International Journal of Soft Computing 2 (5): 580-587, 2007
ISSN: 1816-9503

Simulation of Large Scale Wireless Systems

[1]R. Latha and [2]S.P. Rajagopalan
[1]Department of Computer Applications, St. Peter's Engineering College,
Avadi, Chennai-600 054, India
[2]Department of Computer Science and Engineering, Dr.MGR University,
Chennai-600 095, India

Abstract: The parallel and distributed simulation approach can be a valuable solution to reduce the computation time in the simulation of ad hoc and sensor networks and to support model components' modularity and reuse. In this study a testbed evaluation of the ART`IS middleware for the simulation of large scale wireless systems is presented. To realize a testbed evaluation of the considered framework a set of wireless systems models were implemented and investigated. Specifically, two classes of widely investigated wireless models were identified: Mobile ad hoc and static sensor networks. In this study the performances of the simulation framework, with respect to the heterogeneous set of execution architectures and the modeled systems'characteristics are presented. Results demonstrate that the framework leads to increased model scalability and speed-up, by transparently adapting and managing at runtime the communication and synchronization overheads and the load balancing.

Key words: Testbed, mobile ad hoc, sensor networks, RTI system, middleware, wireless systems

INTRODUCTION

Simulation makes more practical the modeling and investigation of complex and dynamic scenarios, often characterized by multiple correlated factors, "memory effects" of the system states and dynamic causal effects. Under such conditions, a simulation model would allow a level of detail that exceeds the detail level that could be obtained in most tractable mathematical models. A modular simulation modeling makes it possible the model components reuse and composition and works in favor of a correct system design, together with the possibility of a preliminary functional-test and interoperability analysis that would result in a fast system deployment. Wireless networks' architectures and wireless sensor systems are currently under analysis to obtain insights and guidelines governing many relevant design issues: As an example, system architecture and management choices, protocols' design, dynamic self-configuration and adaptation to system dynamics, systems and protocols' interoperability and co-existence, system scalability and system fault-tolerance and lifetime.

To obtain valuable insights of the investigated indices, intuitive, accurate and fine-grained methodologies and tools for the analysis is required. Because of mathematical intractability and the model complexity, simulation-based investigation of wireless systems is often preferred to numerical and analytical resolution methods (Rao and Wilsey, 2000). On the other hand, two problems appear to limit the adoption of simulation techniques for the analysis of complex systems:

- The limitation of affordable cost simulation execution architectures (Rao and Wilsey, 2000).
- The scarce possibility of model components' reuse and model composition among heterogeneous simulation models and tools.

As a consequence, the research for tools and new methodologies for standard- and module-based modeling and simulation of large-scale and complex wireless networks has received a great attention by the research community and has led to some interesting results. Currently, it is widely recognized that many of the most adopted tools for simulation of wireless systems (Network Simulator ns2, 2004) suffer the memory limitation of the execution architecture. Also, they suffer the great computation time required to complete the simulation processes (Naoumov and Gross, 2003). Large scale and complex simulation models are often impractical to simulate on a single-processor execution unit, because of huge memory requirements and large amount of time

Corresponding Author: R. Latha, Department of Computer Applications, St. Peter's Engineering College, Avadi, Chennai-600 054, India

required to complete the simulation runs (Rao and Wilsey, 2000). Parallel and distributed models and architectures may be a viable alternative to reduce memory bottlenecks through distributed memory hierarchies and to obtain simulation speed-up (Boukerche and Fabbri, 2000; Fujimotto, 2000).

The investigation of new features and services and the lack of open source runtime implementations are the main motivations behind the design and implementation of the ARTIS (Advanced RTI System) middleware. The main bottleneck arising in a distributed simulation framework is given by the communication overheads to realize the event-message distribution and synchronization services between a set of distributed model entities. The communication overhead due to the message passing required for the parallel and distributed simulation could nullify all the performance gain obtained by parallel executions (Fujimotto, 2000). The ART`IS framework is designed to realize the dynamic adaptation of the interprocess communication layer to the heterogeneous communication support offered by possible different simulation-execution architectures.

