Accelerating Parallel Agent-based Epidemiological Simulations

Dhananjai M. Rao
CSE Department
Miami University, Oxford, OHIO, USA
raodm@miamiOH.edu

ABSTRACT

Background: Simulations play a central role in epidemiological analysis and design of prophylactic measures. Spatially explicit, agent-based models provide temporo-geospatial information that cannot be obtained from traditional equation-based and individual-based epidemic models. Since, simulation of large agent-based models is time consuming, optimistically synchronized parallel simulation holds considerable promise to significantly decrease simulation execution times.

Problem: Realizing efficient and scalable optimistic parallel simulations on modern distributed memory supercomputers is a challenge due to the spatially-explicit nature of agent-based models. Specifically, conceptual movement of agents results in large number of inter-process messages which significantly increase synchronization overheads and degrades overall performance.

Proposed solution: To reduce inter-process messages, this paper proposes and experimentally evaluates two approaches involving single and multiple active-proxy agents. The Single Active Proxy (SAP) approach essentially accomplishes logical process migration (without any support from underlying simulation kernel) reflecting conceptual movement of the agents. The Multiple Active Proxy (MAP) approach improves upon SAP by utilizing multiple agents at boundaries between processes to further reduce inter-process messages thereby improving scalability and performance. The experiments conducted using a range of models indicate that SAP provides 200% improvement over the base case and MAP provides 15% to 25% improvement over SAP depending on the model.

Categories and Subject Descriptors

I.6.8 [**Simulation and Modeling**]: Types of Simulation—*Discrete event, Parallel*

SIGSIM-PADS'14, May 18–21, 2014, Denver, CO, USA.

ACM 978-1-4503-2794-7/14/05 ...$15.00.
http://dx.doi.org/10.1145/2601381.2601387.

General Terms

Algorithms; Performance

Keywords

Time Warp; Logical Process Migration; Ghosting; Performance Improvement

1. INTRODUCTION

Humanity continues to face a multitude of global socioeconomic challenges due to annual epidemics and punctuated pandemics of highly virulent zoonoses such as avian influenza (H5N1, H7N9) and the 2009 swine flu (H1N1) pandemic. Moreover, diseases such as seasonal Influenza are of global importance because they annually affect ~90 million people globally, causing about 250,000 to 500,000 human fatalities [28] and billions of dollars of annual losses due to recurrent epidemics both in humans and livestock.

Importance of modeling and simulation in epidemiology: Epidemiological Modeling and Simulation (M&S) plays a pivotal role in study and analysis of diseases, phylodynamics, and design of prophylactic measures to contain epidemics [24, 18, 27]. Encouraged by advancement in computing technology and catalyzed by the need for detailed epidemiological analysis of emergent diseases, Agent-based Modeling (ABM) is gaining importance over traditional epidemiological modeling [24, 18, 27]. Figure 1 illustrates an ABM viewed in SEARUMS, the M&S environment used in this study, and Section 3 provides additional details on ABM.

Need for Parallel Simulation: Unfortunately, the advantages of ABMs are realized at the cost of significantly higher simulation execution time because ABMs are computationally demanding. For example, a single threaded (not counting garbage collector and other background Java threads) simulation of 3,088 (namely model `M4` discussed in Section 9) mobile agents in SEARUMS requires about 25 hours on a 3.5 GHz Intel i7-3770K CPU. The issue of long simulation execution times is magnified by the need to conduct a large number of repetitions (~200,000 repetitions are not uncommon [8]) with different parameters to analyze various scenarios.

A PDES Environment (prior research): Parallel simulation holds considerable promise to significantly decrease simulation execution times and enabling effective use of ABMs. Accordingly, our earlier investigations focused on developing a modeling, simulation, and analysis environment called SEARUMS for study and analysis of the role of migratory

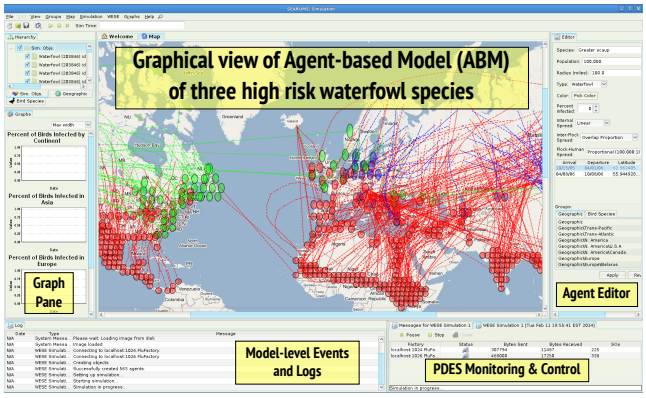


Figure 1: Screenshot of an Agent-based Model (ABM) viewed in SEARUMS, the M&S environment used in this study (see Section 2 for details.)

waterfowl in intercontinental spread of Avian Influenza [23, 22]. The graphical frontend is in Java while the simulation infrastructure operates as an optimistically synchronized Parallel Discrete Event Simulation (PDES) in C++. Figure 1 shows a screenshot of SEARUMS [23] while Section 2 presents an overview of the environment.

1.1 Motivation: Scalability Problems

The PDES infrastructure of SEARUMS provided good performance improvement over the initial conservatively synchronized, Java-based multithreaded kernel as reported in our earlier publication [21]. However, the PDES did not efficiently scale as the number of mobile agents in the model was increased even when optimism was throttled. The root cause of the issue was experimentally identified to be large number of Inter-Process Messages (IPMs) that increased synchronization overheads and degraded performance. IPMs arise because agents are initially partitioned to different compute nodes (on the compute cluster used for PDES) and they do not *physically* move (the C++ class for an agent is fixed on a compute node), but only move *logically* by changing state. However, as illustrated in Figure 2, as agents logically move they have to interact with agents on a different compute node giving raise to IPMs. Consequently, as simulation time advances from t_0 to t_2 ($t_0 < t_1 < t_2$), the communication patterns change increasing IMPs.

IPMs negatively impact synchronization resulting in degraded scalability and performance as illustrated by the charts in Figure 3. The graphs in Figure 3 also illustrate the strong correlation (Pearson correlation coefficient $r = 0.98$, p-value < 0.0001) between IPMs (Figure 3(a)) and rollbacks (in Figure 3(b)) as well as a strong correlation ($r = 0.96$, p-value < 0.0001) between inter-process messages and simulation execution time (Figure 3(c)).

1.2 Solution Overview & Paper Outline

In continuation with the foregoing discussions, the volume of Inter-Process Messages (IPMs) arising due to movements of agents had to be reduced to improve scalability and performance. Reducing IPMs requires that interacting agents must be predominantly on the same compute node – implying that as agents *logically* migrate they must be correspondingly repartitioned or relocated to a different compute node.

