Simulating the Spread of Infectious Disease over Large-Scale Social Networks

Summary by Erin Carson

Bio: I am a first year PhD student in the Computer Science Department at UC Berkeley. I am advised by James Demmel and Armando Fox. My research interests include numerical linear algebra, high performance computing, and programming languages and tools. Application areas of interest include computational materials science and computational epidemiology. I am taking this course to further my knowledge of parallel computing constructs and gain experience writing parallel code, as well as prepare for the Preliminary Exam.

Aiding in Public Policy Decision-Making

The Field of Computational Epidemiology

Infectious and communicable diseases account for a third of all deaths worldwide [2]. Governmental organizations, on both the national and international level, seek to devise public policies related to potential intervention strategies in order to minimize the death toll from a disease outbreak. Because experimental results on the efficacy of a specific intervention are impossible to obtain short of intentionally infecting a population, researchers have turned to modeling and simulation to help answer these questions. This need birthed the field of Computational Epidemiology, which involves using computer applications to both understand disease spread dynamics and to evaluate potential intervention strategies.

Although differential equation models, such as the standard SIR model [6], are relatively accurate in describing general disease dynamics, evaluating potential intervention strategies requires spatial and temporal information, as well as demographic information about individual agents. Public policy officials would ideally like detailed information about the effects of closing schools, only vaccinating health care officials, etc. For this reason, researchers have turned to agent-based simulations involving social contact networks based on US Census data. A typical compartmental disease spread model is depicted in Figure 1.

Such a simulation involving the entire contact network structure for the US population as well as

![Figure 1](image-url)

Figure 1. A simple disease model. Ovals represent disease states that an individual passes through, with lines representing probabilities for transitioning between states. [1]
demographic information for each individual in the network requires a vast amount of computing power. Simulations were typically performed for a small representative population. Recent advances and changes to algorithmic structure, however, have allowed computational epidemiologists to exploit parallelism in their applications, and have enabled much larger-scale, detailed simulations.

Goals of the Application

Many different research groups have devised simulations to help determine the effectiveness of intervention strategies [3,4]. We focus our discussion here on EpiSimdemics, an application from the Network Dynamics and Simulation Science Laboratory at Virginia Tech which was presented at SuperComputing ’08 [1]. EpiSimdemics has been validated and used in real studies for the Department of Homeland Security, Department of Defense, and Department of Health and Human Services [1].

The primary design goal of EpiSimdemics is to evaluate the success of both pharmaceutical interventions (vaccinations) and non-pharmaceutical interventions (quarantine, isolation, etc). Different levels of each intervention are possible, and can be simulated. For example, pharmaceutical interventions might include mandatory mass vaccination, targeted vaccination, ring vaccination, and voluntary vaccination [3]. By analyzing simulation results and complex interactions between agents for different simulated interventions, public policy officials can gain insights which help form appropriate intervention policies.

Enabling Scaling through Parallelism

Adapting the Algorithm

Given a set of individual agents, a set of locations, and knowledge of when individuals are at specific locations, the probabilities of disease spread between individuals can be calculated. The formal model consists of [1]:

1. A collection of entities (people) with state values and local rules for state transitions through the various disease stages
2. An interaction graph describing the dependency of an entity on its neighbors
3. An update sequence such that local mappings represent the causality in the system
4. A bipartite graph representing the social network (Edges between people and locations they visit. There can be multiple edges for multiple visits)

In order to allow for a large number of agents (e.g., the population of the U.S.) to be simulated, parallelization is necessary. The idea is to partition the data to allow decoupling of spatial and temporal interdependencies between network nodes, targeting a distributed memory cluster. To partition, the total time to be simulated is divided into n phases. As long as the length of a phase is less than the latent disease period, all locations and interactions between individuals at these locations can be simulated in parallel. In fact, the state of an individual when they enter a location can be precomputed at the beginning, and the overall effect of the interaction between entities at the visited location within a phase can be decomposed (similar to parallel prefix computation properties). “Message brokers” are used for coordination between entities (both people and locations) in the simulation (see Figure 2).
The algorithm used in EpiSimdemics is a variation of a generic Parallel Discrete Event Simulation (PDES) solution. Many recent advances have been made in this area. Despite causality constraints inherent in Discrete Event Simulations, insights have allowed for new algorithms which can scale to a large number of processors without large overheads from lookahead-based or rollback-based concurrency [5].

Parallel Architecture

In the EpiSimdemics simulation, results are reported for an MPI implementation run on a Linux cluster with 112 nodes. Each node has two AMD Opteron dual-core processors (448 cores total). The cluster used Myrinet (Myri-10G) for interconnections.

Performance Evaluation

Reported results indicate that this PDES algorithm is capable of achieving acceptable speedups. Linear speedups were reported, for both strong scaling and weak scaling, on a variety of mid-sized HPC platforms [1].

However, in experiments varying the number of cores per node that are used, results show that performance decreases once the number of cores per node is increased past two. More work is needed to determine the exact cause of this performance degradation (memory contention, cache poisoning, network contention, lack of processor or core affinity, etc).

Challenges and Contributions

Successes

Although there is still more work to be done in the area of parallelizing epidemiological simulations, the EpiSimdemics work is promising. This is one of the first parallel algorithms with acceptable performance for simulating epidemics using larger contact networks. The major successes here are the recent advances...
in PDES technology, as Discrete Event Simulations inherently contain mainly spatial and temporal dependencies which must be synchronized. Parallelization in this area is not intuitive.

**Weaknesses**

A weakness here, as in many epidemiological simulations, is a lack of convincing validation. Due to a vast amount of inherent uncertainty about the dynamics and mechanisms of particular infectious diseases, correctness can only be evaluated against historical outbreak datasets. No uncertainty quantification or analysis is performed.

**Areas for Improvement/Challenges**

It should be noted that developing HPC-based approaches to support agent-based models is a challenge itself, and an active area of research. Many factors contribute to this challenge [1]:

1. These simulations often involve large, irregular, and dynamic social contact networks
2. A large number of runs are required due to the stochastic nature of the model, and
3. Capturing diversity amongst agents is necessary in order to understand disease dynamics

The most significant challenge for this application specifically is making effective use of multi-core architectures. Experiments show that for this implementation, communication cost is 20% of the computation time for a network with a few million nodes. As the data in Figure 3 indicates, however, as population size or number of processing elements increases communication costs increase. In order to effectively scale to a larger social network, different partitioning strategies must be considered. Researchers are also considering the use of hybrid programming model (a mix of MPI and multithreading) to improve performance on multi-core architectures, allowing for more overlap between communication and computation.

The authors of EpiSimdemics also hope to add multiple co-circulating diseases, enhanced sociological modeling in the agents, and the addition of more complex interventions such as contact tracing [1].

**Figure 3.** Execution times for the EpiSimdemics algorithm with varying number of cores. 120 day simulation of Alabama. Strong scaling experiment demonstrating the increasing cost of computation. [1]
REFERENCES