Proceedings of the 2015 Winter Simulation Conference
L. Yilmaz, W. K. V. Chan, I. Moon, T. M. K. Roeder, C. Macal, and M. D. Rossetti, eds.

ON THE SCALABILITY OF AGENT-BASED MODELING FOR MEDICAL NANOROBOTICS

Elvis S. Liu

School of Computer Engineering
Nanyang Technological University
50 Nanyang Avenue
639798 SINGAPORE

ABSTRACT

Nanorobotics is an emerging field of research in robotics technology, which may someday benefit clinical medicine by delivering both drugs and diagnostics into the human body. Potential applications of medical nanorobotics include early diagnosis of cancer, neutralisation of viruses, precise and incisionless surgery, targeted drug delivery, and monitoring and treatment of diabetes. To further study the application of nanorobotics in medicine, this paper focuses on the design of an agent-based simulation system, which is a computer simulation composed of multiple interacting intelligent agents within an environment. Agent-based modeling has been used to solve problems that are difficult for an individual agent to solve. A swarm of simulated robots that has individual skills and partial information about a medical problem can work collaboratively to develop a solution to it. We particularly want to investigate the scalability requirements of data distribution and communication techniques under different constraints at the nano scale.

1 INTRODUCTION

A nanorobot is a type of active structure capable of actuation, sensing, manipulation, propulsion, signaling, information processing, intelligence, and swarm behavior at the nanoscale $(10^{-9}m)$ (Mavroidis and Ferreira 2013). Nanorobotics is a major development pathway of nanomedicine and may someday benefit clinical medicine by delivering both drugs and diagnostics into the human body. Medical nanorobotics should be able to treat, if not eliminate, most of the major diseases of the 20th century. Its potential applications include early diagnosis of cancer, neutralisation of viruses, precise and incisionless surgery, and monitoring and treatment of diabetes (Cavalcanti, Shirinzadeh, and Kretly 2008). Targeted drug delivery (TDD) (Lenaghan et al. 2013; Cavalcanti et al. 2007; Dadkhah1 and Yoon 2013), which uses nanoparticles functionalised with targeting moieties such as antibodies and DNA aptamers, has been used effectively in the clinic to direct drugs and diagnostic molecules to sites of interest, such as a solid tumor or an atherosclerotic plaque. Amongst the various possible medical endeavours described, TDD bears the most promise and research interest in nanomedicine.

To further study the application of nanorobotics in medicine, this paper introduces an agent-based model, which is a computational model for simulating the actions and interactions of autonomous agents. Agent-based modeling and multiagent system (MAS) (Wooldridge 2009) have been used to solve problems that are difficult for an individual agent to solve. A swarm of simulated robots that has individual skills and partial information about a medical problem can work collaboratively to develop a solution to it. These agents are organised without any centralised control unit and can act according to simple local rules. The agent-based model is an essential tool for exploring the human body, giving a better understanding on the requirements, constraints, and working principles of nanorobots through the use of virtual reality technologies.

This cross-disciplinary research lies at the intersection of mechanical and biomedical engineering, clinical medicine, artificial intelligence, and virtual reality. In this paper, we focus on the communication

requirements of the agent-based model. Some typical application scenarios such as TDD were developed and studied, which will provide a stepping stone for more research into nanomedical applications. Furthermore, we also propose a system model, which exploits parallelism and therefore enhances the computational efficiency of the simulations.

The remainder of this paper is organized as follows: Section 2 briefly reviews the background and related work of medical nanorobotics and agent-based simulations. Section 3 presents the details of the proposed simulation models. Section 4 evaluates the performance of a TDD simulation scenario. Finally, Secion 5 presents the conclusions of this paper and briefly describes our future work.

2 BACKGROUND AND RELATED WORK

This section briefly reviews the background and related work of medical nanorobotics and agent-based modeling.

2.1 Medical Nanorobotics

Nanorobotics is the technology of creating machines or robots at or close the scale of a nanometer (Mavroidis and Ferreira 2012). It is a still largely hypothetical nanotechnology engineering discipline of designing and building nanorobotic devices that are constructed of molecular components. One of the primary useful applications of nanorobotics might be in medical technology. Freitas's book (Freitas 1999) presents a historical overview of the nanorobotic concept for medical applications.

