

Proximity Detection in Distributed Simulation of Wireless Mobile Systems



Gabriele D'Angelo

Lorenzo Donatiello

Luciano Bononi

Michele Bracuto

**Department of Computer Science
University of Bologna**

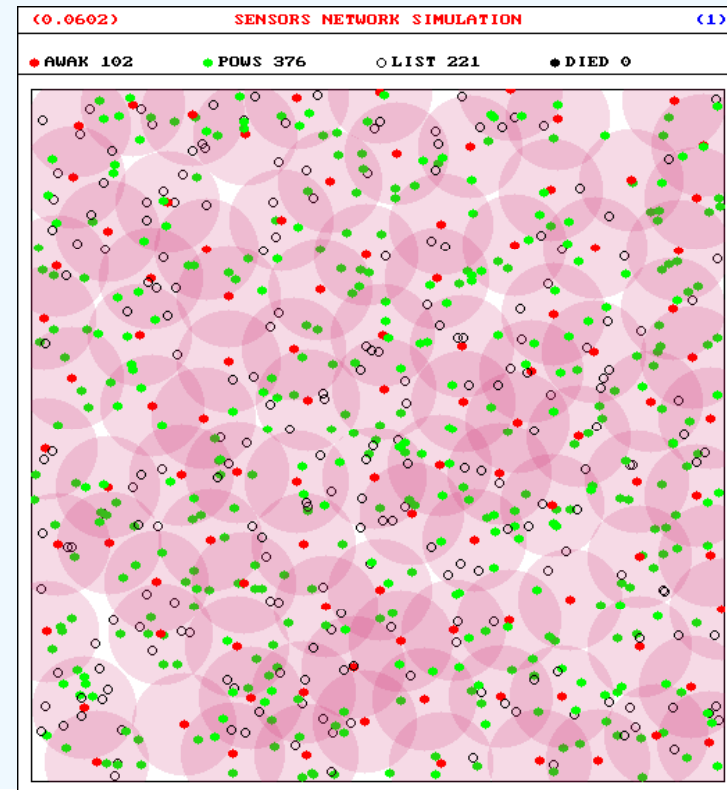
Torremolinos, 03/10/2006

Presentation outline

- Proximity Detection of Wireless Mobile Systems
- Proximity Detection algorithms + enhancements
 - All-pairs algorithm (brute force)
 - External and internal squares
 - Grid-based data structure
 - Results caching
- Experimental environment
- Simulation results
- Conclusion and future work

Proximity Detection of Wireless Mobile Systems

- A common problem of many simulations is to determine what entities are involved in the notification and processing of a given event
- In a wireless simulation a new frame transmission is translated as the delivery of a transmission event from a sender host to a whole set of potential receiver entities
- The wireless signals decay with distance depending on propagation model assumptions



Proximity Detection of Wireless Mobile Systems

- For performance reasons, it make sense to distribute the transmission event messages only to the subset of neighbor hosts that are reached by the transmission
- Since usually the hosts are mobile, such subset is very dynamic
- Under the simulation viewpoint, this problem can be translated in the dynamic identification of the recipient entities of each event message
- Some work has been done to address the proximity detection over single processor architectures and multi-processors with shared memory but mainly for cost-reasons the distributed simulation architectures are more and more interesting and diffused

The all-pairs algorithm (brute force)

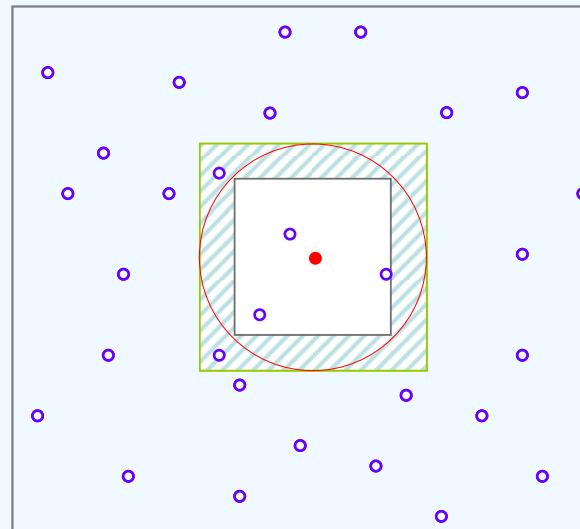
- Given a set of wireless devices moving in a simulated scenario, at each timestep, the simulator has to determine the whole set of hosts affected by the ongoing transmissions
- In a simplistic way this problem can be solved by computing the distance between each pair of nodes. This algorithm is usually referred to as “**all-pairs**” (brute force) and has a complexity of $O(n^2)$ per timestep
- Proposed proximity management solutions:
 - External and internal squares
 - Grid-based data structure
 - Results caching

- The all-pairs algorithm computes $n(n-1)$ Euclidean distance values between nodes coordinates

$$d(x, y) = |x - y| = \sqrt{\sum_{i=1}^n |x_i - y_i|^2}$$

- By exploiting the "*homogeneous limited horizon distance*" assumption of wireless transmissions, it is possible to reduce the effective cost of the all-pairs algorithm
- By assuming a 2-D space, in the worst case four comparisons are necessary to determine if a point falls within a square: the evaluation of the Euclidean distance is much more costly!

