REPORT - STOCHASTIC SIMULATION ALGORITHM FOR EFFECTIVE SPREADING DYNAMICS ON TIME-EVOLVING ADAPTIVE NETWORX (SSATAN-X)

Giorgia Bucciarelli, Chiara Campanelli, Adriano Voltolini

Introduction

The article we selected for the scope of our project and whose description and critical analysis we aim to conduct, describes the implementation of an algorithm for the exact stochastic simulation of a pathogen spreading process, represented through the use of an adaptive network.
In this first, introductory part, the theoretical concepts lying at the basis of the above-mentioned study are reported, in order to give a more precise and theoretical background of the biological, mathematical and network modeling context.
First, the definition of what epidemiology is and the processes implicated in the pathogen spreading are roughly introduced. Secondly, a description of the mathematical models used to describe the transmission of communicable diseases is given, together with the definition of which network models are mainly employed to represent epidemiological spreading.
Third, the simulation algorithms employed in the implementation of the proposed algorithm or used to perform the performance analysis and comparisons are introduced theoretically.

Epidemiology and pathogen spreading

Epidemiology is the method used to find the causes of health outcomes and diseases in populations. In epidemiology, the patient is the community and the individuals are reviewed collectively. By definition, it is the study (scientific, systematic and data-driven) of the distribution (frequency, pattern) and determinants (causes, risk factors) of health-related states and events in specified populations. It is also the application of this study to the control of health problems. [1]
The traditional epidemiologic triad model holds that infectious diseases result from the interaction of agent, host and environment. More specifically, transmission, which is defined as the passing of a pathogen from an infected host to a particular individual or group, occurs when the agent leaves its reservoir (or host) through a portal of exit and enters through a portal of entry in the susceptible host to infect.
There are different modes of transmission divided into two main classes: direct and indirect.
In direct transmission, the infectious agent is transferred by direct contact, which occurs through skin-to-skin contact, kissing and sexual intercourse, or droplet spread, which refers to spray with relatively large (> 5 microns), short-range aerosols produced by sneezing, coughing or talking.
Indirect transmission refers instead to the transfer of the infectious agent by suspended air particles (airborne transmission), inanimate objects (vehicles) or animate intermediaries (vectors).
Airborne transmission occurs when infectious agents are carried by dust or droplet nuclei, which are dried particles of less than 5 microns in size, suspended in air. In contrast to droplets that fall to the ground within a few feet, droplet nuclei may remain suspended in the air for long periods of time and may be blown over great distances.
Knowledge about the modes of transmission of a disease provides the basis for determining appropriate control measures directed at controlling or eliminating the agents at source of transmission, protecting the portals of entry, or increasing the host's defenses.

Mathematical and Network Modeling of pathogen spreading

Communicable diseases have always been an important part of human history. Since the beginning of recorded history there have been epidemics that have invaded populations, often causing deaths

before disappearing, possibly recurring years later or possibly diminishing in severity as populations develop immunity.
The goal of epidemiologists is first to understand the causes of a disease, then to predict its course, and finally to develop ways of controlling it, including comparisons of different possible approaches.

The beginnings of compartmental models

In order to describe a mathematical model for the spread of a communicable disease, it is necessary to make some assumptions about the means of spreading infection. In the modern view the spread occurs by contact through a virus or bacterium.
In 1906 W. H. Hamer proposed that the spread of infection should depend on the number of susceptible individuals and the number of infective individuals.
Compartmental models are very general modeling techniques, and the most basic ones used to describe the transmission of communicable diseases are contained in a sequence of three papers by Kermack and McKendrick, the first of which refers to epidemic models.

Developments in compartmental models: SIR and SIS

In the mathematical modeling of disease transmission there is always a trade-off between simple models which omit most details and are designed only to highlight general qualitative behavior, and detailed models, usually designed for specific situations including short-terms quantitative predictions.
Detailed models are generally difficult or impossible to solve analytically and hence their usefulness for theoretical purposes is limited, although their strategic value may be high.
Very simple models for epidemics predict that an epidemic will die out after some time, leaving a part of the population untouched by the disease, and this is also true of models that include control measures. This qualitative principle is not by itself very helpful in suggesting what control measures would be most effective in a given situation, but it implies that a detailed model describing the situation as accurately as possible might be useful for public health professionals.
It is important to recognize that mathematical models needed in a public health setting require a great deal of detail in order to describe the situation accurately and the increased availability of high-speed computing in recent years has made the use of such models possible.
In the study of compartmental disease transmission models the population under study is divided into compartments and assumptions are made about the nature and time rate of transfer from one compartment to another.
In a SIR model we divide the population being studied into three classes labeled S, I and R. We denote S(t) the number of individuals who are susceptible to the disease, that is, who are not (yet) infected at time t. I(t) denotes the number of infected individuals, assumed infectious and able to spread the disease by contact with susceptibles. R(t) denotes the number of individuals who have been infected and then removed from the possibility of being infected again or of spreading infection. Usually, diseases caused by a virus are of SIR type.
In many diseases though, infectives return to the susceptible class on recovery because the disease confers no immunity against reinfection. Such models are appropriate for most diseases transmitted by bacterial or helminth agents, and most sexually transmitted diseases. We use the terminology SIS to describe a disease with no immunity against re-infection, to indicate that the passage of individuals is from the susceptible class to the infective class and then back to the susceptible class.

Each disease has its own properties which should be included in a model for this disease. For example, for influenza, there is a significant fraction of the population which is infected but asymptomatic, with lower infectivity than symptomatic individuals. There are seasonal outbreaks which may be the same strain as the previous year but modified by mutation of the strain, and there is some cross-immunity protecting individuals who were infected by a similar strain in a previous year. Also, influenza models may include the effect of a partially efficacious vaccination before an

outbreak and antiviral treatment during an outbreak. In HIV/ AIDS instead, the infectivity of an individual depends very strongly on the time since infection.

Stochastic models

There are serious shortcomings in the simplest Kermack-McKendrick compartmental epidemic model, especially with the description of the beginning of a disease outbreak, for which a very different kind of model is required. As a matter of fact, the simple Kermack-McKendrick compartmental epidemic model assumes that the sizes of the compartments are large enough that the mixing of members is homogenous. However, at the beginning of a disease outbreak there is a very small number of infective individuals and the transmission of infection is a stochastic event depending on the pattern of contacts between members of the population.

A stochastic branching process description of the beginning of a disease outbreak begins with the assumption that there's a network of contacts of individuals, which may be described by a graph with members of the population represented by vertices (nodes) and with contacts between individuals represented by edges.

One possible approach to a realistic description of an epidemic would be to use a branching process model initially and then make a transition to a compartmental model when the epidemic has become established and there are enough infectives that mass action mixing in the population is a reasonable approximation. Another approach would be to continue to use a network model throughout the course of the epidemic. It's possible to formulate this model dynamically and the limiting case of this dynamic model as the population size becomes very large is the same as the compartmental model.[2]

Adaptive vs static networks for modeling spreading processes

A network consists of a number of nodes connected by links. The specific pattern of connections defines the network's topology. Network models incorporate more realism, explicitly considering interactions (edges) between individuals and locations (nodes). Within this context, mostly static networks have been studied in the past.

Static models can be defined as models that represent a phenomenon at a given point in time or that compare the phenomenon at different points in time (comparative static models).

For many applications it's not necessary to capture the topology of a given real-world network exactly in a model; rather, in many cases the process of interest depends only on certain topological properties.

However, in static networks, the extent of linkage of nodes is determined only after integrating information derived from observing the contact (or spreading) process over a period of time. Several examples highlight that analysis of static networks lacks important temporal information about causal paths that underlie the spreading process, consequently yielding false conclusions for the control of the spreading process.

In order to capture the temporal causality of the underlying system, different **time-evolving network models** have been introduced recently.

