Concurrent Replication of Parallel and Distributed Simulations

Gabriele D'Angelo

joint work with

Luciano Bononi
Michele Bracuto
Lorenzo Donatiello

University of Bologna
Department of Computer Science

PADS 2005 – Monterey (CA)

Presentation outline

- Goal and basic assumptions
- State of the art
- The Concurrent Replication of PADS (CR-PADS) mechanism
- The ARTìS middleware
- The CR-PADS logical structure and implementation
- Performance evaluation: Ad Hoc network model
    - Parallel and distributed environments
- Conclusions and future work

Goal and basic assumptions

- Goal: to **increase the simulation speed**, reduce the **Wall Clock Time (WCT)** required to complete the simulation runs

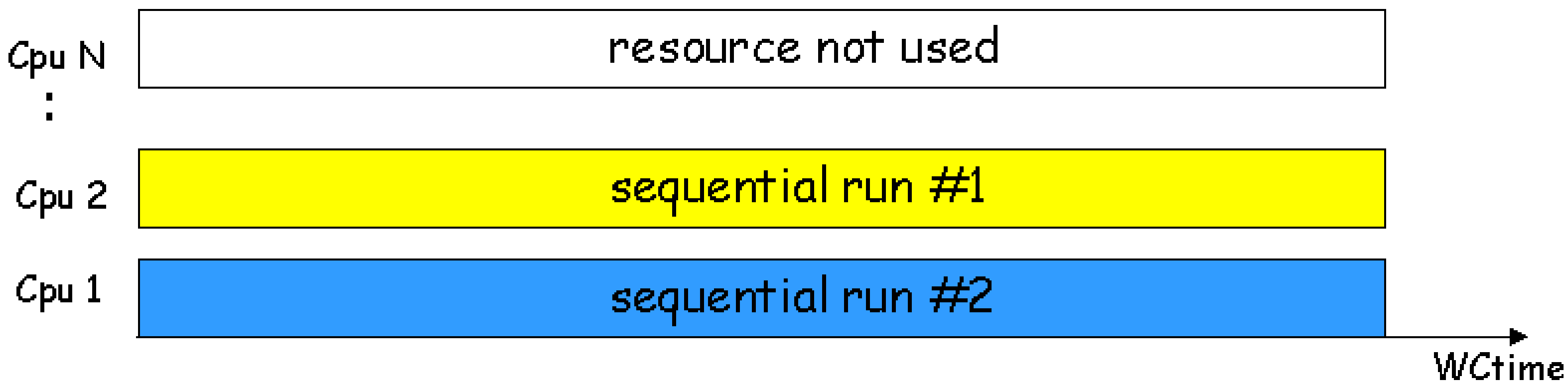
- We focus on Parallel and Distributed Simulations based on conservative synchronization algorithms (time-stepped, CMB)

- A typical simulation-based investigation requires to collect **many independent observations** for a correct and significant statistical analysis of results

- And so we are interested in the WCT necessary to complete **all** the simulations runs (not a single run)

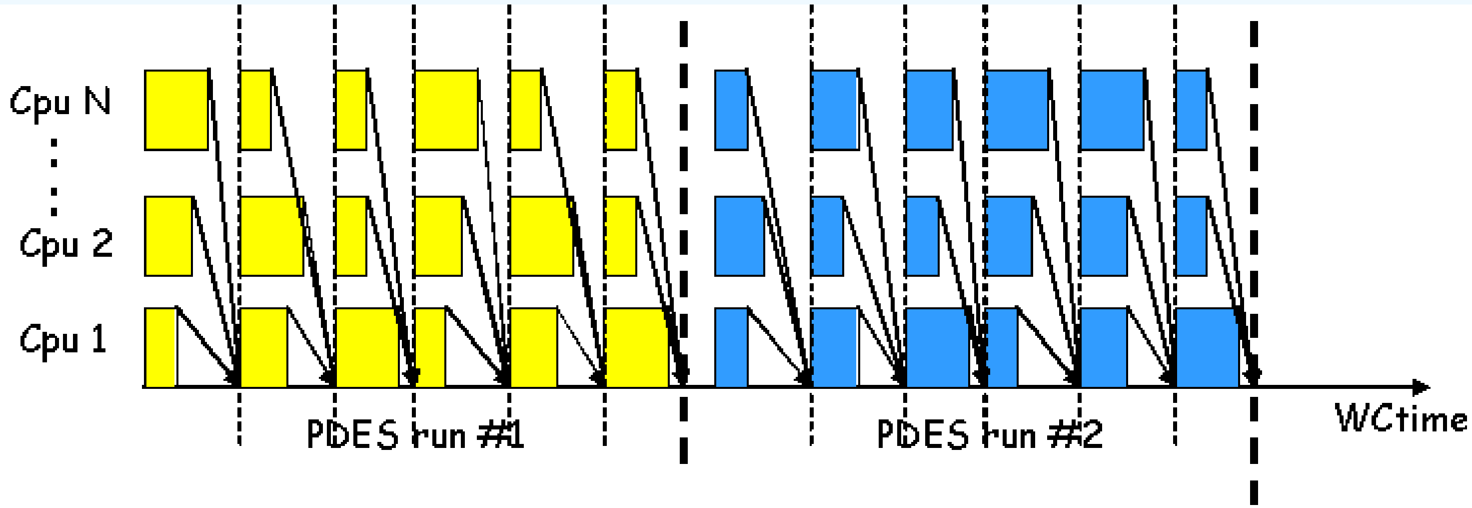
MRIP: Multiple Replications in Parallel

- MRIP approach: launching **multiple runs** of independent sequential simulations in parallel over a set of concurrent CPUs
- Every simulation run is executed from the beginning up to the end on the same CPU (**monolithic sequential simulation**)
- Some computational resources may remain **unexploited**



Legacy PDES approach

- The whole simulation is composed by the execution of a **linear sequence of multiple parallel or distributed simulation run**
- The standard Parallel or Distributed event simulation (PDES) approach may introduce advantages: every independent run could exploit the whole computation architecture
- A single run may complete in less time than a sequential run