ART`IS has been also integrated in a framework with another middleware, named Generic Adaptive Interaction Architecture (GAIA). Basically, GAIA implements a simple model components migration mechanism, preliminary proposed on the top of HLA-based distributed simulations. The composition of ART`IS and GAIA realizes a framework for parallel and distributed simulation, characterized by adaptive reactions to dynamic systems behavior and oriented to the communication-overhead reduction. In this study, the prototype implementation of the ART`IS and GAIA framework is outlined. A set of testbed-evaluation results are presented to express the potential of ART`IS with and without GAIA over heterogeneous execution architectures and for two classes of relevant wireless systems models: Wireless mobile ad hoc networks and wireless sensor networks.

In this study, first the guidelines and motivations for the modeling and implementation of distributed simulation of wireless systems is outlined then the GAIA framework is introduced. Next, two wireless system's models are described then a set of simulation results is presented followed by the conclusions.

DISTRIBUTED SIMULATION OF WIRELESS SYSTEMS

Wireless systems are highly dynamic systems where the interactions are subject to fast changes driven by the system evolution. Every autonomous model component (e.g., a wireless node or sensor) is required to mimic the interactions (i.e., the causal effects of events) with all the neighbor components in the modeled scenario. Two inherent characteristics of wireless systems are: The hosts'mobility and the open broadcast nature of the wireless transmissions.

Given two or more neighbor hosts sharing the wireless medium, the causal effect of a signal interference event due to the "open broadcast" nature of the wireless transmissions could result in a chain of local-state events up to the high protocols layers. In our modeling approach, a model entity as the data structure defined to model a Simulated Mobile Host (SMH) is designed.

A high degree of causality in the simulation of the wireless hosts communication is driven by the local-topology interaction (i.e., transmissions) between neighbor hosts. If a SMH changes its position, it will eventually interact with a new community of neighbor hosts. A certain degree of time-locality among neighbors communications can be considered an acceptable assumption in many wireless system models, depending on the communication load and the mobility model assumptions. The system and model dynamics can be influenced by motion model and speed and also by the SMHs density. Complex systems with detailed and fine-grained simulation models can be considered communication-intensive under the distributed simulation approach. As a result, interprocess communication may become the bottleneck of the distributed simulation paradigm. The way interprocess communication can be supported in distributed systems would mainly depend on the execution units and on the simulation system resources, architectures and characteristics. The physical clustering of interacting model components on shared memory architecture could result in the advantage to exploit the most efficient message passing implementation.

No background optimization is based on the heterogeneous characteristics of any available communication infrastructure. The event-message distribution of a distributed simulation requires a dynamic definition of publishing/subscribing lists, or the implementation of a complete statesharing information system. On the other hand, a dynamic approach for the eventdistribution and state-information-updates (e.g., dynamic lists and subscription groups) would lead to additional communication and management overheads.

THE SIMULATION EXECUTION TESTBED

The simulation testbed consists of a distributed, discrete-event simulation of model components. Model components are executed as logical processes over a set of Physical Execution Units (PEUs), connected by a

physical LAN network. The execution architecture for our experiments is realized by 3 PEUs each one equipped by Pentium IV 2800 Mhz, with 1GB RAM, all connected by a Fast Ethernet (100 Mb s^{-1}) LAN and all equipped with Debian GNU/Linux OS with kernel version 2.6. The design approach is mainly focused on the adaptive communication-reduction between the PEUs where Logical Processes (LP) are executed. Every LP is statically allocated and executed on a single PEU. Specifically, one single LP cannot be split over two or more PEUs, more LPs can be executed over a single PEU and LPs cannot be migrated between PEUs.

Every LP is managed by a runtime simulation core (ART`IS) as a single simulation component. On the other hand, a single LP is implicitly formed by a set of threads, each one managing and updating the state (i.e. local data structures) of a set of Simulated Mobile Hosts (SMHs). A communication between wireless hosts can be modeled as a set of interactions (i.e., message-events) between any couple of adjacent SMHs. Since a wireless communication must be always modeled as a broadcast within a limited local transmission range, this requires that each SMH within a variable range would be notified with the transmission-related eventmessages.