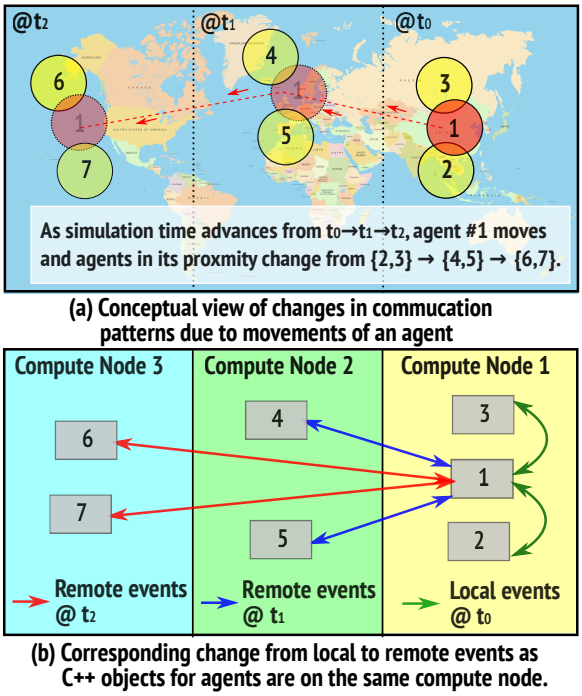


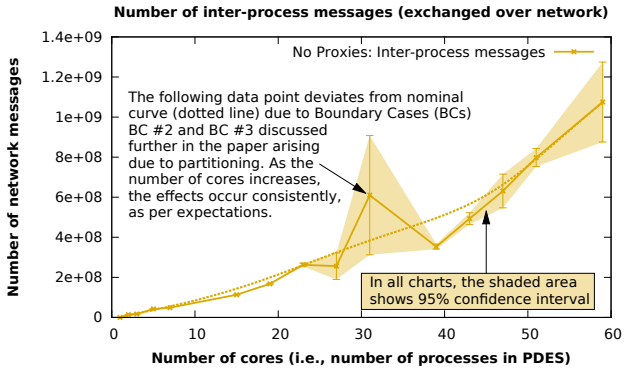
Figure 2: Overview of Problem: Increase in Inter-Process Messages (IPMs) due to conceptual movement of agents

Accordingly, the notion of *proxy* agents for each mobile agents was introduced on each compute node used for PDES. As an agent logically migrates across process boundaries, it logically deactivates itself after activating an appropriate proxy on a different compute node. Therefore, only one proxy-agent is active at any given time. This strategy, called Single Active Proxy (SAP) approach, is discussed in further detail in Section 5. SAP essentially accomplishes logical (rather than physical [16]) process migration, events thereby reducing IPMs.

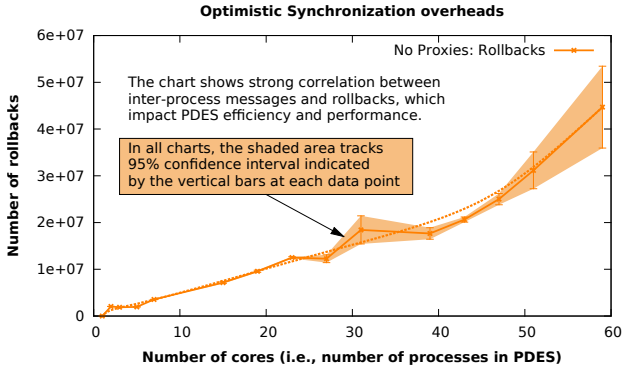
However, as detailed in Section 6, the SAP approach still experienced performance issues at boundary cases when an agent's neighborhood spans two more partitions. Consequently, a Multiple Active Proxy (MAP) approach is proposed to address three different boundary cases that arose in SAP approach. The MAP approach extends the SAP approach by permitting multiple proxies to be active at the boundary cases when an agent spans two or more partitions to minimize IPMs. Section 7 presents the MAP approach followed by Section 8 that contrasts SAP and MAP with related research investigations. Section 9 discusses results from various experiments conducted to assess SAP and MAP approaches. Section 10 concludes the paper by summarizing the outcomes and inferences drawn from this paper.

2. SEARUMS: M&S ENVIRONMENT

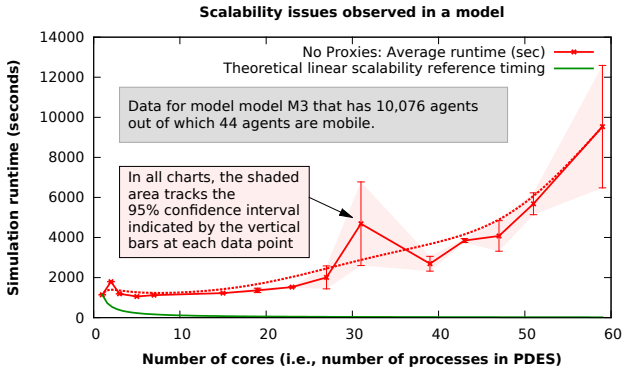
SEARUMS is the epidemiological modeling, parallel simulation, and analysis software system used in this study. An architectural overview of SEARUMS is shown in Figure 4 and a screenshot of its Graphical User Interface (GUI) is



(a) Inter-Process Messages (IPMs)



(b) Rollbacks



(c) PDES execution time

Figure 3: Experimental data (*without proposed solution* from model M3 discussed in Section 9) illustrating lack of scalability in PDES execution

shown in Figure 1. The GUI and associated frontend components have been developed in Java. The frontend modules enable execution of simulations in an offline, non-GUI mode to ease experimental analysis. These modules also handle partitioning of model using several different strategies.

The backend parallel simulation infrastructure of SEARUMS has been developed in C++ using WESE. WESE is a general purpose, web-enabled, Time Warp synchronized framework that eases development of parallel and distributed simulations [19]. It has been developed by suitably extending the WARPED simulation kernel [5] which provides the core Time Warp infrastructure. WESE further customizes WARPED to provide web-based features for modeling, parallel simulation, I/O stream centralization, monitoring, and control.

In WESE a `Factory` is deployed on a compute node and acts as an agent repository as well as a simulation server that is part of a PDES. The `communication subsystem` can be configured to operate using different network protocols and handles the tasks of interacting with other WESE factories used for PDES. A `session manager` (on each `Factory` used for PDES) is used to coordinate, monitor, and control a PDES. The `session manager` also handles the task of creating local agents partitioned to execute on the `Factory`.

Agents executing on a `Factory` share a scheduling system for management of virtual timestamped events generated during PDES. Therefore, events exchanged between agents on the same WESE `Factory` never cause rollbacks. However, the agents are not coerced into synchronizing with each other. Conversely, inter-`Factory` events, that are exchanged over the network give raise to straggler events resulting in rollbacks. The Time Warp infrastructure of WESE uses traditional state saving and rollback-based mechanism to recover from causal violations. A Global Virtual Time (GVT) based approach is used for garbage collection and management of optimistic I/O streams. Further details on the various software modules constituting the frontend and backed of SEARUMS is available in the literature [21, 23].

3. BACKGROUND: EPIDEMIC MODELS

Modeling and Simulation (M&S) plays a pivotal role in epidemiological analysis, phylodynamics, bionomics, and design of prophylactic measures to contain epidemics [24, 18, 27]. Epidemiological models are classified into Equation-based models (EBMs) (also called compartmental models), Individual-based Models (IBMs), and Agent-based Models (ABMs) [24, 18, 27]. EBMs use Ordinary Differential Equations (ODEs) to model transition of population between disease states or compartments such as: Susceptible → Infected → Recovered (SIR) [3]. Unlike EBMs that model the population as homogeneous aggregate compartments, IBMs model individuals in the population to explore heterogeneity, phylodynamics, antigenic drift, and other important epidemiological phenomena. However, IBMs are stochastic models and typically do not embody geospatial characteristics of diseases [18, 27].