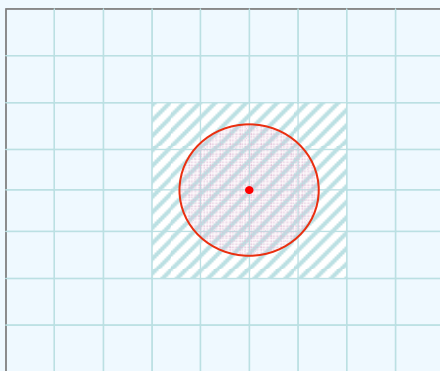
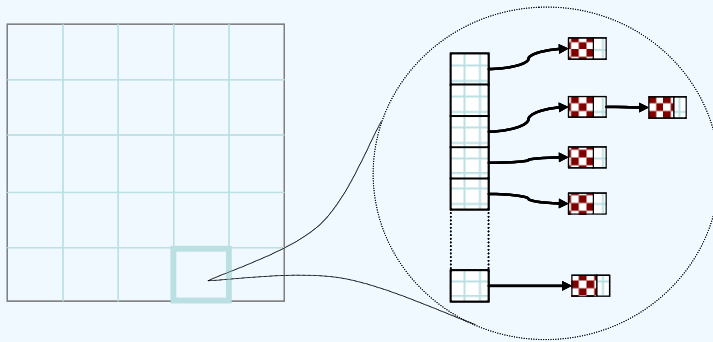
- Hosts allocated outside of the external square can be assumed to be out of the reception range of the central SMH, and can be discarded without any further evaluation



- The transmission events originated by the SMH can be delivered without any further examination to other SMHs within the inner square. *Only the dashed zone has to be further evaluated*

Grid-based data structure

- An overlay grid divides the simulated space in cells. Each square of the grid (or cell) represents only a small portion of the simulated scenario and has a fixed size



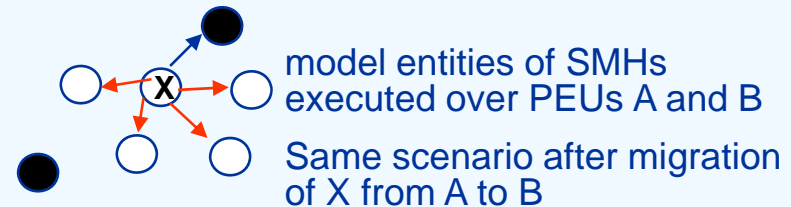
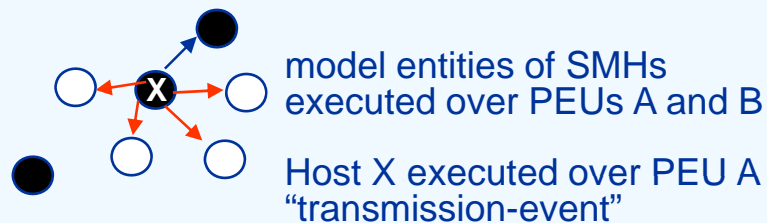
- The overlay grid is composed by a matrix of separate chaining hash tables
- Only the grid squares whose area has non zero intersection with the transmission range circle have to be checked
- In this way the number of checks can be drastically reduced, but the data structure management introduces some overhead

Results caching

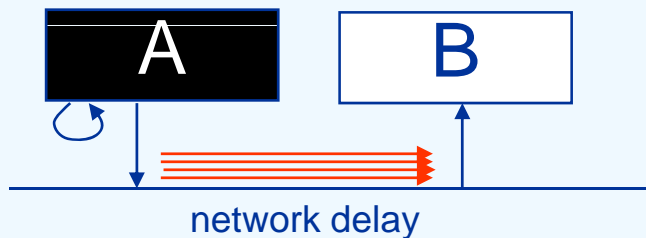
- The Euclidean distance is a commutative binary operator: given two points x and y we have that $d(x,y) = d(y,x)$
- In a centralized simulation-execution architecture it would be possible to reduce the number of Euclidean distance operations to $n(n-1)/2$
- On the other hand, in a distributed simulation, each LP manages only a subset of the SMHs and a **global knowledge of the simulated environment is missing**
- A prerequisite to apply the commutative optimization is: in a pair (SMH1, SMH2) **both simulated nodes** have to be allocated in the same LP. Otherwise the results caching can not be applied

