EpiSimdemics: an efficient algorithm for simulating the spread of infectious disease over large realistic social...

Xizhou Feng

Supercomputing Conference

EpiSimdemics: an Efficient Algorithm for Simulating the Spread of Infectious Disease over Large Realistic Social Networks

Christopher L. Barrett, Keith R. Bisset, Stephen G. Eubank, Xizhou Feng, Madhav V. Marathe
Network Dynamics and Simulation Science Laboratory, Virginia Tech, Blacksburg, VA 24061
{cbarrett, kbisset, seubank, fengx, mmarathe}@vbi.vt.edu

***Abstract*—Preventing and controlling outbreaks of infectious diseases such as pandemic influenza is a top public health priority. We describe EpiSimdemics – a scalable parallel algorithm to simulate the spread of contagion in large, realistic social contact networks using individual-based models. EpiSimdemics is an interaction-based simulation of a certain class of stochastic reaction-diffusion processes. Straightforward simulations of such process do not scale well, limiting the use of individual-based models to very small populations. EpiSimdemics is specifically designed to scale to social networks with 100 million individuals. The scaling is obtained by exploiting the semantics of disease evolution and disease propagation in large networks. We evaluate an MPI-based parallel implementation of EpiSimdemics on a mid-sized HPC system, demonstrating that EpiSimdemics scales well. EpiSimdemics has been used in numerous sponsor defined case studies targeted at policy planning and course of action analysis, demonstrating the usefulness of EpiSimdemics in practical situations.**

I. Introduction

Pandemic diseases such as avian influenza are extremely serious threats to global public health security. Pandemic influenza strains have demonstrated their ability to spread worldwide quickly and to cause infections in all age groups. According to the World Health Report 2007, every year human influenza rapidly spreads around the world within weeks, resulting in an estimated three to five million cases of severe illness and between 250,000 and 500,000 deaths [1]. Although, the final number of infections, illnesses, and deaths is unpredictable, and could vary tremendously depending on multiple factors, it is certain that without adequate planning and preparations, an influenza pandemic in the 21st century has the potential to cause enough illnesses to overwhelm the public health system at all levels. Certain modern trends will likely exacerbate the situation and further underscore the need for developing appropriate technologies to support public health preparedness. These include: (i) a larger global population and increased urbanization leading to a higher density of individuals within cities; (ii) higher levels of long distance travel, including international travel; (iii) increased numbers of elderly individuals and individuals with chronic medical conditions.

Broadly speaking, computational epidemiology is defined as the development and use of computer models to understand the spatio-temporal diffusion of disease through populations. Until recently, aggregate, or collective, computational epidemiology models were used extensively. These models usually assume that a population is partitioned into a few subpopulations (e.g., by age) with a regular interaction structure within and between subpopulations. Although useful for obtaining analytical expressions for a number of interesting parameters such as the total numbers of infections, these models can neither capture the complexity of human interactions that serve as a major transmission mechanism for infectious disease, nor provide any causal explanation. Additionally, the number of different subpopulation types considered is small and parameters such as mixing rate and reproductive number are either unknown or hard to observe.

Over the past several years, large-scale, individual-based, disaggregate models have been studied [2]–[5]. These new models use an endogenous representation of individual agents together with explicit interactions between these agents to generate and capture the disease spread across social networks. Developing HPC-based modeling approaches to support individual-based models is challenging for a number of reasons. First, the underlying social contact network is extremely large, irregular and dynamic. This lack of symmetry and constantly changing structure rule out the possibility of using model reduction techniques used often for physical systems. Second, due to the stochastic nature of our models, a typical experimental design used to support a public policy question requires a large number of runs. The size of the design often implies that we need computational steering of experiments. Third, unlike physical systems, the diversity amongst agents is crucial in understanding the spatial and temporal spread of the disease.

A. Summary of Contributions and Significance.

In this paper, we introduce *EpiSimdemics* – a new parallel algorithm for simulating the spread of infectious diseases in large realistic social contact networks. We believe that com-

This work has been partially supported NSF Nets GrantCNS-062694, HSD Grant SES-0729441, CDC Center of Excellence inPublic Health Informatics Grant 2506055-01, NIH-NIGMS MIDAS project5 U01 GM070694-05, and DTRA CNIMS Grant HDTRA1-07-C-0113.Computational support for the work was provided in part by theNational Science Foundation through TeraGrid resources provided byNCSA, TACC, and PSC.

SC2008 November 2008, Austin, Texas, USA 978-1-4244-2835-9/08 $25.00 ©2008 IEEE

putational epidemiology represents a new application domain for high performance computing. EpiSimdemics is one of the first parallel algorithms (and implementation) with good performance for simulating epidemics on *large & realistic* social contact networks. It is also one of the few algorithms that can be mapped efficiently onto a distributed memory cluster. The major design goal of EpiSimdemics is to explore the effects of complex pharmaceutical interventions (PIs) and non-pharmaceutical interventions (NPIs) on the spread of infectious disease through realistic populations.

We briefly describe our methodology to generate realistic high resolution social contact networks; see [3], [6]–[9] for more details. Our synthetic population is derived and statistically indistinguishable from the United States Census. Each individual in the population is unique, and is tagged with up to 163 demographic variables from the census. Housing units are individually represented, and are obtained by integrating the Census and NAVTEQ street data. Non-housing locations are taken from a commercial database provided by Dun and Bradstreet, which represents most of the business locations in the United States. School data, including locations and enrollment, is taken from the Digest of Education Statistics provided by the National Center for Education Statistics. Each individual has a daily activity list with specific locations, generated from the National Household Transportation Survey. These data sets are combined using integrated statistical and machine learning methods to produce an input data set of 100GB in size for the United States. This effectively represents a labeled social contact network, discussed in the next section.

Recently, a number of other researchers have also proposed the use of realistic social contact networks in studying the spread of contagion [10]–[12]. To the best of our knowledge, these networks do not have the same level of resolution as the social network that we construct. The role of social networks in epidemic simulations is twofold. First, the structure of the social contact networks greatly influences how disease spreads on these networks; see [6], [13], [14] for more discussion. Second, the network structure plays an important role in determining the efficiency of the parallel simulations by affecting the inter-processor communication patterns of a parallel algorithm and its implementation.

EpiSimdemics, in conjunction with an appropriate decision support environment, provides public health officials new ways to understand epidemics and for formulating and analyzing public policies. It supports a wide range of disease models, intervention strategies, agent diversities, and dynamic agent behavior modeling. Each of these aspects are crucial when using EpiSimdemics in practical settings.

We use a variant of finite state machines, called probabilistic timed transition systems (PTTSs) to represent the within-host disease progression. The model is general enough to represent various types of airborne diseases. In the past, we have modeled influenza, smallpox and SARS. Our ongoing work [6] seeks to extend the model to vector borne diseases such as malaria, as well as other reaction diffusion processes such as the spread of social norms and fads [15]. Another important feature is its ability to represent complex interventions. An illustrative example of this is a *triggered* PI, the administration of vaccines to all school children when 5% of school children report sick. This intervention is not static – the exact time when it will be enacted depends on how the disease propagates. Another consideration, individual behavior such as *primary school closures* is an important NPI. A realistic representation of school closure requires modification of the activity patterns (behaviors) not only of school children but also of their caregivers. Representing such interventions using aggregate models or even percolation-based models is usually not possible.

We have performed extensive experimental analysis of EpiSimdemics. Our results indicate that EpiSimdemics achieves linear speedup (both in terms of strong scaling and weak scaling) on several mid-sized HPC platforms and instances that we have studied so far. For example, in our performance test on a commodity Linux cluster , EpiSimdemics achieves over 309 times speedup on 320 PEs (relative to the execution time on 32 processors) for the California network with roughly 39 million nodes and 182 million edges. This scaling was obtained by developing a number of new algorithmic ideas, and their efficient implementations, which exploit the specific structure of the contagion processes associated with infectious diseases. This allows us to decouple the temporal and spatial interdependencies between network nodes leading to improved parallelism. Two important technical ideas to consider are: (i) use of generic individual models combined with context-based attribute evaluation for capturing the heterogeneity amongst agents, (ii) implicit representation of the social contact network, combined with differences in the rate of disease evolution and network evolution, to obtain highly parallel implementations. These ideas will be discussed later in the paper.