Agent-based Models (ABMs) further extend IBMs to provide additional details in interactions between individuals, including geospatial characteristics, enabling in depth analysis of various phenomena and prophylactic measures [18, 24]. In contrast to EBMs, ABMs are descriptive rather than prescriptive models. Consequently, ABMs are most effective in epidemiological analysis of emergent diseases whose characteristics are not well understood [18, 24]. Furthermore, unlike EBMs and IBMs, ABMs can be readily extended to embody other aspects of the system. Moreover, ABMs enable more vivid and intuitive temporo-geospatial visualization for various analysis. However, the advantages of ABMs are realized at the cost of increased simulation execution times because ABMs are computationally demanding. Nevertheless, ABMs are gaining importance because PDES methodologies coupled with proliferation of affordable supercomputing provide an effective solution to meet computational demands making ABM an attractive approach. The investigations reported in this paper focus on improving scalability

and performance of parallel simulation of spatially explicit, epidemiological ABMs.

4. MODELS IN SEARUMS

The SEARUMS models used in this study are agent-based, spatially-explicit models of global epidemiology of avian influenza. The Agents embody the classical SIR (Susceptible → Infected → Removed) mathematical models [3] and are used to describe the epidemiological behaviors of the the three salient entities, namely: waterfowl, poultry, and humans. The agents are specifically designed to ease effective use of real world statistical data. The conceptual design of each agent is based on discrete time Markov processes [23] that implement the SIR mathematical model. Figure 5 presents an overview of the Markov processes along with the SIR mathematical model constituting the three main agents in the models used in this study. Various discrete state transitions and inter-agent interactions are accomplished via virtual timestamped events.

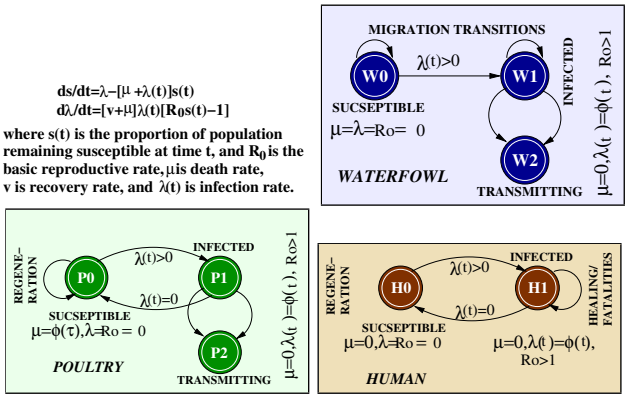


Figure 5: Overview of Susceptible → Infected → Recovered (SIR) state transitions in various agents.

The spatial interactions between agents are modeled using one or more `EcoArea` agents that represents Earth's surface. Typically, in a PDES the number of `EcoArea` agents correspond to the number of compute nodes used for simulation. The Earth's surface is evenly subdivided into non-overlapping regions and assigned to each `EcoArea`. `EcoArea`s receive updates from agents overlapping its area whenever an agent changes selected attributes, such as: current coordinate, infection percentages, and population changes. The `EcoArea` components tracks agents in its purview and uses the information to detect and trigger interactions between overlapping agents by scheduling events.

4.1 Model Generation and Validation

The models used in this study have been developed using real-world statistical data on: waterfowl migration [11], high risk waterfowl species [9], global poultry population and distribution [10], and global human population distribution information [25]. We have already published approaches for verification of the models using bioinformatics and viral isolates [20] as well as statistical analysis against temporo-geospatial data on epidemic outbreaks reported by the World Health Organization (WHO) [22]. Readers are referred to our earlier publications [23, 22, 20] for details on model generation, verification, and various analysis. However, pertinent sections of our prior publications are included in the appendix.

5. SINGLE ACTIVE PROXY (SAP)

In continuation with the foregoing discussions in Section 1.1 about the problem, the volume of Inter-Process Messages (IPMs) arising due to movements of agents had to be reduced to improve scalability and performance. Reducing IPMs requires that interacting agents must be predominantly on the same compute node – implying that as agents *logically* migrate they must be correspondingly repartitioned or relocated to a different compute node. Accordingly, the notion of *proxy* agents was introduced in the simulation. Proxy agents are agents that are automatically created for each mobile agent on each compute node used for PDES as shown in Figure 6. As the center of an agent logically moves across `EcoArea` boundaries, it logically deactivates itself after activating an appropriate proxy on a different compute node. Therefore, only one proxy-agent is active at any given time and consequently this strategy is called Single Active Proxy (SAP) approach.

An simplifed pseudo code for the SAP approach is shown in Algorithm 1. Initially, one deactivated proxy agent is automatically created on each PDES-process for every mobile agent in a given model. Deactivated agents do not per-

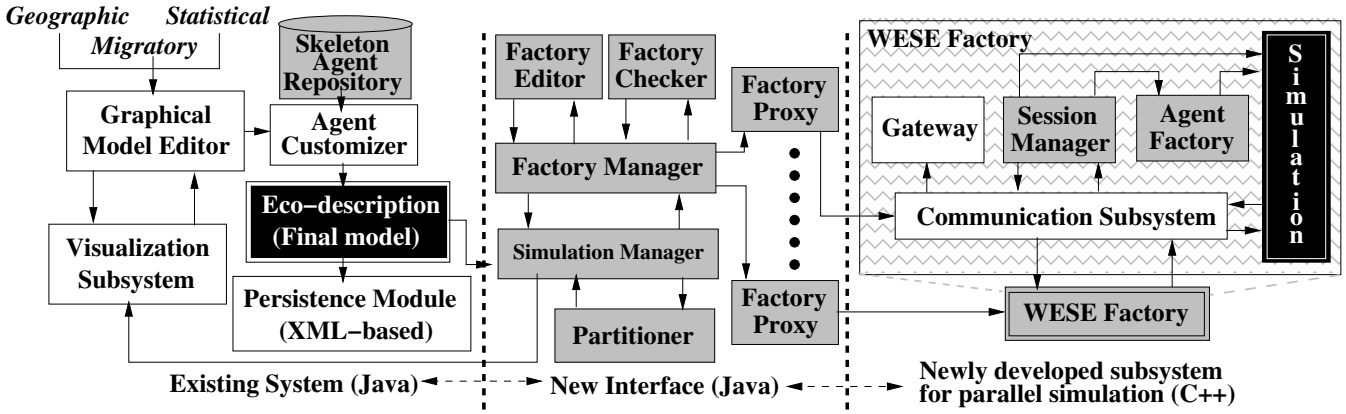


Figure 4: Architectural overview of SEARUMS showing core software subsystems.