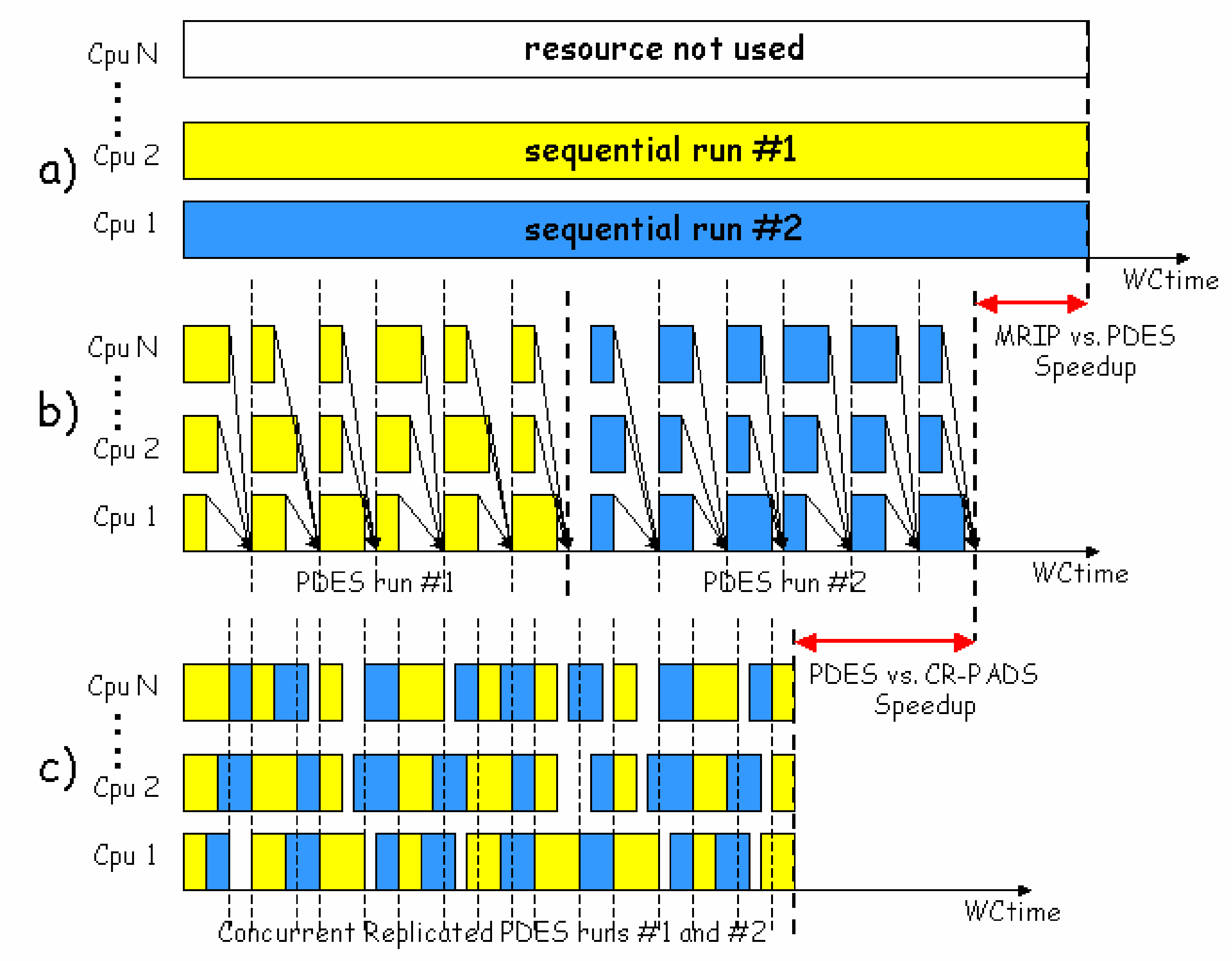


Typical PDES problems

- **Frequent synchronizations** are required among the model components
- Each simulation component swings between **computation and communication phases**
- The whole set of processes advance with the **speed of the slowest**
- Some phases (usually before the synchronization barriers) could be communication intensive and may led to network congestions (further increasing the communication overhead)

Our idea is to obtain a more **fluent computation and communication** by concurrently merging the execution of more PADS replicas

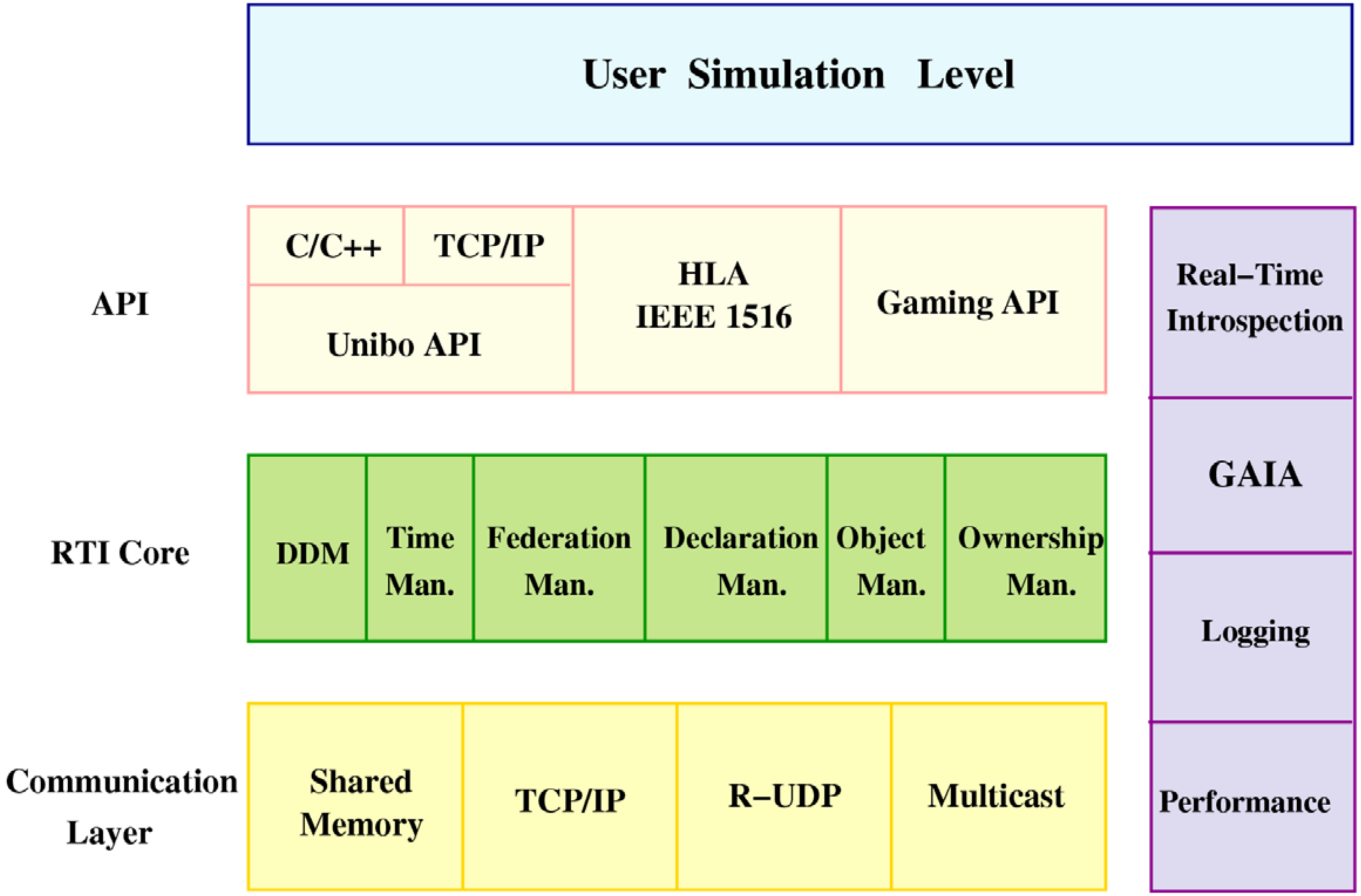
The CR-PADS approach



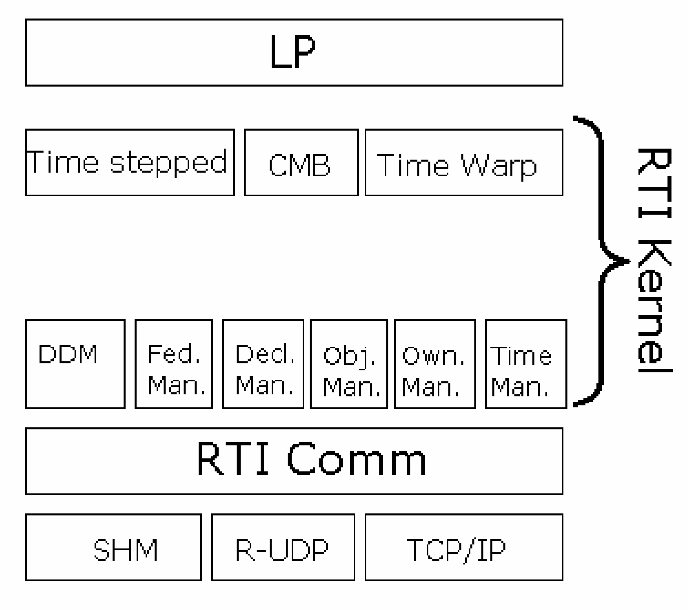
Concurrent Replication of PADS (CR-PADS)