Each event would result in a multiple set of one-to-one interactions (i.e., event messages) among local SMHs. If the sender SMH and its neighbors belong to the same LP (i.e., they are executed on the same PEU), or if they belong to different LPs implemented over the same PEU, then their interactions can be considered local (e.g. shared memory communication) and do not involve any physical network communication. On the other hand, every interaction involving participants implemented over foreign LPs (e.g., LPs implemented over different PEUs) may require time-expensive physical network communication. By reducing the physical network communication the synchronization delays can be reduced. By clustering neighbor SMHs within the same LP, or within the LPs executed over the same PEU, the causal interactions and system communication within the PEU where the interacting LPs (and their respective SMHs) are executed is closed. In addition, clustered interacting SMHs would limit interactions with the management layers of the ART`IS middleware, by further reducing the computation and communication overheads. To sum up, by limiting the network communication in favor of the local (shared memory) communication, the wall clock time required by the simulation runtime to achieve full synchronization would be reduced. This would make it possible to obtain a simulation speed-up.

A static approach could be adopted to optimally distribute the SMHs within the LPs in the simulation initialization phase. The optimal solution for allocation is hard to find and could be defined in many ways, depending on the targeted overheads' reduction. Typically, the optimality is defined with respect to latency (to reduce the physical network communication cost) or computation (to obtain an optimally balanced execution parallelism). Anyway, this should be explicitly performed offline by the modeler, on the basis of the modeling assumptions. Moreover, the model dynamics (e.g. the SMH mobility) would make the initially optimal distribution less effective after few simulation steps. This result may translate in performance degradation for the simulation speed-up, mainly due to the increasing cost of communication and synchronization required between distributed model components (logical processes).

In our approach, the optimization is dynamically performed at runtime, by the proposed simulation middleware, by migrating the SMHs between LPs. In this way, the modeler is relieved by the optimization task and the system converges towards a balanced, tuneable and pseudo-optimal model components distribution driven by the model interaction scheme. If a time-locality is assumed in the interaction between neighbor hosts, it could be convenient to migrate the foreign SMH with the LP (and to the PEU) where its new neighbors are located, by reducing the cost of successive interactions.

THE DISTRIBUTED SIMULATION FRAMEWORK

The Advanced RTI System (ARTIS): The main purpose of ART`IS is the efficient support of complex simulations in a parallel and distributed environment. Distributed model components would simply export their interfaces and interactions (i.e. messages) with the simulation middleware and Runtime Infrastructure (RTI) implementing a distributed simulation. a distributed simulation is conceptualized that could be performed over TCP/IP or Reliable-UDP/IP network protocol stacks, like in web-based simulations. The most natural and efficient execution scenarios for PADS often involve Shared Memory (SHM) and/or LAN as the infrastructures supporting inter-process communication and synchronization services. The Georgia Tech RTI-kit implementation has been realized by introducing some elasticity and optimization in the exploitation of the shared memory execution-system architecture, whereas many other implementations still rely on UDP or TCP socket-based interprocess communication even on a single execution unit.

ART`IS is composed by a set of logical modules organized in a stack-based architecture. The communication layer is located at the bottom of the middleware architecture and it is composed by a set of different communication modules. The ART`IS middleware is able of adaptively select the best interaction module with respect to the dynamic allocation of Logical Processes (LPs) in the execution environment. The current scheme adopts an incremental straightforward policy: Given a set of LPs on the same physical host, such processes always communicate and synchronize via shared memory.

THE GENERIC ADAPTIVE INTERACTION ARCHITECTURE (GAIA)

Generic Adaptive Interaction Architecture (GAIA) provides the interaction to the simulation core, the location and distribution data management, the random number generator, tracefile-logging and other simulation facilities. The target of GAIA is to provide migration and service APIs to the simulation developer. Because of the unavailability of DMSO RTI source-code, the GAIA facilities were initially provided as an external middleware on top of the DMSO RTI. The development of ART`IS middleware has permitted to merge the GAIA framework within the runtime core, still reducing the runtime execution overheads. The SMH models are implemented as code with data structures to define and maintain the SMH state information. GAIA migrates the "data structure", i.e., the state information of SMHs between LPs. This required to design and to implement a migration layer for the "state" of the SMH model entities between LPs.

The ART`IS runtime has been extended to execute static models and to exploit migration by means of a small set of Application Programming Interfaces (APIs) providing migration services for migration-enabled models. The ART`IS runtime has been designed and implemented as an alternative to HLA runtime and the GAIA middleware has been completely reimplemented, with both the migration and the load-balancing heuristics completely redesigned. Moreover, the composition of GAIA with ART`IS results in lower management overheads and greater speed-up than the ART`IS framework without the migration support.