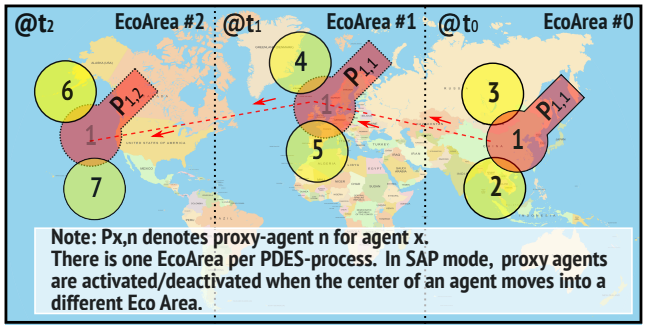


Figure 6: Overview of Single Active Proxy (SAP) approach (with 3 `EcoAreas` on 3 PDES-processes)

form any operations and remain dormant in the simulation. However, based on the initial geographical location of an agent, the appropriate proxy activates itself and performs the normal operations of the agent. When the agent migrates across its local `EcoArea`, the active proxy first deactivates itself *logically* removing itself from the simulation. The proxy then schedules an activation event (which includes the current state) to the appropriate proxy object on a different `EcoArea`. For example when agent #1 in Figure 6 moves from `EcoArea #0` to `EcoArea #1` proxy agent $P_{1,0}$ deactivates itself while activating proxy agent $P_{1,1}$. The activated proxy then resumes life cycle activities for the agent and as a consequence, the SAP approach essentially accomplishes logical (rather than physical [16]) process migration.

The dormant proxies have an extra role of forwarding events (as they may already be scheduled) to the active proxy. Activation, deactivation, and forwarding of events are performed using sub-simulation cycles. Specifically, simulation time is defined as a double precision value and fractional times are used to represent subcycles. Consequently, the overall resulting state of various agents is identical with the base case (without SAP). Moreover, tools for post-simulation analysis of simulation logs ignore the subcycles. Consequently, the filtered simulation logs used for various analysis are identical in all cases.

Algorithm 1: Single Active Proxy (SAP) Approach

```
1  begin initialization(state)
2      if inLocalEcoArea( state->latitude,
       state->longitude) then
3          state->isActive = true;
4          registerWithEcoArea();
5          scheduleLifeCycleEvents();
6      else
7          state->isActive = false;
8      end if
9  end initialization
10 begin processEvent(state, event)
11     if state->isActive then
12         proessEvent(event);
13         if ! inLocalEcoArea( state->latitude,
           state->longitude) then
14             state->activeProxy =
               unregisterFromEcoArea();
15             activateProxy(state, state->activeProxy);
16             state->isActive = false;
17         end if
18     else
19         if isProxyEvent( event) then
20             activate();
21             state->isActive = true;
22         else
23             rescheduleEvent(state->activeProxy,
               event->recvTime + DELTA);
24         end if
25     end if
26 end processEvent
```

6. BOUNDARY CASES IN SAP

The Single Active Proxy (SAP) approach provided significant improvement in scalability and performance over the base case as discussed in Section 9. However, the simulations continued to experience some performance degradation as the number of mobile agents in the model increased. Experimental analysis indicated that the SAP approach still experienced increased inter-process messages and rollbacks when an agent spans two or more partitions, as enumerated in the following three different Boundary Cases (BCs):

1. **BC #1**: The first case occurs when agents transitioning from one `EcoArea` to another across PDES-prosess boundaries as illustrated for agent #1 in Figure 7. In this transitional period only one proxy is active in SAP and has to interact with agents in two different PDES-processes resulting in increased Inter-Process Messages (IPMs) and degrades performance in models with many mobile agents.
2. **BC #2**: The second case occurs when a mobile agent is resting but happens to spans across two `EcoAreas` as illustrated by the set of agents {#6, #7, #8} in Figure 7. Since agents span two or more `EcoAreas`, it gives raise to copious amounts of Inter-Process Messages (IPMs) which results in degraded performance. The frequency of this scenario steadily increases as the number of `EcoAreas` or partitions increases. This issue conspicuously manifests itself even without SAP (as expected) as highlighted by the chart in Figure 3(a).
3. **BC #3**: The third BC occurs when an agent's movement oscillates between two or more partitions, similar to the pathway of agent #9 shown in Figure 7. In this scenario, proxy agents often experience cascading rollbacks which degrades performance.

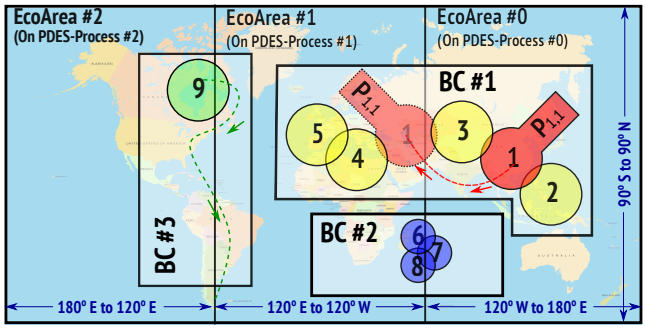


Figure 7: Illustration of three Boundary Cases (BCs) that impact SAP approach

7. MULTIPLE ACTIVE PROXY (MAP)

The Multiple Active Proxy (MAP) approach is proposed to extend SAP approach and address the three boundary cases (BCs) discussed in Section 6 to provide a more comprehensive solution. The MAP approach extends the SAP approach by permitting multiple proxies to be active at the boundary cases when an agent spans two or more partitions. Recollect that an agent has one proxy preallocated on each `EcoArea` (*i.e.,* PDES-process). Consequently, in MAP approach two or more proxies can be active when boundary cases arise. The pseudo code in Algorithm 2 provides additional details about the MAP implementation.

In MAP approach, each active proxy "ghosts" operations of related proxies and handles interactions with other agents on its local partition, thereby eliminating Inter-Process Messages (SAP) that occur at boundary cases in SAP mode. Moreover, each proxy agent also performs various life cycle tasks by scheduling events to itself. Ghosting of life cycle tasks by active procexies does increase the net number of events in the simulation. However, these events are local events that do not significantly degrade performance.

At the end of handling local interactions, the active proxies exchange messages (which are inter-process messages) to consistently update the states of all active proxies. Once the agent no longer spans multiple partitions, relevant proxies deactive themselves eliminating "ghosting" and the simulation continues to proceed in SAP mode. MAP minimizes the number of Inter-Process Messages (IPMs) at boundary cases thereby improving upon the efficiency gained by utilizing SAP. In this context, it must be noted that the final resulting state in both MAP and SAP approaches is the same as that of the original agent's state. Consequently, the results from the simulations used for various analysis are identical in all cases.

8. RELATED RESEARCH

The proposed research is a novel integration several major topics in of Modeling and Simulation (M&S), including: agent-based epidemic modeling, spatially-explicit models, and optimistic PDES. Several investigations have been reported on these topics and this section compares the proposed research with some of the closely related investigations. Readers are referred to the references and literature for a more comprehensive survey of related works [23, 22].

Agent-based models (ABMs) have been employed for large-scale epidemiological analyses for various diseases [6, 15].