- The CR-PADS is a mechanism that **duplicates** the logical processes (LPs) of PADS runs starting from the **initialization phase** of every single run
- Every replica is based on the same model definition, realizes an **independent execution** based on local initial parameters, variable factors of the analysis and different random number generator seeds
- CR-PADS is absolutely **different** from **simulation cloning**
- The risk of this mechanism is to spend too much time in switching processes' execution, in the creation of communication bottlenecks and live-locks, resulting in trashing effects

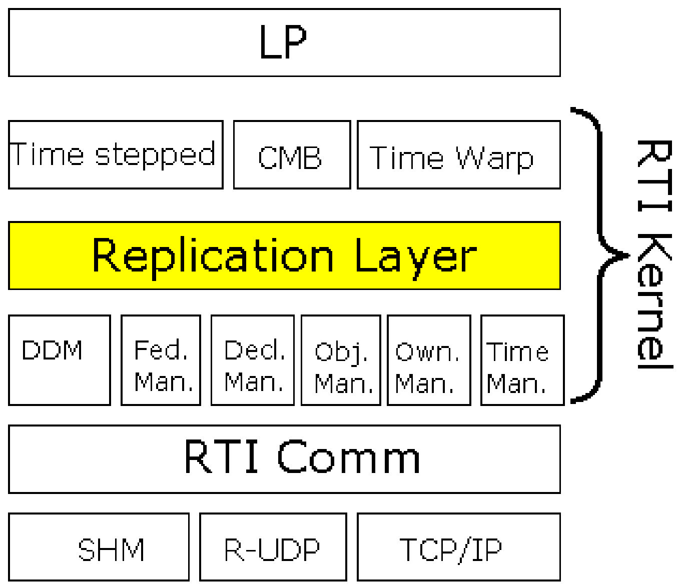
ARTìS: Advanced RTI System



The ARTìS and Replication architecture



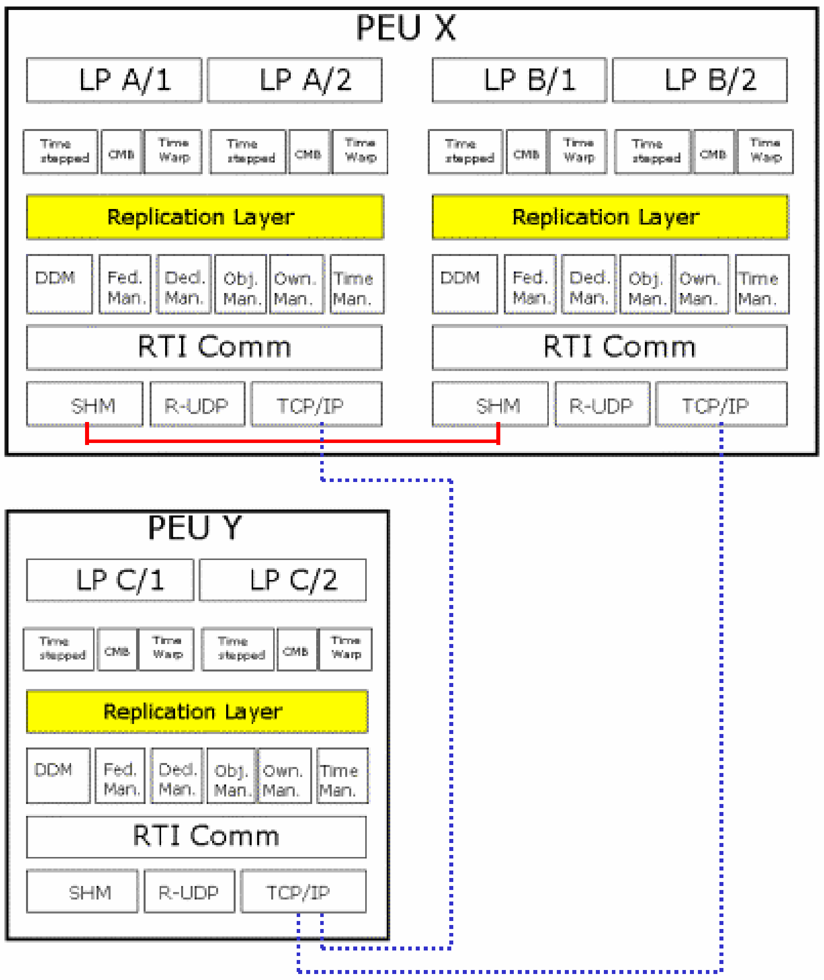
LP standard logical structure



LP CR-PADS logical structure

- The Replication layer is the **funnel** for replicas over the RTI Communication layer. It is required to maintain transparency and to optimize the communication performances

Parallel and Distributed CR-PADS architecture



- **PEU = Physical Execution Unit**
- A set of LPs on the same PEUs communicate via low latency Shared Memory (SHM)
- Two or more LPs located on different hosts (i.e. no shared memory available) rely on standard TCP/IP connections
- The whole structure is based on a set of inter-communicating threads

Performance evaluation: Ad-Hoc wireless network model

Simulated model:

- A set of **Simulated Mobile Hosts** (SMHs)
- Mobility model:
  - Random Mobility Motion model (RMM)
  - uncorrelated SMHs' mobility
- Traffic model:
  - ping messages (CBR) by every SMH to all neighbors within the wireless communication range (250 m)
- Propagation model
  - open space (neighbor-SMHs within detection range)

Ad-Hoc network model characterization

Computation and communication issues:

