DOI: https://doi.org/10.46354/i3m.2019.mas.019

AGENT-MODEL BASED ON FLAME GPU FOR ASSESSING PUPAL PRODUCTIVITY OF THE TRANSMITTING VECTOR OF AEDES AEGYPTI INFECTIOUS DISEASES

Erica Montes de Oca[a], Remo Suppi[b], Laura De Giusti[c], Marcelo Naiouf[d]

[a],[c],[d]Computer Science Research Institute LIDI (III-LIDI), School of Computer Science, National University of La Plata, Scientific Research Agency of the Province of Buenos Aires (CICPBA), La Plata, Buenos Aires, Argentina
[b]Universitat Autònoma de Barcelona, Department of Computer Architecture and Operating Systems, School of Engineering, Campus Bellaterra, Cerdanyola del Vallès, Barcelona, 08193, Spain

[a]emontesdeoca@lidi.info.unlp.edu.ar, [b]remo.suppi@uab.cat, [c]ldgiusti@lidi.info.unlp.edu.ar
[d]mnaiouf@lidi.info.unlp.edu.ar

ABSTRACT
Dengue, Zika and chikungunya are among the infectious diseases that have emerged in recent years. The common denominator to these three is their transmitting vector: the Aedes aegypti mosquito. Due to sanitary reasons, it is highly important that the vector for transmitting these diseases be controlled through the implementation of strategies specifically designed for each situation. In this article, the creation of an agent-based simulation model that allows assessing control strategies and policies through parallel computing on GPU is proposed. High Performance Computing is necessary due to the large volume of data that has to be processed (hundreds of thousands of agents) to obtain results within an acceptable time frame. Model validation was done at small scale with an analogous model on CPU and NetLogo and using data from an real system. In this article, the implementation, scalability and potential of this model as decision support system (DSS) are presented.

Keywords: Infectious Diseases, Agent-Based Model, GPU, FLAME GPU

1. INTRODUCTION
The influence of human activity on the environment has resulted in the degradation of nature. The relation between climate change and the effects on human health are evident in the proliferation of infectious diseases on different locations around the globe (Barba-Evia 2016). The World Health Organization (WHO) declared a public health emergency situation, describing it as “an extraordinary event which is determined [...] to constitute a public health risk to other States through the international spread of disease; and to potentially require a coordinated international response” (Maguiña-Vargas 2016). Dengue, Zika and chikungunya are among the infectious diseases that have emerged in recent years. The spread of these diseases has become a complex situation with high mortality levels (Álvares, Torres, Torres, Semper, Romeo 2018). The three diseases are transmitted through the Aedes aegypti vector, a mosquito species typical of tropical regions. Due to its adaptability, it has spread all over the globe through commercial and tourist routes. Currently, the three diseases that are transmitted by this mosquito are among those of greatest concern for public health (López-Latorre, Neira 2016).
Due to the importance for public health of controlling this vector, control strategies that are specifically designed for each particular situation are required. To this end, it is highly important to know mosquito populational characteristics and how they propagate (Albrieu-Llinás, Chiappero, Rodán-Dueñas, Gardenal 2016).
The modelization of complex systems through classic models based on continuous and derivable functions is not enough to describe the complexity of their components and relations. For this reason, computational models have gained relevance as a solution to study and research living entities systems (Ginovart 2015).
One of the tools that can assist in the decision-making process and help prevent diseases is simulation. Complex-system Agent-Based Modeling (ABM) has emerged to simulate the collective behavior of individuals in different scenarios, allowing researchers to study their characteristics in a short period of time (Izquierdo-Espinosa 2016). Agents are autonomous individuals that are familiar with the environment where they are, which allows them to interact with other agents. Agent behavior is determined by their inner state, a set of data that allows them to store information and act through temporary and historic perceptions. The environments where agents interact can be cities, towns, or any location in the real world that the researcher wants to model (López 2017). This allows this type of model to know system behavior through agent characteristics and behaviors as well as understanding

how the system itself affects the individuals it contains (Cantergiani, Gómez-Delgado 2016). Agents control their own behavior, but this behavior is not without any limits, but conditioned by the behavior of other agents. Agents are social actors with heuristic reasoning based on simple rules, and their behavior can be modified through learning from previous experiences. This means that they have limited memory to remember interactions and the results of their actions and strategies, and they use these to decide on new actions.

The amount of information generated through ABMs requires enough computational power in all its stages – screening, analysis and visualization. Simulating most complex problems requires using parallel or distributed solutions to obtain results within an acceptable time frame.

In recent decades, the advance of Graphics Processing Units (GPUs) in general-purpose programming (GPGPU), has allowed speeding up numerous applications that can be adapted to this architecture (Montes de Oca, De Giusti, De Giusti, Naiouf 2018). GPU is a *manycore* architecture whose approach is based on running parallel applications, with core number being doubled in each new generation. These devices have evolved and increased the calculation power in each floating point per second (initial design feature), which allows carrying out mass calculations of this type and in parallel. This parallel architecture has become one of the most widely accepted architectures by the scientific community, since its monetary cost is acceptable in relation to its computation power for the development of applications with high calculation demands (Wu, Deng, Jeon 2018).

GPU programming was benefited by the launch of CUDA (Compute Unified Device Architecture) by Nvidia in 2007, which freed programmers from having to consider parallel expressions (Aguilera, Silva-Aceves, Torres-Arguelles, Martínez-Gómez, Bravo-Martínez 2018). However, to exploit GPU performance to its maximum, programmers need to be familiar with the underlying architecture and optimize it as relevant, for instance, by using shared memory to reduce global memory latency time, kernel synchronization at the host, etc. (Nvidia 2019).