EpiSimdemics has been extensively validated and used in numerous real studies for DHS, DoD, and DHHS, among others. These studies have led to novel scientific findings and insights in the prevention and response to disease outbreaks.

In the next section, we provide a formal abstraction for the problem of simulating the process of disease spread across social contact networks. Then, in Section II, we describe the EpiSimdemics algorithm and prove its correctness. We present the parallel implementation in Section III, followed by performance analysis in Section IV and an overview of its applications in several real studies in Section V. In Section VI, we provide a brief description of the work related to EpiSimdemics. Finally, we summarize the findings and conclusions of our work.

B. The Formal Model

We will use a discrete dynamical system framework as a formal framework to describe the system [6], [8]. The basic framework consists of the following components: (i) a collection of entities with state values and local rules for state transitions, (ii) an interaction graph capturing the local dependency of an entity on its neighboring entities, and (iii)

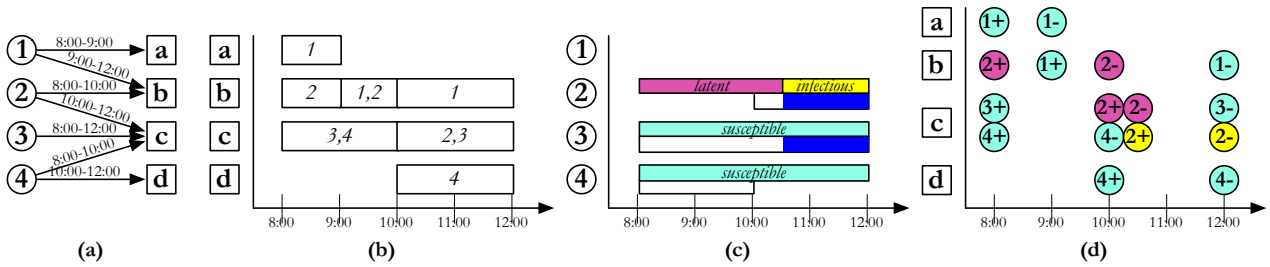

Fig. 1. An example social contact network: (a) the bipartite graph representation G_{PL} with people 1-4 and locations a-d; (b) the temporal and spatial projections G_P, i.e., temporal people-to-people contacts; (c) the occupancy of location c overlaid with the health states of its occupants. The dark blue area shows the time of possible disease transmission. (d) The event based representation of the information ($+$ for arrival, $-$ for departure). The color of the events represents the health state of the associated individuals, and correspond to those in Figure 2. A change in health state is modeled as a departure, followed by an arrival with the new health state.

an update sequence or schedule such that the causality in the system is represented by the composition of local mappings. We can formalize this such that a Networked Discrete Dynamical System (NDDS) $\mathcal{S}$ over a given domain $\mathcal{D}$ of state values is a triple $(G(V,E),\mathcal{F},W)$, whose components are as follows: $G(V,E)$ is the basic underlying social network; V is a set of vertices. For each vertex v_i there is also a local state transition function; the set of these functions is denoted by $\mathcal{F}$. The function f_{v_i} at a vertex v_i is used to map the state of vertex v_i at time t to its state at time $t+1$, and the input to this function is the state sub-configuration induced by the vertex and its neighbors in the contact network at time t, denoted by $N(i,t)$. The final component is a string W over the alphabet $\{v_1, v_2, \ldots, v_n\}$. The string W is a schedule. It represents an order in which the state of a vertex or the possible edges incident on the vertex will be updated. f_{v_i} is usually stochastic and also depends on time. As detailed below, we use a probabilistic timed transition system (PTTS) to represent the local function. The formalism serves as a mathematical abstraction on which EpiSimdemics is based. From a modeling perspective, each vertex represents an agent. Here, we will assume that the states of the agent come from a finite domain $\mathbb{D}$. The maps f_{v_i} are generally stochastic. A step of an NDDS (i.e., the transition from one configuration to another), involves updating the state associated with a subset of vertices based on the schedule W.

Computationally, the functions f_i in our NDDS are used to decide if the disease will be transmitted between two individuals. In the following, two components of $\mathcal{S}$, namely $G(V,E)$ and $\mathcal{F}$, will be described as part of our algorithm.

1) Social Network Representation: In EpiSimdemics, the social network is represented by a labeled bipartite graph, G_{PL}, where P is the set of people and L is the set of locations. If a person $p \in P$ visits a location $\ell \in L$, there is an edge $(p, \ell, label) \in E(G_{PL})$ between them, where $label$ is a record of the type of activity and the time of the visit. Each node (person and location) can also have labels. The labels attached to persons correspond to his/her demographic attributes such as age, income, etc. The labels attached to locations specify the location's attributes such as its x and y coordinates, the types of activity performed, maximum capacity, etc. It is important to note that there can be multiple edges between a person and a location which record different visits.

Figure 1 shows an example of a social contact network that captures the interaction between individuals moving through a region [3]. Figure 1a is the bipartite graph representation of the social network, G_{PL}. It has four person nodes, denoted by circles, four location nodes, denoted by squares, and seven labeled edges. Figure 1b shows a temporal projection of the network, G_P, which provides location occupancy by time of day, implying person-to-person contacts. Figure 1c shows the occupancy of location c, along with the health states of the occupants. In the system, both the contact network and the health states of individuals change over time. Additionally, a person's health state change will affect his/her succeeding activities, leading to a new network. This co-evolution is important [8].

2) The Disease Model: EpiSimdemics uses a disease model to specify the form of the local transition functions $\mathcal{F}$ explicitly. The dynamics of most infectious disease is captured by two connected processes: within-host disease progression and between-host disease transmission [3].

The within-host disease progression is modeled as a PTTS, an extension of the well known finite state machine (FSM) with two additional features: the state transitions are probabilistic and timed. The model has the following components: (i) each individual infected by a disease will progress through a series of disease states, which are characterized by a set of attributes such as susceptibility to disease, infectiousness, severity of non-specific symptoms (prodromes), severity of disease specific symptoms, and incapacitation; (ii) an individual stays in a state for a period of time (the dwell time) and then transitions to a succeeding state. Both the dwell time and the state transition are probabilistic, following distributions that correspond to the statistics of the disease being studied. (iii) The parameters of the distributions vary with the individual's demographic characteristics (e.g., age, sex, and nutrition level), treatments received, as well as the effectiveness of the treatments. (iv) Once infected, the progression of states and dwell times is purely a local calculation and is not affected by any other individual. It can, however, be affected by pharmaceutical interventions, such as antiviral drugs.

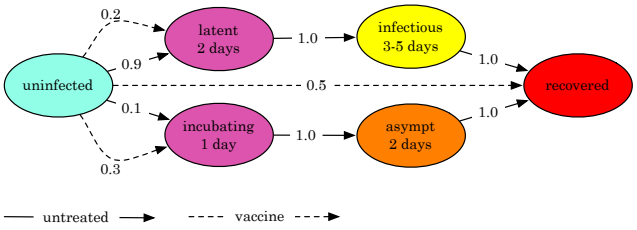

Fig. 2. A simple disease model. Ovals represent disease state, while lines represent the transition between states, labeled with the transition probabilities. The line type represents the treatment applied to an individual. The states contain a label and the dwell time within the state.