WIRELESS SYSTEMS' MODEL DEFINITION

Two classes of wireless systems' models was considered in the testbed evaluation of the ART`IS and GAIA framework. To obtain results about the optimization and speed-up achieved based on the exploitation and adaptation to many variable model characteristics.

The mobile Ad Hoc network's model: To study the effect of host mobility and high communication loads assumptions under the modeling viewpoint, a highly scalable number of Simulated Mobile Hosts (SMHs), each one following a Random Mobility Motion model (RMM) was assumed. This motion model is synthetic and far from reality, but the choice was driven by the unpredictable and uncorrelated mobility pattern of SMHs. This is the worst case analysis for the presented mechanism, because any heuristic definition cannot rely on any assumption about the motion correlation and predictability of SMHs. The only correlation effect exploited in our mechanism is given by the "time-locality" of communication sessions between neighbor-hosts.

The RMM model is defined by SMHs swinging between mobile and static epochs. At the beginning of each epoch, every SMH decides to stay or to change its mobile or static status, by following a geometric distribution with parameter p=1/2. When entering a mobile state, new, uncorrelated and uniformly-distributed direction and speed are randomly selected and maintained up to a static epoch. The cycle is repeated for the whole simulation run by every SMH.

Sometimes motion sub-models related to the motion speed, i.e. high speed (25 spaceunits/timestep) and lower speed (10 spaceunits/timestep). To stress the migration scheme, it was also used an extreme sub-model with very high speed (100 spaceunits/timestep). Space is modeled as a torus-shaped 2-D grid-topology, 10.000×10.000 spaceunits, populated by a constant number of mobile SMHs. SMHs are randomly and uniformly distributed in the simulated area (Fig. 1).

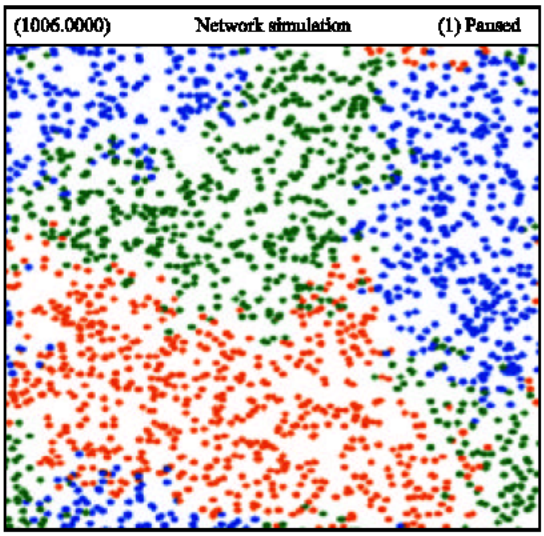

Fig. 1: A snapshot of a Mobile Ad Hoc network with 1000 SMHs dynamically allocated by GAIA over 3 PEU. Dot colors define the PEU where the SMH is executed

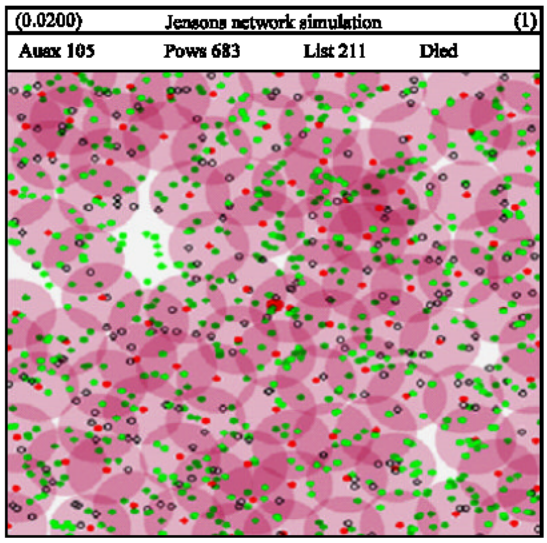

Fig. 2: A snapshot of a wireless sensor network with 1000 sensors. Dot colors refer to sensor state: Red=active, green=power saving, white= listening

