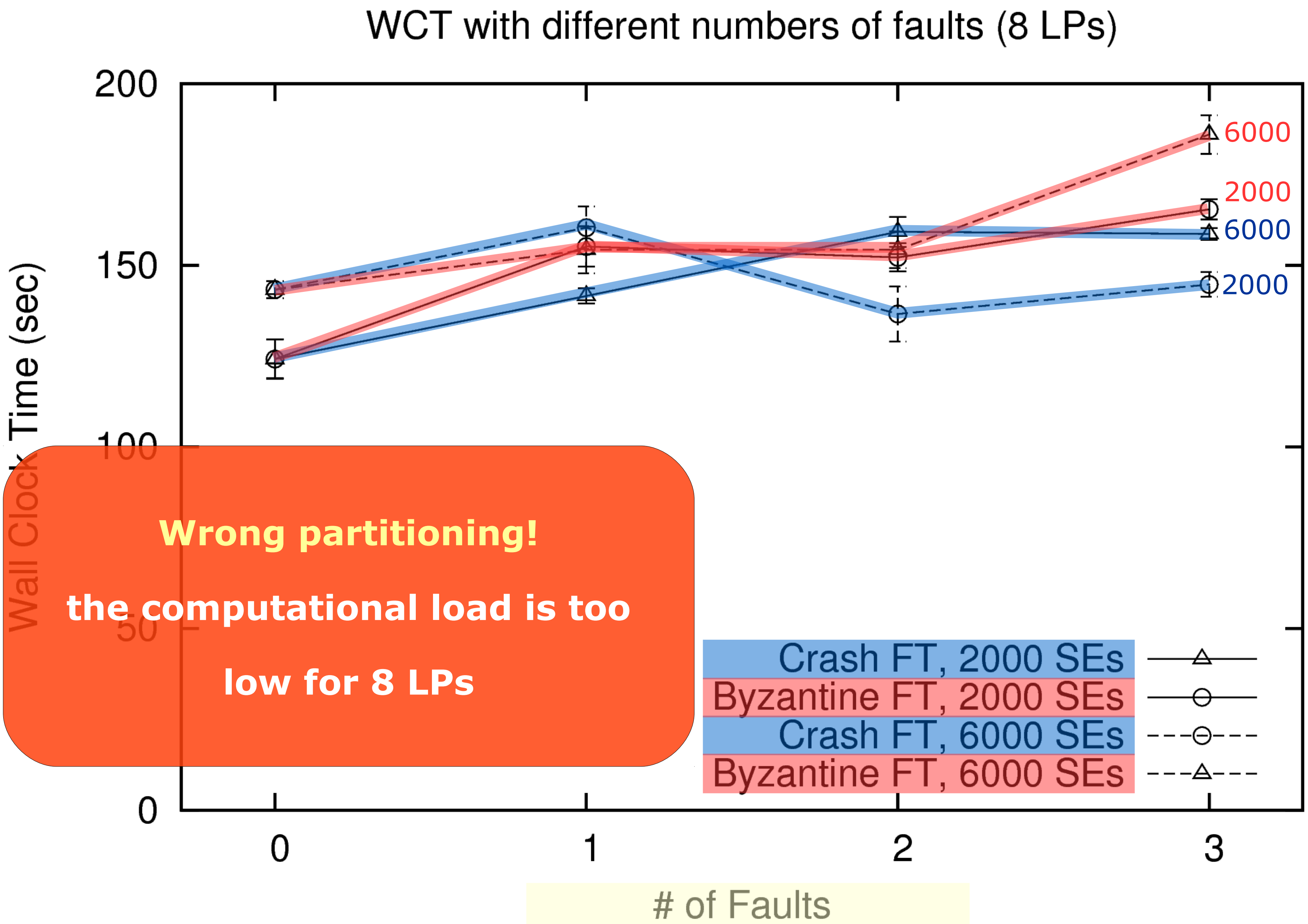
Impact of the number of failures (5 LPs)



Impact of the number of failures (8 LPs)