The implementation of a complex system such as the one presented here requires great dedication and effort to develop the ABM simulation kernel. Therefore, we decided to use the FLAME GPU framework (Chimeh, Richmond 2018). With this framework, we can disregard parallel simulation and focus on the simulation model for infectious diseases transmitting vector reproduction, obtaining results within an acceptable time frame.

FLAME GPU is a framework that allows simulating real-world individuals or objects into virtual agents. It is an extension of the FLAME (Flexible Large-scale Agent-based Modeling Environment) version. Its main goal revolves around expanding framework features to allow modeling GPU agents. It is specifically designed for high-performance parallel architectures, and it can create models with a massive number of agents (Richmond, Chimeh 2017). Agents are defined as finite state machines (with memory). Agents communicate via messages. Agent behavior is reflected on the functions it can perform, which change the agent's internal memory through inbound or outbound messages (Heywood, Maddock, Casas, García, Brackstone, Richmond 2018). Each agent's memory is permanent for each step of the simulation, but its message list is not. This provides internal communications that reflect the global behavior of the virtualized world.

The model to simulate is specified using a format called X-Machine Markup Language (XMML). It is about using XML to describe the agent and its behavior, the list of messages used by agents to communicate to each other, and the resulting control flow of the simulator (Chimeh, Heywood, Pennisi, Pappalardo, Richmond 2018).

The model defined in XMML generates the code for a set of templates through Extensible Stylesheet Transformations (XSLT). From those templates, a set of API dynamic templates is generated and added to agent features to produce the simulator (FLAME GPU 2018).

2. BACKGROUND

Aedes aegypti is a species of mosquito originally form Africa; it is believed that it used to breed inside tree trunk holes in the woods in the north of this continent. A period of intense drought forced the species to migrate to populated areas, favoring its fast adaptation in urban areas (Ruíz-López, Gonzalez-Maso, Vélez-Mira, Gómez, Zuleta, Uribe, Vélez-Bernal 2016). This adaptation to anthropic environments is due to the availability of the necessary food for their development and survival (Rossi, Almirón 2004). Currently, it has spread all around the globe due to commercial and tourist activities, coupled to the fact that mosquito reproduction control and elimination health programs are inefficient and there is an accelerated and non-planned growth of the population (Álvares, Torres, Torres, Semper, Romeo 2018).

The life cycle of Aedes aegypti has four stages: egg, larva, pupa and adult. Female mosquitoes need to take blood to make the eggs. They prefer low luminosity hours or night time to deposit their eggs in containers with water. Once the eggs are submerged in water, depending on temperature and humidity conditions, they hatch on the second or third day. They are resistant to drying out and can survive for months or years in dried out containers (Byttebier, De-Majo, Fischer 2014).

Larvae are aquatic and feed on the organic matter found in the container, including larvae from other species (or even their own). After fourteen days, at temperatures between 25ºC and 29ºC, larvae transform into pupae, and remain at this stage for two to three days. In this state, they do not eat and use the energy accumulated during the larva period to transform. Finally, they experience anatomic and physiological changes and emerge as winged, dimorphous adults that seek humid places where there are no wind drafts. After 24 hs, they can already mate, and can live between 35-40 days (Rossi, Almirón 2004; Valendia-Romero, Olano, Coronel-Ruíz, Cabezas, Calderón-Peláes, Castellanos, Matís 2017).

The different types of containers they can use to breed depend on water supply, availability of food for the larvae, exposure to sunlight, and whether the container has a cover on top (Ngugi, Mutuku, Ndenga, Musunzaji, Mbakaya, Aswani, Irungu, Mukoko, Kitron, LaBeaud 2017).

When an adult mosquito bites, before it sucks any blood, it injects its own saliva, which contains a mix of anesthetic, blood thinners and histamine so that the host does not detect the bite and the blood does not coagulate, helping the mosquito get more of it and faster by reducing the amount of time required to be in contact with the host (Rossi, Almirón 2004).

Mosquitoes acquire the infection when they feed on a viremic person (Albrieu-Llinás, Chiappero, Rodán-Dueñas, Gardenal 2016). The virus is transmitted only through the mosquito bite, not from one individual to another. However, there have been some cases of Zika virus where transmission occurred through the sexual act or during gestation (from the mother to the fetus) (Gorodner 2016).

Aedes aegypti uses a large number of natural or artificial containers as habitat for its larvae. Some of these containers are more productive than others, meaning that vector control efforts should be aimed at eliminating the most productive ones, since these are the ones with higher epidemiological relevance. These strategies are correlated both vector local ecology and resident habits and attitudes in relation to containers (WHO 2009).

Any containers manufactured by man that are suitable for the development of mosquitoes in their non-mature stages are called "artificial microenvironments." Their distinctive characteristics are (Grech, Ludueña-Almeida 2016):

- Small size in comparison with natural environments such as swamps, irrigation channels, retention ponds, etc;
- They support a low number of species with reduced populational sizes;
- They do not generate organic matter;
- They are temporary environments with a lower frequency of predators.

The World Health Organization (WHO) recommends surveillance of the mosquito population as a method for analyzing and assessing prevention and control actions for the diseases transmitted by this vector (Cromwell, Stoddardt, Barker, Van-Rie, Messeer, Meshnick, Morrison, Scott 2017). Entomological surveillance is used for operational and research ends, to determine the following (WHO 2009):

- Vector geographical distribution;
- Obtaining measurements about vector population;
- Facilitating appropriate and timely decisions in relation to interventions.