At present, there is no universally accepted design for practical nanorobots. Some prior work (Patel et al. 2006; Lenaghan et al. 2013) describes a number of key components of a fully functional nanorobot. Typically, it is desirable that medical nanorobots have the following essential characteristics and abilities. The nanorobots would be present in large numbers and use these abilities to collaboratively perform a given task in a human body.

- Swarm intelligence (Bonabeau, Dorigo, and Theraulaz 1999): the collective behaviour of decentralised, self-organized nanorobots;
- Propulsion: the nanorobots would be able to move and navigate in a human body;
- Power supply: the nanorobots would consume energy when performing their various actions;
- Sensing: the nanorobots would be able to sense the sites of interest (e.g., differentiate tumor and normal cells);
- Control: the nanorobots would control the release of drugs;
- Communication: nanorobots would need to communicate with each other and with external devices;
- Decision making: if the nanorobots detect the sites of interest, it would move towards the site.

2.2 Communication Techniques for Nanorobots

Early work such as (Lewis and Bekey 1992) suggests that a chemical communication is feasible at the nano scale. Events or objects can be made noticeable by a nanorobot through the release of diffusing chemical substances, which can attract other nanorobots through the gradient associated to the signal intensity. As observed in (Cavalcanti et al. 2006), following gradient with attractant signal is a practical method for orientation and coordination of nanorobots. It has enabled a better performance for nanorobots to detect and reach targets. It can also be useful in medical applications that requires detailed examination and intervention in a patient's body.

Another approach proposed by (Cavalcanti et al. 2006) is based on the assumption that collision at the nano scale could be negligible. Therefore, communication between nanorobots can happen through direct physical interactions. If the nanorobots cluster around cancer cells and the cluster is sufficiently large, it can easily be captured by x-ray computerised axial tomography.

(Loscrí et al. 2012) proposed a communication technique called NanoBee. If a nanorobot discovers a cancer cell, it eliminates the cell and starts to 'dance' (sends signals), which will attract other nanorobots to the signal source. This technique allows nanorobots to communicate through vibration (waggle dance for the bees), which generate acoustic waves that propagate in an elastic medium and can be detected from a acoustic detector. The authors claimed that the NanoBee approach is more effective to face cancer than the Lew and Bekey's approach.

2.3 Agent-based Simulations

Agent-based simulation is a well-established field of research that is positioned in application domains including telecommunications, business process modeling, computer games, control of mobile robots, and military simulations (Logan and Theodoropoulos 2001). A typical agent can be regarded as a self-contained and autonomous entity that can sense its environment, including other agents, and use this information in making decisions. It also has attributes and a set of basic rules that determine its behaviours. Interoperability is one of the primary advantages of agent-based simulations. It allows independently developed components to interoperate in a heterogeneous environment, such as distributed virtual environments (Steed and Oliveira 2010) and military simulations (DMSO 1998). A comprehensive review of agent-based simulations can be found in (Macal and North 2009).

In this paper, we focus on the scalability of the nanorobotic simulation. A TDD application scenario is presented, which aims to simulate a virtual environment that contains tens of thousands of agents. The computational requirements of MAS at such scale may far exceed the capabilities of conventional computer systems. Therefore, we propose a solution that exploits the parallelism inherent in MASs. The simulation can be run on multiple processors, which are able to work concurrently and thus enhances the overall scalability of the simulation.

2.4 Interest Management and Scalable Data Distribution

Data distribution for parallel and distributed simulations have been studied extensively in the literature. The simplest data distribution approach is to have each agent broadcast the states it maintains. These states are received by every agent in the simulation, and are used to update their local copy of the world states. This approach works acceptably in a small scale simulation. However, a large-scale nanorobotic simulation may involve thousands of agents, which will generate a large amount of data communications. Given the limited resources available for the nanorobotic system, state broadcasting is obviously an infeasible approach. Therefore, scalable data distribution services, or commonly known as interest management, have been development to satisfy the scalability requirement.