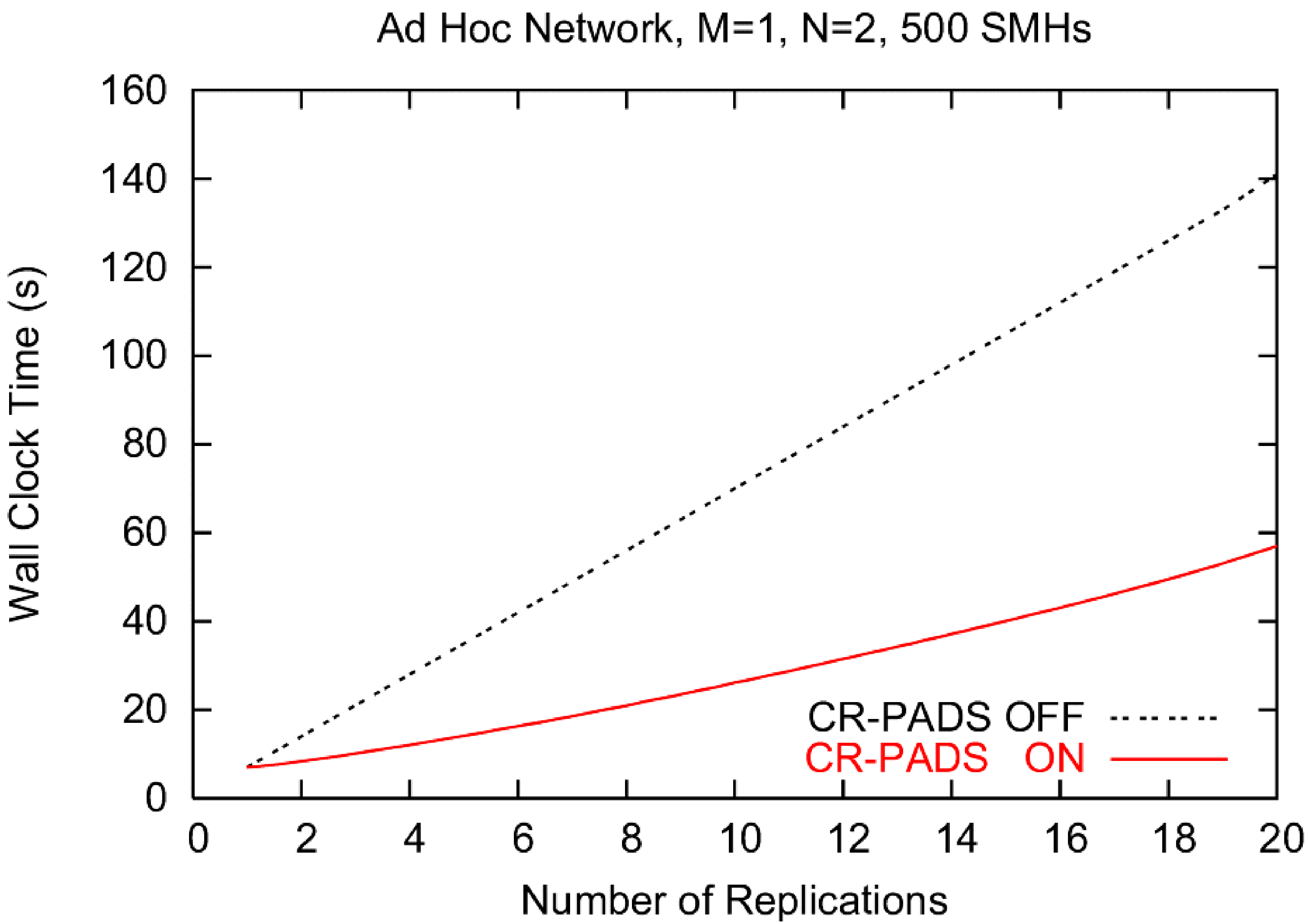
- The **computation** required for each SMH per time-step is in the order of **O(#SMH^2)**: to determine the neighbor set
- The communication required among SMHs is in the order of **O(K*#SMH)** per time-step, with K defined as a constant value based on SMHs density (assumed as constant)

Simulation test-beds

Two different environments:

- **Parallel**: a single Physical Execution Unit (PEU)
- **Distributed**: a set of PEU, interconnected by a Fast Ethernet LAN (100 Mb/sec)
- Each PEU is an Intel Dual Xeon Pentium IV 2800 MHz, with 3 GB RAM, Debian GNU/Linux OS with kernel version 2.6.x
- Conservative time-stepped simulation: 300 time-steps