The majority of recent studies follow two different lines of research: one is concerned with the *dynamics of networks* where the topology of the system itself is regarded as a dynamical system which changes in time according to specific, local rules; the second focuses on the *dynamics on networks*, where each node of the network represents an individual dynamical system which is then coupled to the others according to the network topology. Thus, the topology of the network remains static while the states of the nodes change dynamically.

Important processes studied within this framework include synchronization of the individual dynamical systems and contact processes such as epidemic spreading. These studies have made it clear that certain topological properties have a strong impact on the dynamics.

Until a few years ago these two lines of network research were pursued almost independently but now it is clear that in most real-world networks the evolution of the topology is invariably linked to the state of the network and vice versa. Hence, a feedback loop between the state and topology of

the network is formed which can give rise to a complicated mutual interaction between a time varying network topology and the nodes' dynamics.
Networks exhibiting such a feedback loop are called **coevolutionary or adaptive networks**.
Since adaptive networks occur over a large variety of scientific disciplines, they are currently investigated from many different directions. In adaptive networks for modeling of pathogen spreading, the dynamical rules that shape the network structure change in response to the dynamics of the spreading process. Examples are the concurrency of sexual partnership which is important for HIV spread or measures of self-isolation for individuals diagnosed with SARS-CoV-2.
This creates a feedback loop between the spreading dynamics on the network and the dynamics of the network itself, leading to emergent complex behavior. Epidemic spreading has been extensively studied on these types of networks to understand the influence of social contact structure on disease prevalence. However, the dynamics of contagion spreading on adaptive networks are usually complex, and analytical results can only be obtained in special cases. This makes numerical studies based on stochastic simulation indispensable.[3]

Simulation Algorithms

Systems biology has recently emerged as a new discipline whose aim is to understand how the reactions occurring between the species composing a system can lead to a specific resulting behavior, or biological response. Hence, it provides a system-wide perspective of biological phenomena and tries to give an order to its activities and properties[4].
The systematic understanding of biological systems poses a great challenge due not only to the large number of possible reactions that can occur within a system, but also to their nonlinear dynamics; as a matter of fact, for these systems the stationary and time-invariant assumptions are often violated.
Computational tools play a crucial role in the development of systems biology.
Models are used to represent in a more or less precise way the biological system under study with the species of interest (states) and the reactions between them (state transitions); but providing an explicit encoding of the knowledge about the system and a mathematical description of its assumptions, is still challenging and much depends on the features that the modeler is interested in capturing.
Given a model, a computer simulation can evaluate it together with additional information to realize its temporal evolution using a simulation algorithm.
In the context of our paper a **stochastic simulation algorithm** is developed and employed to perform the simulation.
Stochastic kinetics is used to provide a probabilistic description of the time evolution of a system, in which the discreteness in population of species and the randomness of reactions are treated as an intrinsic part. The dynamical behavior of the biochemical reactions is exactly described by the **chemical master equation (CME)** which is a collection of differential equations in which each differential equation represents the probability of each possible state of the system at the time t, so providing a complete description of the time evolution of the grand probability (probabilities of all reachable states of the system at time t given the initial state).
CME can be partially solved using the stochastic simulation approach by producing possible realizations of the grand probability function.
Doing so, we explore only the possible states in the state space each time and we obtain as a result a trajectory showing the evolution of the biological system over time.
The mathematical basis of the stochastic simulation is the reaction probability density function (pdf), which is a joint distribution of two variables respectively showing the index μ of the reaction firing R_μ and the time τ to the firing, knowing that the system is at state $X(t) = x$ at time t.

$$p(\tau, \mu | \mathbf{x}, t) = a_\mu(\mathbf{x}) \mathrm{e}^{-a_0(\mathbf{x})\tau}$$

Stochastic simulation algorithm (SSA)

At the basis of the stochastic simulation approach is the stochastic simulation algorithm (SSA) which is a stochastic, discrete event simulation strategy where a reaction is randomly selected to update the system state. It produces an exact realization of the temporal dynamics of the reactions in the system and the heart of the SSA is represented by the Monte Carlo step for sampling the next reaction firing and its firing time from the joint reaction probability density function without introducing approximation.

Algorithm 1 Stochastic Simulation Algorithm (SSA) - General Sketch

Input: a biochemical reaction network of M reactions in which each reaction R_j, $j = 1,\ldots,M$, is accompanied with the state change vector $\mathbf{v}_j$ and the propensity a_j, the initial state $\mathbf{x}_0$ at time 0 and the simulation ending time T_{max}

Output: a trajectory of the biochemical reaction network which is a collection of states $X(t)$ for time $0 \leq t \leq T_{max}$.

initialize time $t = 0$ and state $X = \mathbf{x}_0$
while $(t < T_{max})$ **do**
  set $a_0 = 0$
  for all (reaction R_j) **do**
    compute a_j
    update $a_0 = a_0 + a_j$
  end for
  sample reaction R_μ and firing time τ from pdf $p(\tau, \mu|\mathbf{x}, t)$ in Eq. (2.16)
  update state $X = X + \mathbf{v}_\mu$
  set $t = t + \tau$
end while

The input of SSA is a reaction network of M reactions in which each reaction R_j is characterized by a state change vector v_j and the propensity function a_j; the initial state x_0 which denotes the initial population of each species at time t = 0 and a specified time T_{max} which represents the ending time to stop the simulation.
The population of each species at each time point $t \leq T_{max}$ is stored in the state vector X.

Procedure:
The algorithm begins assigning the initial state x_0 to the state X and then goes into the main simulation loop, where at each iteration, computes the propensity a_j of each reaction R_j and the total propensity a_0. It then proceeds to the Monte Carlo sampling step, where the next reaction R_μ and its firing time τ are sampled from the pdf, which may require the generation of uniformly distributed random numbers.
Finally, the system is updated to a new state $X = X + v_\mu$ and the time is advanced to $t = t + \tau$.
The simulation loop is repeated until the time t is greater than or equal to T_{max}.

There are many different Monte Carlo implementation techniques providing different versions of the SSA, each of which focuses on different aspects of the network with the aim of overcoming specific computational limitations, thus trying to improve the simulation performance.

Tau-leaping

In the context of our paper, the tau-leaping algorithm, as originally proposed by Cao and Gillespie and later modified by Anderson to incorporate post-leap checks with the aim of guaranteeing the accuracy, is used to perform bulk updates of the contact network dynamics.
Exact simulation of complex biochemical systems is often prohibitively expensive due to their intrinsic stochastic and multiscale nature[4].
These computational challenges motivate the development of approximate algorithms to improve the simulation efficiency by sacrificing the accuracy.

The tau-leaping method is an approximate stochastic simulation algorithm which aims to discretize the time into intervals and to approximate the number of reactions firing within each interval. The simulation then leaps from a time interval to the next performing many different reactions in parallel. The time intervals have length τ, which is called the leap time and which is not needed to be a fixed value, hence it can be adaptively defined during the simulation.
The mathematical basis for the simulation of the tau-leaping method is the joint probability density $P\{k_1, \ldots, K_m\}$ that gives the number of firings of reactions during the time interval $[t , t + \tau]$, given the state $X(t) = x$ at time t. Finding an exact formula to solve this probability problem is difficult, but an approximation for it can be derived assuming that the changes in propensities of all the reactions after the reaction firings in the time interval are insignificant. This represents the leap condition.

$$\mathbb{P}\{k_1, \ldots, k_M | \tau, \mathbf{x}, t\} = \prod_{j=1}^{M} \mathbb{P}\{k_j | \tau, \mathbf{x}, t\}$$

The shown equation provides the mathematical basis for the implementation of the tau-leaping. Knowing the firing times k_j of reactions, the tau-leaping method leaps down the time t by an amount τ to the new time $t + \tau$ and updates the state by the following equation:

$$X(t + \tau) = \mathbf{x} + \sum_{j=1}^{M} k_j \mathbf{v}_j = \mathbf{x} + \sum_{j=1}^{M} \mathtt{Poi}(a_j(\mathbf{x})\tau)\mathbf{v}_j$$

which shows that if the number of firings k_j of reaction R_j with j = 1,..,M during the time interval $[t , t + \tau]$ is sufficiently large then the tau-leaping method is faster than the exact simulation.