The basic idea of interest management is to have all agents receive only the data that are of interest to them. This involves an interest matching process, which determines what data should be sent to the agents as well as what data should be filtered. Much of the influential work in interest management has been reviewed and discussed in Liu and Theodoropoulos's comprehensive survey (Liu and Theodoropoulos 2014). In this paper, we focus on the aura-based interest management approach, which was originally proposed in (Bassiouni, Williams, and Loper 1991) and was later employed in the DIVE system (Carlsson and Hagsand 1993). The basic idea of this approach is to use 'auras' to represent the interests of simulation entities. When two auras overlap, a connection between their owners is established and messages are exchanged through the connection. This approach provides a precise data filtering mechanism, however, the simulation system needs to periodically check the overlap status for the auras, resulting in increased computational costs. In Section 3.2, we present a parallel approach to reduce the computational costs of this matching process.

3 SIMULATION MODELS

In this section, we present the design principles of a TDD nanorobotic simulation. The proposed simulation system consists of an application model and a system model, which are described in the following two subsections, respectively.

3.1 Application Model

The application model describes the attributes of nanorobots. Each robot has two sensors with various ranges, which allow the robot to sense cancer cells and other robots within a circular aura (hereafter referred to as the 'sensor aura') at its current position. This is similar to the aura-based interest management approach presented in (Liu and Theodoropoulos 2014). When the robot sees a cancer cell, it will engage the target and deliver the drugs. The NanoBee approach described in Section 2.2 is also incorporated in our design, which allows the robots to attract other robots by sending signals. To implement this design, an additional circular aura (hereafter referred to as the 'signal aura') would be associated with the robot. When a sensor aura and a signal aura overlap, the robot that owns the sensor will engage the signal source in order to collaborate with other robots to eliminate the cancer cells. Table 3.1 describes the nanorobot model. The parameters are set based on the estimations presented in (Hogg and Freitas Jr. 2012).

Table 1: Nanorobot Model.

Robot size	0.5μm radius
Sensor aura	One sensor aura is associated with each nanorobot. Its radius is set to 2.5μm
Signal aura	One signal aura is associated with each nanorobot. Its radius is set to 100μm
Robot actions	Robot actions are based on the following three modes: • Search: the robot moves randomly around the virtual space searching for a cancer cell or a signal source. If it encounters an obstacle, it moves away in a different random direction. If a target appears, it then changes to Engage mode. • Engage: the robot moves towards the target. If it reaches a cancer cell, it then changes to the Eliminate mode and starts sending signals to attract other robots • Eliminate: the robot deliver drugs to the cancer cell and eliminates it. Once the target is eliminated, the robot changes to the Search mode.

The simulation system has to match the sensor and signal auras to determine which pairs of auras overlap. Since both of them are circular auras, a simple circle to circle overlap test is carried out. Let R_1 be the radius of the sensor aura, R_2 be the radius of the signal aura, and D be the distance between the two robots. If $D < R_1 + R_2$, then the two auras overlap and the robot with the sensor aura should move towards the signal source. If the robots are static, the computational cost of this process would be insignificant. However, in our simulations the robots move frequently during runtime in order to search for

cancer cells. Therefore, the matching process should be carried out at every simulation time-step, which introduces an $O(n^2)$ computational overhead. (Liu and Theodoropoulos 2014) has reviewed a number of interest matching algorithms to reduce this overhead. However, most of these approaches are designed as sequential processes appropriate for execution on a single processor. Since, nowadays, most of the workstations are shared-memory multi-core machines, the performance gain would be limited if we deploy the sequential algorithms to these workstations. In the next subsection, a spatial decomposition approach is presented, which can reduce the computational overhead of the matching process by distributing the workload across multiple processors.

3.2 System Model

The system model describes how and when agents send and receive information, and also defines the updates and the rules of data distribution that each processor will perform over time. The preliminary design of the data distribution algorithm is as follows: it divides the data distribution process into two phases. In the first phase it employs a spatial data structure called 'uniform subdivision' (Liu and Theodoropoulos 2009) to efficiently decompose the virtual environment into a number of subdivisions, and hashes the agents into them. In the original approach, rectangular auras (coined 'regions') are used to represent the interests of the simulation entities and the hashing process is based on a hash value $H(v)$, where v is a vertex of a region. Moreover, if not all vertices of a region are hashed into the same hash table slot, then the region exists in multiple slots. In the proposed application model, however, we use a circular aura to represents the agent sensors. For the sake of simplification, we only compute the hash value of $H(c)$, where c is the position of an agent (i.e., the centre of its sensor). Therefore, no agent would exist in more than one hash table slot (space subdivision). The hash function is defined as follows:

$$H : \mathbf{R}^n \rightarrow \mathbf{Z}^n, H(x_i) = \lfloor \frac{x_i}{l_i} \rfloor, i = 1, 2, ..., n$$

where x_i and l_i are the position of an agent and the length of space subdivision, respectively, on the corresponding dimension i.