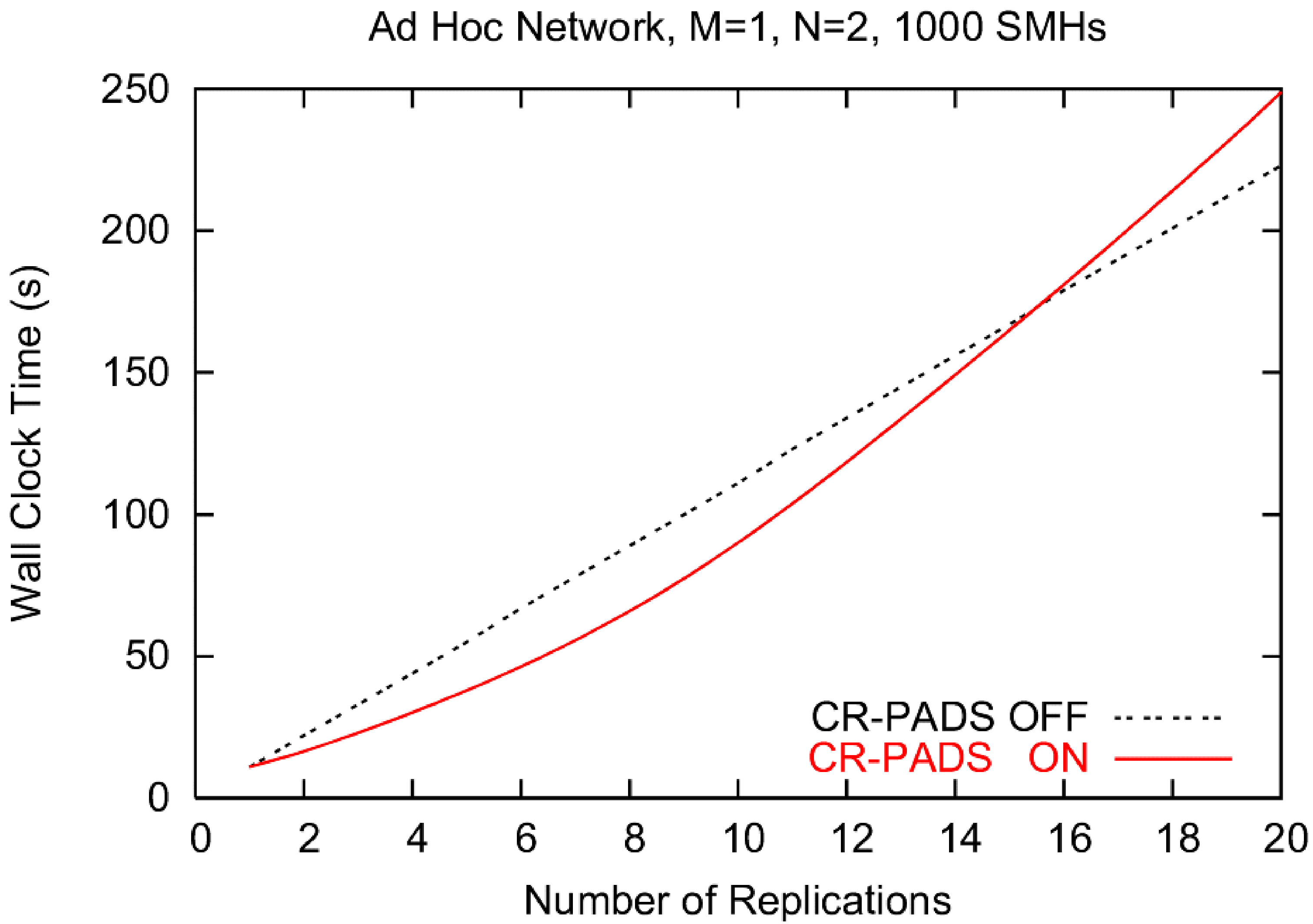
Parallel environment: 500 SMHs



M = # PEU

N = # LP

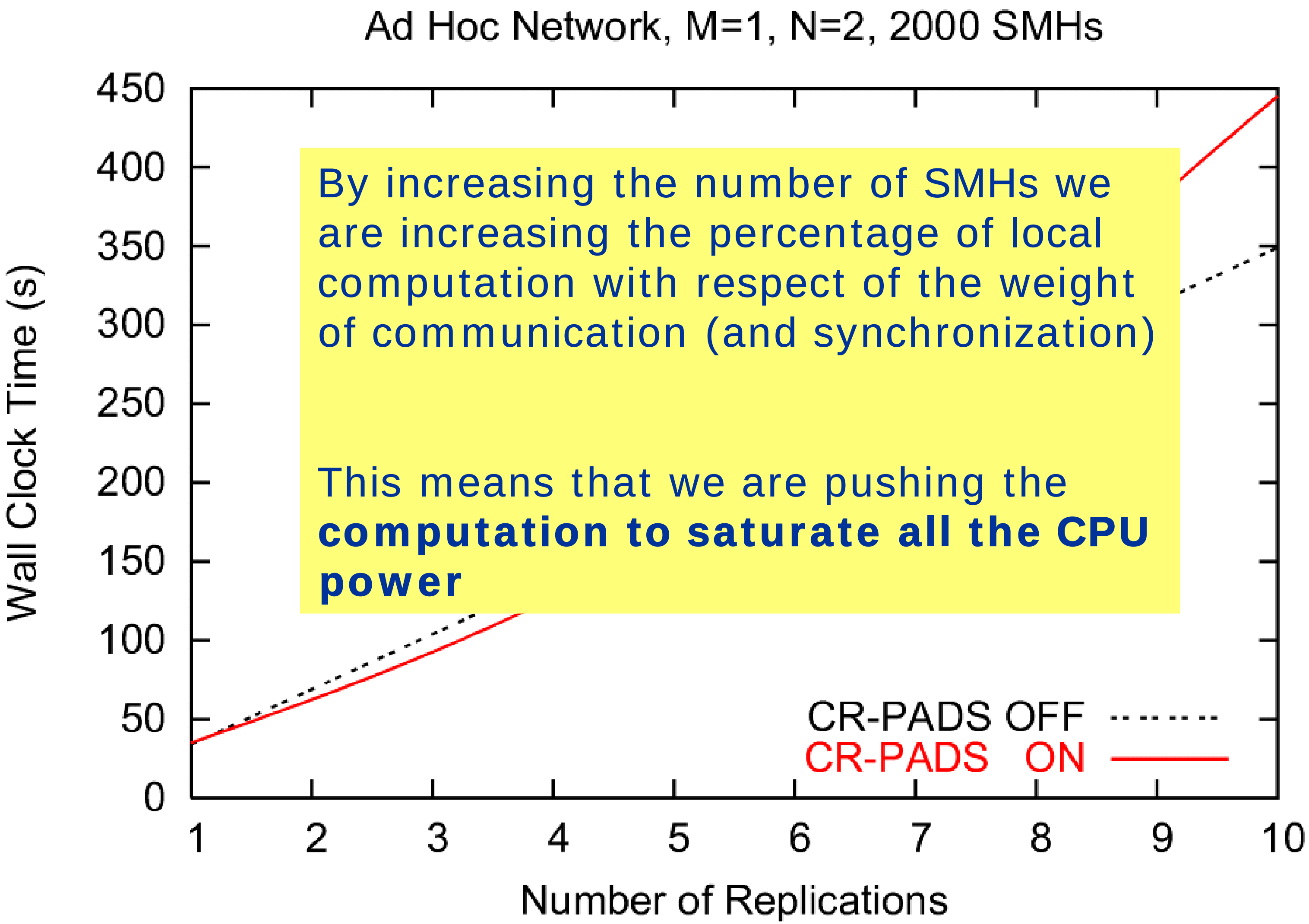
Parallel environment: 1000 SMHs



M = #PEU

N = #LP

Parallel environment: 2000 SMHs



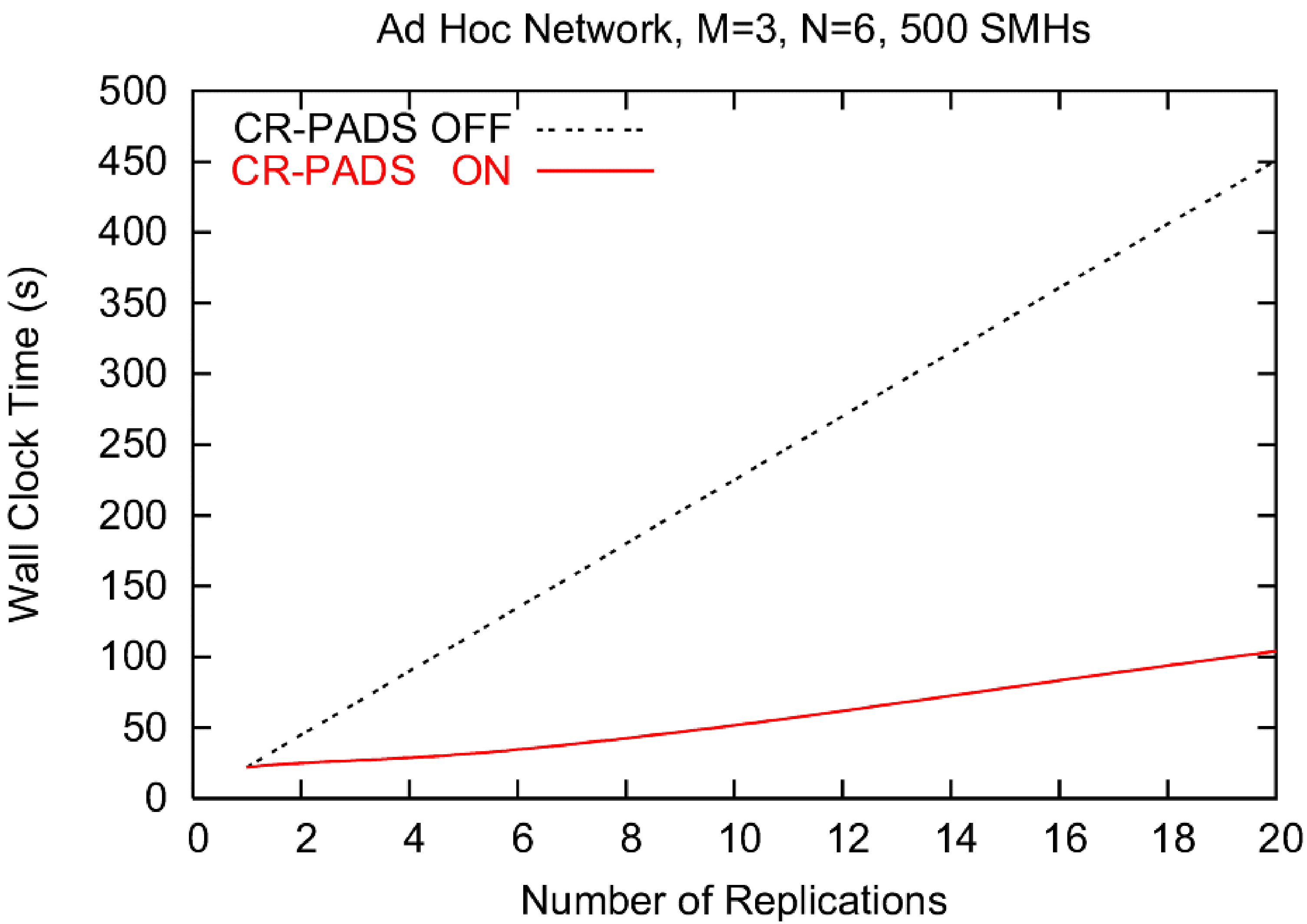
By increasing the number of SMHs we are increasing the percentage of local computation with respect of the weight of communication (and synchronization)