There are several surveillances methods, and using each of them depends on the availability of skills and resources and, oftentimes, the level of infestation. The WHO recommends a sampling frequency of "weeks to months." Sampling methods vary based on the stage in the life cycle of the mosquito (Cromwell, Stoddardt, Barker, Van-Rie, Messeer, Meshnick, Morrison, Scott 2017). The most common surveillance methodologies use sampling procedures for larvae or pupae, rather than capturing eggs or adult mosquitoes. The basic sampling unit is the household, where searches are carried out to detect containers filled with water. Typically, a laboratory analysis is required to confirm the species. To regulate infestation levels, the following indexes are normally used:

- Household Index (HI): percentage of houses infested with larvae and/or pupae.
- Container Index (CI): percentage of water-holding containers infested with larvae and/or pupae.
- Breteau Index (BI): number of positive containers per 100 houses inspected.
- Pupae Per Container Index (PCI): total number of pupae for every 100 containers inspected.

The main recommendation for dengue campaigns is eradicating the most productive breeding environments based on the use of local evidence of pupal productivity in the containers. This strategy has been successfully used in many countries (Villegas-Trejo, Che-Mendoza, Gonzalez-Fernandéz, Guillermo-May, Gonzalez-Bejarano, Dzul-Mansilla, Ulloa-Gracia, Danis-Lozano, Manrique-Saide 2011).

2.1. Classic Epidemiological Models

Numeric modelization is a useful and pretty accurate tool that allows studying, analyzing and drawing conclusions about the system being considered, which allows making decisions towards controlling epidemic outbreaks (Medina-Arce, Ramos-Tapias 2017). The most commonly used models are compartment-based ones. Compartments represent a region with a group of evenly distributed individuals. While the individuals are static, it is the disease that moves in space through states (López 2017), which are based on the states through which the individuals go through. Available literature describes several mathematical models aimed at simulating infectious disease propagation. In relation to the transmission of dengue, Zika and chikungunya, the following references can be found, for example:

- Ross-MacDonald model, based on ordinary differential equations, describing the dynamics for the relation between mosquitoes and humans, to model dengue disease in Cali, Colombia (Sepúlveda-Salcedo, Vasilieva, Martínez-Romero, Arias-Castro 2015);
- Mathematical model that uses ordinary differential equations to describe the evolution in time of human and mosquito populations in relation to the chikungunya virus in Guatemala (Ponciano, Chang, Quiroga 2018);
- Compartment-based mathematical model on a subpopulational network focused on dengue, Zika and chikungunya diseases (Anzo-Hernández, Velásquex-Castro, Bonilla-Capilla, Soto-Bajo 2018).

2.2. Agent-Based Epidemiological Modeling

Agent-based models are models that can be implemented on computers and allow describing problems whose analytical solution results are difficult to interpret or problems where finding a solution takes too much time (Rodriguez-Zoya, Roggero 2015). They have been widely used by the scientific community to understand how diseases are transmitted. References of epidemiological models based on agents developed for GPU architecture were gathered for this work; the most relevant are:

- ABM to simulate an activated sludge reactor on GPU (Pereda-García 2014);
- ABM modeling of an immune system using the FLAME GPU framework (Tamraker 2015);
- Implementation of a hybrid model using Verlet integration method and ABM to simulate agents involuntary and intentional interactions within a given mass; simulation results being presented on a GTX 750 GPU (Gutierrez-Milla, Borges, Suppi, Luque 2016);
- ABM for Reynolds' Boids simulation using GPU; a comparison with already implemented models is made. The results obtained show great speedup and a better understanding of agent behavior (Hermellin, Michel 2016);
- ABM that allows finding a treatment for vocal chord inflammation. The ABM proposed is run on GPU and, specifically in this work, a 3D visualization feature is added to the model (Seekhao, Jaja, Mongeau, Li-Jessen 2017);
- Simulation architecture called ParaCells, used to model biological problems. It uses the concepts of ABM and leverages the parallelism in GPU (Song, Yang, Lei 2018);
- Simulation of the pilgrimage to Makkah using an underlying model with agents, implemented with CUDA for GPU (Majid, Hamid, Rahiman, Zafar 2018).

There were no references in the literature reviewed for this work to models similar to the one presented here implemented on a high-performance architecture, based on GPU.

3. MODELING AND SIMULATION

In this article, an agent-based model is proposed for assessing the pupal productivity of Aedes aegypti mosquito. The model considers mosquito reproduction, which is dependent on container productivity. Results obtained in (Borges, Gutierrez-Milla, Suppi, Luque, Brito-Arduino 2015) show the effects on infectious disease propagation if container productivity is considered when developing prevention and control actions. As a first approximation, (Borges, Gutierrez-Milla, Suppi, Luque, Brito-Arduino 2015) developed a model on NetLogo; however, given the characteristics if this environment (based on Java), obtaining results for a large number of individuals and different scenarios within an acceptable time frame is not possible. To overcome these issues, a new agent-based model that can be run on a high-performance architecture, based on the FLAME GPU environment, is proposed. This new model, suitable for high-performance parallel environments, will allow modeling larger areas and simulate different scenarios within acceptable time frames. In the following sections, both approximations are presented, and the results obtained with the new version of the model proposed here are discussed.