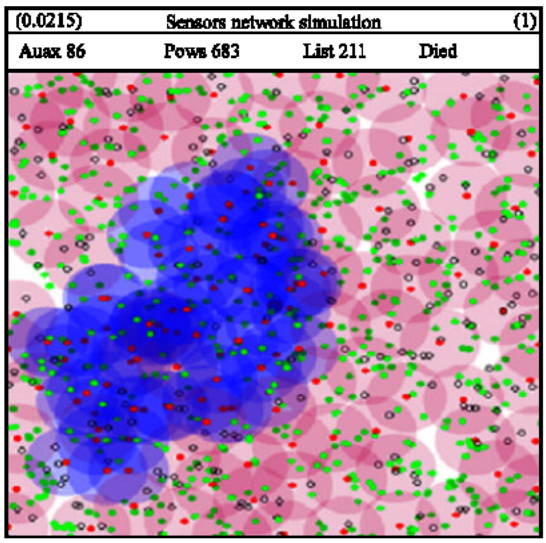

Fig. 3: A snapshot of a wireless sensor network with 1000 sensors while propagating an alert message

The torus space topology, indeed unrealistic, is commonly used by modelers to prevent non-uniform SMHs concentration in any area. This allows evaluating the mechanism behavior in a worst case scenario, where the clustering of SMHs is not trivially determined by high concentration in small areas.

The simulated space is wide and open, without obstacles. The modeled communication between SMHs is a constant flow of ping messages (i.e. constant bit rate), transmitted by every SMH to all neighbors within a wireless communication range of 250 spaceunits. Again, this choice is stressing the migration mechanism under the mobility effects of continuously transmitting SMHs. In the defined scenario, since the SMH migration policy is evaluated on the basis of the local and remote interaction (i.e. communication), no communication translates in no migration needs, hence no additional communication, synchronization and migration overheads. The rate of ping messages is constant because it is the control parameter for communication: Increasing/reducing the ping rate would be equivalent to change the interaction rate. Currently, the host model implements the Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) Medium Access Control protocol of the IEEE 802.11 Distributed Coordination Function. Adopting more detailed and complete protocol stacks implementations over the SMH model entities, results in additional advantages of parallel execution.

The sensor network model: In this model to test the model scalability, up to 40.000 static sensors results are shown. They are randomly placed with uniform distribution in the simulated area (Fig. 2 and 3). For maintaining the protocol behavior and average connectivity, the area size is variable such that the sensor density is constant in all experiments (approximatively 1 sensor in 10×10 space units). The communication range of each sensor is 15 space units.

Every sensor implements a "pressure variation" detector and sends broadcast alerts that are flooding towards a set of target detection points. A sensor can be active, listening and in power saving state. Under the modeling and simulation viewpoint this model is complete and provides detailed information about the system behavior, both for sensor network management, communication, resources utilization and network lifetime indices.

EXPERIMENTAL RESULTS

The testbed simulation experiments have been performed over heterogeneous execution infrastructures and scenarios and have involved 2 different classes of wireless systems models. The target indices to be evaluated include: The observation of migration overhead related to model dynamics under GAIA, the advantages obtained by GAIA and ARTˆIS under the adaptive communication overheads reduction and the speedup obtained by the framework under variable execution scenarios and under variable modeling assumptions for

the simulated wireless systems. Multiple runs of each experiment were performed and the confidence intervals obtained with a 95% confidence level are lower than 5% the average value of the performance indices shown. In the following it is defined as M the number of Physical Execution Units (PEUs) supporting the simulation execution and as N the total number of Logical Processes (LPs) implemented. With "migration ON" or "migration OFF" is identified a distributed simulation with the GAIA migration heuristic turned ON and OFF, respectively.

All the performed experiments were started with a pseudo-random, uniform distribution of a variable number of SMHs for both the ad hoc and the sensor networks models. Initially, the set of SMHs is randomly allocated over the set of PEUs, without any optimal allocation. The choice of the initial random distribution allows analyzing the transient dynamic effect of the GAIA migration mechanism. The random distribution would be asymptotically obtained if migration is disabled, starting from any initial (and optimal) allocation scheme and due to the SMHs mobility. Most of the figures presented show transient behavior of the performance indices, because this describes the dynamics and fast convergence effect of the proposed mechanisms.

Mobile ad hoc network's simulation: Figure 4 shows the transient number of model components (SMH) migrations performed by the GAIA middleware during the initial phase of a distributed simulation of the mobile ad hoc network model. The model is composed by 5000 SMHs, randomly distributed in the simulated area and randomly allocated over three PEUs.