A simple example of this disease model is shown in Figure 2. This disease model has three paths from the uninfected state to the recovered state. The upper path represents a disease progression with a two day latent period, followed by three to five days of illness, followed by recovery. In this model, once recovered, an individual cannot become infected a second time. A transition from the recovered state to the uninfected state can be added to allow for reinfection. The lower path represents a disease progression with a shorter latent period, followed by two days in which the individual does not show any symptoms, but can still infect others. For untreated individuals, the upper path is taken 90% of the time, while the bottom path is taken 10% of the time. For individuals who have been vaccinated, there is a middle path which represents the immunity conferred by the vaccine. The disease model used in the results presented in Section IV was used for the studies in [16]–[18], and has 17 states, 44 edges, and three edge types.

Between-host transmission specifies the effects of interactions between individuals, i.e., how does an uninfected individual become infected after being exposed to the disease by an infected individual. The effects are probabilistic and captured by the probability function shown in (1) [13]:

$$p_i = 1 - exp(\tau \sum_{r \in R} N_r ln(1 - r s_i \rho)) \quad (1)$$

This equation specifies the probability that a particular susceptible individual i is infected at a given location, where τ is the duration of exposure, R is the set of infectivities of the infected individuals at the location, N_r is the number of infectious individuals with infectivity r, s_i is the susceptibility of i, ρ is the transmissibility, a disease specific property defined as the probability of a single completely susceptible person being infected by a single completely infections person during one minute of exposure. The probability function is q-symmetric, meaning it only cares about which of the q equivalence classes an infectious individual that comes in contact belongs to (depending only on infectivity level r). This fact will turn out to be crucial in our parallel algorithm. We observe the following disease specific properties:

1) Between-host transmission and within-host progression can be viewed as two connected but independent processes. Between-host transmission triggers the start of within-host progression by causing an uninfected individual to transition to an infected state. The disease progress of the infected individual is then fully determined by the local function governing the within-host progression.
2) There is a latent period between the time an individual gets infected and the time he/she has the capability to infect others. In other words, during an individual's latent period, his/her infectivity is 0. We use D_{min} to denote the minimum latent period length for all individuals.

As described in Section VII, we plan to extend the system to support multiple interacting PTTSs.

II. The EpiSimdemics Algorithm

A. The Simple DES simulation

This NDDS can be simulated with a simple discrete event simulation (DES) algorithm in which the system only changes its state upon the occurrence of an event. We will refer to this as `simple-DES`. As shown in Figure 1d, there are two types of events in the system: *Arrive Events* (person p arrives at location l at time t_{arrive}) and *Depart Events* (person p leaves location l at time t_{depart}).

To ensure correctness, `simple-DES` has to adhere to the following causality constraint: *If an individual i leaves location L_A at time t_{depart} and arrives at location L_B at time t_{arrive}, his/her health state when arriving at L_B (denoted by $s_i(t_{arrive})$) has to be decided prior to calculating the states of other individuals at L_B after time t_B.* This causality constraint leads to temporal and spatial dependencies among nodes in the simulated system.

For simplicity of exposition, travel between locations is shown as instantaneous. In the actual system, there is a delay between leaving one location and arriving at the next location. This delay can be calculated with varying degrees of accuracy within Simdemics. Typically, disease transmission on public transportation vehicles has been ignored, but can be easily modeled by adding buses, trains, etc., as additional locations.

By sorting all events by a global clock and updating the health states of all individuals (according to the disease model) as well as their next destination locations (according to their original schedule and dynamic responses to the environment), we can obtain an implementation for simulating a coupled PTTS. However, for a large population consisting of tens of millions of individuals, the computational resources provided by a single machine are insufficient.

One alternative is to implement a parallel discrete event simulation (PDES) [19]. A second alternative, and the approach taken in EpiSimdemics, is to design a novel algorithm that takes advantage of the semantics of the underlying system. An interesting question is how the performance of EpiSimdemics compares to a generic PDES solution to the problem. A previous system, EpiSims, used a conservative PDES approach with a relaxed causality constraint. Like EpiSimdemics, EpiSims uses realistic social contact networks to simulate the spread

of infectious diseases. The primary difference between the two simulations is their computational implementation. In EpiSims, an arrival event is allowed to be processed up to 15 minutes (simulation time) late before considering it a causality violation. Even with this addition, the system showed poor scaling when run with a population containing more than 10 million individuals. Recently, a number of important advances have been made in the area of PDES [20]. These advances have demonstrated the ability of generalized PDES systems to scale to very large number of processors, despite the generally known limitations of lookahead-based concurrency or rollback-based overheads. Investigating the application of these methods for simulating epidemics is an interesting topic for future research.

B. A Semantics-aware, Sequential DES Algorithm

We first describe a semantics-aware DES algorithm called `Seq-EpiSimdemics`. This will allow us to better explain our parallel algorithm. `Seq-EpiSimdemics` uses an interaction-based hybrid approach approach explained below,

1) Individuals can only affect other individuals through interactions that occur when they are at the same location at the same time.
2) An individual's health state changes are deterministic, and can be precomputed.
3) There is a minimum latent period, D_{min}. This is the amount of time that must pass between a person becoming infected, and a person being able to infect others.

The above observations led to a semantics-oriented problem decomposition. The existence of a latent period for newly infected individuals in the disease model provides a basis for relaxing the global clock. If the time period to be simulated is divided into n phases, and if the *length of single simulation phase* is less than D_{min}, then all locations can be concurrently considered and interactions between individuals at these locations can be simulated in parallel. This is the basis of the EpiSimdemics algorithm `EpiSimdemics`, illustrated in Figure 3. The benefit of this is that only $O(n)$ synchronizations are needed. n is generally small (120-360). A suitable length for a single phase is 24 hours of simulation time. In fact, (i) the state of an interactor when it enters a location can be precomputed at the start of a phase, and (ii) the overall effect of the interaction between an interactor with other individuals at each of the visited locations in a given phase can be *decomposed*, i.e., the possible within-host state transitions at the start of a phase can be computed by a simple computable function of the state transitions that are suggested by the interactions at each of the locations visited. The second property is intuitively similar to the property used by parallel prefix computations. Formally, let $x_v(t)$ denote the state of an interactor v at time t and let $v_1, v_2, \ldots v_k$ denote its neighbors at all the locations in a given phase. Then, a sufficient condition for the within host disease progression is that the update function $x_v(t+1) \leftarrow f_v(x_v(t), x_{v_1}(t), \ldots, x_{v_k}(t))$ can be written as

$$f_{v,d_v}(x_v(t), x_{v_1}(t), \ldots, x_{v_k}(t)) \equiv g(x_v(t) \oplus x_{v_1}(t) \ldots \oplus x_{v_k}(t))$$

where $\oplus$ is an associative and commutative operator and g is a simple computable function. For example, g could be a probabilistic threshold function that has been often used for modeling the spread of norms or fads [15]. Note that Equation 1 used to compute state transitions from susceptible to infected has this desirable form. All other state transitions happen locally and are time triggered.