Algorithm 2: Multiple Active Proxy (MAP) Approach

```
1  begin initialization(state)
2      ecoAreas=ecoAreas(state->latitude,
       state->longitude);
3      if localEcoArea ∈ ecoAreas then
4          state->isActive = true;
5          registerWithEcoArea();
6          scheduleLifeCycleEvents();
7      else
8          state->isActive = false;
9      end if
10 end initialization
11 begin processEvent(state, event)
12     if state->isActive then
13         prevEcoAreas=ecoAreas(state->latitude,
           state->longitude);
14         proessEvent(event);
15         if ! inLocalEcoArea(state->latitude,
           state->longitude) then
16             state->activeProxy =
               unregisterFromEcoArea();
17             state->isActive = false;
18         else
19             newEcoAreas=ecoAreas(state->latitude,
               state->longitude) - prevEcoAreas;
20             if newEcoAreas != φ then
21                 activeProxies(newEcoAreas);
22             end if
23         end if
           // Synchronize active proxy states
24         activeProxyStateCoherence();
25     else
26         if isProxyEvent(event) then
27             activate();
28             state->isactive = true;
29         else
30             rescheduleEvent(state->activeProxy,
               event->recvTime + DELTA);
31         end if
32     end if
33 end processEvent
```

Parker and Epstien [15] discusses the issues involved in design and development of a Java-based, Global-Scale Agent Model (GSAM) distributed platform for epidemiological modeling using over 6 billion interacting agents. Similar to GSAM, the proposed research avoids "physical" process migration but unlike GSAM the proposed research involves "logical" process migration using proxy agents. However, in contrast to agents in GSAM, the mobile agents in our model explicitly embody migration and inter-agent interactions without relying on contact matrices or contact networks [15]. Moreover, the agents in our research are different in that they do not represent a single entity but a collection of collocated entities, striking a better balance between model resolution and execution costs. Our modeling approach is useful for M&S of 21 billion migrating waterfowl using surveillance and satellite telemetry data for that is available only aggregate forms [11].

The use of ABMs with explicit agent mobility distinguishes the proposed research from work from those reported by Barrett *et al* [1], Bisset *et al* [2], and Perumalla *et al* [17]. These three investigations use a reaction-diffusion approach involving contact graphs for epidemic modeling. The use of contact graphs or social networks for epidemic modeling has been discussed by other researchers as well [8, 14, 12]. Although, the migratory flyways for waterfowl are specified in our models, the contacts between agents are not preordained but discovered based on probabilistic movements of agents. However, similar to these three investigations we aim to utilize PDES to accelerate performance of epidemic simulations.

The proposed Single Active Proxy (SAP) and Multiple Active Proxies (MAP) approach discussed in Section 5 and Section 7 essentially accomplishes logical process migrations in contrast to physical process migration that has been extensively investigated in conservative as well as optimistic PDES [13, 16]. However, unlike the application-specific, logical migration accomplished by SAP, almost all of the efforts reported in the literature focus on providing a generic infrastructure for physical process migration. Consequently, the earlier investigations involve extensions to the underlying simulation kernel. On the other hand, the proposed investigations focus on model-level extensions that can be applied to both conservative and optimistic PDES. Nevertheless, both of these approaches are distantly related to "ghosting" in multi-resolution simulations in which multiple representations of entities are maintained [26]. However, MAP approach is different then concept of "ghosting" in that the agents are not at different resolutions and in fact strive to be identical copies of each other.

9. EXPERIMENTS

The experiments conducted to evaluate the effectiveness of the Single Active Proxy (SAP) and Multiple Active Proxies (MAP) mode were performed using four different models. Table 1 shows the number of agents constituting each models. `Waterfowl` agents indicate the number of mobile entities while `Poultry` and `Human` agents are stationary. The migratory flyways of the waterfowl and their population has been generated from GROMS database [11] [22, 11] and validated using statistical and bioinformatics analysis [22, 20].

The models `M1`, `M2`, and `M3` have a varying number of humans and poultry agents at different scales of detail. Agent data at were generated from poultry and human Geographic Information Systems (GIS) data obtained from NASA's SEDAC database [25]. Unlike the first three models in Table 1, model `M4` serves as a stress-test case in which all agents are mobile.

All the experiments were conducted on a distributed memory super computing cluster running Red Hat Enterprise Linux (RHEL 6.5). Each compute node had two hex-core Intel Xeon X5650 CPUs @ 2.67 GHz (with 12 MB L2 cache) yielding 5333 bogomips on each of the 12 cores. Each compute node had 48 GB of RAM averaging to 4 GB per core. The nodes were interconnected using QDR infiniband interconnect. Note that the experiments were performed using the non-GUI mode supported by SEARUMS to avoid GUI overheads. Furthermore, the Java frontend was executed on a separate compute node and model-logs were turned off to eliminate any resource contentions or I/O overheads for all performance tests reported in this section.

Table 1: Characteristics of models used for conducting the experiments

ID	Number of Agents			Total
	Waterfowl (mobile)	Poultry (stationary)	Humans (stationary)	
M1	44	1314	1314	2672
M2	44	4251	2160	6455
M3	44	4763	5269	10076
M4	3088	0	0	3088

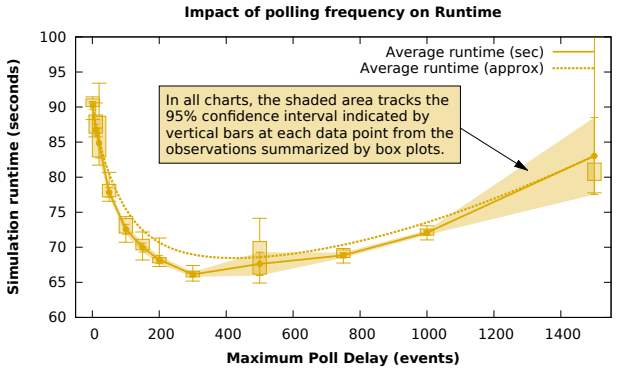
9.1 Calibration

The objective of the first phase of experimentation was to identify the optimal simulation configuration for the base case with did not utilize the SAP or MAP optimization. The objective was to identify suitable settings to obtain the best possible performance for base case comparisons. The medium sized model `M2` was used for these experiments. For all of the tests the surface of Earth was vertically partitioned because most migratory flyways are in North-South direction. Consequently, vertical partitioning minimizes inter-process messages even in the base case. Effects of different strategies for partitioning this model have been reported in our earlier publication [21]. The number of `EcoArea`s correspond to the number of processes used for PDES with one `EcoArea` assigned to each PDES-process. Furthermore, each `EcoArea` handles agents in an evenly divided, non-overlapping vertical region of Earth's surface.

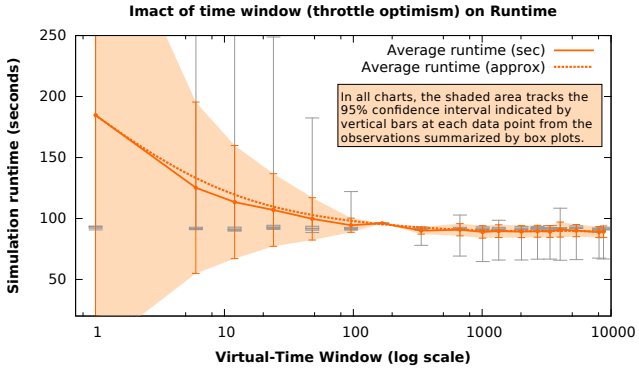
The charts in Figure 8 illustrate the effect of three Time Warp parameters that influence overall performance of the simulations [21]. The calibration experiments enable identification of suitable settings for these three parameters for a given computational platform and minimize noise in the experimental data. The graph in Figure 8(a) illustrates the effect of changing the maximum delay between successive checks for Inter-process Messages (IPM) arriving over the network. Consistent with expectations, neither small nor large settings are effective with ideal value lying at the inflection point of the characteristic curve.