The migration heuristic of GAIA begins to migrate the SMH model components between PEUs after a warmup (observation) phase of 50 timesteps. Figure 4 shows the transient number of migrations performed between the three PEUs in every timestep, based on the average speed value of SMHs in the simulated mobile ad hoc network (i.e., 10, 25 and 100 m s^{-1}). The SMH speed here is just a modeling factor to stress the simulation and it is not expected to be realistic. It can be observed that in the initial phase the GAIA middleware induces a peak of model components reallocation aiming to cluster the interacting SMHs over the same PEU. The resulting model component allocation over PEUs at timestep 1000 would be similar to the distribution shown in Fig. 1. In Fig. 1, one dot represents a SMH in the simulated area and the color of dots indicates the PEU where the SMH is executed. At the steady state, the SMH (dots) mobility would smoothly require a continuous adaptation and migration of SMHs moving out of the context of SMHs executed over the

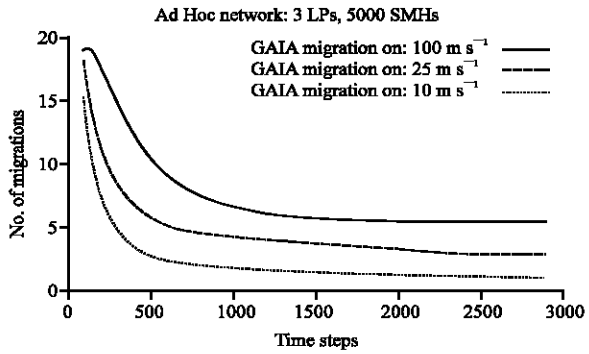

Fig. 4: Transient number of GAIA migrations per timestep, with respect to modeled mobility parameters

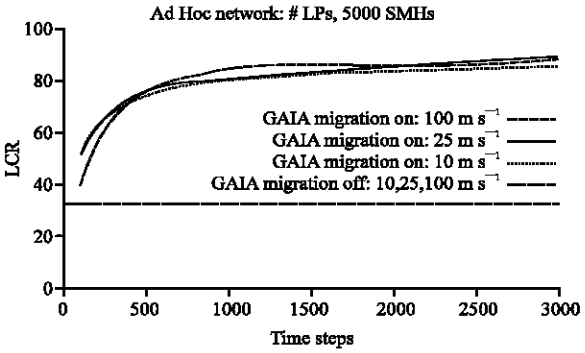

Fig. 5: Transient percentage of local communications per timestep, with respect to modeled mobility parameters, with and without GAIA migration

local PEUs. Figure 4 indicates that the higher the mobility (speed) of SMHs, the higher is the reallocation rate required by GAIA, to optimize the degree of local communications within the PEUs. Figure 5 shows the Local Communication Ratio (LCR) of messages originated by the simulation in the same scenario considered in Fig. 1 and 4. The LCR is intuitively defined as the percentage of message passing required by the simulation execution which is local to each one of the three PEUs adopted for the execution. Given the ART`IS design and assumptions, local message passing translates in efficient shared memory communication within each PEU, as an alternative to less efficient and time consuming network communications. Figure 5 shows that the GAIA migration allows to obtain a steady-state percentage of local communications around 85% in the considered scenario. Figure 5 also indicates that the mobility of SMHs has less or no effect on the LCR index when GAIA migration is active.

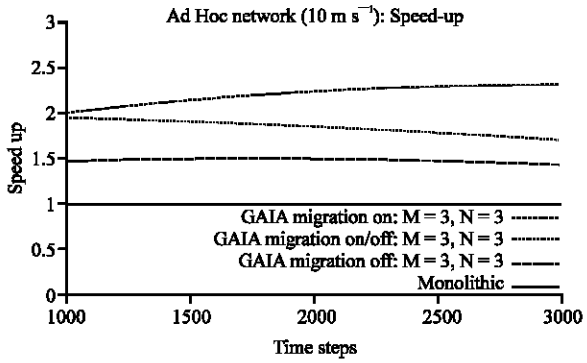

Fig. 6: Transient speed-up effect over ART`IS with and without the GAIA migration mechanism. Average SMH speed: 10 m s^{-1}