Input: A social network $G = (P, L, V)$, a disease manifestation M, an initial system state $S(0)$, a set of scenarios C, a simulation phase $\triangle t$, and a total simulation time T
Output: A sequence of system states over time $S(\triangle t), S(2\triangle t), \ldots, S(T)$

initialization
for $t = 0$ **to** T increasing by $\triangle t$ **do**
  foreach individual $i \in P$ **do**
    compute a set of visits $v_{i,j} = (l_{i,j}, t_{i,j}, d_{i,j}, s_{i,j}), 1 \leq j \leq N_i$, where N_i is the number of locations i will visit during the time interval $(t, t + \triangle t)$, and $l_{i,j}$, $t_{i,j}$, $d_{i,j}$ and $s_{i,j}$ denote the target location, the start time, the duration, and i's health state during its j^{th} visit
    foreach $v_{i,j}$ **do**
      if $s_{i,j}$ changes value to $s'_{i,j}$ at time t' and $t_{i,j} < t' < t_{i,j} + d_{i,j}$ **then**
        split $v_{i,j}$ into two visits $v^a_{i,j} = (l_{i,j}, t_{i,j}, d^a_{i,j}, s_{i,j})$ and $v^b_{i,j} = (l_{i,j}, t', d^b_{i,j}, s'_{i,j})$, where $d^a_{i,j} = t' - t_{i,j}$ and $d^b_{i,j} = d_{i,j} - d^a_{i,j}$
      end if
    end for
    foreach $v_{i,j}$ **do**
      send $v_{i,j}$ to location l_j
    end for
  end for
  foreach location $l_j \in L$ **do**
    compose a serial DES
    foreach visit $v_{i,j}$ **do**
      add an arrive event $e^A_{i,j}$ at time $t_{i,j}$, and a depart event $e^D_{i,j}$ at time $t_{i,j} + d_{i,j}$ to a priority queue Q_j in which events are ordered by processing time
    end for
    foreach event $e \in Q_j$ **do**
      remove e from the head of Q_{L_j}
      compute the outcome $r_{i,j}$ of e
      send $r_{i,j}$ to person p_i
    end for
  end for
  foreach individual $i \in P$ **do**
    receive and combine all results $r_{i,j}$
    update i's health state
  end for
end for
finalization

Fig. 3. A sequential version of the EpiSimdemics algorithm `Seq-EpiSimdemics`

In the Seq-EpiSimdemics algorithm, the simulation process is broken into an initialization phase and a sequence of simulation phases with simulation phase interval $\triangle t$. During each simulation phase, the coupled PTTS is simulated with a collection of serial DESs. Instead of passing events from location to location, all events that will occur during a phase are pre-computed by each individual, concurrently, during the beginning of each simulation phase. Dependencies between two successively visited locations are decoupled and all locations are simulated concurrently. At the end of each simulation phase, each individual combines the outcomes of all of interactions it was involved in and updates its local state.

C. The Correctness of the Algorithm Seq-EpiSimdemics

The correctness of Seq-EpiSimdemics is supported by the following theorem:

Theorem 1: Given an NDDS described in Section I-B and a disease M specified by the model in Section I-B2, Algorithm Seq-EpiSimdemics simulates the dynamics described by simple-DES correctly if and only if $\triangle t \leq D_{min}$, where $\triangle t$ is the simulation phase interval, D_{min} is the minimum latent period length.

Proof: Let us define $V_i = \{v_{i,1}, \ldots, v_{i,j}, \ldots, v_{i,N_i}\}$ as the set of visits for an individual i during the simulation phase $[t, t + \triangle t]$, and $s_i = \{s_{i,1}, \ldots, s_{i,j}, \ldots, s_{i,N_i}\}$ as the set of health states during the visits. All variables here have the same meanings as those described in Figure 3.

The fundamental difference between the Seq-EpiSimdemics and simple-DES algorithms is that Seq-EpiSimdemics precomputes all $s_{i,j}$ before any visit during the current phase, while DES computes $s_{i,j}$ when visit $v_{i,j-1}$ is completed. It is trivial that Seq-EpiSimdemics and simple-DES are equivalent if and only if $\forall i \in P$, the lookahead mechanism used by Seq-EpiSimdemics does not change the effects of any visit, i.e., i's interaction with other individuals in the system. Let's define $s_{i,0} = s_i(t)$ as person i's starting state before any visit in current simulation phase. The proof consists of two parts:

Part I: $\forall i \in P$, the lookahead used by Seq-EpiSimdemics does not impact the correctness of the computation of other individual's health states during each simulation phase.

We show this statement is true for all three possible cases listed as below:

Case 1: $s_{i,0}$ belongs to one of the infected states. According to the disease model, once i becomes infected, i will not be susceptible to any further between-host transmission and is only determined by within-host progression. Thus all $s_{i,j}$ can be correctly precomputed from i's local information prior to any visit. Also, as all $s_{i,j}$ for $1 \leq N_i$ is known, V_i can be also determined at the beginning of phase $[t, t + \triangle t]$. This means the network structure in current simulation phase also will be known in advance.

Case 2: $s_{i,0}$ is in an uninfected state and i has not been infected during any visit in current simulation phase. It is obvious that all $s_{i,j}$ can be correctly precomputed.

Case 3: $s_{i,0}$ is in an uninfected state and i becomes infected during a visit s_{i,j_c}. Then for $j < j_c$, $s_{i,j}$ will be same as case 2; otherwise the actual $s_{i,j}$ will be different from the one precomputed. However, since $\forall j$ such that $j_c \leq j \leq N_i$, we have $t_j - t_{j_c} < D_{min}$. Thus, i will be in the latent period and still have zero infectivity. Thus the discrepancies between the precomputed health states and actual health states will not impact the correctness of computing any other person's health states.

Part II: $\forall i \in P$, Seq-EpiSimdemics updates his/her health states as well as infection time correctly at the end of each simulation phase.

As discussed above, only Case 3 results in discrepancies between the precomputed health states and actual health states. Let's denote $t_{infect,i}$ as the occurring time of the infect event and $t_{infect,i,j}$ the occurring time of the infect event in visit $v_{i,j}$, both for individual i. We also use the notation $t_{infect,i,j} = \infty$ if p does not become infected in visit j.

The disease model has two properties: (i) all the between-host transmission events are independent of each other; (ii) an individual can only get infected once during a simulation phase. The first property guarantees that Seq-EpiSimdemics computes each $t_{infect,i,j}$ correctly; the second property guarantees that $t_{infect,i}$ will always be smallest $t_{infect,i,j}$, or the one output by DES. Once $t_{infect,i}$ is known, Seq-EpiSimdemics can apply the within-host progression to update i's health states after $t_{infect,i}$ to their correct values.

Combining I and II, we conclude that Seq-EpiSimdemics and DES are equivalent for the same coupled PTTS problem. Thus the correctness of Algorithm Seq-EpiSimdemics is proved. ■

D. The Complexity of Seq-EpiSimdemics

We denote k^P the average degree (i.e., the number of locations visited by the person) for all person nodes, k^L the average degree (i.e., the number of persons who visit the location) for all location nodes, both for the bipartite graph G_{PL}. The computation of the Seq-EpiSimdemics consists of two major parts: persons computing their schedules, and locations simulating their local DESs. For a given disease, the cost of computing state transition from one state to another state after a certain amount of time can be treated as constant. Similarly, the cost of simulating each arrival or departure event is proportional to the number of infectious individuals as described in (1). By denoting $n = |P|$ as the number of person nodes, $m = |L|$ as the number of location nodes, and $z = \frac{T}{\triangle t}$ as the number of total simulation phases, as well as assuming a constant number of infectious individuals per location, we can approximate the time complexity of Seq-EpiSimdemics as $o(z \cdot n \cdot k^P + z \cdot m \cdot k^L)$.

From the outline of the algorithm, it is trivial that each person node only needs to store his/her own profile as well as all locations he/she will visit. Similar, each location node only needs to store the location information and schedules it receives. The size of schedule data is much smaller then the

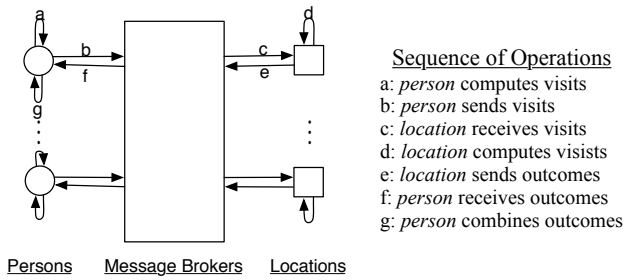

Fig. 4. The computational structure of the sequential EpiSimdemics algorithm.

personal profile data. Thus, we roughly approximate the space complexity of `Seq-EpiSimdemics` as $o(n \cdot k^P + m \cdot k^L)$.