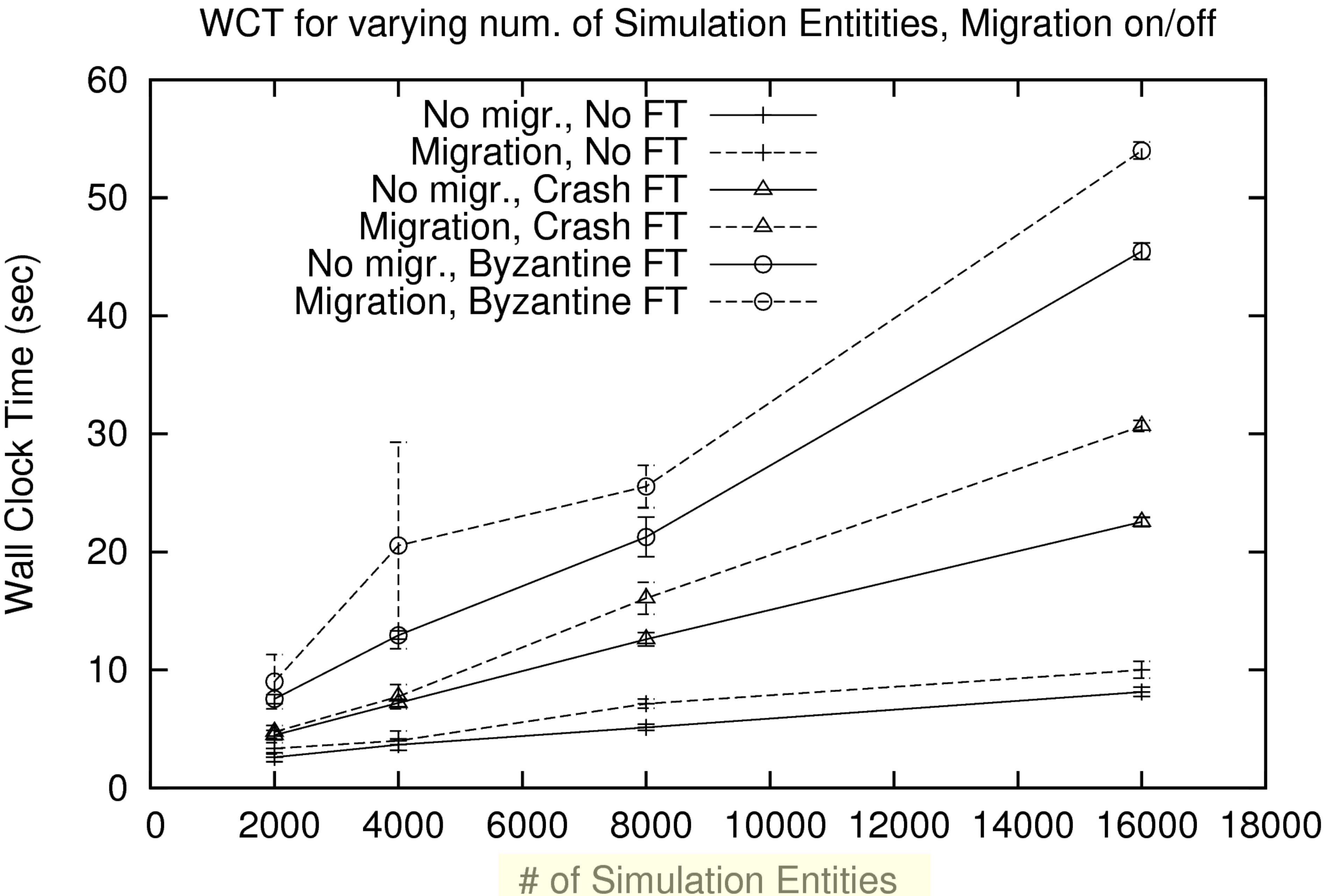
Impact of the number of failures (8 LPs)