This method potentially provides a significant speed-up gain but it also exposes many issues:

- efficiency and accuracy of the tau-leaping method are strongly dependent on how to choose a leap tau satisfying the leap condition. In the case where propensities of all reactions are independent of the state, and hence constant, the leap condition is always satisfied for any value of tau and in this case the method is an exact method. In general though, propensities are state-dependent and change anytime a reaction fires; in this case the choice of the leap is a trade-off between the accuracy and the performance.
- given a leap tau that satisfies the leap condition, the simulation must ensure that the generated Poisson-distributed random numbers don't cause too many firings of reactions which results in negative population of the reactant species, which makes the simulation meaningless and severely compromises its accuracy.
- a robust condition to switch from the tau-leaping to the exact simulation is needed in case the leap tau is very small since in this case the cost for generating the Poisson-distributed random numbers is computationally expensive and it should be better to use the exact SSA instead.

Algorithm 28 τ-leaping method

Input: a biochemical reaction network of M reactions in which each reaction R_j, $j = 1,\ldots,M$, is accompanied with the state change vector $\mathbf{v}_j$ and the propensity a_j, the initial state $\mathbf{x}_0$ at time 0, the simulation ending time T_{max}, the error control parameter $0 < \varepsilon \ll 1$, the reduction factor $\alpha < 1$, the threshold parameter k and the number of exact SSA steps parameter p.

Output: a trajectory $X(t)$, $0 \le t \le T_{max}$, of the biochemical reaction network.

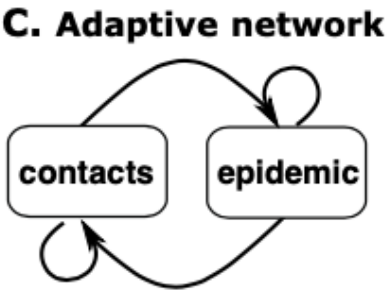
initialize time $t = 0$ and state $X = \mathbf{x}_0$
while $(t < T_{max})$ **do**
  compute a_j for each reaction R_j with $j = 1,\ldots,M$ and $a_0 = \sum_{j=1}^{M} a_j$
  set threshold = k/a_0
  determine τ satisfying leap condition with a leap selection procedure in Section 4.3.1
  repeat
    set acceptedLeap = **true**
    if $(\tau >$ threshold$)$ **then**
      generate M Poisson-distributed random numbers $k_j \sim \texttt{Poi}(a_j(\mathbf{x})\tau)$ with $j = 1,\ldots,M$ (see Appendix B.2.6)
      update $X = X + \sum_{j=1}^{M} k_j \mathbf{v}_j$
      set $t = t + \tau$
    else
      perform p SSA simulation steps
    end if
    if (exists a species in state vector X whose population $X_i < 0$) **then**
      roll back state $X = X - \sum_{j=1}^{M} k_j \mathbf{v}_j$ and time $t = t - \tau$
      reduce $\tau = \alpha\tau$
      set acceptedLeap = **false**
    end if
  until acceptedLeap
end while

The input of the algorithm is a biochemical network consisting of N species and M reactions. It requires an error parameter to enforce the leap condition, a reduction factor α to reduce the leap time τ when negative populations arise, the multiplicative factor k and the number p of exact SSA steps. The output of a simulation of tau-leaping is a trajectory of the reaction network starting at time t = 0 and initial state x_0.

Procedure:
The main loop of the tau-leaping simulation is repeated until the simulation time t passes a predefined time T_{max}. For each iteration the largest tau value that satisfies the leap condition applying a τ selection procedure is determined and is compared against the threshold k/a_0 to decide whether the simulation should switch to exact SSA. If the check is true then a leap is performed and the number of firings k_j of each reaction R_j is generated by sampling from the poisson distribution $(a_j\tau)$. The time is then advanced to $t + \tau$ and the state X is updated simultaneously by all M reactions to move to the new state.
If the leap time tau is smaller than the threshold then the exact SSA is applied.
Also, a check to detect whether there are negative populations of species after the leap and if they occur, the current leap is rejected and the state and time are rewinded in order to perform a new leap trial with a leap value reduced by a factor α.

Paper analysis

In this article the authors developed and presented a claimed-to-be efficient and exact stochastic simulation algorithm, which can allow the prediction of the contagion spreading both when the contact dynamics and the spreading dynamics are governed by inter-dependent stochastic processes. The model on which they based the simulation of the implemented algorithm is that of a simple spreading process but which evolves on a time-dependent and adaptive contact network.



The considered population for the model is composed of susceptible (S), infected (I) and diagnosed (D) individuals each of which is assigned with two rates, one describing the making of new contacts and the other describing the loss of already existing contacts, that together define the contact dynamics of the model, hence the number of contacts connected to each individual at each time point.
The infection is made possible only through contact occurring between individuals and only from an infected to a susceptible or from a diagnosed to a susceptible individual. The definition of contact depends on the considered disease; in case of an airborne pathogen like influenza or SARS-CoV-2, it denotes a period of time during which the individuals are close to one another (i.e. in proximity, within the same room), whereas in case of sexually transmitted infections like HIV, it denotes a sexual intercourse.
The discontinuous adaptive behavior is added in the model when considering the case of an individual who gets diagnosed with the infection and becomes aware of his/her status consequently changing his/her behavior. In this case, the model assumes that the individual cuts all his/her contacts, taking the number of existing contacts related to him/her to 0, and the individual's rate of establishing new contacts drops to 30% of the original value before the diagnosis.
This simple model describes an adaptive dynamics where the contact dynamics change and is changed by the epidemic dynamics in a discontinuous way and vice versa. This feedback loop represents the main feature of the adaptive networks.

This quite simple model was then represented by the authors as an undirected weighted temporal Contact Network whose description we report in the following as described in the paper.

The **contact network** is described as $G(t) = \{E(t), V(t)\}$ where:

- V is the set of nodes (with S, I, D individuals) each of which has a rate λ_i^+ at which a new contact is made and a rate λ_i^- at which contacts are lost
- The set of edges is denoted by $E(t) = \{e_{jk}(t)\}$ with $e_{jk} = (v_j, v_k; \gamma)$, $j \neq k$ if v_j and v_k are in contact.
  The parameter γ represents the transmission rate from j to k, which takes value greater than zero if node j is infected or diagnosed and node k is susceptible, and equal to zero otherwise.

A **continuous-time Markov process** is considered to evolve in time the contact network G(t) and the possible events that can occur are classified into two main groups:

1. Contact Dynamis $R = \{r_{jk}\}$
   - *Assembling of a new contact*.
     For each pair of nodes (v_j, v_k) where $j \neq k$ which is not connected by an edge, the rate of assembling an edge is defined by the product of the assembling rates of the two nodes.
     $$\lambda_{jk}^+ = \lambda_j^+ \cdot \lambda_k^+$$

- *Disassembling of an existing contact*.
  For each pair of nodes (vj, vk), where j ≠ k and which are connected by an edge, the rate of disassembling is defined as the product of the disassembling rates of the two nodes.
  $$\lambda_{jk}^{-} = \lambda_{j}^{-} \cdot \lambda_{k}^{-}$$

2. Epidemic Dynamics A = {a_1}
   - Infection from an undiagnosed, infected I to a susceptible S occurs with a rate gamma greater than 0 if node j and k are connected.
     $$I_j + S_k \xrightarrow{\gamma} I_j + I_{N_I+1}$$
   - Infection from a diagnosed D to a susceptible S occurs with a rate gamma/2 greater than 0 if node j and k are connected.
     $$D_i + S_k \xrightarrow{\gamma/2} D_i + I_{N_I+1}$$
   - Infected individuals I may be diagnosed with the infection
     $$I_j \xrightarrow{\delta} D_{N_D+1}, \text{ for all } j = 1...N_I$$
   - Individuals may die
     $$I_j \xrightarrow{\beta} \varnothing, \text{ for all } j = 1...N_I$$
     $$D_i \xrightarrow{\beta} \varnothing, \text{ for all } i = 1...N_D$$

The model presented by the authors was then firstly simulated using the stochastic simulation algorithm (SSA) of which we report the adapted implementation as proposed in the study.