Figure 1. illustrates an example of spatial hashing. In the figure, three agents are hashed into space subdivision (0,1); three agents are hashed into (1,2); two agents are hashed into (2,0); all other space subdivisions are empty.

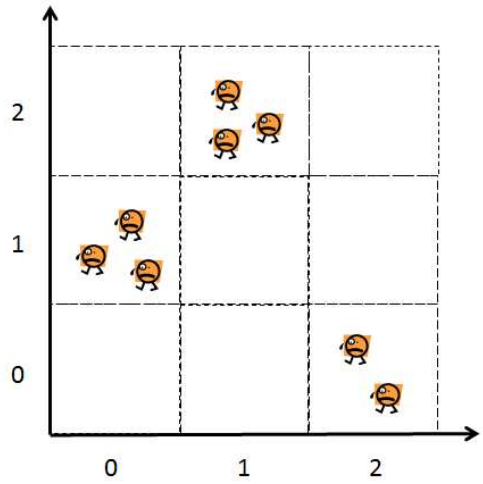

Figure 1: Example of Spatial Hashing.

After the hashing phase, each slot of the hash table represents a space subdivision which will be distributed across different processors and can be processed concurrently. The system then uses a task queue to organise the unprocessed subdivisions. During the second phase, each processor fetches a subdivision from the queue and performs simulation for it. Since only one processor has the authority to manage each space subdivision, there will be no ambiguous simulation results. Moreover, since every space subdivision is isolated during the simulation process, lock mechanisms for each subdivision are not required. Locking is only needed when a processor tries to fetch a subdivision from the task queue.

The task queue is desired for task distribution and provides a very good workload utilization. When a processor finishes processing a subdivision, it would fetch another subdivision from the task queue immediately unless the queue is empty. Therefore, no processor would be idle until all subdivision are fetched.

The performance of the system model is evaluated experimentally in the next section. Theoretically, in the best case, when all robots are distributed uniformly in the virtual space, the computational complexity of the simulation is $O(\frac{n}{p})$, where n is the number of robots and p is the number of processors. The worst case happens only when all robots reside in a single subdivision. In this situation, the workload cannot be shared, a single processor would be responsible for performing simulations for all robots. The computational complexity would become $O(n)$.

4 SIMULATION RESULTS

This section describes the evaluation of the parallel simulation model presented in this paper. The evaluation is based on the computational performance of the simulation. We carried out several experiments to compare the following two simulation systems:

1. SLS: a simulation system constructed based on the application model presented in Section 3 but not the system model
2. PLS: a simulation system constructed based on both the application model and the system model presented in Section 3

For the sake of simplification, no robot would be out of power during the simulations. The systems were implemented in C++. All of the experiments were executed on a HP Z420 workstation with a Intel Xeon E5-1620v2 3.7GHz 4-Core processor and 16GB main memory. The experiments were conducted based on the follow set-ups:

- Robot distribution: the robots were initially deployed at a single point in the virtual space
- Number of cells: 10^6 cells
- Cancer size: $10,000$ cells
- Number of nanorobots: $10,000$ - $50,000$

The first set of experiments measured the average elapsed time per simulation time-step of SLS and PLS with the number of virtual entities extending from 10,000 to 50,000. All four cores of the Intel Xeon E5-1620v2 were enabled. Figure 2 shows the simulation results. It is not difficult to see that PLS is much more computational efficient than SLS. Their difference becomes significant when the number of robots is gradually increased.