3.1. Model on NetLogo

In (Borges, Gutierrez-Milla, Suppi, Luque, Brito-Arduino 2015), an agent-based model that allows assessing the pupal productivity of Aedes aegypti mosquito species was proposed. It is based on the control actions proposed by the WHO, considering the traditional method proposing division into areas, called strata, containing houses and breeding sites for Aedes aegypti. The mosquito gets the infection if it bites an infected individual and then, when it bites another individual who is not infected, it may transmit the disease. Only female mosquitoes can bite; they use the blood they suck to lay their eggs. Considering container productivity can help sanitation agents come up with more efficient actions, since they can focus on removing those containers whose productivity is higher, reducing the population of Aedes aegypti and, therefore, minimizing the risk for contagion.
The model was validated with data obtained from the real system through a study carried out in São Sebastião, which is on the northern coast of the state of São Paulo, 220 Km away from the state capital (Brito-Arduino 2014). In this study, a percentage of container productivity was obtained through field work. Containers were classified into removable and non-removable. This percentage is compared with the

average number of pupae produced in each container in several simulation runs.
The main limitations of the model developed on NetLogo revolve around its inability to simulate large areas. In (Brito-Arduino 2014), the study area encompassed 400.4 km^2. NetLogo was able to simulate 30,600 m^2, which represents a very small fraction of the total. Also, NetLogo only allows sequential runs, which means that a very long time is required to reach results statistical stability, since this type of models has a very high computational demand when the number of individuals is increased.

3.2. Model on FLAME GPU

Even though the types of agents to be represented and their interactions are similar to those implemented on the NetLogo model, their development is completely different, since the language used for modeling and interaction is entirely different. The same as NetLogo, FLAME GPU allows creating multiple agents (with internal memory and a set of functions that represent their actions) that interact through messages.
The framework, using an XML file and a *functions.c* file (containing agent actions implementation), is responsible for creating all necessary files (with *.cu* extension) that will form the simulator.
The developer only has to focus on implementing the details that are specific to the problem to be modeled, since the framework not only translated the code into CUDA language, but it also carries out all necessary optimizations to achieve maximum performance for the underlying GPU architecture.
The model simulates three types of agents: people, mosquitoes and containers. Some important parameters in the model are: the number of people and/or mosquitoes initially infected, the sizes of the populations of people and mosquitoes, the number and types of containers, the mosquitoes life time (by stages), the time of incubation of the disease, the size of the virtual world, the number of days of simulation, among others. The mosquitoes, in the initial state, are in the egg-stage, after completing this period, go through the larva-stage and then to pupa-stage and adult-stage. An adult mosquito will move through the virtual world in a maximum of 100 meters from the container where its birth, looking for a person to bite. Then, the mosquito, will explore for containers within the radius of action to deposit eggs.
In Figure 1, shows the three types of agents in the model. The types of containers can be tires, vases, glass containers, plastic, or metal, among others. If an uninfected mosquito bites a person not infected, neither the person nor mosquito acquire any disease. If the person is infected and the mosquito is not infected, the person will infect the mosquito, and after the incubation period, when the mosquito bites a not infected person, this person, after an incubation period, will be infected. An infected mosquito will remain (all his life) infected, contributing to spread the disease.

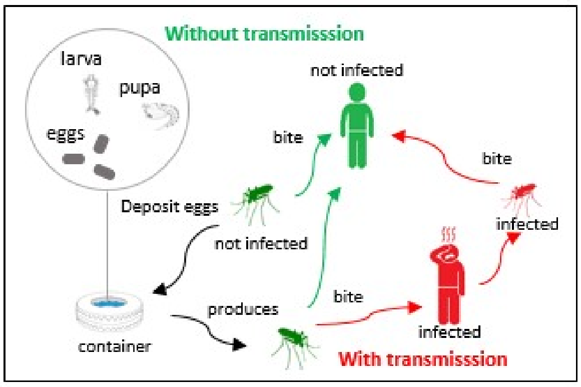

Figure 1: Model Agents and transmission cycles

The model can be configured to define the productivity of the containers. This productivity will determine the amount of eggs that a mosquito can put in each container. The productivity is used in the model to compute the pupal index. The pupal index (or the larval index) are used to estimate the number of mosquitoes that will reach the adult stage. The model can be configured to use any of these indices, however, the choice of the pupal index is more accurate since the mortality rate of the pupal stage is much lower than that of the larval state. On the other hand, if the productivity of the containers is not considered, the amount of eggs placed in each container will transform in a 100% of adult mosquitoes (worst case).

3.3. Experimental work and results

To verify the results of the model developed on FLAME GPU with the one implemented on NetLogo, 50 simulation runs were executed on both languages using the same number of individuals and model conditions. The number of pupae obtained in the experiments correspond to an initial setup of 300 mosquitoes and a total of 100 simulated days; the number of pupae in the container was established after each run and then averaged for the 50 runs. The models consider two possible scenarios: In the first scenario, the container has a certain productivity percentage that represents the number of non-mature stages of the mosquito that can reach adulthood. The second scenario does not consider container productivity – it allows 100% of the eggs laid in it to reach adulthood. Figure 1 shows the average of pupae per container for the 50 simulation runs carried out for each of the environments (bars) and their variance (*+ marks).