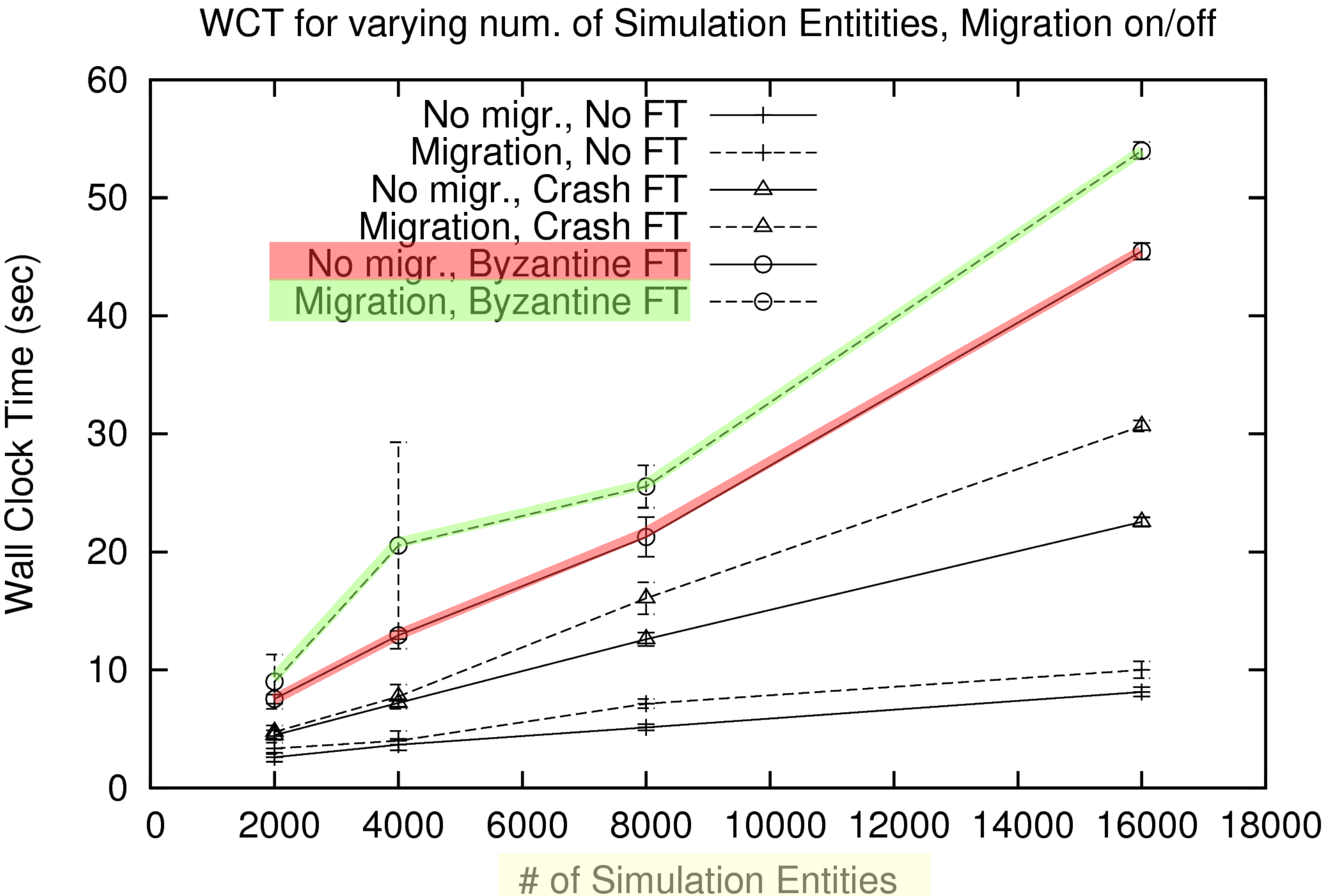
Wrong partitioning!