Simulated Mobile Hosts (SMHs) migration

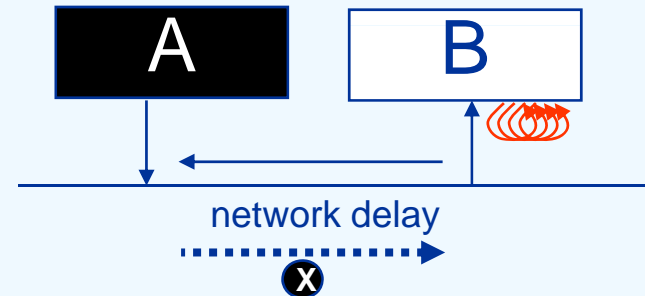
Wireless ad hoc network scenario: (evaluating migration of SMH x)



Physical Execution Units for the simulation



Physical Execution Units for the simulation



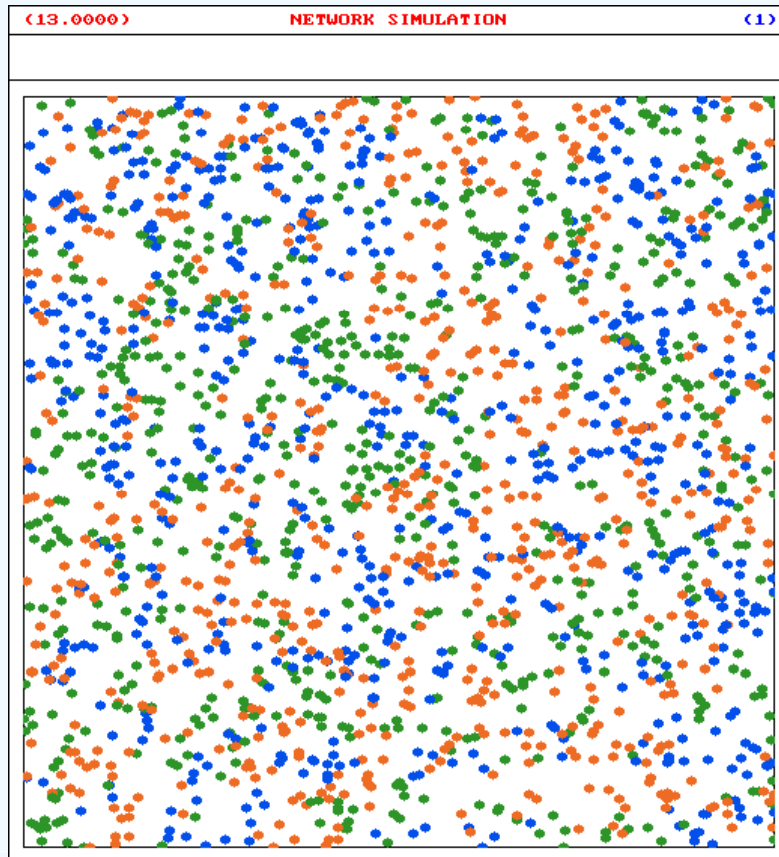
X's "transmission-event" must be notified to the 4 model entities executed over B

After X's migration, X's "transmission-event" must be notified to one model entity executed over A

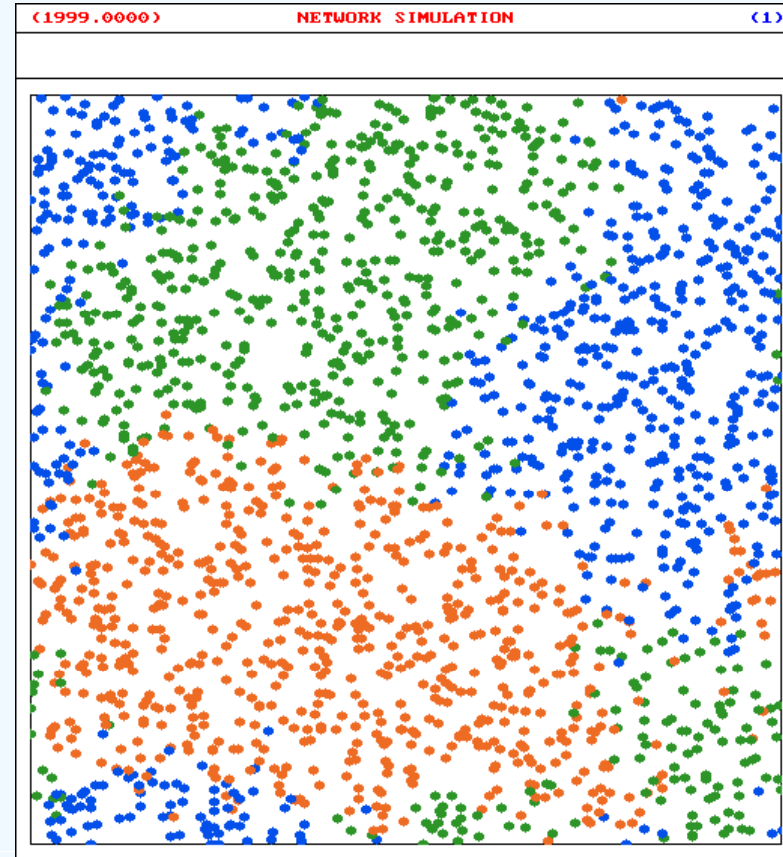
SMH = Simulated Mobile Host
PEU = Physical Execution Unit