Roughly speaking, `Seq-EpiSimdemics` has the same order of sequential complexity as `simple-DES`, but is more efficient because of the use of a larger simulation phase interval. More importantly, because `Seq-EpiSimdemics` exploits disease specific semantics in order to decouple the temporal and spatial dependencies among network nodes, it leads to a better parallel algorithm with substantially improved parallel time complexity.

III. Parallelization of EpiSimdemics

For an epidemiological study, parallel computing serves three objectives: (i) enabling a large geographic area to be studied (10^7 to 10^8 individuals); (ii) reducing the run time for a single simulation run; and (iii) increasing the throughput for a study with many simulation runs (a thousand runs is not uncommon). In this paper, discussion is focused on the first two objectives.

Based on discussions from the previous section, we organize the operations of the sequential EpiSimdemics algorithm during each simulation phase into the structure shown in Figure 4. This structure depicts the three types of computational components: persons, locations, and message brokers, as well as concurrencies in their local computation and communications. In this structure, message brokers mediate the communication between processes, and are discussed in Section III-B.

Consider a parallel system consisting of N processing elements $\{PE_1, PE_2, \ldots, PE_N\}$. We define a PE as the smallest computing unit to run a single simulation process, which could be a single core on a multi-core processor. We employ the following parallelization strategy:

1) Partition persons into N groups, denoted by $P_1, P_2, \ldots, P_N$;
2) Partition locations into N groups, denoted by $L_1, L_2, \ldots, L_N$;
3) Duplicate the message broker with N copies, denoted by $MB_1, MB_2, \ldots, MB_N$;
4) Distribute the object set (P_i, L_i, MB_i) to process element PE_i;
5) For each PE_i, implement a person manager PM_i for P_i and a location manager LM_i for L_i.

initialization
for $t = 0$ **to** T increasing by $\triangle t$ **do**
  foreach individual $i \in P_i$ **do**
    a. compute a set of visits for the period $(t, t + \triangle t)$
    b. send visits to message broker MB_i
  end for
  c. MB_i accept, retrieve, and forward visit messages
  s1: synchronize
  foreach location $l_j \in L_i$ **do**
    d. compose a serial DES by turning each visit message into an arrive event and a depart event and process events (infections are computed for periods of constant location occupancy)
    e. send outcomes to message broker MB_i
  end for
  f. MB_i accept, retrieve, and forward outcome messages
  s2: synchronize
  foreach person $p_i \in P$ **do**
    g. combine received outcomes and update health state
  end for
  s3: synchronize
end for
finalization

Fig. 5. A parallel version of the EpiSimdemics algorithm `EpiSimdemics`

6) For each PE_i, run a parallel version of `EpiSimdemics` shown in Figure 5 on its local data (P_i, L_i).

The parallel algorithm implements three global synchronize operations. The first one is to guarantee that every location has received all of its visits before computing the local DESs; the second one is to guarantee that every person has received all interaction outcomes before combining them to compute the updated health states; the third one is required for updating global simulation state (e.g., the total number of infected individuals) and triggering dynamic interventions, which are not discussed in this paper.

A. The network partition

For the above EpiSimdemics parallel algorithm, partitioning the person and location nodes and mapping them to available PEs mainly affects the amount of communication required. Ideally, an optimal partition strategy should minimize the amount of messages between PEs, while keeping the computational load balanced across PEs. This is difficult without preprocessing the graph, especially when there are significant number of edges corresponding to long-distance trips. Currently, we distribute the data in a round-robin fashion. This choice was made due to the fact that the initial studies performed with this system were done on metropolitan areas, which have a much more uniform geographic distribution of travel. In our previous studies, we found that the communication cost is roughly 20% of the computation time for a network with a few million nodes (e.g., the Alabama network

TABLE I
CHARACTERISTICS OF THE TWO BENCHMARK NETWORKS

Networks	# of Persons	# of Locations	# of Daily Visits	# of Daily Visit/Person	# of Daily Visit/Location
Alabama	4,333,172	1,300,548	32,672,940	7.54	25.12
Data Size	120 MB	40 MB	1.2 GB		
California	33,153,148	5,521,932	181,670,319	5.48	32.90
Data Size	890 MB	170 MB	9.5 GB		

described in the following section) when running at 96 PEs. This partially justifies the appropriateness of the round-robin distribution schema. Additional benefits of using this round-robin distribution include relatively even load balance among computing nodes and simpler data management.

However, as the results show in the following section, we do find that as the number of PEs or the population size increases, communication cost starts to dominate the total execution time. As part of our ongoing work, we are looking at several partitioning strategies that group the locations according to their geographic location and then place a person on the PEs that holds most of the individual's visited locations. As we study larger and larger areas, an effective partitioning scheme will become more important for good performance.

B. The Message Broker

A message broker connects to the person manager and the location manager on its local PE and to a set of remote message brokers. Each message broker keeps an incoming and outgoing message buffer for each PE with which it has direct communications. If an incoming message has a target location or person hosted by a remote PE, the message broker forwards the message to the corresponding message broker; otherwise, it passes the message to local person manager or local location manager, depending on the message type.

Essentially, the message broker is used to simplify the EpiSimdemics parallel implementation by hiding the complexity of routing interactions across many PEs. A message broker works like a blackboard on which other components (persons or locations) write messages without knowing how and when the message will be delivered. This strategy leaves room for further optimizations of the message broker implementation. This approach is especially suited to EpiSimdemics, as the simulation is very insensitive to network latency. Any message sent during one of the three inner loops in Figure 5 is not needed until after the next synchronization point.

In EpiSimdemics, we use a fully-connected network to organize message brokers on all PEs. This is sufficient for small and medium sized systems with up to several hundred PEs, but insufficient for larger systems. We plan to convert to a layered topology in a future version of the system. In a layered topology, the local message brokers are organized into groups that communicate with a single high-level message broker. The topology of the high-level messages brokers can be optimized for the topology of the underlying communications network (e.g., a tree or hypercube), or can be fully connected.

C. Computation/communication overlapping

To reduce the overhead due to message passing between two PEs, we overlap the computation and communication as follows: (i) pipelining the sequence operations as shown in Figure 4; (ii) using non-blocking send and receive to overlap network communication with computation; and (iii) combing active executions for operations generating messages (e.g., operations *a*, *d*, and *g*) and event-based execution for operations depending on incoming messages (e.g., operations *c* and *f*). To reduce the overhead of sending many small messages, we group messages that have the same destination PE into larger messages before sending them out. This takes advantage of EpiSimdemic's insensitivity to latency.

It is widely expected that the number of cores on a single chip will continue to grow. As a result, we are interested in further investigating how to improve the communication of EpiSimdemics on multi-core architectures, by adopting a hybrid programming model (e.g., mixed use of MPI and multi-threading). The current MPI approach uses one single-threaded MPI process per core. Each process has a group of assigned people and locations, and an MPI subsystem responsible for interprocess communication. A hybrid approach would have $k + 1$ threads on a k core node. One thread would contain a work queue of people or locations to be processed, and handle all MPI related functions. The remaining k threads would process jobs taken from the work queue. This approach would allow more overlap between computation and communication, and achieve a better load balance across the k cores. It would also eliminate contention for access to the single network interface. Again, this takes advantage of the insensitivity of the simulation to network latency.

IV. PERFORMANCE EVALUATION

We have evaluated the performance of an MPI-based implementation of EpiSimdemics on several tera-scale systems. However, due to space limitations, we mainly discuss results on a 112 node (448 core) Linux cluster. The cluster uses Myrinet (Myri-10G) technology for interconnections. Each node is an IBM Server Blade with two 2.2 GHz AMD Opteron Model 275 dual-core processors and 8 GB memory, running CentOS Linux with kernel 2.6.9. For all benchmark runs, timings are collected by instrumenting the code with `gettimeofday` function calls.