The need for controlling optimism is pervasive to spatially-explicit, optimistic simulations running on both shared and distributed memory architectures [4, 7, 21]. The chart in Figure 8(b) illustrates the effect of varying the time window used for throttling. Small time windows curtail optimism too aggressively and consequently the PDES cannot use available optimism effectively. Consequently, a significant variation in runtimes is observed because the simulation becomes network-bound, waiting for messages to arrive. On the other hand, when the time window is large the simulation experiences more overheads due to rollbacks which degrade performance. Accordingly, an intermediate value of 144 hours (in simulation-time units) corresponding to the minimum variance point has been used.

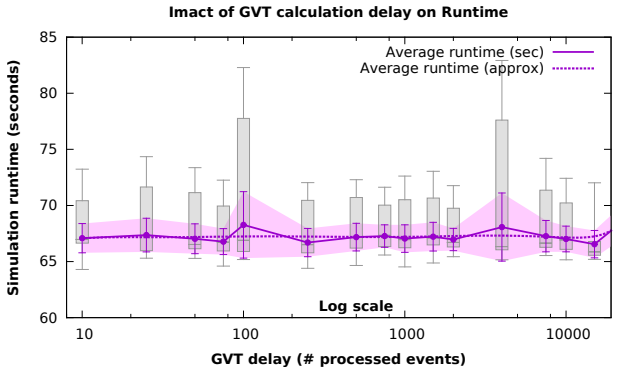
The Global Virtual Time (GVT) computation delay parameter had the least impact on the simulation as shown in Figure 8(c). The impact of GVT period is not pronounced because the compute nodes have large caches and RAM. However, as expected, the variance in timings increases for small and large GVT values. Therefore a intermediate value of 4000 events has been used for experimentation. The charts in Figure 8 indicate that the PDES is most sensitive to polling delay and least sensitive to GVT delay.



(a) Calibration of Polling delay



(b) Calibration of Time Window



(c) Calibration of GVT duration

Figure 8: Experimental calibration (*without proposed solution*) of influential parameters to identify optimal settings for base case measurements. The tests were conducted using 12 PDES-processes (on 4 compute nodes, 3 processes per node)

9.2 Experimental Evaluation of SAP & MAP

The influential parameter settings identified via the calibration discussed in Section 9.1 were used to conduct all the experiments for evaluating the proposed Single Active Proxy (SAP) and Multiple Active Proxy (MAP) modes. The observations collated from experiments conducted using a range of configurations with for different models (see Table 1) are shown by the charts in Figure 9 and Figure 10. The x-axis of all the charts corresponds to the number of parallel PDES-processes (*i.e.,* Linux processes), with each process running on an independent core. The solid lines on each one of the charts tracks average values obtained from 10 runs. The corresponding dotted lines show the approximate curve fitting for the observations to emphasize general trends in the data due to the occurrence of Boundary Cases (BCs) as discussed in Section 6. The error bar at each of the data points indicates the 95% confidence interval computed from the 10 runs and the lightly shaded region tracks the confidence interval to highlight statistical significance of the observations.

In all the charts in Figure 9 and Figure 10, the `base case` configuration corresponds to simulations conducted without the use of proxies. However, the base case curves are not included in the charts for model `M4` because the configuration without proxies was practically unusable due to long simulation execution times (over 24 hours with over 32 cores) and the experiments had to be abandoned. The long simulation execution times arise because of the large number of mobile agents in the model. Note that, although model `M4` has fewer total number of agents than even `M2` model, all of its agents are mobile waterfowl agents. Consequently, model `M4` is a "worst case" model for simulating with the base case configuration (*i.e.,* without proxies) resulting in long execution times.

The charts for model `M1` in Figure 9 already highlight the effectiveness of the Single Active Proxy (SAP) and Multiple Active Proxies (MAP) approaches. However, for `M1` both the SAP and MAP approaches are pretty close in runtime for most configurations, with the MAP approach performing about 10% to 15% better. The small difference is expected because the MAP approach operates as SAP except for boundary cases. Furthermore, the `M1` model is the smallest and lacks sufficient workload to effectively utilize the available compute power. Consequently, even with SAP and MAP the number of rollbacks offset advantages of PDES and the time for simulation actually increases.

The graphs for model `M2` in Figure 9 and model `M3` in Figure 10 illustrate the advantages of SAP and MAP approach. Furthermore, the advantages of MAP is a bit more prominent in these two models. The SAP approach consistently outperforms the the base case by about 2x while the MAP approach provides another 15% to 25% performance boost on top of SAP. In these two models the overall effect of MAP is still muted because the models do not have many mobile agents in them. Nevertheless, the advantages of MAP are a bit more prominent as the number of PDES-processes (synonymous to number of cores used for PDES) are increased. The MAP approach provides much better scalability that SAP for both `M2` and `M3` models. On the other hand, the SAP approach starts experiencing degradation in scalability around 20 cores and 40 cores for models `M2` and `M4` respectively. The degradation in scalability occurs sooner for `M2` than for `M3` because `M2` is a smaller model with less workload and with increased availability of compute power the model optimistically advances only to suffer from increased rollbacks, as evidenced by the charts in Figure 9.

The advantages of MAP approach is more pronounced in the case of model `M4` that has a large number of mobile agents. The abundance of mobile agents increases the chances of Boundary Cases (BCs) discussed in Section 6 in SAP as the number of PDES-processes are increased. The approximated runtime curve for `M4` shown in Figure 10 shows that initially the runtime decreases with increase in PDES-processes demonstrating good scalability. However, the scal-