The visual effect of the migration based mechanism

Migration OFF



Migration ON



Migration & results caching

- The migration based approach can reduce the communication cost, clustering together the highly interacting entities
- A side-effect of this mechanism is to allocate the highly interacting entities within the same LP
- In a wireless simulation the interacting entities are usually quite close in the simulated topology
- As a consequence, the migration mechanism increases the probability that two interacting mobile hosts are executed within the same LP. Hence, the migration mechanism increases **the probability to apply with success the results caching**

Performance evaluation: Ad-Hoc wireless network model

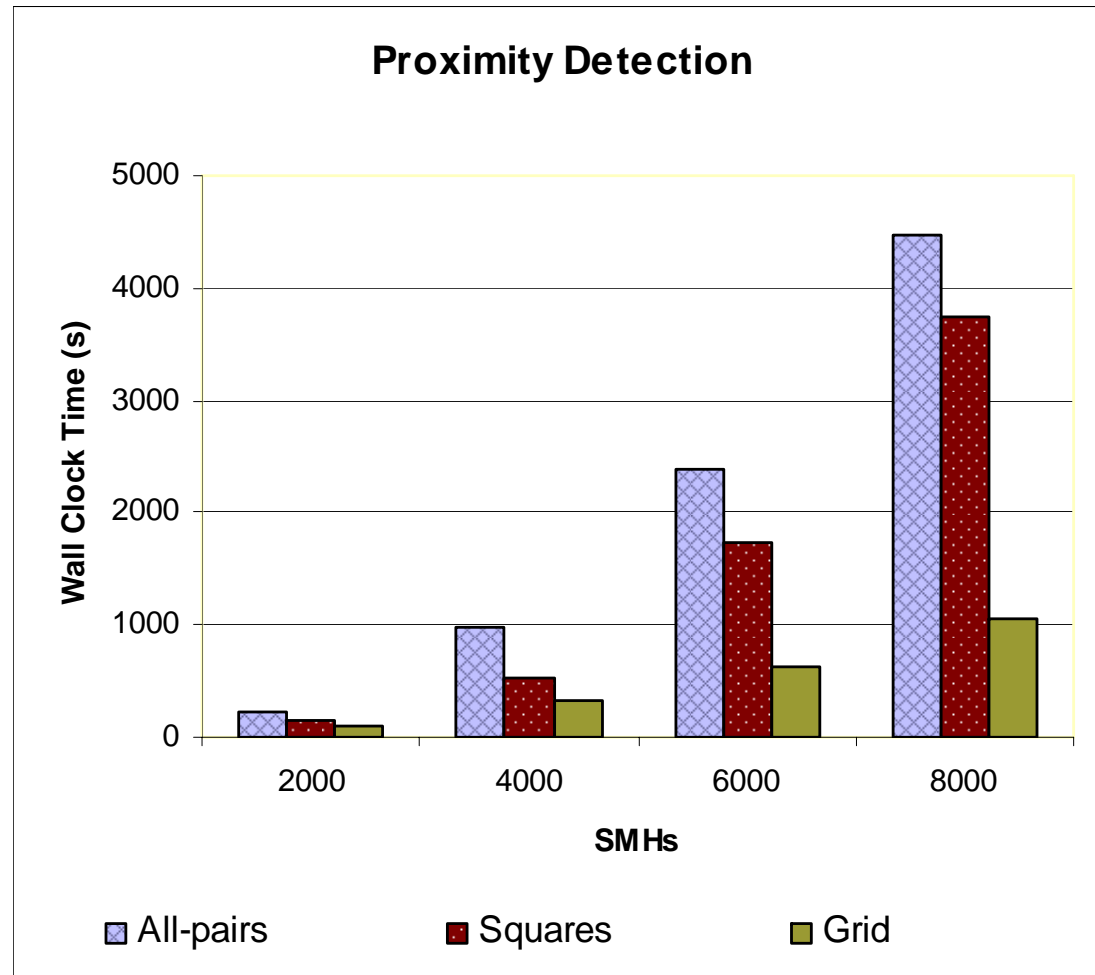
Simulated model:

- 6000 **Simulated Mobile Hosts** (SMHs) on a torus-shaped bidimensional grid-topology, 10.000x10.000 space units
- Mobility model:
 - Random Waypoint Model (max speed 10 m/s)
 - 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)

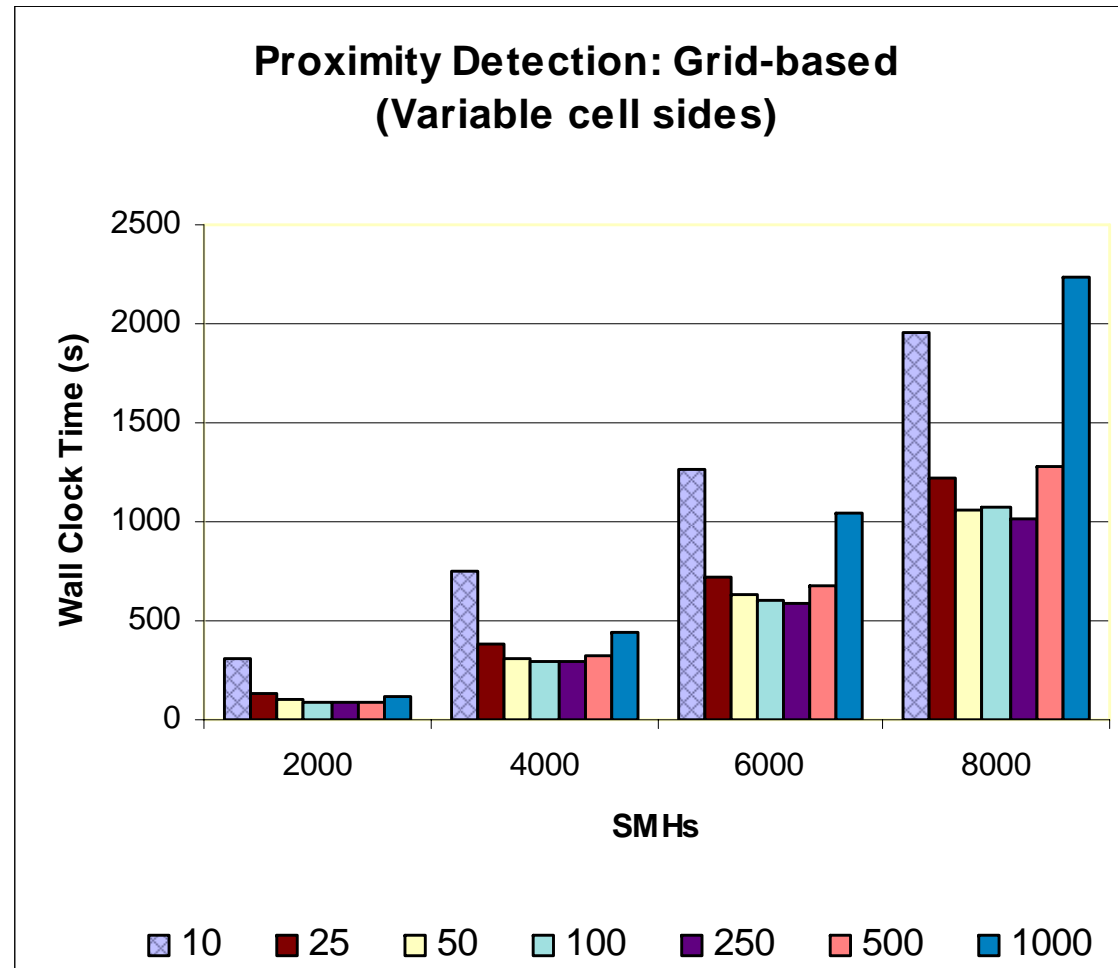
Experimental environment and performance evaluation

- PEU equipped an Intel Dual Xeon Pentium IV 2800 MHz, with 3 GB RAM, Debian GNU/Linux OS with kernel version 2.6.10
- Conservative time-stepped simulation: 1000 time-steps
- Performance evaluation:
 - All-pairs algorithm vs. proposed enhancements
 - The effect of variable grid-side on performances
 - The performance of the caching mechanism
 - The effect of the entities migrations on the proposed algorithms and enhancements