We first present performance results for the social contact networks of two states in the United States, Alabama and California. The characteristics of these two social contact networks are summarized in Table I. These networks were created by our research group using methods described in [6],

[9], and chosen to provide a good range of network sizes. A similar dataset for Portland, OR containing 1.6 million people has been made available from ndssl.vbi.vt.edu [21]. In the benchmark runs, we simulate the spread of the H5N1 avian influenza virus across each network. We set the simulation duration as 120 days and set simulation phase interval as 24 hours. All simulations are run without additional interventions and have an attack rate (total number of people infected in the simulation) of about 40%.

We present results that investigate the following areas: strong scaling (constant problem size with an increasing number of PEs), weak scaling (scaling problem size proportionately with the number of PEs), and the effects of multicore architectures on performance. Strong scaling includes discussion of the running time, memory footprint, and the ratio of communication to computation.

Strong Scaling. The running time and memory usage for both networks are shown in Figures 6 and 7. All figures, unless otherwise noted, include results using from one core per node (cpn=1) to four cores per node (cpn=4). For Alabama, increasing the number of PEs by 48 times decreases the running time by 42 times, while for California, increasing the number of PEs by 13 times decreases the running time by 19 times, achieving roughly linear speedup. Since the system used for testing contains 8GB of memory per node, the minimum number of PEs used is determined by the size of the population used. Due to memory constraints and the lack of a serial version of the code, true speedup numbers are not available. The running time is apportioned to computation and communication in Figures 9 and 10. While the computation time shows a decrease across the entire range of PEs, the time taken by communication reaches a plateau around 100 processors for Alabama and 200 processors for California. This is also apparent in Figure 8, which shows the relative speedup for both networks. While further study is needed, one thought is that the synchronization overhead becomes dominant once a certain number of processors are in use. A proposed improvement to the communication architecture is discussed in Section III-B. Additionally, the results are shown for a simulation without disease interventions which, when included, will increase the amount of computation time.

Multicore performance. The system used for testing has dual processor, dual core nodes, for a total of four cores per node. Figure 11 shows the effects of using various numbers of cores per node. The default OS process scheduling algorithm was used, and no attempt to alter either processor or core affinity was made. The scaling results are calculated by assuming that the smallest case (eight nodes, cpn=1) has perfect speedup. The Alabama network is shown, results for the California network are similar. The cpn=1 and cpn=2 cases show nearly identical performance, while performance decreases as the number of cores per node is increased past two. Possible reasons for the performance degradation when multiple cores per node are used include memory contention, cache poisoning, network contention, and lack of processor or core affinity. More study is required to determine the contribution, if any,

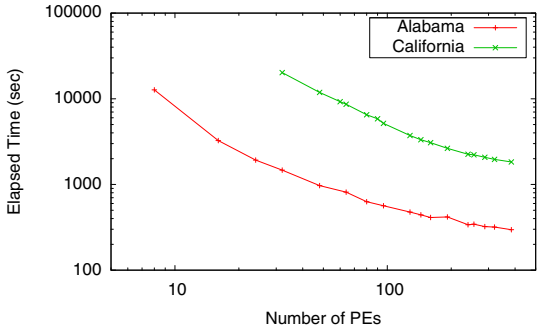

Fig. 6. The execution time for a 120 day simulation of the Alabama and California networks when using different numbers of PEs. Results are shown for different values cpn, the number of cores per node that are used for the simulation.

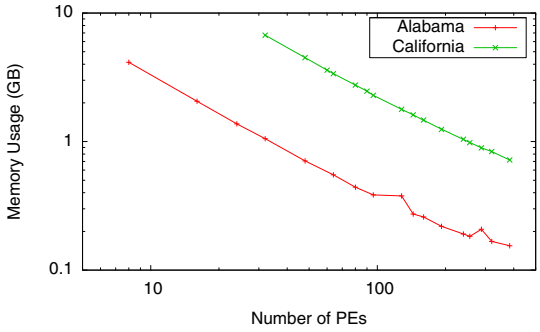

Fig. 7. Memory consumption per PE for a 120 day simulation for the Alabama and California networks.

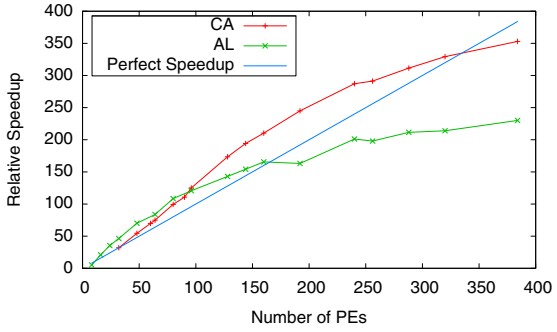

Fig. 8. Speedup relative to smallest parallel configuration.

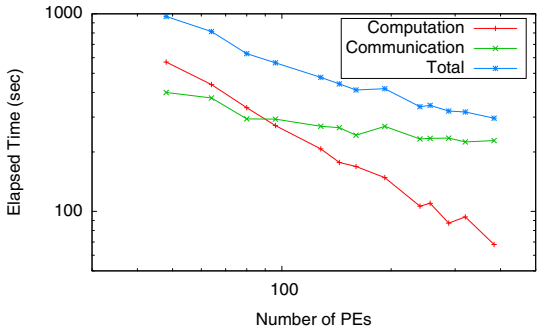

Fig. 9. The breakdown of the execution time of EpiSimdemics for a 120 day simulation for the Alabama network.

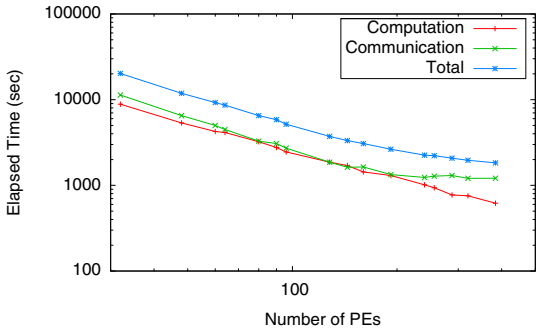

Fig. 10. The breakdown of the execution time of EpiSimdemics for a 120 day simulation for the California network.

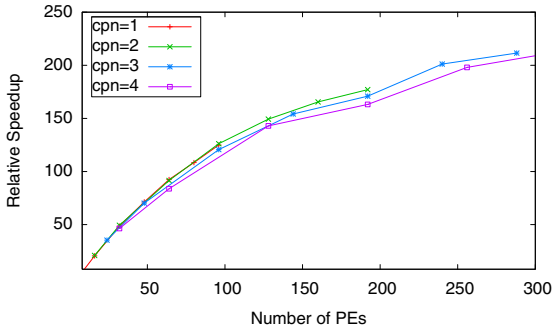

Fig. 11. For the Alabama network, we show the relative speedup for each of the various cpn values. The scaling results are calculated by assuming that the smallest case (8 nodes, cpn=1) has perfect speedup.

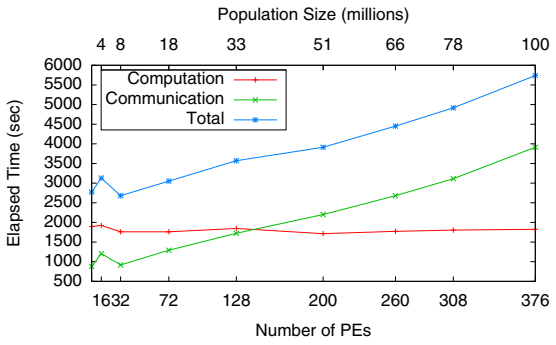

Fig. 12. The elapsed time for a range of population sizes. PEs were chosen so that there were approximately 250,000 people per PE. The times are normalized to exactly 250,000 individuals per PE.

of each of these possibilities.

