SHARETRACE: CONTACT TRACING WITH THE ACTOR MODEL

by

RYAN TATTON

Submitted to the Department of Computer and Data Sciences
in partial fulfillment of the requirements for the degree of

Master of Science in Computer Science

at the

CASE WESTERN RESERVE UNIVERSITY

August 2022

Case Western Reserve University
School of Graduate Studies

We hereby approve the thesis/dissertation of

Ryan Tatton

candidate for the degree of Masters in Computer Science.

Comittee Chair

Erman Ayday

Committee Member

Youngjin Yoo

Committee Member

Harold Connamacher

Committee Member

Michael Lewicki

Date of Defense

31 July 2022

*We also certify that written approval has been obtained for any proprietary material contained therein.

SHARETRACE: CONTACT TRACING WITH THE ACTOR MODEL

by

RYAN TATTON

Submitted to the Department of Computer and Data Sciences
on July 31, 2022, in partial fulfillment of the
requirements for the degree of
Master of Science in Computer Science

Abstract

Proximity-based contact tracing relies on user-device interaction to estimate the spread of disease. ShareTrace is one such approach that has been shown to provide improved efficacy in tracking the spread of disease by also considering direct and indirect forms of contact. In this work, we aim to provide an efficient and scalable formulation of ShareTrace by utilizing asynchronous, concurrent message passing on a temporal graph. We also introduce a unique form of reachability, message reachability (MR), that accounts for the dynamic nature of message passing on the temporal graph. Our evaluation on both synthetic and real-world graphs indicates that correct parameter values optimize for algorithmic accuracy and efficiency. In addition, we demonstrate that MR can accurately estimate the risk of a user on their contacts.

Thesis Supervisor: Erman Ayday
Title: Assistant Professor

Acknowledgments

This work made use of the High Performance Computing Resource in the Core Facility for Advanced Research Computing at Case Western Reserve University.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

ShareTrace is a privacy-preserving contact-tracing solution [5]. Unlike other approaches that rely on device proximity to detect human interaction, ShareTrace executes iterative message passing on a factor graph to estimate a user's marginal posterior probability of infection (MPPI). To indicate its similarity to belief propagation, we refer to the ShareTrace algorithm as *risk propagation*. By considering both direct and indirect contact, [5] demonstrates that risk propagation is more effective than other proximity-based methods that only consider former.

Building upon the efforts by [5], we provide an efficient and scalable formulation of risk propagation[1] that utilizes asynchronous, concurrent message passing on a temporal graph [23, 21]. While message passing has been studied under specific epidemiological models [27, 33], our formulation allows us to contextualize risk propagation as a novel usage of a temporal graph that does not require such assumptions to infer the transmission of disease. As a result, we introduce a form of reachability that can uniquely characterize the dynamics of message passing on a temporal graph. Our formulation of risk propagation aligns with its distributed extension, as introduced by [5], which has connections to the actor model of concurrent computing [6, 2] and the "think-like-a-vertex" model of graph algorithms [37].

[1]`https://github.com/share-trace`

Our evaluation aims to (1) describe the efficiency of risk propagation on both synthetic and real-world temporal graphs; (2) validate the accuracy of our new form of reachability on both synthetic and real-world graphs; (3) and briefly quantify the scalability of this implementation of risk propagation on synthetic graphs. To keep the scope of this work focused, we defer to [5] on the privacy and security aspects of ShareTrace.

Chapter 2

Related Work

Since the beginning of the COVID-19 pandemic, there has been a copious amount of research in mobile contact tracing solutions, most notably being the joint effort by Apple and Google [4]. External reviews and surveys provide extensive comparison of existing solutions through the lenses of privacy, security, ethics, adversarial models, data management, scalability, interoperability, and more. References [3] and [36] provide thorough reviews of existing mobile contact tracing solutions with discussion of the techniques, privacy, security, and adversarial models. The former offers additional detail on the system architecture (i.e., centralized, decentralized, and hybrid), data management, and user concerns of existing solutions. Other notable reviews with similar discussion include [44, 41, 11, 15, 35]. Reference [31] provides a formal framework for defining aspects of privacy for proximity-based contact tracing.

Chapter 3

Proposed Scheme

3.1 Preliminaries

We assume a system model in which each user owns a smart mobile device that has device-proximity detectability (e.g., Bluetooth). Furthermore, we assume that proximal interactions between user devices subsequently allow their devices, or a digital proxy thereof, to exchange messages over several days.

In risk propagation, the computation of infection risk is an inference problem in which the task is to estimate user MPPI. The prior probability of infection is derived from user symptoms [38], so we shall refer to it, along with the timestamp of its computation, as a *symptom score*. Because the posterior probability of infection also considers direct and indirect contact with other users, we call it an *exposure score*. In general, a *risk score* (r, t) is a timestamped probability of infection where $r \in \mathbb{R}_{[0,1]}$ is the *value* of the risk score and $t \in \mathbb{R}_{\geq 0}$ is the *time* of its computation.