This means that we are pushing the **computation to saturate all the CPU power**

M=#PEU

N=#LP

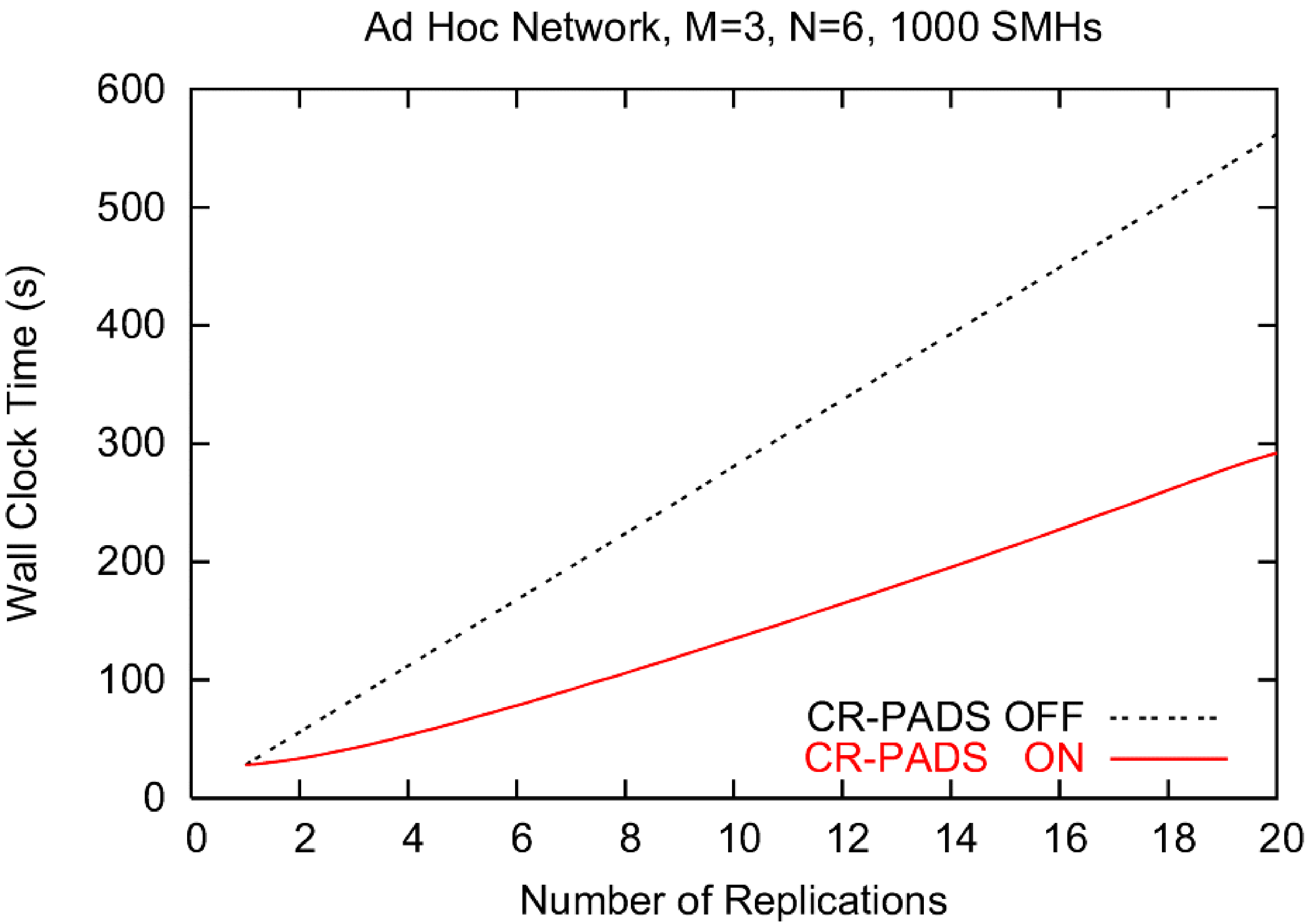
Distributed environment: 500 SMHs



M = #PEU

N = #LP

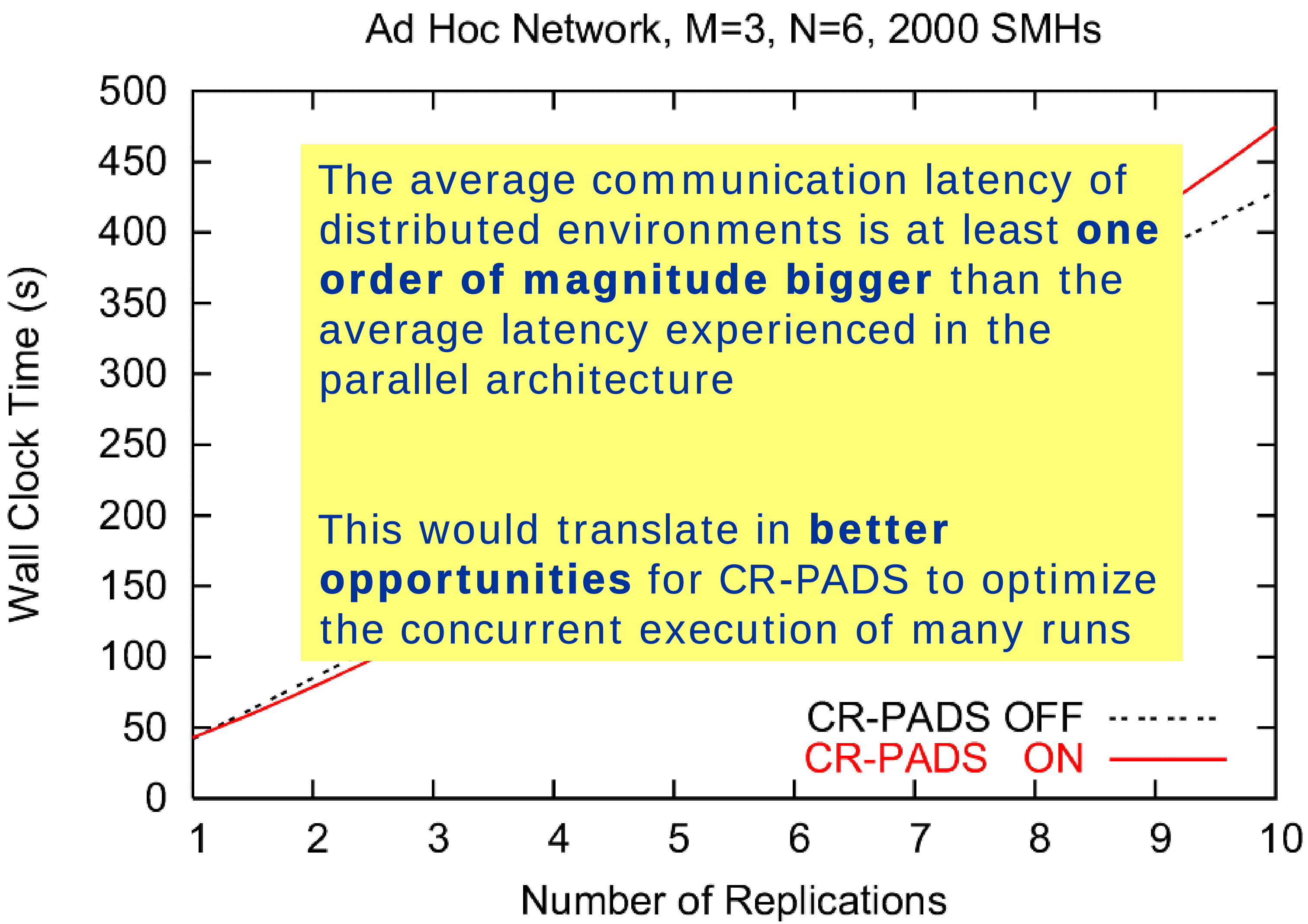
Distributed environment: 1000 SMHs



M = #PEU

N = #LP

Distributed environment: 2000 SMHs