Algorithm 1: SSA

initialize: Network $G(t)$, reaction rate functions $r \in \mathcal{R}, a \in \mathcal{A}$, final time T_F

1 $t = 0$
2 **while** $t < T_F$ **do**
3 Compute reaction propensities $r_{(\cdot,\cdot)}$ and $a_{(\cdot)}$ based on the current contact network $G(t)$
4 Compute the sum of the reactions propensities
5 $r_0 = \sum r_{jk}$
6 $a_0 = \sum a_\ell$
7 $\Delta t \sim Exp\,(a_0 + r_0)$
8 Choose the next reaction, *e.g.* a *contact dynamics* reaction with probability:
9 $\frac{r_{jk}}{r_0 + a_0}$,
10 or an *epidemic dynamics* with probability:
11 $\frac{a_\ell}{r_0 + a_0}$.
12 Execute chosen reaction
13 $t = t + \Delta t$
14 **end**

The algorithm takes in input the contact network G(t), the reaction rate functions ‘r’ belonging to the contact dynamics R and the reaction rate functions ‘a’ belonging to the epidemic dynamics A. A final time at which ending the simulation is also provided and stored in the variable T_F.

Procedure:
The algorithm initializes the time at t = 0 and then proceeds to execute the main loop; in it, the reaction propensities of both the contact dynamics and the epidemic dynamics are computed together with their respective sum and stored respectively in the variables r_0 and a_0 (lines 3-6). The algorithm then proceeds sampling the time to the next event from an exponential distribution with rate parameter given by the sum of r_0 and a_0 (line 7).

The next reaction to fire is chosen, and it would be an event either related to the contact dynamics or related to the epidemic dynamics.
Finally, the chosen reaction (or event) is executed and the time is updated to $t = t + \Delta t$.
This main loop, hence its Monte Carlo step which represents the main part of the algorithm, is executed as long as the value of the time t is less than the provided final time T_F at which the simulation is ended.

The authors state that if the main interest of the simulation is in the assessment of the epidemic dynamics, this approach could lead to a considerable computational overhead, especially in the case when the contact dynamics are much faster than the epidemic dynamics. This means that when only a few contacts are actually leading to infection, the amount of time needed to compute the simulation makes this algorithm not particularly efficient. Hence, the computational overhead represents the main issue that the authors decided to address. To overcome this problem the algorithm implemented by the authors is based on the idea of dividing the contact dynamics and the epidemic dynamics into two different processes and performing **bulk updates** of the contact dynamics.
The bulk updates of the R reactions contact dynamics are performed exploiting the **tau-leaping algorithm**.
With the aim of making the simulation computationally more efficient, the authors proposed the employment of a key adaption considering all reactions of one type, namely addition or deletion of edges, simultaneously. The number of edges available for deletion is clearly limited by the number of existing edges in the network E(t) at time t. In total, the number of edges present on a network can't exceed $N(N-1)/2$ for a finite population of N individuals; therefore, the number of edges available for addition is limited by $N(N-1)/2 - |E(t)|$, which can be readily computed by considering a complement network G' = {E' , V'}. The authors have decided to use this additional data structure in order to apply a less complex search algorithm, but in our proposed implementation, we decided to only consider the number of edges that would exist in the complement network E', storing it in a numerical variable.
The tau-leaping algorithm for the bulk updates implemented by the authors is reported below.

Algorithm 3: τ-leaping for bulk-updating the contact network

initialize: Network $G(t)$, *contact dynamics* reaction functions $r \in \mathcal{R}$, final time T_F, $0 < p < p^* < 1$ and $0 < q < 1$. $q^* > 1$, $\epsilon < 1$

1 Set $Q^\xi = C^\xi = 0$, $S^\xi = [0, 0]$, t = 0
2 Calculate propensities $r^\xi_{(\cdot,\cdot)}$ and for each reaction *type* $\xi = \{+, -\}$ based on the current network state $G(t)$. Compute $r^\xi_0 = \sum r^\xi_{jk}$, as well as $r_0 = \sum_\xi r^\xi_0$; Calculate τ according to equation (2.1)
3 **while** $t < T_F$ **do**
4 $\tau = \min(\tau, T_F - t)$
5 **if** $\tau < \frac{10}{r_0}$ **then**
6 Execute SSA 100 times
7 Return to **line 2**
8 **else**
9 **for** ξ **do**
10 B^ξ = number of rows in S^ξ
11 **if** $r^\xi_0\tau + Q^\xi \geq S^\xi[B^\xi, 1]$ **then**
12 $M^\xi \sim \mathcal{P}(r^\xi_0\tau + Q^\xi - S^\xi[B^\xi, 1]) + S^\xi[B^\xi, 2] - C^\xi$
13 $row^\xi = B^\xi$
14 **else**
15 Find K^ξ such that $S^\xi[K^\xi - 1, 1] \leq r^\xi_0\tau + Q^\xi < S^\xi[K^\xi, 1]$
16 $u = \frac{r^\xi_0\tau + Q^\xi - S^\xi[K^\xi-1,1]}{S^\xi[K^\xi,1] - S^\xi[K^\xi-1,1]}$
17 $M^\xi \sim \mathcal{B}(S^\xi[K^\xi, 2] - S^\xi[K^\xi - 1, 2], u) + S^\xi[K^\xi - 1, 2] - C^\xi$
18 $row^\xi = K^\xi - 1$
19 **end**
20 **end**
21 **if** *leap conditions equation* (2.4) *holds* **then**
22 **for** ξ **do**
23 Delete all rows from S^ξ less than or equal to row^ξ
24 Add new first row $[r^\xi_0\tau + Q^\xi, C^\xi + M^\xi]$
25 $Q^\xi = Q^\xi + r^\xi_0\tau$
26 $C^\xi = C^\xi + M^\xi$
27 **end**
28 $t = t + \tau$
29 **if** *leap would have failed the leap condition for* $\epsilon' = 0.75\epsilon$ **then**
30 $\tau = \tau p^*$
31 **else**
32 Set $\tau = \tau^q$ if $t < 1$ and otherwise $\tau = \tau^{q^*}$
33 **end**
34 Bulk-update network by executing M^ξ reactions using Algorithm 4
35 Recalculate $r^\xi_{(k,l)}$, r^ξ_0 and r_0 based on updated network state
36 **else**
37 **for** ξ **do**
38 Add row $[r^\xi_0\tau + Q^\xi, C^\xi + M^\xi]$ between row^ξ and $row^\xi + 1$
39 **end**
40 $\tau = \tau p$
41 **end**
42 **end**
43 **end**