Weak Scaling. In addition to studying the strong scalability of EpiSimdemics, we are also particularly interested in the capability provided by EpiSimdemics for analyzing the spread of disease across very large contact networks. Figure 12 shows the results of running various sized networks, adjusting the number of PEs so that the population size per PE remains roughly constant. The results were further normalized to exactly 2.5×10^5 individuals per PE. The experiment is summarized in Table II. Unsurprisingly, the computation remains constant across all problem sizes. The time spent in communication increases as the problem size grows, however, the increase is linear and total runtime only increases by a factor of 2.5 for a 50 fold increase in problem size. We believe that the planned changes described in Section VII, along with better partitioning of the data, will be able to increase the efficiency of communication for large problem sizes.

To summarize, by considering the semantics of the disease, EpiSimdemics is able to partition and distribute the workload evenly among available processors and thereby has the potential to scale linearly up to a reasonably large number of processors and support massive social contact networks. This conclusion has been further confirmed by the above experimental results. The performance results also indicate that multicore architectures may present a performance impediment to EpiSimdemics, and that potential optimizations on the communication part are both possible and necessary. As described is the next section, the current version of EpiSimdemics scales well enough to enable interesting and important work in computational epidemiology.

V. Scientific Results

As a benefit of using a highly-resolved population model, EpiSimdemics has the capability to provide fine details about the disease spread in a population including information such as, which specific set of individuals were infected, where were they infected, and who infected them. This information not only provides information about the severity of the overall epidemic, but also its impact on a specific subpopulation, such as school children or health care workers. Thus, the users can identify the demographics of individuals most likely to become infected, and those most likely to spread the infections, and the locations where the disease is spread most easily. Such information provides valuable insights to public health researchers investigating the most effective interventions. Figure 13 shows a snapshot of day 31 of a simulation run of EpiSimdemics for a social contact network of individuals residing in the New River Valley in southwestern Virginia, using the pandemic influenza disease model. The figure illustrates the kind of information that one can obtain from simulations using EpiSimdemics.

Over the past several years, we have used EpiSimdemics in several user defined case studies, including recent pandemic planning studies undertaken for DHS, DoD and DHHS [3], [16]–[18].

For example, at the request of federal agencies involved in preparing for an influenza pandemic, EpiSimdemics was used by the NIH funded MIDAS group to analyze the effectiveness of combined intervention strategies. Results of the MIDAS analyses were reviewed in a Letter Report by the Institute of Medicine, Modeling Community Containment for Pandemic Influenza [22]. The MIDAS study considered both pharmaceutical and non-pharmaceutical interventions targeted at those parts of the population where they might most effectively control the spread of disease [4].

Another study done for the DoD studied the military health preparedness in the event of pandemics [17]. The results demonstrated important differences between public health and military health preparedness problems. When investigating

TABLE II

EXPERIMENTAL DATA FOR WEAK SCALING. STATES AND PES WERE CHOSEN SO THAT THERE WERE APPROXIMATELY 250,000 PEOPLE PER PE. COMP. AND COMM. ARE THE WALLTIME USED FOR COMPUTATION AND COMMUNICATION, RESPECTIVELY. TOTAL IS THE SUM OF THESE TWO VALUES, AND NORM. IS THE TOTAL EXECUTION TIME NORMALIZED TO EXACTLY 250,000 INDIVIDUALS PER PE, AND IS USED IN FIGURE 12.

States	PEs	Individuals	Ind per PE	Locations	Visits	Comp.	Comm.	Total	Norm.
nv	8	1,973,281	246660	440,172	10,881,347	1869	867	2735	2773
co	16	4,213,168	263323	831,482	23,325,370	2025	1271	3296	3129
nj	32	8,063,771	251993	1,419,727	44,475,688	1776	925	2701	2680
ny	72	18,208,506	252896	2,916,179	79,706,069	1782	1306	3088	3053
ca	128	33,153,148	259009	5,520,197	181,681,732	1914	1785	3699	3570
ny, ca	200	51,361,654	256808	8,436,376	261,387,801	1760	2261	4021	3914
ny, ca, fl	260	65,640,135	252462	10,342,750	330,544,083	1790	2705	4495	4451
ny, ca, fl, il	308	77,761,415	252472	12,697,360	397,360,131	1822	3145	4967	4918
ny, ca, fl, il, mi, nj	376	99,602,570	264900	16,090,253	494,559,758	1934	4147	6081	5739

the efficacy of sequestering a military sub-population, we found that sequestration might lead to more infections rather than controlling them. Although counter-intuitive at first, a more detailed analysis showed that such a situation occurred due to a delicate inter-play between two competing factors: (i) sequestration leads to fewer social contacts, but leads to increased social contact within a smaller group of individuals sharing military quarters, and (ii) diseases such as influenza have the property that individuals can be infectious before being symptomatic. It can thus happen that individuals who are infectious can be unknowingly put in close proximity with susceptible individuals for long periods of time. As a result, depending on the size of the sequestration groups, it may be more effective not to implement sequestration.

VI. RELATED WORK

Mathematical modeling and simulation provide powerful tools to study how the course of disease spread will be changed by interventions [22]. Several recent reviews on currently used models can be found in [23], [24]. Roughly speaking, these models can be classified into two categories: population-based compartmental models and individual-based networked models. Compartmental models divide a population into subgroups according to people's disease status (e.g., susceptible, exposed, infected, and recovered) and demographics, and then use differential equations to describe the evolution of disease in the system under the assumption of uniform mixing of the population. In contrast, individual-based networked models make explicit use of the idea of individual interaction and only allow disease transmissions between persons who have contacts. Over the last several years, it is becoming increasingly clear that such models provide a new and useful approach to computational epidemiology [2], [3].

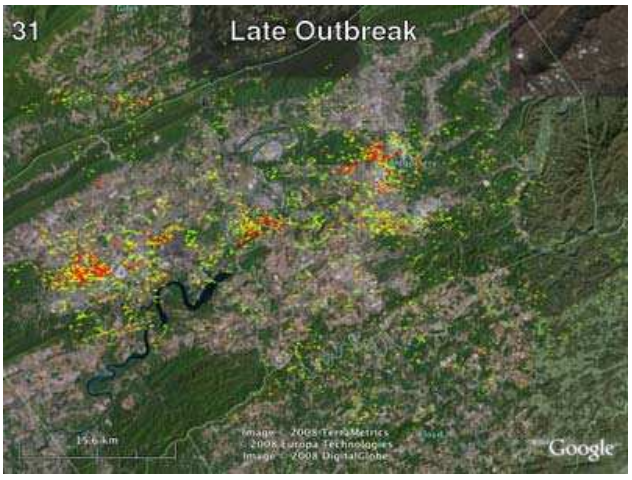

Fig. 13. Snapshot of simulation of Pandemic Influenza in the New River Valley in southwestern Virginia. The colored dots represent the number of infected individuals who live in that geographic area, green for low through red for for high.

EpiSimdemics is closely related to EpiSims — an earlier model co-developed by members of our group [2], [3] and described in Section II-A. Two other simulation methods that are closely related to EpiSimdemics are the parallel simulation systems developed by Longini et al [25], and Ferguson [24]. The system described by Ferguson is implemented to be executed on a shared memory platform and, thus, is limited by the amount of available shared memory. Emulated shared memory machines can be used, but very few machines at present exist that can store very large social networks in such a form. The work of Longini et al is indeed a parallel simulation like EpiSimdemics, but uses a very simple and structured social contact network. The locations in these social contact networks are not real but simply surrogates for simple location types such as school, home, etc. This results in a structured social contact network that is more amenable to efficient parallel computation, but which, arguably, is less representative of real-world social networks.