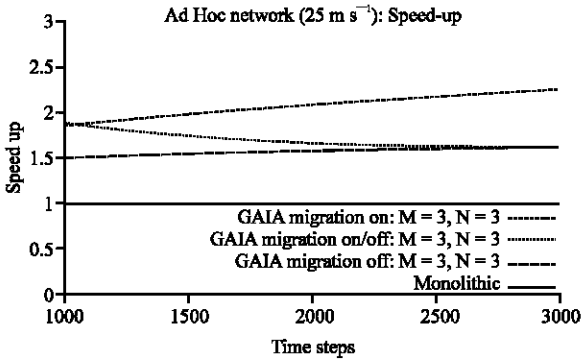

Fig. 7: Transient speed-up effect over ART`IS with and without the GAIA migration mechanism. Average SMH speed: 25 m s^{-1}

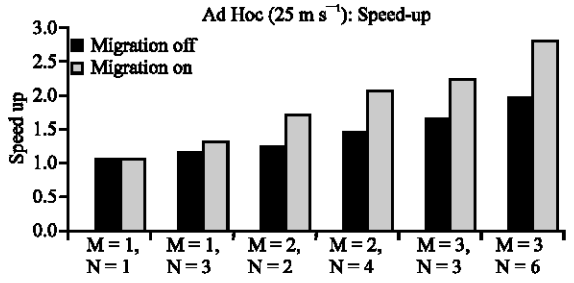

Fig. 8: Speed-up investigation of ART`IS and GAIA over many execution system architectures

This is due to the adaptive effect of GAIA migrations at runtime, shown in Fig. 4. The same simulation scenarios with GAIA migration OFF result in a LCR index which is around 33%, as expected when interacting SMHs are randomly allocated over three PEUs. Figure 6 shows results about the speed-up of the distributed simulation under the mobile ad hoc modeling scenario considered above. The speed-up is shown as a transient index, by averaging the consecutive speed-up indices calculated over separated and adjacent simulated time windows. This allows evaluating the transient effect of the speed-up in the initial phase and when GAIA is switched OFF at runtime. The monolithic scenario is considered as the normalization value for speed-up evaluation. A monolithic simulation is intended as a single Logical Process (LP) executed over a single PEU.

In our implementation, when analyzing the simulator performances, it is considered the monolithic execution platform as equivalent to a single sequential simulator. This assumption is not completely true in practice, because a really small biasing effect is introduced by the ART`IS middleware in background, anyway the biasing is really low. When the simulation is executed over PEUs (M = 3) and each PEU implements a single LP (N=3) with GAIA migration OFF, the speed-up obtained is around the value 1.5 with respect to the monolithic execution scenario. In the same scenario with GAIA migration ON, the speed-up starts around the value 2 and it increases up to 2.3 by the effect of GAIA dynamic reallocation and the increase of local communications.

The curve labeled "GAIA Migration ON/OFF" shows the effect of degradation of the speed-up obtained when, after the initial reallocation of GAIA, the GAIA migration is switched off: it is clear how the dynamic effect of SMH mobility (whose average speed is 10 m s^{-1}) realizes a transient mutation of interactions (i.e. communication) from local to external for the PEUs, by decreasing the speed-up. Figure 7 shows the same indices of Fig. 6, with the only difference given by the average speed of the modeled SMHs: From 10 m s^{-1} in Fig. 6-25 m s^{-1} in Fig. 7.

As expected, the high modeled speed translates in less "time-locality" effect of local interactions. This reduces a little the speed-up index obtained, because GAIA introduces less local communication advantages. On the other hand, the relative differences among the considered scenarios and mechanisms remain valuable, as in previous case. The same consideration about "time-locality" can be applied to explain why the speed-up degradation when the GAIA migration is switched ON/OFF at runtime is faster in Fig.7 than in 6. Figure 8 shows the speed-up investigation of many execution system architectures, based on the mobile ad hoc network model characterized by 5000 SMHs with average speed of 25 m s^{-1}. Every bar in the histogram shows the speed-up with respect to the monolithic implementation, with GAIA Migration Off and On, respectively.