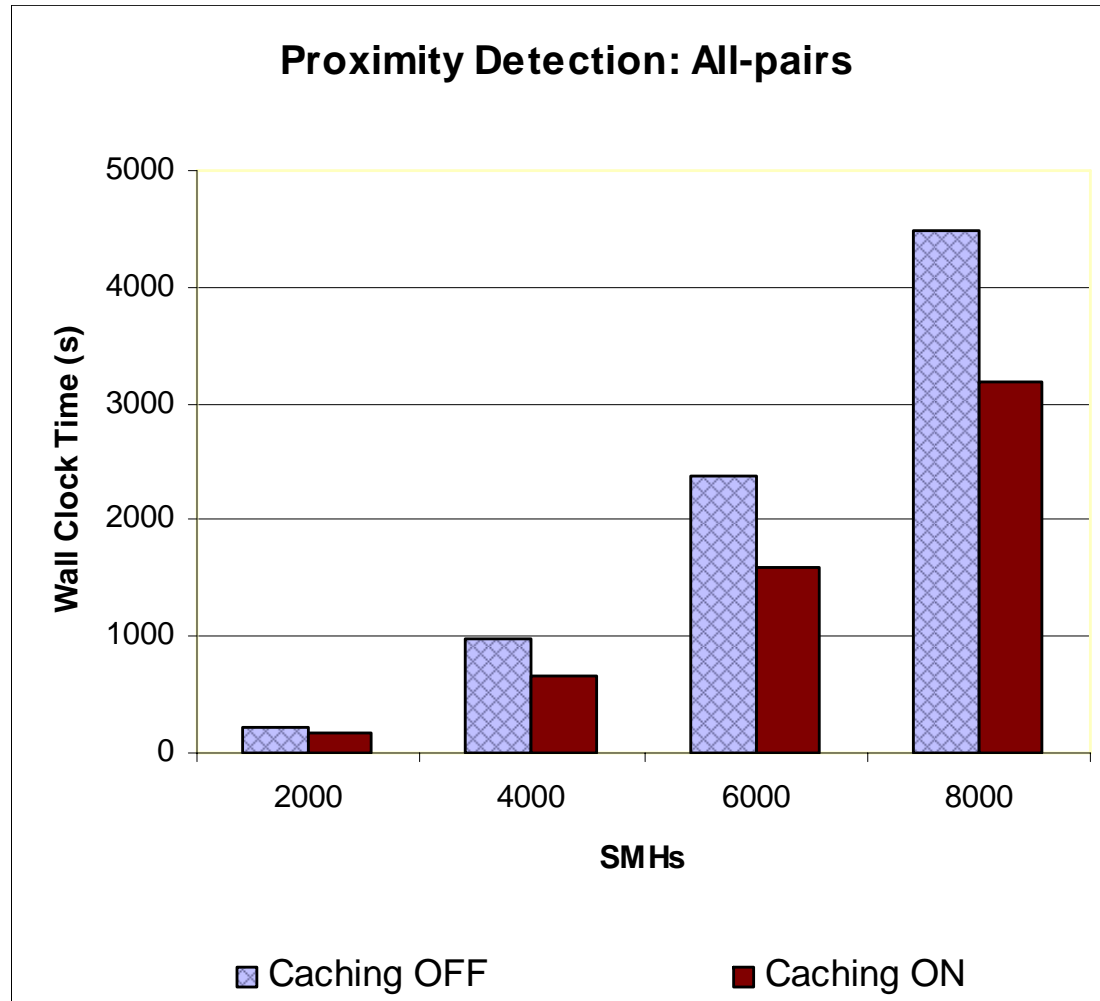
Simulation results: proximity detection



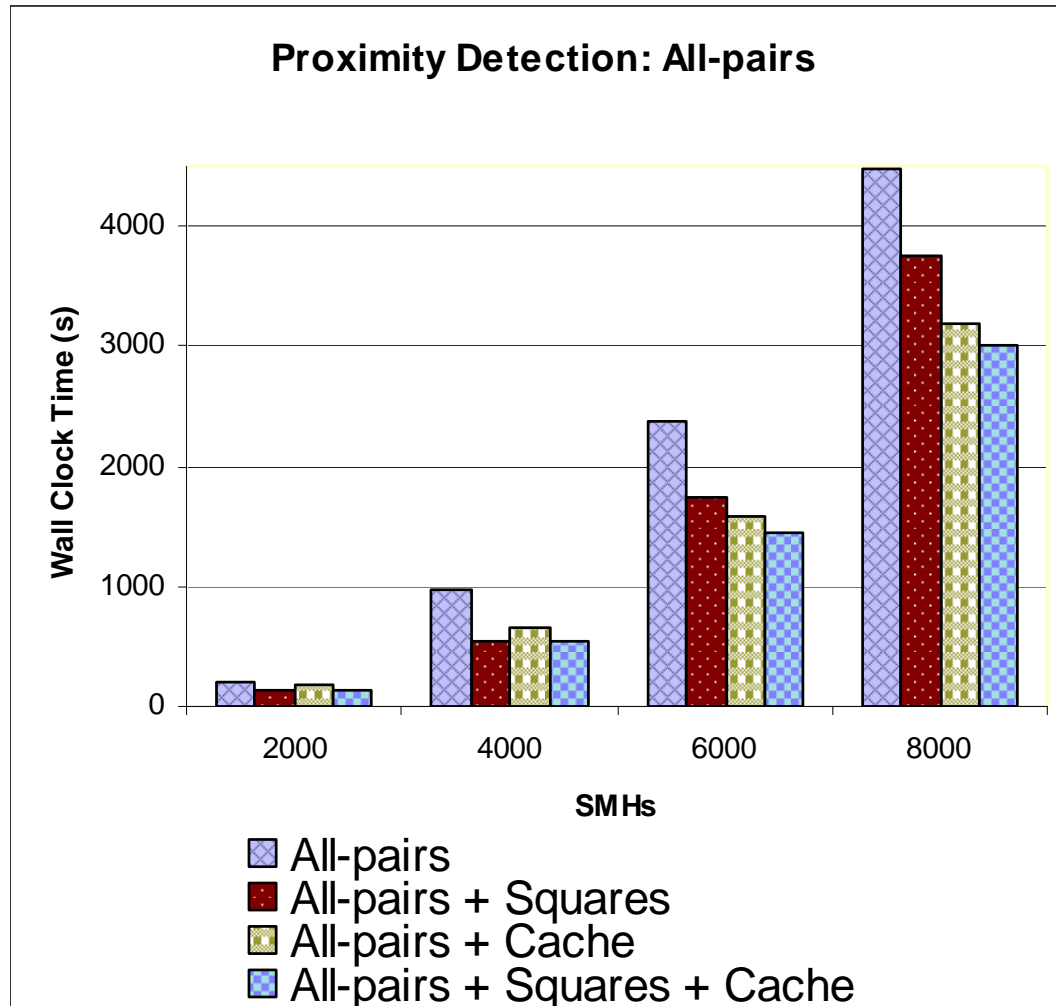