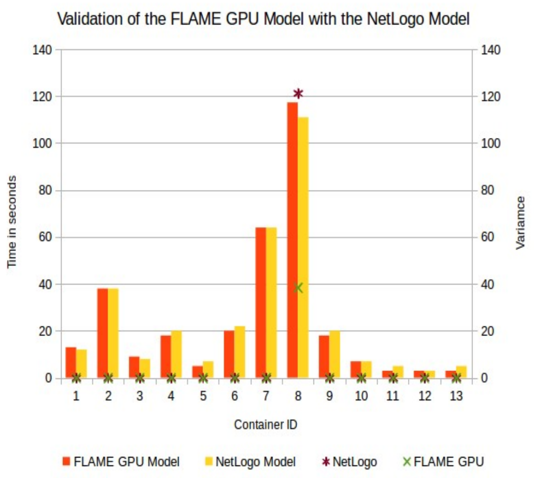

Figure 2: FLAME GPU vs. NetLogo. Average pupae per container for both models, and their variances

The number of eggs that a mosquito lays in a given container is randomly generated. It can be clearly seen that the model implemented on FLAME GPU yields results that are similar to those obtained with NetLogo, despite the fact that both models use different probability distributions for the generation of random numbers. NetLogo generates random numbers based on the normal distribution, whereas FLAME GPU uses (after an analysis of equivalences) a uniform distribution. Based on the results obtained, it can be stated that the probability distributions used are equivalent and they have no effect on result.
The variation in the number of pupae per container is reflected on the variance in each model. As it can be observed in Figure 2, the variances for both models converge. The peak observed with NetLogo in container 8 shows results variability in relation to the mean expected value; however, the average values obtained are similar in both environments.
All this confirms that the variability in the number of pupae per container is given by productivity, with randomness in the number of eggs that a mosquito can lay in each container being a contributing factor as well. To validate this, the percentages of pupae per container obtained with these models were compared with those resulting from the field study detailed in (Brito-Arduino 2014). To obtain the percentage of pupae per container in (Brito-Arduino 2014), containers were inspected in the months during which mosquito eggs, larvae and pupae are produced, in consecutive seasons of vector proliferation, between 2002-2004.
As it can be seen in Figure 3, the largest difference between the compared results is 3.9% in container 8, followed by container 7 with a difference of 1.4%. In all remaining containers, this difference is smaller than 1%. These results show that the model proposed is viable and capable of yielding results that will be similar to those of a real system (model of reference in the fig. 2).

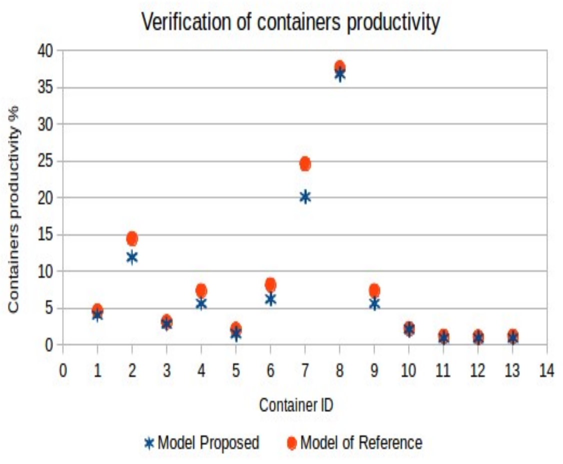

Figure 3: Comparison between pupal productivity in the containers of the real system (reference model) with the average percentage of pupae per container obtained with the model being proposed

After this validation, a new experiment was carried out, this time to show the advantages of the use of GPU as platform to run the model. Figure 4 shows the results in relation to time measurements for runs done on both models (average for the 50 runs). The initial number of mosquitoes was changed, simulating a total of 80 days. As it can be seen, for 30,000 mosquitoes, the simulation without-productivity could not be run on NetLogo, since the model exceeded memory capacity (Java Virtual Machine limitations).

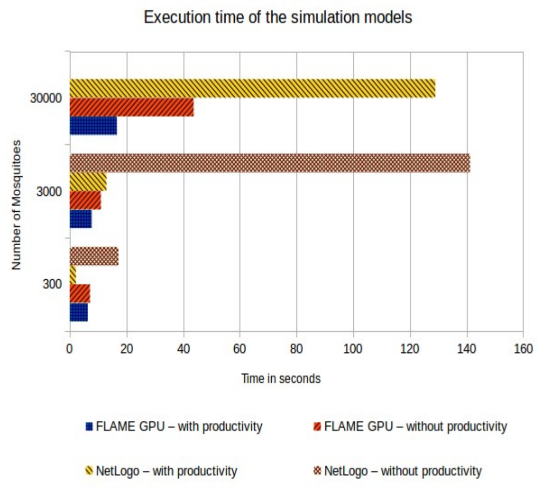

Figure 4: Measured execution time (in seconds) for both simulation models, increasing the initial number of mosquitoes in the simulation

It can be observed that the model developed on FLAME GPU, when the initial number of individuals is small, yields a higher execution time (considering container productivity) than that of the simulation run on NetLogo. In a GPU architecture, all cores must be working and if the number of thread is greater than the data to process, the GPU will distribute the work in order to avoid not idle threads. Therefore, the time-increase in the FLAME GPU model occurs because the total number of individuals in the simulation is reduced due to container productivity. In this case, the use of a GPU architecture is not recommended. However, when the initial number of mosquitoes is increased, execution time increases, yielding better results when container productivity is not considered. The limitations of the model developed on NetLogo is given by the limitations that are inherent to the environment itself, since it limits the number of agents that can be used to experiment and, consequently, only very small areas can be simulated.