The first couple of bars on the left are just a reference of the monolithic normalized speed-up. In general, the GAIA migration has a positive effect on the speed-up indices, with many PEUs, by increasing the speed-up indices up to 25%. More specifically, the second couple

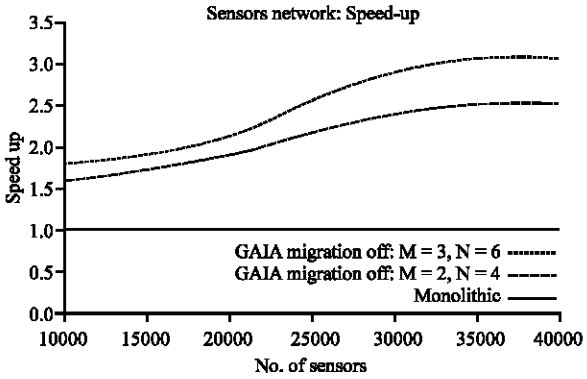

Fig. 9: Speed-up and scalability investigation of ART`IS for a massive sensor network model

of bars show the speed-up obtained by 3 LPs over 1 PEU. This indicates that ART`IS is able to exploit the shared memory, dual processor architecture of the PEU, when implementing the simulation splitted on 3 LPs. In the same way, as reported on the histogram, by increasing the number of PEUs (M value) in the execution architecture, ART`IS and GAIA show to scale and to support more LP executions by gaining simulation speed-up. In addition, in all the execution scenarios, GAIA dynamically recovers the overhead of network communication due to model assumptions (SMH mobility), resulting in additional speed-up.

Sensor network's simulation: Figure 9 shows the scalability and speed-up obtained by the simulation of the sensor network model previously defined. The effect of GAIA here is not considered because we are testing the scalability of the model and simulation implementation for a model composed by static wireless sensors. Again, every PEU considered here is a shared memory, dual processor architecture. It was considered 3 execution scenarios.

The monolithic scenario is realized by one LP over one PEU. The scenario labeled (M=3, N=6) is realized by 3 PEUs connected by the Ethernet LAN, each one with 2 LPs executed over the dual processor, shared memory architecture. The scenario labeled (M=2, N=4) is realized by 2 PEUs connected by the Ethernet LAN, each one with 2 LPs executed over the dual processor, shared memory architecture.

The figure shows the speed-up obtained as a function of the number of modeled sensors, i.e. the ART`IS speed-up and scalability under the model complexity viewpoint. The speed-up obtained increases with the number of simulated sensors. This can be explained because a huge number of sensors could exploit the potential for parallel computation expressed by the multiple numbers of processors in the execution architecture. The speed-up obtained by 3 PEU outperforms the speed-up obtained with only 2 PEUs, as expected. When the number of sensors is really high (i.e. around 40.000) the speed-up index reaches the top value and does not show reductions, indicating the good scalability achieved by the simulator performance.

As a marginal note, by considering that a state occupancy of a sensor model entity in our experiments was around 250 bytes, we executed a single experiment for a simulation of 1.000.000 sensors without having evidence of any problem. Additional investigation will be performed on the evaluation of GAIA reallocation mechanism under the sensor network scenario, in the initial phase. This would contribute to further optimize and increase the local communication and speed-up obtained.

CONCLUSION

In this study, a testbed evaluation of the ART`IS middleware for the simulation of large scale wireless systems has been presented. Two classes of widely investigated wireless models were simulated: Mobile ad hoc and static sensor networks. In the first case has been demonstrated as the integration of the GAIA framework ("simulated entities migration") leads to increased model scalability and speed-up. In the case of static sensor networks an optimal static allocation could be performed at bootstrap and so the GAIA framework is not required. In this case the middleware has proved to be scalable and our testbeds further enlarge the number of simulated entities involved. Our ongoing work includes the definition of new models for dynamically interacting systems like multi-agent systems, P2P models, scale free networks, complete protocol stacks for ad hoc and sensor models, biology-inspired models and molecular systems.

REFERENCES

Boukerche, A. and A. Fabbri., 2000. Partitioning parallel simulation of wireless networks'. In: Proc. 32nd Conf. Winter simulation, Soc. Comput. Simulation Int., pp: 1449-1457.

Fujimoto, R.M., 2000. Parallel and Distributed Simulation Systems. John Wiley and Sons, Inc., (1st Edn.), pp: 247-256.

http://www.isi.edu/nsman/ns/, 2004.

Naoumov, V. and T. Gross, 2003. Simulation of large ad hoc networks. In Proc. 6th international workshop on Modeling analysis and simulation of wireless and mobile systems, ACM. Press, pp: 50-57.

Rao, D.M. and P.A. Wilsey, 2000. An ultra-large scale simulation framework. J. Parallel and Distributed Computing, 10:18-38.

UCB/LNBL/VINT: the ns-2 network simulator.