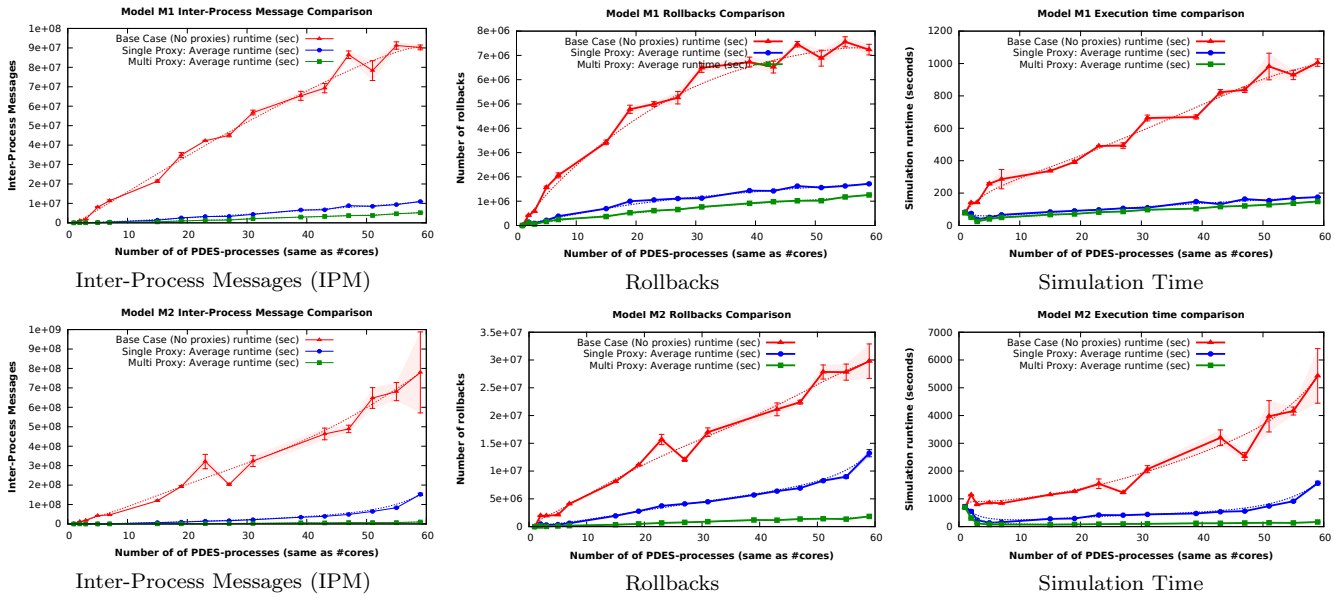


Figure 9: Comparison of Inter-Process Messages (IPM), rollbacks, and simulation execution time for the models `M1` and `M2` shown in Table 1 in the following three configurations: no proxies (base case), Single Active Proxy (SAP) approach, and Multiple Active Proxies (MAP) approach. Note that number of PDES-processes is the same as the number of CPU-cores used for simulation (one PDES-process per CPU-core).

ability tappers off around 40 cores and further increase in PDES-processes worsens the runtime. However, the MAP approach continues to provide much better scalability and about 20% to 25% performance improvement in the configurations with more than 25 PDES processes. Reprising the discussion from earlier paragraph, the base case data for model `M4` are not shown in in Figure 10 as the experiments were abandoned due to extremely long simulation times that exceeded 24 hours in several configurations.

The experimental results shown in Figure 9 and Figure 10 also illustrate the strong correlation between Inter-Process Messages (IPMs), the number of rollbacks, and simulation execution time. Increase in rollbacks transitively degrades scalability of the PDES. However, the SAP approach is effective in eliminating a large fraction of IMPs by essentially accomplishing logical process migration using a set of proxies. Furthermore, the MAP approach improves upon SAP by further reducing IPMs and consequently providing more reliable scalability.

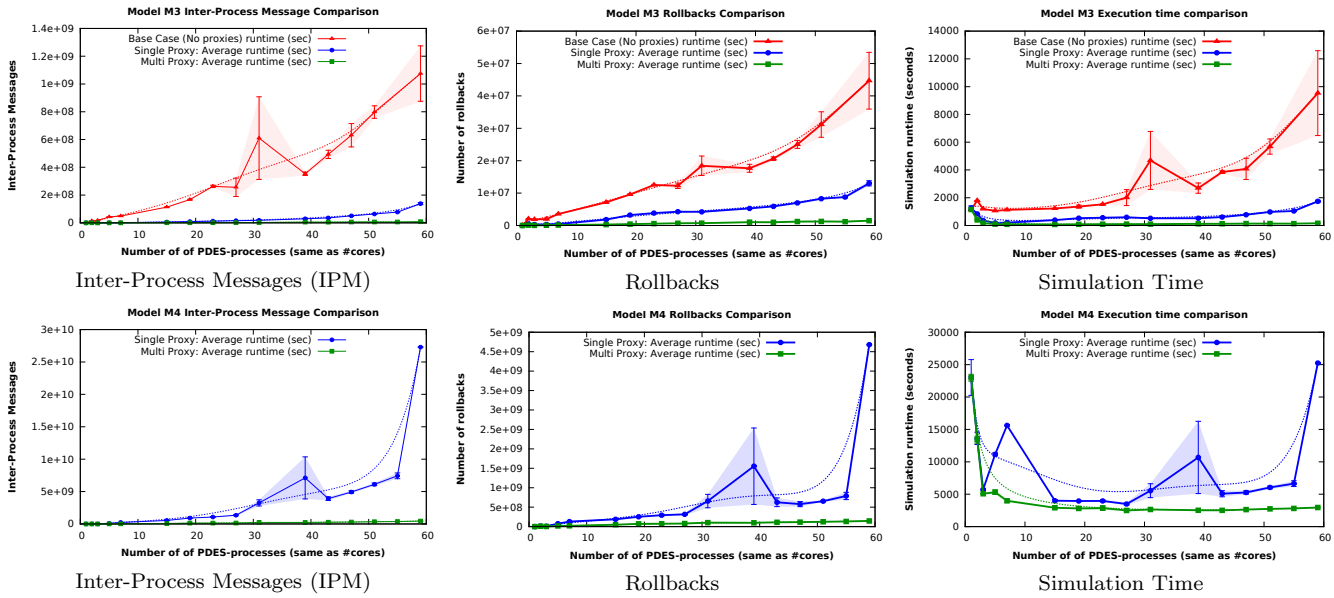
10. CONCLUSIONS

The application of simulation-based analysis using spatially-explicit, agent-based models is gaining momentum for epidemiological analysis of emergent diseases such as avian influenza [24, 18, 27]. We have developed a modeling, parallel simulation, and analysis environment called SEARUMS to meet the computational demands of agent-based models. In SEARUMS, Parallel Discrete Event Simulation (PDES) is accomplished by partitioning various agents and the geospatial components used to coordinate interactions between them. The PDES uses Time Warp algorithm for optimistic synchronization to effectively utilize inherent parallelism in the model [21] and operated well for models with many stationary agents but few mobile agents.

However, the optimistically synchronized PDES backend of SEARUMS experienced performance degradation when the number of mobile agents in the model were increased. The investigation reported in this paper focused on identifying and addressing the scalability and ensuing performance degradation. Exploratory investigations (presented in Section 1.1) identified that the large volume of Inter-Process Messages (IPMs) exchanged over the network was magnifying synchronization issues resulting in degraded scalability. The IPMs arise because as the mobile agents *logically* move they have to interact with agents on a different compute node causing increase in volume of IPMs.

The investigations proposed and assessed the effectiveness of using *proxy* agents on each PDES-process for each agent. Specifically, as an agent migrates across process boundaries, a proxy deactivates while activating another proxy on a different process. Initially, only one proxy was permitted to be active at any given time resulting in a Single Active Proxy (SAP) approach. The SAP approach provided significant performance improvements as indicated by the experimental results discussed in Section 9. However, the SAP approach did not address bottlenecks arising due to three different Boundary Cases (BCs) as discussed in Section 6. Consequently, a Multiple Active Proxy (MAP) approach was proposed and assessed.