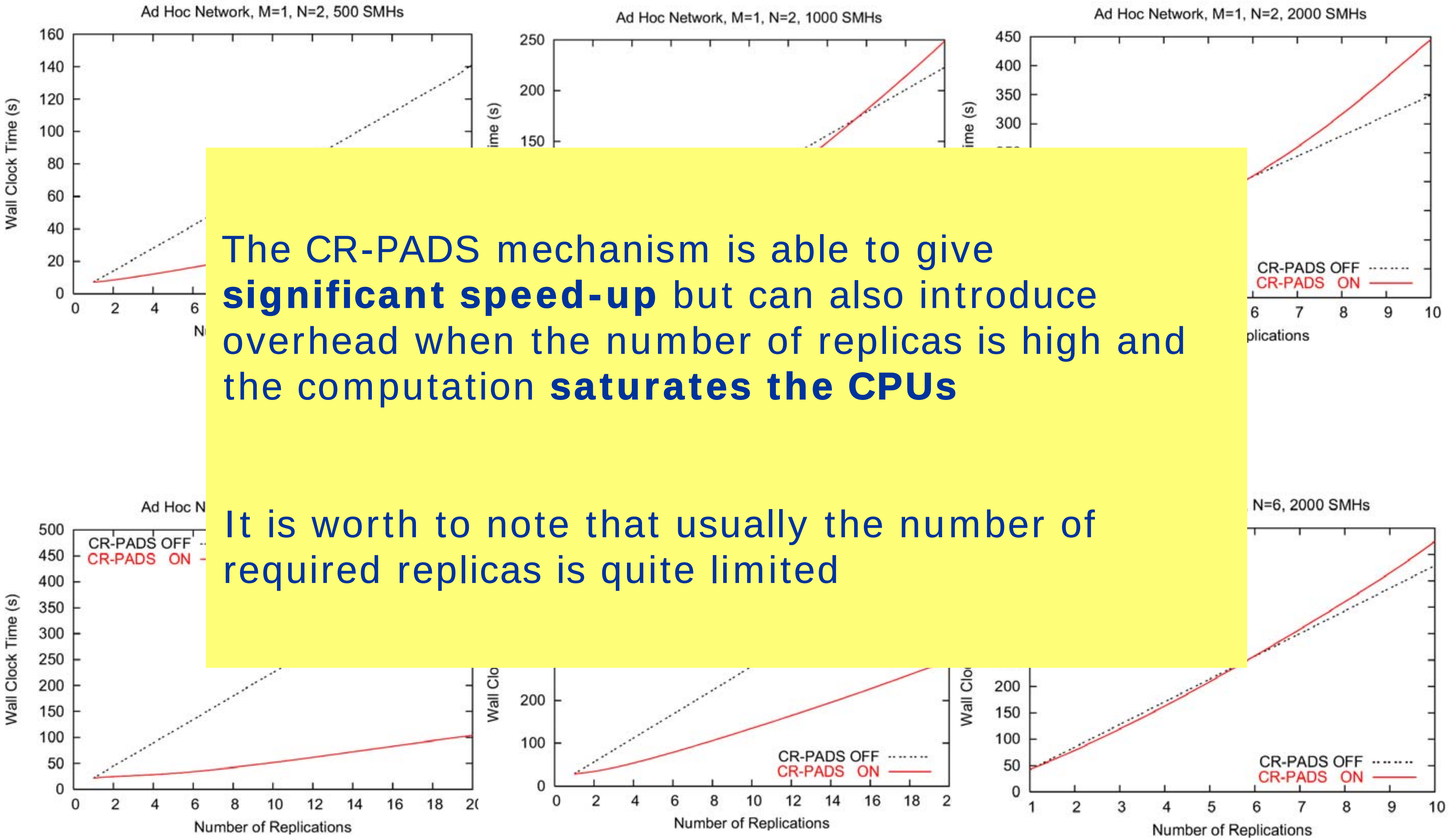
The average communication latency of distributed environments is at least **one order of magnitude bigger** than the average latency experienced in the parallel architecture

This would translate in **better opportunities** for CR-PADS to optimize the concurrent execution of many runs