These results help consider the scalability of the model proposed on FLAME GPU in relation to the initial number of mosquitoes. Figure 5 shows how the models respond when container productivity percentage is changed, with a fixed initial number of mosquitoes (set to 3,000). To do this, the most productive containers (2, 7 and 8) were selected. The result is the average of the times measured, in seconds, for 50 simulation runs in both models, with the following settings: considering original productivity, Case A (12.6%, 21.5% and 32.9%, for containers 2, 7 and 8, respectively), increasing productivity for one container at a time (considering all possible combinations: Case B-1:50%, 21.5% and 32.9%; Case B-2: 12.6%, 50% and 32.9 %; and Case B-3: 12.6%, 21.5% and 50%, for containers 2, 7 and 8, respectively), increasing the productivity for two containers at the same time (with the following combinations: Case C-1: 50%, 50% and 32. 9%; Case C-2: 50%, 21.5% and 50%; Case C-3: 12.6%, 50% and 50%, for the three containers mentioned above, as applicable), and lastly, Case D, where productivity was increased to 50% for all three containers simultaneously.

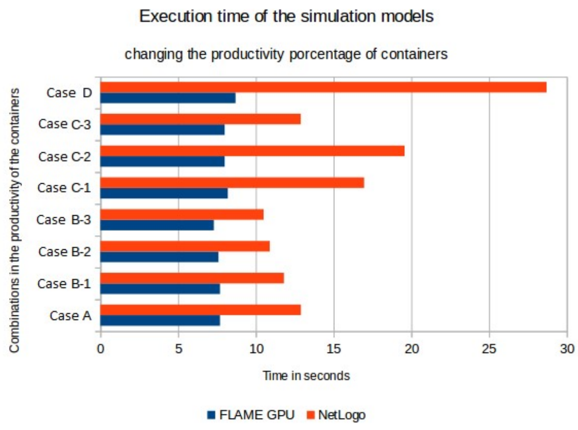

Figure 5: Measured execution time (in seconds) for both simulation models, increasing the productivity of containers 2, 7 and 8

Based on the runs carried out, the model on FLAME GPU is more stable in relation to changes in container productivity. The model on NetLogo presents an increase in the average simulation time (especially in these combinations: Case C-1: 12.6%, 50% and 50%, and Case C-2: 50%, 50% and 32.9%). Even though this increase is explained by container distribution in NetLogo, since container are close to each other and, as a consequence, they affect the number of pupae in other containers, this increase in simulation time is not proportional to the time measured for other types of containers.

4. CONCLUSIONS

The agent-based model developed with FLAME GPU offers clear advantages over its predecessor, implemented on NetLogo. With FLAME GPU, the developer can focus entirely on modeling the problem instead of having to devote time and effort to implement a complex system on GPU.

The implementation of an agent-based model on FLAME GPU to analyze how infectious diseases transmitted by Aedes aegypti spread exhibited an excellent behavior from the point of view of high-performance simulation and was successfully validated against a real system. The results obtained yielded values that are very close to the actual numbers of pupae per container produced by Aedes aegypti. Being able to reduce simulation time allow running larger simulations, be it in relation to the area to be simulated or to the number of individuals included in the simulation.

As open research lines and future work, the following topics of interest can be mentioned: a) assessing energy consumption for high-performance simulations on FLAME GPU; b) fine-tuning and validating the model using different geographical areas in Argentina; c) interactive visualization and adaptation to transform the model into an interactive tool for decision-making; d) migration to the Cloud to offer a high-performance simulation service to the scientific community.

ACKNOWLEDGMENTS

The authors would like to thank Francisco Borges for his contributions as developer of the first model on NetLogo, and especially Silvana Gallo for her collaboration with the implementation of the model on FLAME GPU. This research has been partially supported by the Agencia Estatal de Investigación, Spain and the FEDER-UE (contract TIN2017-84875-P) and collaboration agreement with the Fundación EUG.

REFERENCES

Barba-Evia J.R., 2016. Cambio Climático, globalización y su efecto sobre enfermedades infecciosas. ¿La fiebre por virus Ébola es una amenaza latente?, Rev. Latinoam Patol. Clin. Med. Lab. 63 (3), 124–132.

Maguiña-Vargas C., 2016. Zika la nueva enfermedad emergente en América. Rev. Med. Hered. 27 (1), 3-6.

Álvares E.M.C., Torres Á.A., Torres Á.A., Semper G.A.I., Romeo A.D., 2018. Dengue, Chikungunya, virus de Zika. Determinantes sociales. Rev. Med. Electórnica 40 (1), 120-128.

López-Latorre M.A., Neira M., 2016. La influencia del cambio climático en la biología de Aedes aegypti

(Diptera: Culicidae) mosquito transmisor de arbovirosis humanas. Rev. Ecuatoriana de Medicina Y Ciencias Biológicas 37 (2), 11-21.

Albrieu-Llinás G., Chiappero M.B., Rodán-Dueñas J.C., Gardenal C.N., 2016. Reconstrucción de una invasión: pasado y presente de poblaciones de Aedes (Stegomyia) aegypti en la Argentina. Preceedings of the X Jornadas Regionales sobre Mosquitos, 65-71. 15 y 16 de Septiembre de 2016, Mar del Plata, Buenos Aires, Argentina.

Ginovart M., 2015. ¿Qué pueden ofrecer los modelos basados en agentes en el contexto docente?. Modelling in Science Education and Learning 8 (2), 5-26.