The Multiple Active Proxy (MAP) approach extends SAP approach and permits multiple proxy agents to be active only in boundary case scenarios while operating in SAP mode otherwise. The MAP mode essentially permits "ghosting" of an agent where multiple active proxies handle local interactions occurring on the same PDES-process and using Inter-Process Messages (IPMS) to maintain consistent states. Experimental evaluation of the MAP approach discussed in Section 9 indicates that the MAP approach pro-



Inter-Process Messages (IPM) | Rollbacks | Simulation Time

Inter-Process Messages (IPM) | Rollbacks | Simulation Time

Figure 10: Comparison of Inter-Process Messages (IPM), rollbacks, and simulation execution time for the models `M3` and `M4` shown in Table 1 in the following three configurations: no proxies (base case), Single Active Proxy (SAP) approach, and Multiple Active Proxies (MAP) approach. Note that number of PDES-processes is the same as the number of CPU-cores used for simulation (one PDES-process per CPU-core).

vides more reliable scalability along with a modest performance boost of 15% to 25% on top of SAP.

The SAP and MAP approaches are implemented at the model-level using the default infrastructure provided by the PDES kernel. Since the proposed solutions do not depend on any special support from the underlying PDES kernel, they can be readily implemented in any infrastructure, including conservatively synchronized parallel simulations. Nevertheless, the MAP approach involving "ghosting" can be incorporated as general-purpose infrastructure in PDES kernels. The optimized PDES significantly reduces simulation execution times, easing exploratory analyses, and highlights the current and future importance of parallel simulations in epidemiology, bionomics, and related fields.

11. REFERENCES

[1] C. L. Barrett, K. R. Bisset, S. G. Eubank, X. Feng, and M. V. Marathe. Episimdemics: An efficient algorithm for simulating the spread of infectious disease over large realistic social networks. In *Proceedings of the 2008 ACM/IEEE Conference on Supercomputing*, SC '08, pages 37:1–37:12, Piscataway, NJ, USA, 2008. IEEE Press.

[2] K. R. Bisset, J. Chen, X. Feng, V. A. Kumar, and M. V. Marathe. Epifast: A fast algorithm for large scale realistic epidemic simulations on distributed memory systems. In *Proceedings of the 23rd International Conference on Supercomputing*, ICS '09, pages 430–439, New York, NY, USA, 2009. ACM.

[3] F. Brauer and C. Castillo-Chavez. *Mathematical Models for Communicable Diseases*. SIAM, 3600 Market Street, Philadelphia, PA 19104-2688, USA, 2013.

[4] E. Deelman and B. K. Szymanski. Simulating spatially explicit problems on high performance architectures. *Journal of Parallel and Distributed Computing*, 62(3):446–467, Mar. 2002.

[5] T. Dickman, S. Gupta, and P. A. Wilsey. Event pool structures for pdes on many-core beowulf clusters. In *Proceedings of the 2013 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation*, SIGSIM-PADS '13, pages 103–114, New York, NY, USA, 2013. ACM.

[6] J. M. Epstein. Modelling to contain pandemics. *Nature*, 460:687–687, 2009.

[7] S. Eubank. Scalable, efficient epidemiological simulation. In *Proceedings of the 2002 ACM symposium on Applied computing*, pages 139–145, Mar. 2002.

[8] N. M. Ferguson, D. A. T. Cummings, C. Fraser1, J. C. Cajka, P. C. Cooley, and D. S. Burke. Strategies for mitigating an influenza pandemic. *Nature*, 442:448–452, 2006.

[9] M. Gilbert, X. Xiao, J. Domenech, J. Lubroth, V. Martin, and J. Slingenbergh. Anatidae migration in the western palearctic and spread of highly pathogenic avian influenza H5N1 virus. *Emerging Infectious Diseases*, 12(11), 2006.

[10] GLiPHA. Global Livestock Production and Health Atlas (GLiPHA): Animal Production and Health Division of Food and Agriculture Organization of the United Nations, 2014.

[11] GROMS. Global Register of Migratory Species (GROMS): Summarising Knowledge about Migratory Species for Conservation, Jul 2013.

[12] M. E. Halloran, N. M. Ferguson, S. Eubank, J. Ira M. Longini, D. A. T. Cummings, B. Lewis, S. Xu, C. FraserÂğ, A. Vullikanti, T. C. Germann, D. Wagener, R. Beckman, K. Kadau, C. Barrett,

C. A. Macken, D. S. Burke, and P. Cooley. Modeling targeted layered containment of an influenza pandemic in the united states. *Proceedings of the National Academy of Sciences of the United States of America*, 105(12):4639–4644, Mar. 2008.

[13] S. Jafer, Q. Liu, and G. Wainer. Synchronization methods in parallel and distributed discrete-event simulation. *Simulation Modelling Practice and Theory*, 30(0):54 – 73, 2013.

[14] I. M. Longini, A. Nizam, S. Xu, K. Ungchusak, W. Hanshaoworakul, D. A. T. Cummunings, and M. E. Halloran. Containing pandemic influenza at the source. *Sience*, 309(5737):1083–1087, 2005.

[15] J. Parker and J. M. Epstein. A distributed platform for global-scale agent-based models of disease transmission. *ACM Trans. Model. Comput. Simul.*, 22(1):2:1–2:25, Dec. 2011.

[16] S. Peluso, D. Didona, and F. Quaglia. Supports for transparent object-migration in pdes systems. *Journal of Simulation*, 6:279–293, 2012.

[17] K. S. Perumalla and S. K. Seal. Discrete event modeling and massively parallel execution of epidemic outbreak phenomena. *SIMULATION*, 2012.

[18] L. L. Pullum and O. Ozmen. Early results from metamorphic testing of epidemiological models. In *BioMedical Computing (BioMedCom), 2012 ASE/IEEE International Conference on*, pages 62–67, 2012.

[19] D. M. Rao. *Study of Dynamic Component Substitution*. PhD thesis, University of Cincinnati, 2003.

[20] D. M. Rao. Enhancing temporo-geospatial epidemiological analysis of h5n1 influenza using phylogeography. In *Proceedings of the Great Lakes Bioinformatics Conference 2014 (GLBIO'14)*, University of Cincinnati, Ohio, USA, May 2014. International Society for Computational Biology (ISCB). (submitted).

[21] D. M. Rao and A. Chernyakhovsky. Parallel simulation of the global epidemiology of avian influenza. In *Proceedings of the 2008 Winter Simulation Conference*, pages 1583–1591, Dec. 2008.

[22] D. M. Rao and A. Chernyakhovsky. Automatic generation of global agent-based model of migratory waterfowl for epidemiological analysis. In *Proceedings of the 27th European Simulation and Modelling Conference (ESM'2013)*, Lancaster University, Lancaster, UK, oct 2013. EuroSis. Best paper award.

[23] D. M. Rao, A. Chernyakhovsky, and V. Rao. Modeling and analysis of global epidemiology of avian influenza. *Environmental Modelling & Software*, 24(1):124–134, jan 2009.

[24] B. Roche, J. Drake, and P. Rohani. An agent-based model to study the epidemiological and evolutionary dynamics of influenza viruses. *BMC Bioinformatics*, 12(1):87, 2011.

[25] SEDAC. SocioEconomic Data and Applications Center (SEDAC): Gridded Population of the World, Oct 2014.

[26] A. Tolk. *Engineering Principles of Combat Modeling and Distributed Simulation*. Wiley, 2012.

[27] E. M. Volz, K. Koelle, and T. Bedford. Viral phylodynamics. *PLoS Computational Biology*, 9(3):e1002947, 2013.

[28] WHO. Influenza (seasonal) fact sheet, Feb. 2014. Citations for 90 million annual infections and 500,000 annual deaths.