It takes in input the network G(t), the reaction functions 'r' of the contact dynamics R, the final time T_F and a few design parameters.
The additional parameters p and p* both take values between 0 and 1 but with p < p*; q also with value between 0 and 1 and q* with value greater than 1; ε takes value equal to 0.75.
The propensities are then computed for each reaction type (addition and deletion) based on the current network state G(t). Also, both the sum of all reaction propensities for each reaction type and the sum of all the reaction propensities, are computed and stored respectively in the variables r_0^ξ and r_0. Tau τ is calculated according to the equation:

$$\tau = \min\left\{\frac{\max(\epsilon|E|, 1)}{|\mu_E|}, \frac{\left(\max(\epsilon|E|, 1)\right)^2}{\sigma^2_E}, \frac{\max(\epsilon|E'|, 1)}{|\mu_{E'}|}, \frac{\left(\max(\epsilon|E'|, 1)\right)^2}{\sigma^2_{E'}}\right\}$$

The algorithm runs until t reaches T_F and for each iteration we choose τ as the minimum between τ itself and T_F - t. Then, the condition for which we choose to switch to the exact SSA is defined in line 5 and eventually executed in line 6.
If the condition doesn't hold, the tau-leaping proceeds; for each reaction type the number of reactions to be executed is determined by sampling either from a Poisson or a Binomial distribution (lines 11-17).
The algorithm then checks if the conditions for the leap hold; if so, some optimization variables are updated together with t. If the leap would have failed with an ε slightly smaller than the one used, τ is updated by multiplying it by p*, otherwise τ is set as equal to itself to the power of q whether $t < 1$, or to itself to the power of q* whether $t > 1$. ε is therefore considered as the error parameter which determines the threshold for the leap acceptance.
Then the algorithm performs the bulk update of the network by executing the M reactions using the 4th algorithm for the bulk update. The propensities are then recalculated. If the leap condition doesn't hold, the value of tau is reduced by multiplying it by p and the algorithm is attempted again.

Algorithm 4: Bulk Update

1 Calculate the cumulative sums ς^+ and ς^- for reaction types $\xi \in \{+, -\}$. Permute M^- reactions that eliminate an edge and M^+ reactions that create an edge and save the order in the vector O
2 **for** *each contact update in* O **do**
3 Sample $u \sim U(0, 1)$
4 **if** *deletion* **then**
5 set r_0^- to the last element in ς^-
6 Find first y such as: $\sum_{i=1}^{y} \varsigma_i^- \geq u \cdot r_0^-$
7 Remove edge e_{jk} corresponding to index y from the network and remove the corresponding element from ς^-
8 Calculate $tmp = r_{jk}^+ +$ value of last element in ς^+
9 append tmp to ς^+
10 **else if** *addition* **then**
11 $r_0^+ = \varsigma_{\varsigma^+.length}^+$
12 Find first y such as: $\sum_{i=1}^{y} \varsigma_i^+ \geq u \cdot r_0^+$
13 Add edge e_{jk} corresponding to index y to the network and remove the element from ς^+
14 Calculate $tmp = r_{jk}^- +$ value of last element in ς^-
15 append tmp to ς^-
16 **end**

Showing above is the algorithm used by the authors to perform the bulk update.
For each reaction type, an array containing the cumulative sums of the propensities is created. Then, for each reaction taken into account, a random number is computed by sampling it from a uniform distribution between 0 and 1; it is then multiplied by the r_0 of the specific reaction type to find the first element of the array greater than or equal to the defined product in order to choose the next reaction to fire.
Conclusively, the network is updated according to the chosen fired reactions.

Algorithm 2: SSATAN-X envelope algorithm for simulating effective spreading dynamics on adaptive networks.

initialize: Network $G(t)$, *epidemic* reaction functions $a \in \mathcal{A}$, final time T_F

1 $t = 0$

2 **repeat**

3 Define *look-ahead time* $T_L = T_F - t$

4 Define propensity upper bound B_{T_L} as proposed in Appendix S2:

5 Sample $\Delta t \sim Exp\,(B_{T_L})$

6 **if** $\Delta t > T_L$ **then**

`/* reject */`

7 $t = t + T_L$

8 **else**

9 Bulk update the edges of $G(t)$ for time interval $[t, t + \Delta t]$ using Algorithm 3

10 $t = t + \Delta t$

11 Compute *epidemic* reaction propensities a_ℓ based on the current network $G(t)$;

12 Compute $a_0 = \sum a_\ell$

13 Sample $u \sim U(0, 1)$

14 **if** $a_0 \geq B_{T_L} \cdot u$ **then**

`/* accept: */`

15 Choose the ℓ-*th* reaction with probability $\frac{a_\ell}{B_{T_L}}$

16 Execute chosen reaction

17 Execute adaptivity step if required

18 **else**

`/* thin: network remains the same */`

19 **end**

20 **end**

21 **until** $t \geq T_F$

Finally, the comprehensive algorithm composing the SSATAN-X implementation, as proposed in the paper, is reported. As previously mentioned, the authors decided to split the stochastic process into a process Y, comprising all reactions R belonging to the contact dynamics, which we remind again are bulk updated using the tau-leaping algorithm, and a process Z, driven by reactions A of the epidemic dynamics, updated during the execution of the SSATAN-X.
The algorithm takes in input the network G(t), the epidemic reaction functions 'a' of the epidemic dynamics, and the final time T_F.
The initial time is set to 0 and the algorithm iterates until the time is greater than or equal to the final time T_F.
The algorithm first defines the look ahead time T_L for which an upper bound B_{TL} for the network G(t) is computed (line 4).
In the example introduced in the paper, B_{TL} represents the sum of the upper bounds of the propensities for the reactions of transmission, diagnosis and death, which can be defined as:

$$B_{T_L} = B_{S \to I, T_L} + B_{D, T_L} + B_{\varnothing, T_L}$$

The upper bounds for the diagnosis and death events are easily estimated, however the upper bound for the transmission events is computed as an estimation based on a 95% confidence interval and therefore is not always correct. Hence, the total upper bound is not exact and the proposed time to the next reaction, computed with respect to B_{TL}, might be shorter than the actual time. This apparent underestimation is corrected by the "thinning" step in which we observe contact dynamics events but not epidemic events (lines 18-19). Choosing a larger upper bound would make the simulation more accurate but, on the other hand, this could result in many more "thinning" steps which can affect the algorithm efficiency.

The algorithm then proceeds sampling the time to the next epidemic reaction Δt from an exponential distribution with rate parameter B_{TL}. If the proposed time step Δt lies within the look-ahead horizon T_L, the contact dynamics (edges) of the network G(t) are bulk-updated using the tau-leaping algorithm previously described. Then, the time is updated by Δt (line 10).
At this point, the actual propensity functions 'a_l' are recomputed based on the current network and the next reaction to change the state of the nodes is chosen (lines 13-19).

Results

To compare the SSA and the SSATAN-X algorithm, the authors decided to perform the simulations using a population of 200 nodes, composed of 180 susceptible, 20 infected and 0 diagnosed individuals, and 3000 initial contacts (edges).
For the epidemic dynamics, the transmission rate was set to $\gamma = 0.04$ if it considers a S-I contact, and $\gamma = 0.02$ if it considers a S-D contact. The death rate of the infected and diagnosed individuals was set to $\beta = 0.08$ and $\beta = 0$ for susceptible individuals.
The last parameter indicating the diagnosis rate is set to $\delta = 0.5$.
The adaptive behavior of the model occurs when an individual gets diagnosed; in such cases, the individual's existing contacts are deleted and its rate of assessing new contacts drops to 30% of the initial value.

To conduct further comparisons and verify the results obtained by the authors in this study, we decided to do our own implementation of the SSA and SSATAN-X algorithms for studying an adaptive network modeled under the same conditions as the ones reported in the paper.
The proposed algorithms were implemented using the Python3 programming language and relied on several different additional modules such as Numpy, NetworkX, Pandas, Matplotlib and Multiprocessing for parallel computing.
In the following we provide a general description of our proposed implementation of the different algorithms, whose scripts can be found in the supplementary materials of this report:

- Parameters.py: contains all the parameters used in our project. Changing values in here will change the outputs of the other python scripts.