Computing the full joint probability distribution is intractable as it scales exponentially with the number of users. To circumvent this challenge, risk propagation uses message passing on a factor graph to efficiently compute the MPPI. Formally, let $G = (\mathcal{V}, \mathcal{F}, \mathcal{E})$ be a factor graph where $\mathcal{V}$ is the set of variables, $\mathcal{F}$ is the set of factors, and $\mathcal{E}$ is the set of edges incident between the sets $\mathcal{V}$ and $\mathcal{F}$ [30]. A factor $f(u, v)$ represents contact between two users $u, v \in \mathcal{V}$ (i.e., variables), such that $f(u, v)$ is adjacent to u, v. Note that in risk propagation, the aim is to maximize individual MPPIs [5]. This contrasts with belief propagation in which the objective is to maximize the full joint distribution [8].

A *message* $m_{u \to v} = \{(r, t), \ldots\}$ sent from some user u to another user v is a nonempty set of risk scores. We assume that contact with others has a non-decreasing effect on the probability of contracting the disease. Thus, risk propagation is similar to the max-sum algorithm in that each user maintains the value of the maximum risk score it receives [8].

The only purpose of a factor is to compute and relay messages between users. Thus, we can apply one-mode projection onto users such that $u, v \in \mathcal{V}$ are adjacent if there exists a factor $f(u, v) \in \mathcal{F}$ [45]. Upon receiving a message from a neighbor, a user first updates its current value. It then uses the details of its *other* contacts to compute and propagate the message. This modification differs from the distributed extension that [5] proposes in that we do not send duplicate messages to factors. By storing the contact time between users on the edge incident to them, this modified topology is identical to the *contact sequence* representation of a temporal graph or contact network, $\mathcal{C} = \{(u, v, t) \mid u, v \in \mathcal{V}; u \neq v; t \in \mathbb{R}_{\geq 0}\}$, where a triple (u, v, t) is called a *contact* [23]. In the context of risk propagation, t is the starting time at which users u, v *most recently* came in contact.

Our usage of a temporal graph differs from its typical usage in epidemiology, which focuses on modeling and analyzing the spreading dynamics of disease [42, 14, 34, 12, 40, 29, 46]. In contrast, we use a temporal graph to infer user MPPI. As a result, we introduce a new form of reachability that synthesizes message-passing and the temporal dynamics of the graph in Section 3.3. As noted by [23], the transmission graph provided by [42] "cannot handle edges where one vertex manages to not catch the disease." Notably, our usage of a temporal graph allows for such cases by modeling the possibility of infection as a continuous outcome.

We utilize the actor model to achieve scalable performance [6, 2]. Let $K \in \mathbb{N}_{>0}$ be the number of actors, where each actor is a subgraph $G_k \in \mathcal{G}$ of users that is induced by a partitioning algorithm [9] or a clustering algorithm [1]. Formally, we apply a surjective function $\sigma : \mathcal{V} \to \mathcal{G}$ that maps each user to exactly one subgraph or actor. Actors communicate with each other via message passing. Typically, inter-actor communication is slower than intra-actor communication, so using an algorithm that minimizes communication overhead between actors is key to maximizing performance.

We associate with each actor a unique identifier, its *mailing address*, and a buffer, its *remote mailbox*, for storing messages received from other actors. In practice, each actor also has a separate *local mailbox* that it is uses to manage communication between its own users. This local mailbox incurs less overhead than the remote mailbox since the latter typically involves the usage of concurrent primitives (e.g., a lock). To send a message, we must know the mailing address of the receiving actor and the identity of the receiving user. If the mailing address of the sending actor is the same as the receiving actor, then the message is placed in its local mailbox. Otherwise, the message is placed in the remote mailbox of the receiving actor. In addition to maintaining the state of all of the users in its subgraph, an actor also keeps a mapping between mailing addresses and remote mailboxes for all other actors. We do not know *a priori* which actors will need to communicate with each other before partitioning the graph, so we allow an actor the ability to communicate with all other actors.