Izquierdo-Espinosa G.V., 2016. Implementación de un modelo de agentes para estudiar la propagación del virus de la fiebre Chikungunya. Thesis (MD). Universidad San Francisco de Quito.

López L.R., 2017. Modelado de epidemias utilizando sistemas auto-organizados. Thesis (PhD). Universidad Nacional del Litoral.

Cantergiani C., Gómez-Delgado M., 2016. Diseño de un modelo basado en agentes para simular el crecimiento urbano el corredor del Henares (Comunidad de Madrid). Boletín de la Asociación de Geógrafos Españoles 70, 259-283.

Montes de Oca E., De Giusti L., De Giusti A., Naiouf M., 2018. Análisis de Consumo Energético en Cluster de GPU y MultiGPU en un problema de Alta Demanda Computacional. Preceeding of the XXIV Congreso Argentino de Ciencias de la Computación, 103-112. Tandil, Buenos Aires, Argentina.

Wu J., Deng L., Jeon G., 2018. Image autoregressive interpolation model using GPU-parallel optimization. IEEE Transactions on Industrial Informatics 14 (2), 426-436.

Aguilera F.J.E., Silva-Aceves J.M., Torres-Arguelles S.V., Martínez-Gómez E.A., Bravo-Martínez G., 018. Utilización de GPU-CUDA en el procesamiento digital de imágenes. CULCyT 15 (66), 65-79.

Nvidia, 2019. CUDA C Programming Guide. NVIDIA. Available from: https://docs.nvidia.com/cuda/pdf/CUDA_C_Programming_Guide.pdf [accessed April 2019].

Chimeh M.K., Richmond P., 2018. Simulating heterogeneous behaviours in complex systems on GPUs. Simulation Modelling Practice and Theory 83 (2018), 3-17.

Richmond P, Chimeh M.K., 2017. FLAME GPU: Complex System Simulation framework. Preceedings of the International Conference on High Perfomance Computing & Simulation, 11-17. Genoa, Italy

Heywood P., Maddock S., Casas J., García D., Brackstone M., Richmond P., 2018. Data-parallel agent-based microscopic road network simulation using graphics processing units. Simulation Modelling Practice and Theory 83, 188-200.

Chimeh M.K., Heywood P., Pennisi M., Pappalardo F., Richmond P., 2018. Paralell Pair-Wise interaction for Multi-agent inmune systems modelling. Preceedings of the IEEE International Conference on Bioinformatics and Biomedicine, 1367-1373. Madrid, Spain

FLAME GPU, 2018. FLAME GPU Documentation Release. FLAME GPU. Available from: https://buildmedia.readthedocs.org/media/pdf/flamegpu/latest/flamegpu.pdf. [accessed April 2019].

Ruíz-López F., Gonzalez-Maso A., Vélez-Mira A., Gómez G:F:, Zuleta L., Uribe S., Vélez-Bernal I.D., 2016. Presencia de Aedes (Stegomyia) aegypti (Linnaeus, 1762) y su infección natural con el virus dengue en alturas no registradas para Colombia. Rev. Bomédica 36 (2), 303-308.

Rossi G.C., Almirón W.R., 2004. Clave ilustrada para la identificación de larvas de mosquito de interés sanitario encontradas en criaderos artificiales en Argentina. Serie Enfermedades Transmisibles. Buenos Aires, Argentina: Fundación Mundo Sano.

Byttebier B., De-Majo M.S., Fischer S., 2014. Low temperature hatching response of Aedes aegypti eggs: effects of hatching media and storage conditions. Journal of Medical Entomology 51 (1), 97-103.

Valendia-Romero M.L., Olano V.A., Coronel-Ruíz C., Cabezas L., Calderón-Peláes M.A., Castellanos J.E., Matís M.I., 2017. Detección del virus del dengue en larvas y pupas de Aedes aegypti recolectadas en áreas rurales del municipio de Anapoima, Cundinamarca, Colombia. Biomédica 37 (2), 193-200.

Ngugi H.N., Mutuku F.M., Ndenga B.A., Musunzaji P.F., Mbakaya J.O., Aswani P., Irungu N.W., Mukoko D., Kitron U., LaBeaud A.D., 2017. Characterization and productivity profiles of Aedes aegypti (L.) breeding habitats across rural and urban landscapes in western and coastal Kenya. Rev. Parasites & Vectors 10.

Gorodner J.O., 2016. Dengue, fiebre Zika y fiebre Chikungunya. Patologías conminantes y cambio climático en América. Rev. Asociación Médica Argentina 129 (1), 30-32.

WHO, 2009. Dengue: Guidelines for diagnosis, treatment, prevention and control. World Health Organization. Available from: https://apps.who.int/iris/bitstream/handle/10665/44504/9789995479213_spa.pdf;jsessionid=4E7C3EAE339679F1547C9B8CBE44CCB4?sequence=1 [accessed April 2019].

Grech M.G., Ludueña-Almeida F.F., 2016. Mosquitos que crían en microambientes artificiales. Proceedings of the X Jornadas Regionales Sobre Mosquitos, 142-155. Mar del Plata, Buenos Aires, Argentina.

Cromwell E.A., Stoddardt S.T., Barker C.M., Van-Rie A., Messeer W.B., Meshnick S.R., Morrison A.C., Scott T.W., 2017. The relationship between entomological indicators of Aedes aegypti abundance and dengue virus infection. PLOS Neglected Tropical Diseases 11 (3).