- ContactNetwork.py: contains the function "graph_creator" that uses the NetworkX python module to create an initial, randomly generated graph that resembles the ones considered in the paper. This function returns a tuple containing the graph itself, two arrays containing the association and dissociation rates of the nodes, and an array containing the initial statuses of the nodes.

- SSA.py: contains two functions, called "SSA_full" and "SSA_contact". The first function is an implementation of the SSA algorithm that works on NetworkX graphs and it is capable of simulating both the contact and the epidemic dynamics of the system. On the other hand the second function, as the name suggests, is only capable of simulating the contact dynamics. "SSA_contact" is used in the tau-leaping algorithm in case tau is too small. Calling these functions will execute a single reaction.

- Tau_Leaping.py: contains two functions, called "tau_leap_old" and "tau_leap_new". The first function is a basic tau-leaping algorithm implementation that works on NetworkX graphs and is capable of simulating the contact dynamics of the system. The second function is another implementation of tau-leaping but more similar to the one used by the authors of the paper. Calling these functions will execute a single tau-leap.

- SSATANX.py: contains a function called “SSATANX_full”, which is a full implementation of the SSATAN-X algorithm capable of working on NetworkX graphs. Calling this function will execute a single time step.

To make sure that our implementation was functional, we animated a simulation of a randomly generated graph of 10 nodes, 1 of which was initially considered infected (Figure 1). The contact and epidemic dynamics of the systems were updated using the SSA algorithm.

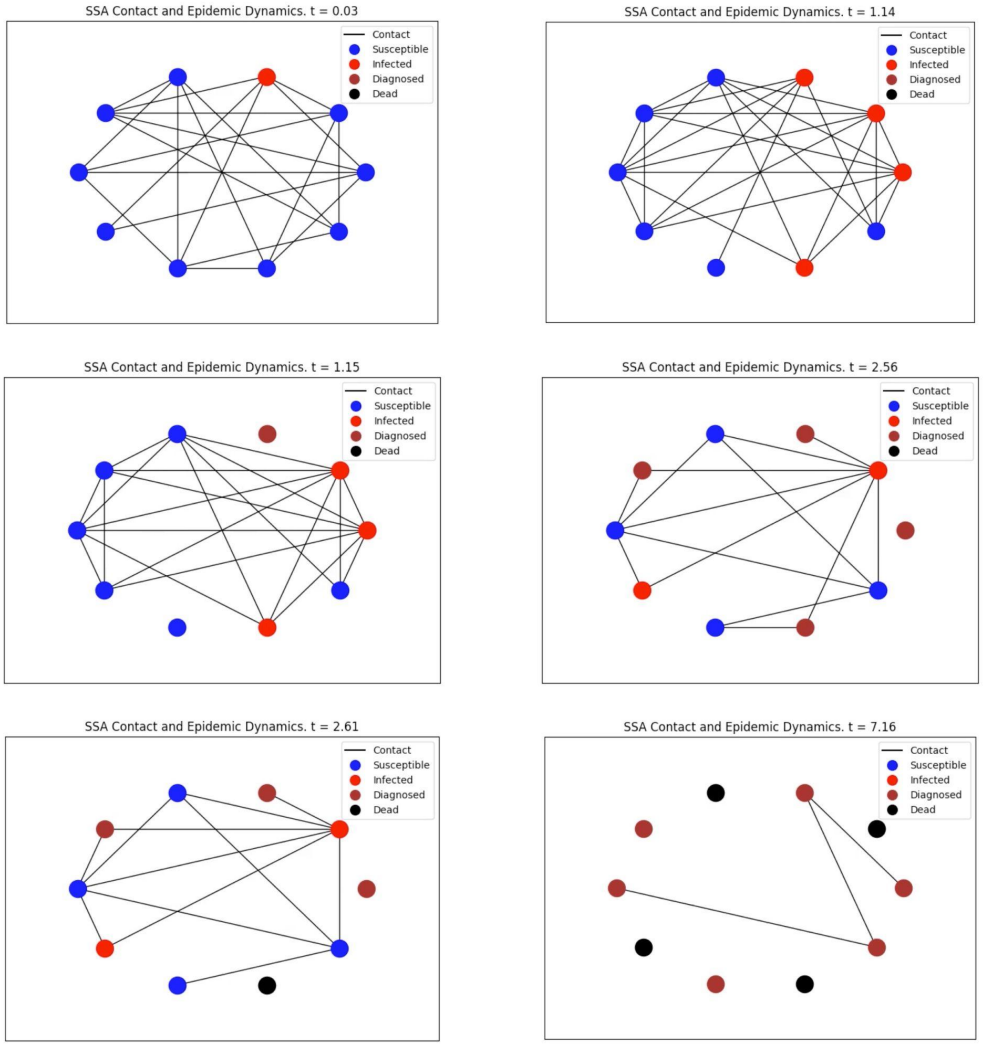


Figure 1. SSA Contact and Epidemic Dynamics simulation

The proposed simulation shows the expected behavior of the system, and confirms the validity of our implementation. This allows us to perform additional analyses and confrontations with the SSATAN-X algorithm. Two different versions of the latter are implemented: one employing a similar version of the optimized tau-leaping algorithm, also used by the authors in the paper, and another one employing an alternative, simplified version constructed following the instructions found in literature.
The main difference between the two tau-leaping implementations is that, in the simplified version, the number of reactions fired in the considered time step is always sampled from a Poisson distribution. Moreover, in the optimized version, tau is modified even when the leap is accepted but would have failed for a slightly lower value of the error parameter ε.
Contrary to what the authors have done, we chose not to use a complementary network in the

optimized tau-leaping algorithm. For this reason, our implementation ends up being slightly slower than what is reported in the paper, but it is more understandable and less memory-hungry.
To obtain more accurate results, our measurements have been obtained by averaging the data of 12 simulations, which is the highest number of simulations we could compute in parallel given the instruments at our disposal. We are aware that it is a much lower number than the 1000 simulations considered by the authors, meaning that our results have a bigger margin of error than the ones reported in the paper.

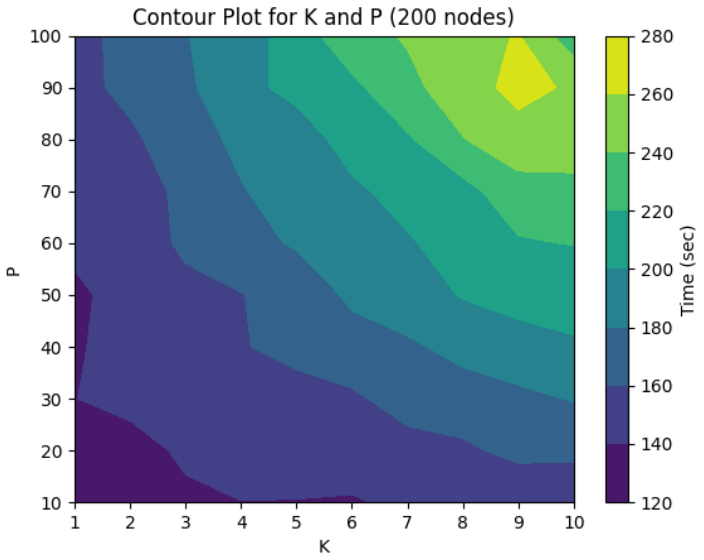


Figure 2: Contour Plot

In our implementation we had to choose appropriate values for the parameters K, which in tau-leaping determines the threshold for τ under which it is faster to use SSA instead, and P, which determines the number of simulated SSA reactions executed when we are under said threshold. These two parameters have been chosen through the analysis of a contour plot shown in Figure 2. This plot gave us a general view of the considered parameter space: there seems to be a local maximum near K = 9 and P = 90, which makes the choice of the authors, being K = 10 and P = 100, not the best fit for our implementation.
We have therefore arbitrarily chosen to use K = 2 and P = 20 for our simulations.
Building the contour plot ended up being highly inefficient, as it required the computation of 1200 simulations. A better way of optimizing the two parameters would have been using a gradient descent algorithm, such as line search, in conjunction with gradient approximation.
It is important to point out that these two parameters, if chosen reasonably, only affect the speed of the algorithm without significantly undermining its accuracy.