3.2 Algorithms

Algorithm 1 defines the main message-passing procedure. We constrain the set of initial risk scores $\mathcal{S} = \{(r, t) \mid t_{\text{now}} - t \leq L\}$ to those that were computed within the last $L = 14$ days, which assumes that a risk score has finite

Algorithm 1 Risk Propagation, Main.

```
1. Create the graph $G$: for each contact $(u, v, t) \in \mathcal{C}$, add an edge between
   users $u, v$ and store the contact time $t$.
2. Partition $G$ into $K$ disjoint subgraphs (actors) w.r.t. a partitioning/clustering function $\sigma(\cdot)$.
3. Partition the initial risk scores $\mathcal{S}$ w.r.t. $\sigma(G)$.
4. Send $\mathcal{S}_k$ to $G_k$ for each $k \in \mathbb{N}_{[1,K]}$.
5. Collect all exposure scores $\mathcal{R} \equiv \cup_k \mathcal{R}_k$.
```

relevance. We constrain the set of contacts $\mathcal{C}$ similarly. Note that the initial risk scores of a user u, denoted $\mathcal{S}(u)$, includes the exposure scores from the last L days and its most recently computed symptom score.

Algorithm 2 describes the behavior of an actor. As in [5], we assume that risk transmission is incomplete by applying a transmission rate of $\alpha = 0.8 \in \mathbb{R}_{(0,1)}$ [19]. Step 2.3c (i.e., step 3c of Algorithm 2) follows from belief propagation in that we marginalize over the factor $f(u, v)$. Because message passing is concurrent, we cannot rely on a global iteration as a stopping criterion, as used by [5]. While convenient, such a criterion requires a synchronization barrier when used in concurrent settings, which can degrade performance [20].

Algorithm 3 describes how we compute and send a message. As indicated by step 2.2, the message $m_{v \to u}$ in step 3.1 is initially the risk scores of user u. Thus, $u = v$ only when sending the first message to each neighbor v'. For all subsequent messages, $u \neq v$ and $m_{v \to u}$ is a singleton that contains the risk score sent from neighbor v. For a singleton message $m_{v \to u}$, we refer to the value and time of the contained score as $r_{v \to u}$ and $t_{v \to u}$, respectively.

In step 3.1, we include a time buffer $B \in \mathbb{R}_{\geq 0}$ of 2 days to account for the possibility that the onset of infection precedes symptoms. We assume that all risk scores with a time later than the buffered time of contact are irrelevant. Assuming that we run risk propagation at least every B days, it is not necessary to persist contacts (i.e., edges) that are older than B days.

Algorithm 2 Risk Propagation, Actor (Main).

```
1. Upon receiving $\mathcal{S}_k$, for each user $u \in G_k$, let
    (a) $m_0(u) = (r_0(u), t_0(u))$ be the initial message; the maximum risk
        score of $u$, scaled by $\alpha$; and
    (b) $c(u)$ be the current value; initially, $\max(\mathcal{S}(u))$.
2. For each user $u \in G_k$, compute and send the message $m_{u \to v}$ using $\mathcal{S}(u)$,
   for each neighbor $v \in \text{ne}(u)$.
3. While a message has been received within $T$ seconds,
    (a) Receive $m_{v \to u}$ s.t. $u \in G_k$ and $v \in \text{ne}(u)$.
    (b) Update user $u$: $c(u) \leftarrow \max(r_{v \to u}, c(u))$ .
    (c) For each $v' \in \text{ne}(u) \setminus v$, compute and send $m_{u \to v'}$.
4. Collect exposure scores: $\mathcal{R}_k \equiv \{(c(u), t_{\text{now}}) \mid u \in G_k\}$.
```

For a given user u and neighbor v, it is impossible for u to send v a risk score higher in value than what it previously sent if it has been more than B days after their most recent time of contact. In other words, the MPPI of user v will already account for any risk score of user u after B days of coming in contact. In this way, we can further improve the efficiency of risk propagation by reducing the communication overhead.

The final aspect of Algorithm 3 is to determine if we should send a computed message. Because we only use contact time as a filter to determine which risk scores to consider, we only need to compare the most recent contact time in step 3.1. That is, given two times of contact t_1, t_2 such that $t_1 \leq t_2$, then any risk score with time $t \leq t_1 + B$ also satisfies $t \leq t_2 + B$. This avoids comparing multiple contact times, as suggested by [5].

The following approach differs from [5] in that we allow no message to be sent, as opposed to sending a "null" message with a risk score value of 0. Avoiding ineffective messages helps lower the communication overhead. The purpose of sending a message is to possibly update the value of some other user in the graph. Sending a risk score with a lower value than the current value of

Algorithm 3 Risk Propagation, Actor (Message).

```
1. Consider only the risk scores in the message $m_{v \to u}$ that may have been
   transmitted:
       $m'_{v \to u} \leftarrow \{(r, t) \mid t \leq t_{uv} + B\}$.
2. Compute the time difference for each remaining score:
       $\Delta \leftarrow \{(r, t, \delta) \mid \delta = \min(t - t_{uv}, 0)\}$.
3. Compute the maximum weighted message:
       $m_{u \to v'} \leftarrow \arg\max_{m \in \Delta} \{\log(r) + \delta/\tau\}$.
4. Scale by the transmission rate: $r_{u \to v'} \leftarrow \alpha \cdot r_{u \to v'}$.
5. Send $m_{u \to v'}$ if $r_{u \to v'} \geq \gamma \cdot r_0(u)$ and $t_{u \to v'} \leq t_0(u)$.
```

the receiving user will not change its value. However, depending on its time, the receiving user may still propagate a message which subsequently results in an update to some other reachable user in the graph. Because we scale the value of a sent risk score by α, it exponentially decreases as it propagates through the graph with a rate constant $\log(\alpha)$. Furthermore, due to how we filter in step 3.1, it is possible that a risk score with a lower value than what a user previously sent can update the value of another user, if it is sufficiently old.