Proximity detection: grid-based



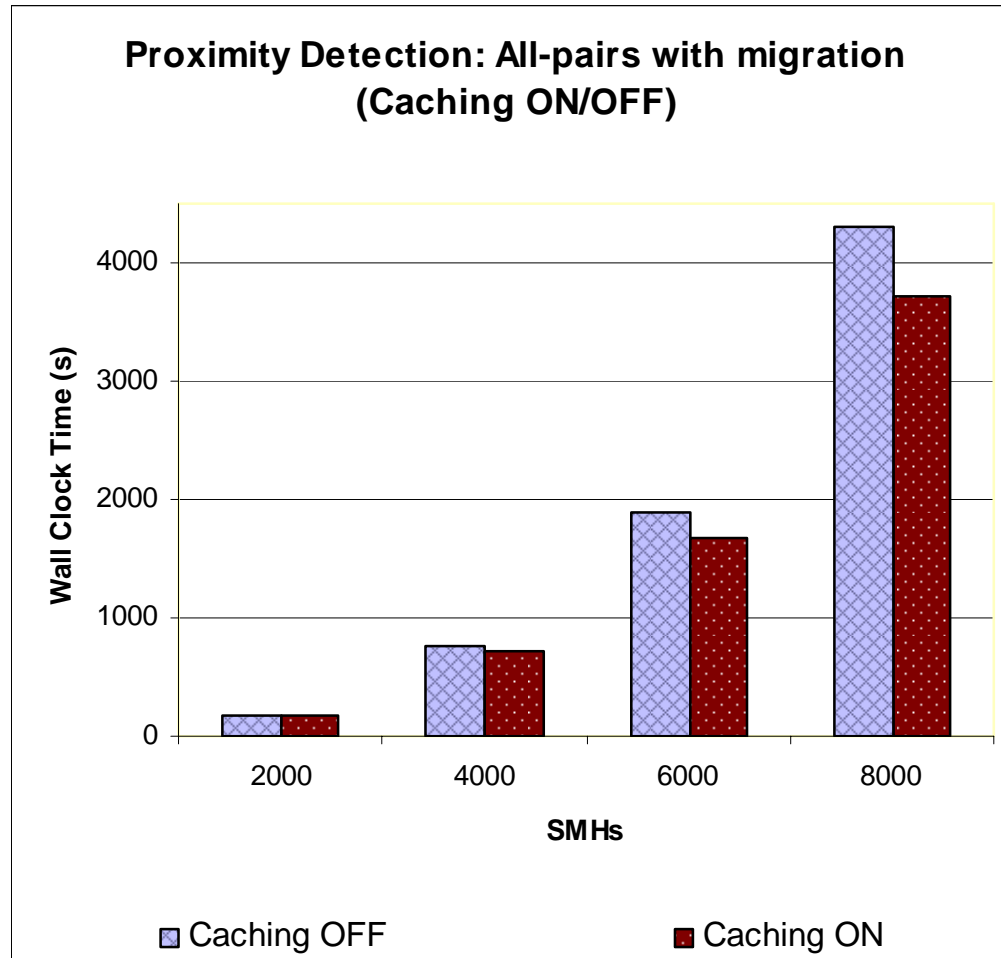
Proximity detection: all-pairs with caching ON/OFF