SSATAN-X accurately captures contact network dynamics

First of all, the authors evaluated the simulations with pure contact dynamics (no epidemic events occur) to assess whether the statistics remained correct between the SSA algorithm and the SSATAN-X.

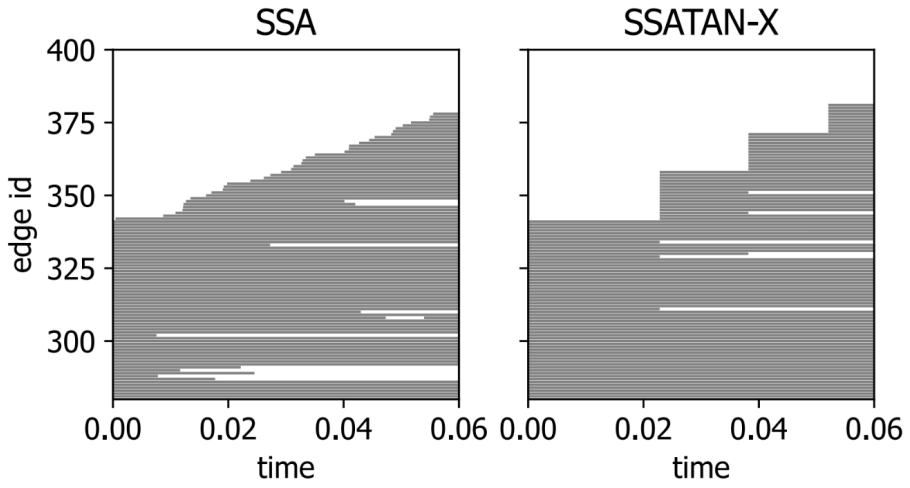


Figure 3. Edge activity plot
Edges of the network depicted in the order of appearance. Each edge is drawn as a bar for the duration of its existence.

As shown in the Figure 3, while the SSA computes each single reaction event one-by-one, the SSATAN-X performs bulk-updates of the contact network while preserving the ability to capture the correct statistics even if it considers larger time-steps.
The degree distribution of the contact network was also evaluated:

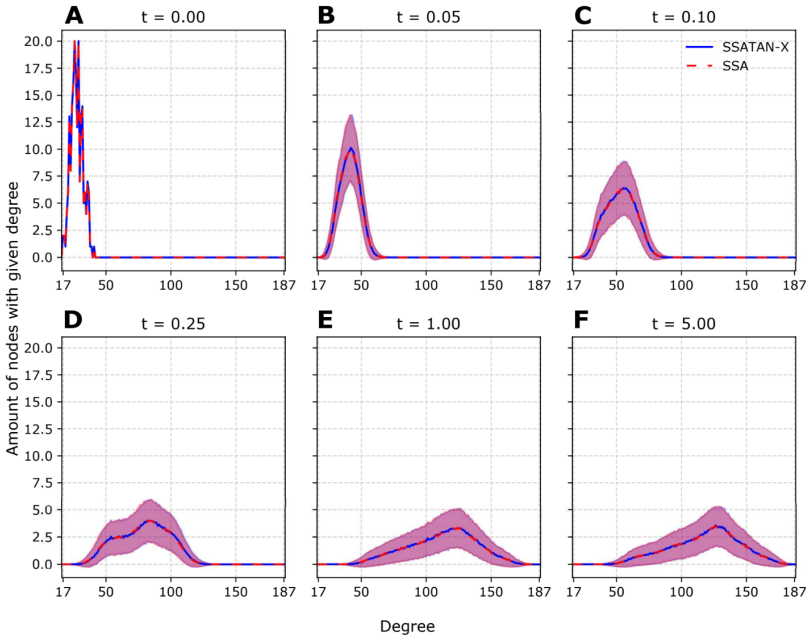


Figure 4. Degree distributions during simulation
The graphic depicts the mean degree distribution of the network during distinct time points within the simulation time.

As can be seen, during the simulation, the degree distribution of the temporal network changes (Fig. 4B–D), until it reaches a stable distribution (Fig. 4E–F). During the entire simulation interval though, both the mean degree distribution (solid blue vs. dashed red lines) and the standard deviations (blue and red shading) remain visually indistinguishable between the two algorithms. This is in further support of the persisting correct statistics between the two algorithms.

Finally, the p-value of the Kolmogorov-Smirnov Test (KST) was computed. The KS test quantifies the distance between the empirical distribution function of two samples. The null hypothesis is that the two samples are drawn from the same distribution. In the paper, the p-value remains close to one throughout the time interval considered, meaning that the two differently simulated contact networks using SSA and SSATAN-X are identical.

SSATAN-X accurately captures epidemic dynamics

The SSATAN-X algorithm is then applied to the adaptive S-I-D model to evaluate the statistics of the epidemic dynamics. Also in this case, both algorithms yield indistinguishable statistics reaffirming the correctness of the SSATAN-X. As can be seen from the histogram in Figure 5, the frequency of the time-step sizes to the next epidemic event of both algorithms remains quite the same.

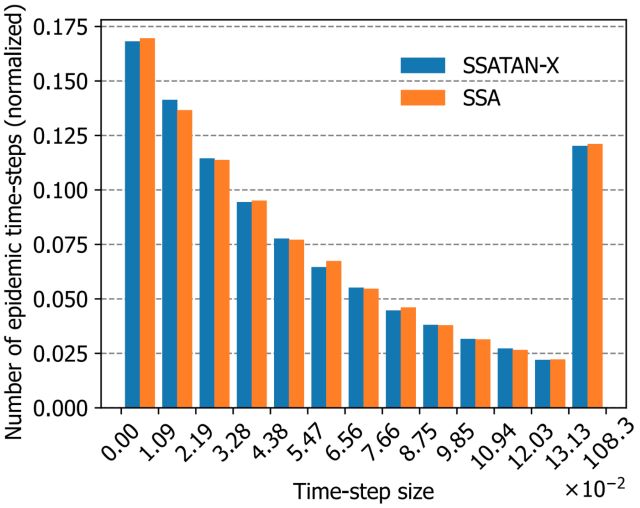


Figure 5. Time-step to the next epidemic event

Figure 6 shows the mean number and standard deviation of susceptible, infected and diagnosed individuals; for each considered subgroup, both statistics are indistinguishable between the two algorithms over the entire duration of the simulation.

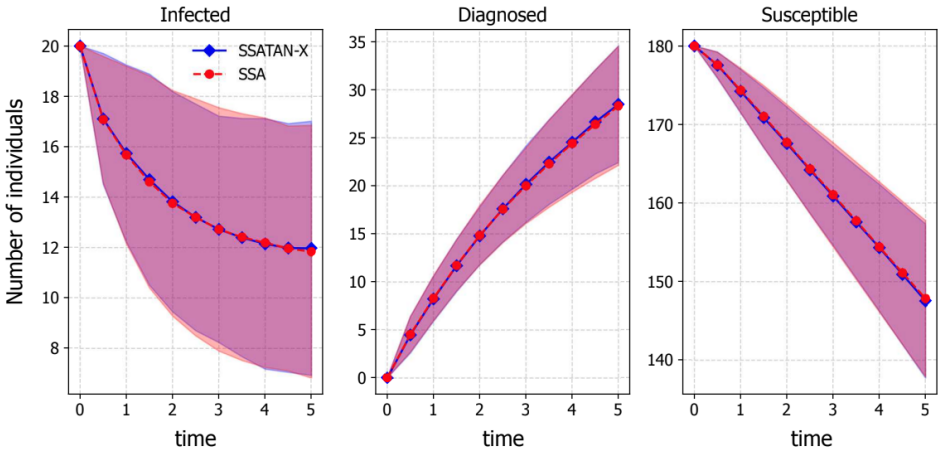


Figure 6. Simulated infection dynamics

To assess the validity and correctness of our proposed implementations we conducted a similar analysis, whose results are reported in Figure 7.