The second set of experiments measured the average elapsed time per simulation time-step of SLS and PLS when running on different number of physical cores. In the case of number of active processor cores less than 4, some of the physical cores of the Intel Xeon E5-1620v2 were disabled. The number of nanorobots was set to constant (50,000). Figure 3 shows the simulation results. Since SLS is designed as sequential algorithm, its elapsed time did not change significantly when the number of active physical cores increased. The elapsed time of PLS, however, decreased gradually due to increased number of active

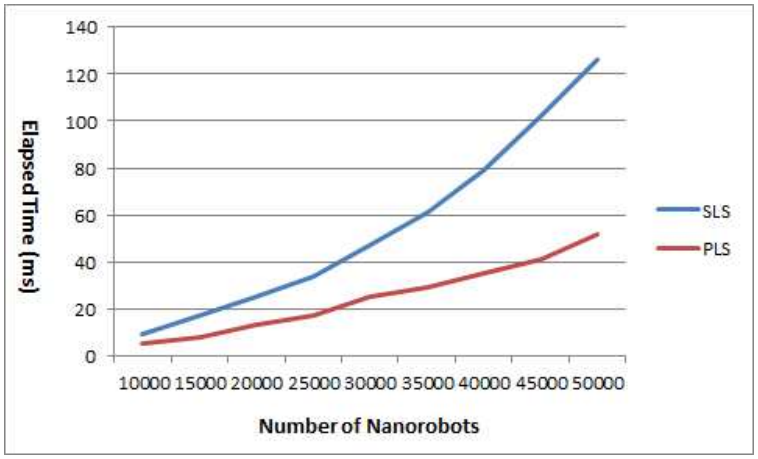

Figure 2: Average Elapsed Time of SLS and PLS (Number of Nanorobots varies).

processor cores. This suggests that PLS is more scalable when running on a 4-core machine since it is able to exploit parallelism and distribute the workload of the matching process across multiple processors.

It is worth to note that, at the time of writing, a 4-core CPU is common as a mainstream workstation CPU. Although our experiments suggest that PLS is more suitable to deploy on this type of machines, we still think it is important to investigate its scalability on a larger scale system. In the future, we plan to perform experiments on a machine with 8 or more cores, in order to further evaluate it and report our findings.

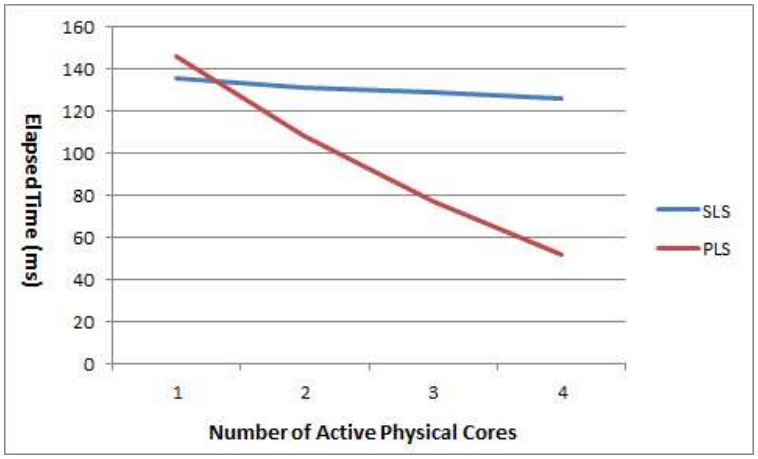

Figure 3: Average Elapsed Time of SLS and PLS (Number of Active Physical Cores varies).

5 CONCLUSIONS AND FUTURE WORK

This paper presented an agent-based simulation model for medical nanorobotics, which used a swarm of nanorobots (agents) in medical applications. A typical TDD scenario was implemented and used to evaluate the performance of the simulation system. We demonstrated how the workload of the simulation can be distributed across shared-memory multiprocessors, which is essential to meet the scalability requirement of large-scale agent-based simulations.

The future work will be focused on performing experiments on a machine with more physical cores and extending the system model so it can be applied on a cluster of distributed-memory processors. We will also extend the application model to simulate the power consumption of the nanorobots.

REFERENCES

Bassiouni, M., H. Williams, and M. Loper. 1991, October. "Intelligent Filtering Algorithms for Networked Simulators". In *Proceedings of 1991 IEEE International Conference on Systems, Man, and Cybernetics. 'Decision Aiding for Complex Systems'*.

Bonabeau, E., M. Dorigo, and G. Theraulaz. 1999. *Swarm Intelligence: From Natural to Artificial Systems*. Oxford University Press.

Carlsson, C., and O. Hagsand. 1993. "DIVE - A Platform for Multi-User Virtual Environments". *Computers & Graphics* 17:663–669.