the computational load is too low for 8 LPs

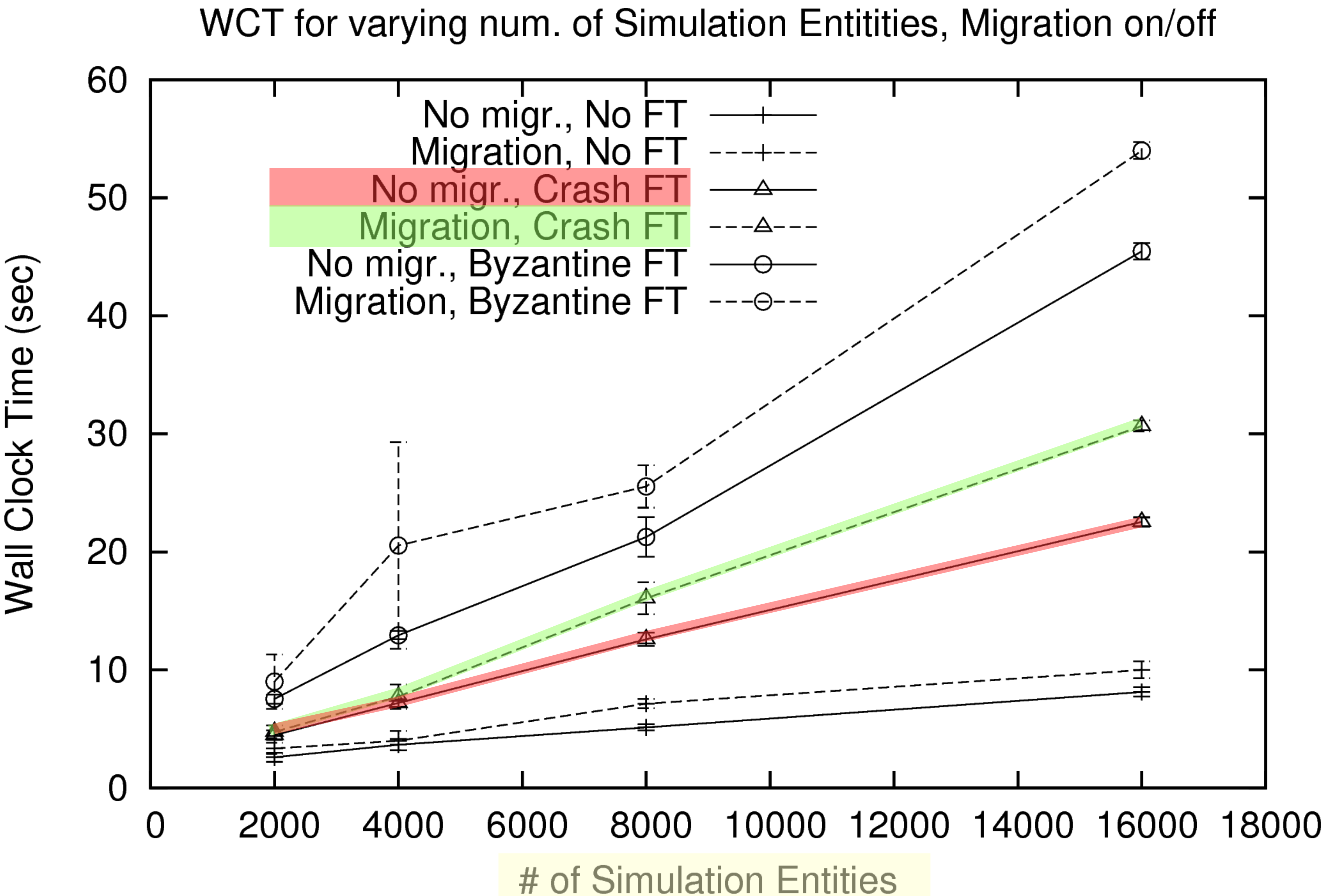
Impact of SEs migration



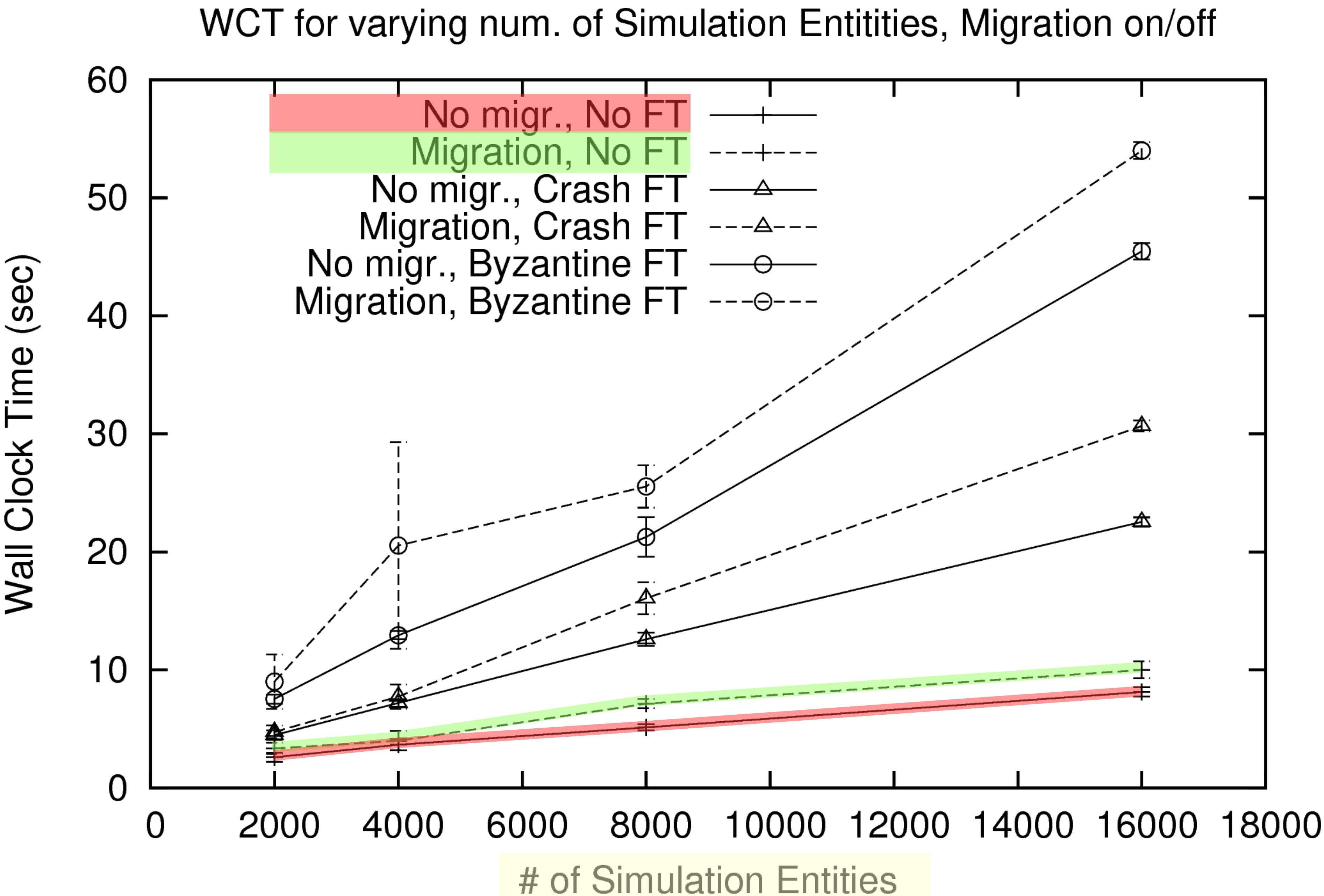
Impact of SEs migration



Impact of SEs migration



Impact of SEs migration



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

Proceedings of the 20-th ACM/IEEE International Symposium on Distributed Simulation and Real Time Applications (DS-RT 2016). London, England, 2016

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 **S**imulation and **R**eal **T**ime Applications (**DS-RT**), 2016