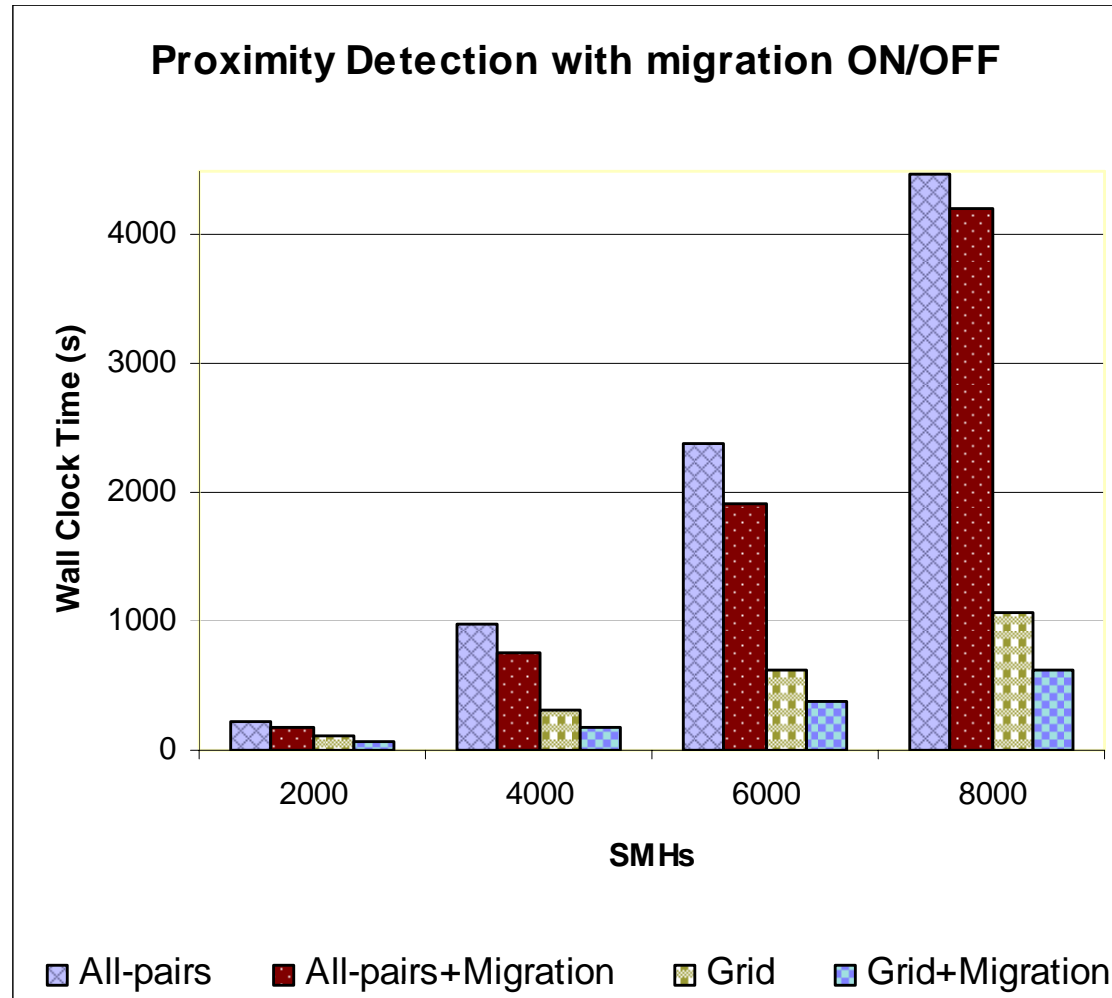
Proximity detection: enhanced all-pairs



Proximity detection: all-pairs with migration and caching ON/OFF



Proximity detection: migration ON/OFF



Conclusions and future work

- The proximity detection algorithms are a key part of the simulation of many mobile wireless systems
- We have analyzed some proposals for proximity detection algorithms being executed in a distributed architecture
- Some enhancements to the classical algorithms, and specifically tailored data structures have been proposed and evaluated
- The composition of migration mechanisms and enhanced proximity detection algorithms can improve the simulators performances
- We plan further investigations of proximity detection schemes (adaptive grid cell-sizes) and related topics (i.e. collision detection)

Proximity Detection in Distributed Simulation of Wireless Mobile Systems



Gabriele D'Angelo

Lorenzo Donatiello

Luciano Bononi

Michele Bracuto

**Department of Computer Science
University of Bologna**

Torremolinos, 03/10/2006