Cavalcanti, A., T. Hogg, B. Shirinzadeh, and H. C. Liaw. 2006, December. "Nanorobot Communication Techniques: A Comprehensive Tutorial". In *Proceedings of the 9th International Conference on Control, Automation, Robotics and Vision (ICARCV 2006)*, 1–6.

Cavalcanti, A., B. Shirinzadeh, and L. C. Kretly. 2008, June. "Medical Nanorobotics for Diabetes Control". *Nanomedicine: Nanotechnology, Biology, and Medicine* 4 (2): 127–138.

Cavalcanti, A., B. Shirinzadeh, M. Zhang, and L. C. Kretly. 2007. "Hardware Architecture for Nanorobot Application in Cancer Therapy". In *Proceedings of the 13th International Conference on Advanced Robotics (ICAR 2007)*.

Dadkhah1, M., and J. Yoon. 2013. "Investigation of Dimensional Parameters Influencing a Nanorobotic Drug Delivery Actuation System Performance". In *Proceedings of the 10th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI 2013)*.

DMSO 1998. "High Level Architecture Interface Specification Version 1.3".

Freitas, R. A. 1999. *Nanomedicine, Volume I: Basic Capabilities* . Landes Bioscience.

Hogg, T., and R. A. Freitas Jr.. 2012, June. "Acoustic Communication for Medical Nanorobots". *Nano Communication Networks* 3 (2): 83–102.

Lenaghan, S. C., Y. Wang, N. Xi, T. Fukuda, T. Tarn, W. R. Hamel, and M. Zhang. 2013, February. "Grand Challenges in Bioengineered Nanorobotics for Cancer Therapy". *IEEE Transactions on Biomedical Engineering* 60 (3): 667–673.

Lewis, M., and G. A. Bekey. 1992, July. "The Behavioral Self-organization Of Nanorobots Using Local Rules". In *Proceedings of the 1992 lEEE/RSJ International Conference on Intelligent Robots and Systems*, 1333–1338.

Liu, E. S., and G. K. Theodoropoulos. 2009, October. "An Approach for Parallel Interest Matching in Distributed Virtual Environments". In *Proceedings of the 13th IEEE/ACM International Symposium on Distributed Simulation and Real Time Applications (DS-RT 2009)*.

Liu, E. S., and G. K. Theodoropoulos. 2014, March. "Interest Management for Distributed Virtual Environments: A Survey". *ACM Computing Surveys* 46 (4): 51:1–51:42.

Logan, B., and G. Theodoropoulos. 2001, February. "The Distributed Simulation of Multiagent Systems". *Proceedings of the IEEE* 89 (2): 174–185.

Loscrí, V., V. Mannara, E. Natalizio, and G. Aloi. 2012. "Efficient Acoustic Communication Techniques for Nanobots". In *Proceedings of the 7th International Conference on Body Area Networks*, BodyNets '12, 36–39.

Macal, C. M., and M. J. North. 2009, December. "Agent-Based Modeling and Simulation". In *Proceedings of the 2009 Winter Simulation Conference*, 86–98.

Mavroidis, C., and A. Ferreira. 2012. *Nanorobotics - Current Approaches and Techniques*. Springer.

Mavroidis, C., and A. Ferreira. 2013. "Nanorobotics: Past, Present, and Future". In *Nanorobotics: Current Approaches and Techniques*, edited by C. Mavroidis and A. Ferreira. New York: Springer.

Patel, G. M., G. C. Patel, R. B. Patel, J. K. Patel, and M. Patel. 2006, February. "Nanorobot: A Versatile Tool in Nanomedicine". *Journal of Drug Targeting* 14 (2): 63–67.

Steed, A., and M. F. Oliveira. 2010. *Networked Graphics: Building Networked Games and Virtual Environments*. Morgan Kaufmann.

Wooldridge, M. 2009. *An Introduction to MultiAgent Systems, Second Edition*. John Wiley & Sons.

AUTHOR BIOGRAPHIES

ELVIS S. LIU is an Assistant Professor in the School of Computer Engineering at Nanyang Technological University, Singapore. He holds a PhD degree in Computer Science from the University of Birmingham, UK. His research interests include distributed virtual environments, parallel and distributed simulations, and computer games. His email address is elvisliu@ntu.edu.sg.