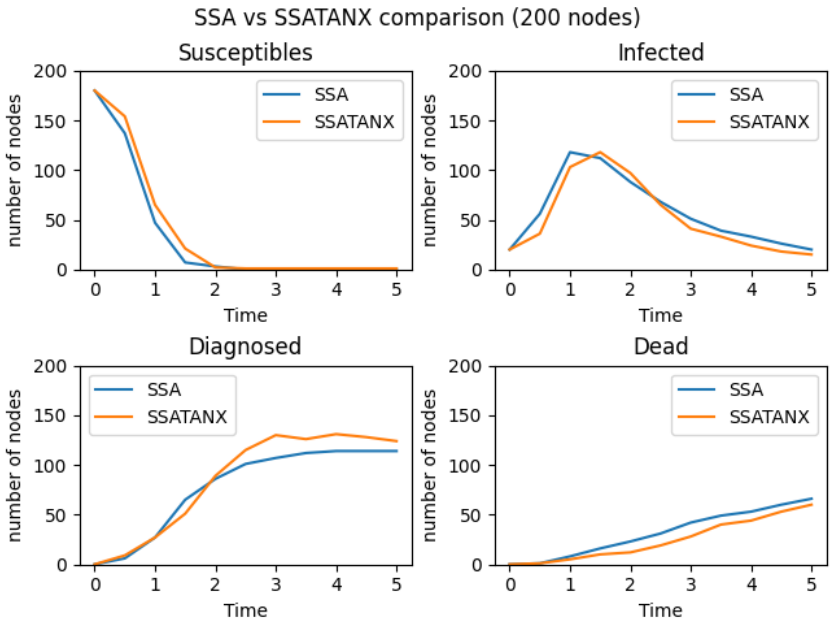


Figure 7. Proposed simulated infected dynamics

As can be seen also for our implementations the statistics for each considered subgroup remains quite the same between the two algorithms.

SSATAN-X speeds up computation time

The authors also analyzed the run time of the SSATAN-X for different population sizes, namely 200, 500 and 1000, to compare it with SSA.

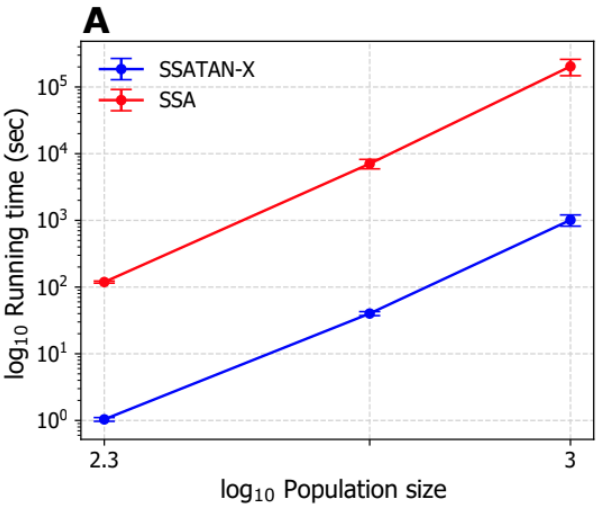


Figure 8. Run time performance

The Figure 8, as reported in the paper, shows that SSATAN-X is consistently 100-times faster than SSA at any population size.
To evaluate the computational speed of our version of the algorithms, we performed a similar analysis but considering a population of 200 individuals and 3000 initial contacts. In our model 10% of the initial population is already infected and we chose to use different parameters that were inferred, as previously stated, to better optimize the performance. Again, we remind that we chose to perform a total of 12 simulations for each algorithm differently from the 1000 performed in the paper.

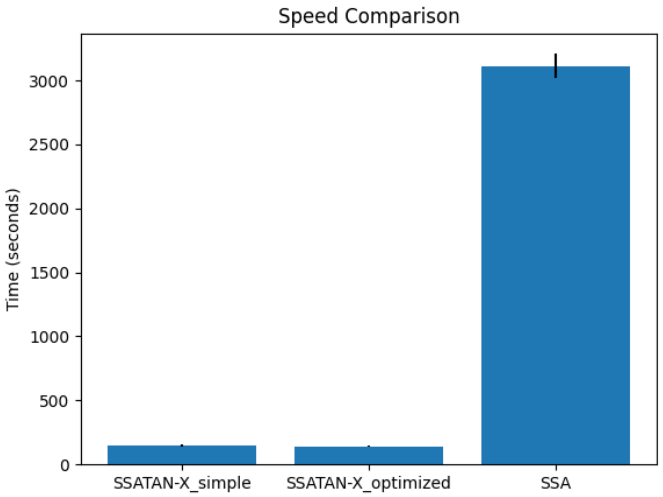


Figure 9. Run-time comparison

Below are reported the results of our analyses:

SSATAN-X with simple tau-leaping	SSATAN-X with optimized tau-leaping	SSA
140.02 ± 8.09 seconds	131.55 ± 4.41 seconds	3013.74 ± 94.24 seconds

In our results, the two implementations of the SSATAN-X algorithm do not show a statistically significant difference in terms of speed. This is due to the fact that the optimizations used by the authors do not affect the slowest step in our implementations of tau-leaping, which is finding the next reaction in the array of cumulative sums of the propensities. We theorize that this issue could be solved by using a more complex data structure than a one-dimensional array, even though it would require more pre-processing steps. Another solution would be sorting the propensities in such a way that the reactions we predict to be more prone to fire will be found in the array in a lesser time than the other ones.
In any case, both our implementations of SSATAN-X managed to be about 20 times faster than the SSA implementation.

Conclusions

To conduct a conclusive analysis of the Stochastic Simulation Algorithm for Effective Time-evolving Adaptive NetworX (SSATAN-X), as stated by the authors and as shown in the results, it seems that the core idea of using bulk-updates of only the contact dynamics partially solves the computational overhead problem, which occurs using the simplest version of SSA and which slows down the performance, without affecting the accuracy.
The speed-up behavior is confirmed, also on the basis of the results we obtained from our version of the SSATAN-X, even if limited by the number of computed simulations.
We noticed that the main bottleneck of SSATAN-X consists in the search for the next reaction firing, for which a possible improvement could be the employment of algorithms exploiting specific data structures for the optimization of this search step (such as tree-based search), or sorting the reaction propensities in a more efficient way.
Additionally, as suggested though, SSATAN-X may be extended to allow for more detailed modeling of interventions of pathogen transmission, as well as regarding the contact dynamics.

References:

1. https://www.cdc.gov/
2. Brauer F. Mathematical epidemiology: Past, present, and future. Infect Dis Model. 2017 Feb 4;2(2):113-127. doi: 10.1016/j.idm.2017.02.001.
3. Gross T, Blasius B. Adaptive coevolutionary networks: a review. J R Soc Interface. 2008 Mar 6;5(20):259-71. doi: 10.1098/rsif.2007.1229.
4. Springer International Publishing AG 2017 225 L. Marchetti et al., Simulation Algorithms for Computational Systems Biology, Texts in Theoretical Computer Science. An EATCS Series, https://doi.org/10.1007/978-3-319-63113-4