Villegas-Trejo A., Che-Mendoza A., Gonzalez-Fernandéz M., Guillermo-May G., Gonzalez-Bejarano H., Dzul-Mansilla F., Ulloa-Gracia A., Danis-Lozano R., Manrique-Saide P., 2011. Control enfocado en Aedes aegypti en localidades de alto riesgo de transmisión de dengue en Morales, México. Salud Pública de México 53 (2), 141-151.

Medina-Arce Y., Ramos-Tapias J.A., 2017. Modelo matemático que explica mejor la afectación eidentifica el patrón relevante en la difusión para el dengue en la zona urbana del municipio de Neiva. Entornos 30 (2), 121-131.

Sepúlveda-Salcedo L.S., Vasilieva O., Martínez-Romero H.J., Arias-Castro J.H., 2015. Ross McDonald: Un modelo para la dinámica del dengue en Cali, Colombia. Revista de Salud Pública 17 (5), 749-761.

Ponciano J.A., Chang J.D., Quiroga F., 2018. Modelo epidemiológico para el estudio regional de la chikungunya. Ciencia, Tecnología y Salud 5 (1), 63-72.

Anzo-Hernández A., Velásquex-Castro J., Bonilla-Capilla B., Soto-Bajo M., 2018. Modelado y simulación de epidemias transmitidas por mosquitos en redes meta-poblacionales. Proceedings of the Congreso Nacional de Ingeniería Biomédica, 445-448. Ciudad de León, Guanajato, México.

Rodriguez-Zoya L.G., Roggero P., 2015. Modelobasado en agentes: aportes epistemológicos y teóricos para la investigación social". Rev. Mexicana de Ciencias Políticas y Sociales LX (225), 227-262.

Pereda-García M., 2014. Agent-based modeling and swarm intelligence in systems enfineering. Thesis (PhD). Universidad de Valladoid.

Tamraker S., 2015. Performance Optimization and Statistical Analysisof Basic Immune Simulator (BIS) Using theFLAME GPU Environment. Thesis (MD). University of Wisconsin–Milwaukee.

Gutierrez-Milla A., Borges F., Suppi R., Luque E., 2016. Crowd turbulence with ABM and Vertet integration on GPU cards. Procedia Computer Science 80, 2428-2432.

Hermellin E., Michel F., 2016. GPU Environmental delegation of Agent Perceptions: Application to Reynolds's Boids. In: Springer, eds. Multi-Agent Based Simulation XVI. Verlag Berlin, Heidelberg, Springer, 71-86.

Seekhao N., Jaja J., Mongeau L., Li-Jessen N.Y.K., 2017. Insitu visualization for 3D Agent-Based Vocal Fold inflammation and Repair Simulation. Supercomput Front Innov. 4 (3), 68-79.

Song Y., Yang S., Lei J., 2018. ParaCells: a GPU architecture for Cell-Centered models in Computational Biology. In: IEEE, eds. IEEE/ACM Transactions on Computational Biology and Bioinformatics. 1-1.

Majid A.R.M.A., Hamid N.A.W.A., Rahiman A.R., Zafar B., 2018. GPU-based optimization of Pilgrim Simulation for Hajj and Umrah Rituals. Pertanika J. Sci. & Technol. 26 (3), 1019 –1038.

Borges F., Gutierrez-Milla A., Suppi R., Luque E., Brito-Arduino M., 2015. An Agent-Based Model for assessment of Aedes aegypti pupal productivity. Proceedings of the 2015 Winter Simulation Conference, 159-169. December 2015, Huntington Beach, CA, USA.

Brito-Arduino M., 2014. Assessment of Aedes aegypti pupal productivity during the Dengue Vector Control Program in a Costal Urban Centre of Sao Pulo State, Brasil. Journal of Insects 2014, 1-9.

AUTHORS BIOGRAPHY

ERICA MONTES DE OCA is Technician in High-Performance Computation and GRID Technology, fellow of the National University of La Plata (UNLP, Argentina) and is currently taking courses for her Doctorate in Computer Science. She is also a member of several research projects of the Computer Science Research Institute LIDI (III-LIDI). Her research is focused on the topics of GPU, GPU clusters, MultiGPU, Agent-based Modeling, High Performance Computing, High Performance Simulation, Green Computing, Energy Consumption and Performance.

REMO SUPPI received his diploma in Electronic Engineering form Universidad Nacional de La Plata (Argentina), and a PhD degree in Computer Science form Universitat Autonoma de Barcelona (UAB) in 1996. At UAB, he spent more than 25 years researching on topics including computer simulation, distributed, systems, high performance and distributed simulation applied to ABM or individual-oriented models. He has published several scientific papers on the topics above and has been an associate professor at UAB since 1997. He is also a member of the High Performance Computing for Efficient Applications and Simulation Research Group (HPC4EAS) at UAB.

LAURA DE GIUSTI has a PhD in Computer Science at Universidad Nacional de La Plata (UNLP, Argentina), and is Professor at the School of Computer Science of UNLP. In addition, she is a member of several research projects at the Computer Science Research Institute LIDI (III-LIDI) and has authored several research works published in national and international journals and conferences.

MARCELO NAIOUF is a Chair Professor at the Computer Science Research Institute LIDI (III-LIDI) at National University of La Plata, Argentina. Head of the PhD Program at the School of Computer Science, supervisor of 8 PhD theses, invited lecturer at various Universities in Argentina, and leader in several research projects. His research interests include high performance computing and parallel algorithms. He has co-authored 3 books and more than 120 full-reviewed technical papers in journals and conference proceedings.