M=#PEU

N=#LP

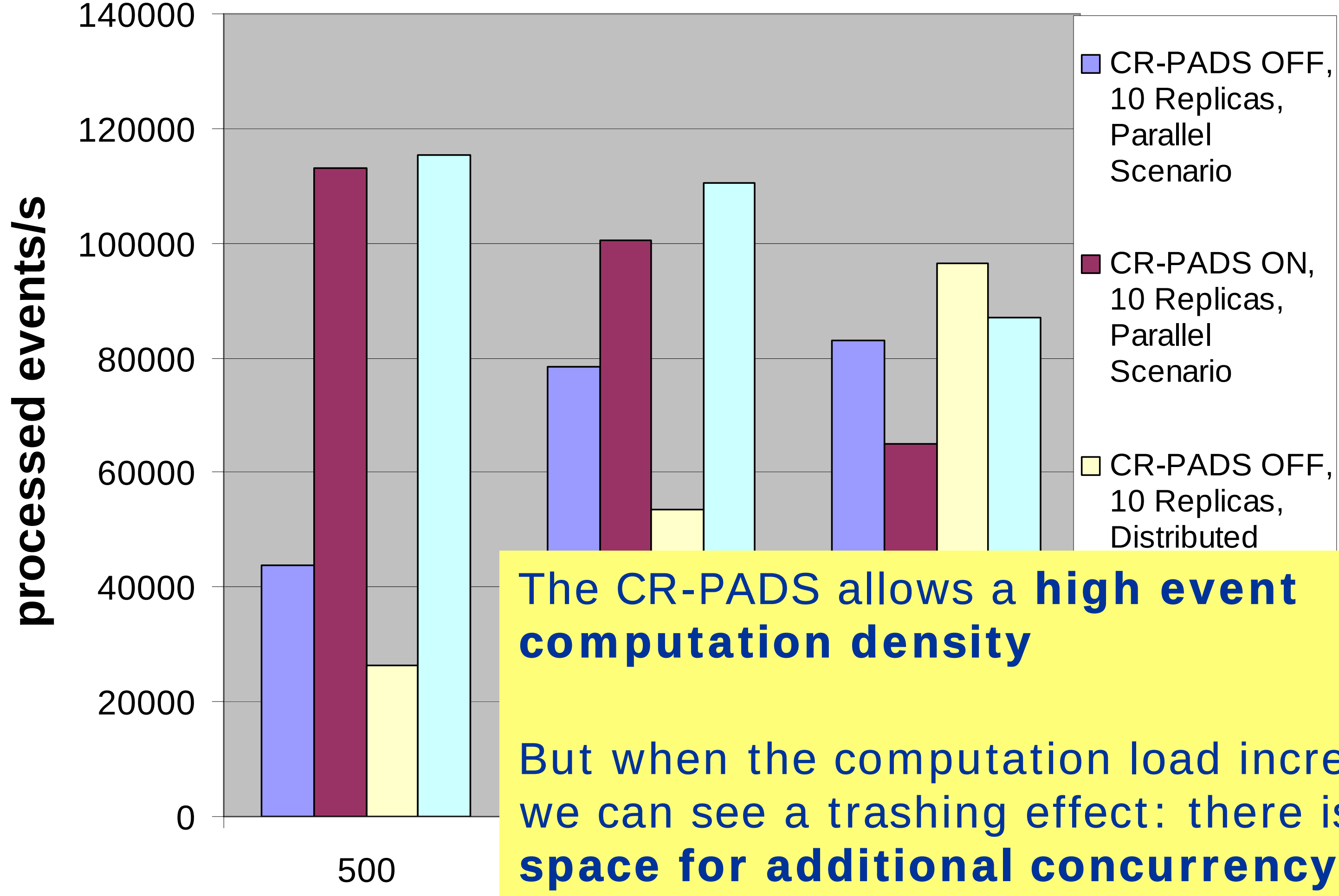
The overall picture



The CR-PADS mechanism is able to give **significant speed-up** but can also introduce overhead when the number of replicas is high and the computation **saturates the CPUs**

It is worth to note that usually the number of required replicas is quite limited

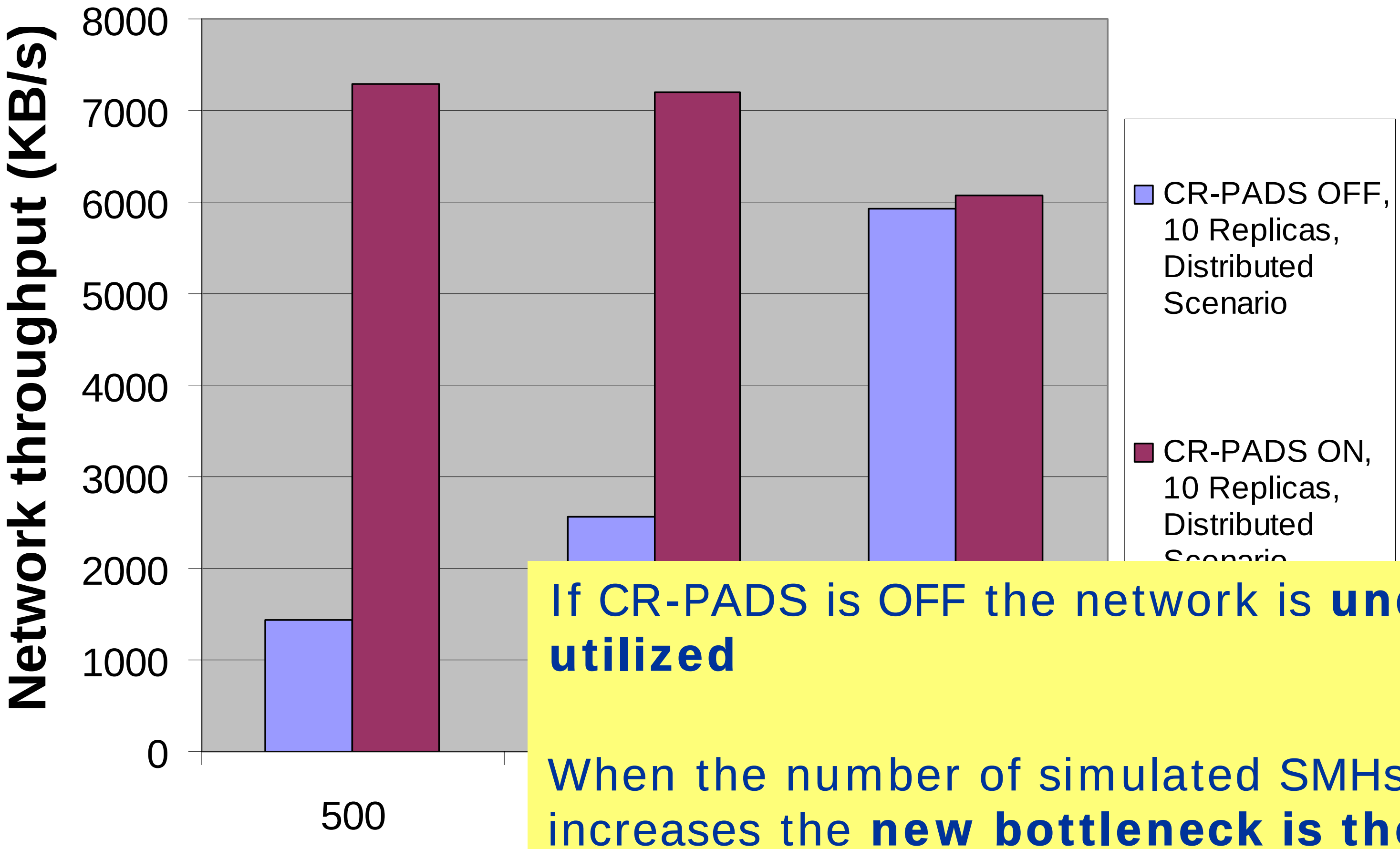
Performance analysis: processed events/sec



The CR-PADS allows a **high event computation density**

But when the computation load increases we can see a trashing effect: there is **not space for additional concurrency** in the computation

Performance analysis: network throughput (KB/sec)



If CR-PADS is OFF the network is **under-utilized**

When the number of simulated SMHs increases the **new bottleneck is the computation** and not the communication

Conclusions: Concurrent Replication of PADS

- Under the Parallel (or Distributed) Discrete Event Simulation (PDES) frequent **synchronizations** are required among the model components
- Every **CPU** swings between computation and **idle**, while the underlying **communication infrastructure** swings between idle and communication periods
- A typical simulation-based investigation requires to collect many independent observations for a correct and significant statistical analysis of results
- **Proposed solution**: launching multiple, independent and concurrent PDES runs over the system, with the goal to reduce idle computation and communication time

Future work

- Further extend the ARTìS middleware
- Improve the CR-PADS implementation
- Test the performances of the CR-PADS mechanism on CMB-synchronized models
- Integrate the CR-PADS with our migration based framework (**GAIA, Generic Adaptive Interaction Architecture**) to improve *load-balancing* and *dynamic model partitioning* (**PADS 2003**)
- Apply the concurrent replication and simulation cloning mechanisms to High Performance Computing (HPC) environments

Concurrent Replication of Parallel and Distributed Simulations

Gabriele D’Angelo

gdangelo@cs.unibo.it

http://www.cs.unibo.it/~gdangelo

University of Bologna

Department of Computer Science

PADS 2005, Monterey (CA)