Physicists have also used a percolation-based approach to simulate the spread of infectious diseases [14]. Such an approach is quite efficient but *only* yields the final outbreak size; the time varying information about disease dynamics (information about transients) cannot be obtained by such methods. But, from the standpoint of epidemic planning and control, transients are the most important parts. For instance, the number of hospital beds required during an epidemic depends on the peak number of infected individuals, not the total number. EpiSimdemics and other simulations discussed above provide time varying information about disease dynamics and are therefore much more useful for studying the effect of interventions and other public policies.

VII. Conclusions and Future Work

EpiSimdemics is a new HPC-based framework to model the spread of infectious disease over large social networks. To our knowledge, EpiSimdemics is one of the few simulation algorithms that have the following unique capabilities. First, it scales to 100 million node realistic social networks, using an interaction-based approach to compute disease dynamics, as opposed to traditional models that are based on differential equations and mean field assumptions. Second, it can be easily modified to study several general reaction diffusion systems over unstructured networks (e.g., diffusion of norms, fads, etc.). Third, it exploits disease-specific semantics in decoupling the temporal and spatial interdependencies existing in the simulation — leading to a scalable parallel implementation for typical distributed memory systems. And fourth, it is specifically designed to be useful in evaluating the efficacy of realistic intervention strategies, e.g., adaptively closing schools, social distancing, etc.

The performance of EpiSimdemics was tested on a cluster with 448 PEs. The results confirm that EpiSimdemics maintains linear scalability within the system size we typically use for studies, showing a speedup of 329 on 300 PEs, relative to the execution time on 32 PEs. This scaling does not continue for larger numbers of processors for the current implementation. Nevertheless, as a large adaptive epidemic experiment usually consists of 100-1000 simulation runs with different parameters, it is more efficient to use a large system to adaptively explore the design space instead of making a single simulation run extremely fast.

Motivated by the performance analysis described in Section IV, there are three main areas for improvement: better partitioning of the data (Section III-A), improved design of the message brokers (Section III-B), and implementing a hybrid MPI/multithreading model (Section III-C). Along with these improvements, the epidemiological models will continue to be improved and generalized. These additions will increase the computational burden of the simulation, while having little impact on the communication load. Additional capabilities that will be added include: multiple co-circulating diseases, enhanced sociological modeling in the agents, and the addition of more complex interventions, such as contact tracing and antiviral stockpiles. These capabilities will be modeled by multiple interacting PTTSs. The Sociological modeling will enable an individual to make choices about the locations that they will visit during the day, based on demographics of the individual, the health state of the individual, family members and coworkers, and the perceived prevalence of disease in the community.

Acknowledgment

We thank the program committee of SC08 for in-depth feedback, and Dr. Kalyan Perumalla for providing constructive comments, suggestions and important references that greatly improved the presentation. We also thank our external collaborators and members of the Network Dynamics and Simulation Science Laboratory for their suggestions and comments.

References

[1] World Health Organization, "World health report 2007: A safe future: global public health security in the 21st century," 2007.

[2] S. Eubank, "Scalable, efficient epidemiological simulation," in *SAC '02: Proceedings of the 2002 ACM symposium on Applied computing*. New York, NY, USA: ACM, 2002, pp. 139–145.

[3] S. Eubank, H. Guclu, M. V. Marathe *et al.*, "Modelling disease outbreaks in realistic urban social networks," *Nature*, vol. 429, no. 6988, pp. 180–184, May 2004.

[4] M. E. Halloran, N. M. Ferguson, S. Eubank *et al.*, "Modeling targeted layeredcontainment of an influenza pandemic in the United States," *Proceedings of the National Academy of Sciences*, vol. 105, no. 12, pp. 4639–4644, 2008.

[5] K. M. Carley, D. B. Fridsma, E. Casman *et al.*, "Biowar: Scalable agent-based model of bioattacks," *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, vol. 36, no. 2, pp. 252–265, 2006.

[6] C. L. Barrett, S. Eubank, and M. V. Marathe, *Interactive Computation: The New Paradigm*. Springer, 2006, ch. An Interaction Based Approach to Modeling and Simulation of Large Biological, Information and Socio-Technical Systems, pp. 353–392.

[7] S. Eubank, V. S. A. Kumar, M. V. Marathe *et al.*, "Structural and algorithmic aspects of massive social networks," in *SODA '04: Proceedings of the ACM-SIAM symposium on Discrete algorithms*. Philadelphia, PA, USA: S. for Industrial and Applied Mathematics, 2004, pp. 718–727.

[8] C. L. Barrett, S. Eubank, and M. V. Marathe, "An interaction based approach to computational epidemics," in *AAAI' 08: Proceedings of the Annual Conference of AAAI*. Chicago USA: AAAI Press, 2008.

[9] C. L. Barrett, R. J. Beckman *et al.*, "TRANSIMS: Transportation analysis simulation system," Los Alamos National Laboratory Unclassfied Report, Tech. Rep. LA-UR-00-1725, 2001.

[10] J. Parker, "A flexiable, large-scale, distributed agent based epidemic model," in *Winter Simulation Conference*, 2007.

[11] N. M. Ferguson, D. A. T. Cummings, C. Fraser, J. C. Cajka, P. C. Cooley, and D. S. Burke, "Strategies for mitigating an influenza pandemic," *Nature*, vol. 442, pp. 448–452, Apr. 2006.

[12] T. C. Germann, K. Kadau, I. M. Longini, Jr., and C. A. Macken, "Mitigation strategies for pandemic influenza in the United States," *Proc. of National Academy of Sciences*, vol. 103, no. 15, pp. 5935–5940, Apr.11 2006.

[13] C. L. Barrett, K. Bisset, S. Eubank, M. V. Marathe, V. A. Kumar, and H. Mortveit, *Modeling and Simulation of Biological Networks*. AMS, 2007, ch. Modeling and Simulation of Large Biological, Information and Socio-Technical Systems: An Interaction Based Approach, pp. 101–147.

[14] M. Newman, "The structure and function of complex networks," *SIAM Review*, vol. 45, no. 2, pp. 167–256, 2003.

[15] P. S. Dodds and D. J. Watts, "A generalized model of social and biological ccontagion," *Journal of THeoretical Biology*, vol. 232, pp. 587–604, 2005.

[16] K. Atkins, C. L. Barrett, R. J. Beckman *et al.*, "Simulated pandemic influenza outbreaks in Chicago: NIH DHHS study final report," NDSSL, Tech. Rep. 06-023, 2006.

[17] ——, "DTRA National Guard study capability demonstration," NDSSL, Tech. Rep. 06-060, 2006.

[18] ——, "An analysis of public health interventions at military bases during a pandemic influenza event," NDSSL, Tech. Rep. 07-019, 2007.

[19] R. M. Fujimoto, "Parallel discrete event simulation," *Commun. ACM*, vol. 33, no. 10, pp. 30–53, 1990.

[20] K. Perumalla, "Parallel and distributed simulation: Traditional techniques and recent advances," in *Winter Simulation Conference*, 2006.

[21] NDSSL, "Synthetic data products for societal infrastructures and proto-populations: Data set 2.0," NDSSL, Virginia Polytechnic Institute and State University, Blacksburg, VA, 24061, Tech. Rep. NDSSL-TR-07-003, 2007, http://ndssl.vbi.vt.edu/Publications/ndssl-tr-07-003.pdf.

[22] C. on Modeling Community Containment for Pandemic Influenza and I. of Medicine, *Modeling Community Containment for Pandemic Influenza: A Letter Report*. Washington D.C.: The National Academies Press, 2006.

[23] S. Riley, "Large-scale spatial-transmission models of infectious disease," *Science*, vol. 316, no. 5829, pp. 1298–1301, 2007.

[24] N. M. Ferguson, M. J. Keeling *et al.*, "Planning for smallpox outbreaks," *Nature*, vol. 425, no. 6959, pp. 681–685, 2003.

[25] I. Longini, A. Nizam *et al.*, "Containing pandemic influenza at the source," *Science*, vol. 309, no. 5737, pp. 1083–1087, 2005.