We combine both of these aspects into a heuristic that allows us to parametrize the trade-off between completeness and efficiency. Let $\gamma \in \mathbb{R}_{[0,1]}$ be the *send tolerance* such that we only send a message $m_{u \to v}$ if $r_{u \to v} \geq \gamma \cdot r_0(u)$. In addition to comparing the value, we must also compare the time to the initial message. Assuming a message satisfies the value condition, then a newer message is less likely to be propagated. Hence, it is only useful to send a message if it is at least as old as the initial message. This send condition is expressed in step 3.5. If $\gamma > 0$, this send condition will eventually cause actors to stop passing messages.

3.3 Message Reachability

A fundamental concept in reachability analysis on a temporal graph is a *time-respecting path*, which is defined as a contiguous sequence of contacts with non-decreasing time. Thus, node v is *temporally reachable* [39] from node u if there exists a time-respecting path from u to v.

In risk propagation, message passing operates on a looser constraint than temporal reachability. In this way, we can define *message reachability* (MR) to be the reachability of an initial risk score for a given user u, denoted $m(u)$. Using the Heaviside step function,

$$H(x) := \begin{cases} 1, & x \geq 0 \\ 0, & x < 0 \end{cases},$$

$$m(u) := \max_{\mathcal{P}} \sum_{(i,j) \in \mathcal{P}} H_c(u,i,j) \cdot H_r(u,i) \cdot H_t(u,i) \tag{3.1}$$

where $\mathcal{P}$ is the set of edges along the simple path $u \to v$ such that the users are enumerated as $0, 1, \ldots, |\mathcal{P}| - 1$,

$$H_c(u,i,j) := H(t_{ij} + B - t_0(u)) \tag{3.2}$$

is the contact-time constraint in step 3.1, and

$$H_r(u,i) := H(\alpha^i \cdot r_0(u) - \gamma \cdot r_0(i)) \tag{3.3}$$

$$H_t(u,i) := H(t_0(i) - t_0(u)) \tag{3.4}$$

are the respective value and time constraints in step 3.5.

A user v is *message reachable* from user u if there exists a path $u \to v$ such that $m(u) > 0$. Because we constrain risk scores to be at most $L \geq B$ days old, $m(u) \geq 1$ for any non-isolated u. We can find the value of (3.1) by applying an augmented shortest-path algorithm [25] such that we start at user

u and iteratively propagate its initial message $m_0(u)$.

By relaxing (3.2) and (3.4), we can define an upper bound on (3.1). For some reachable user v, the *estimated message reachability* of a user u is bounded by

$$\hat{m}(u) \leq 1 + \log_\alpha \left(\gamma \cdot \frac{r_0(v)}{r_0(u)} \right), \tag{3.5}$$

where $\hat{m}(u) = 0$ if $r_0(u) = 0$ and $\hat{m}(u) = \infty$ if $r_0(v) = 0$.

MR is a measure for estimating the size of the induced subgraph (i.e., set of users) that can be impacted by the risk of a user. From the perspective of efficiency, it indicates that a lower send tolerance will generally result in higher MR, at the cost of computing and passing ineffective messages. MR also allows us to quantify the effect of the transmission rate. Unlike send tolerance, the transmission rate is intended to be derived from epidemiology to quantify disease infectivity. Thus, MR allows us to characterize the propagation of risk as a dynamic process on a temporal graph [7].

Chapter 4

Evaluation

4.1 Experimental Design

Risk propagation requires a partitioning or clustering algorithm, as described in Algorithm 1. We configured the METIS graph partitioning algorithm [28] to use k-way partitioning with a load imbalance factor of 0.2, to attempt contiguous partitions that have minimal inter-partition connectivity, to apply 10 iterations of refinement during each stage of the uncoarsening process, and to use the best of 3 cuts.

4.1.1 Synthetic Graphs

We evaluate the scalability and efficiency of risk propagation on three types of graphs: a random geometric graph (RGG) [13], a benchmark graph (LFRG) [32], and a clustered scale-free graph (CSFG) [22]. Together, these graphs demonstrate some aspects of community structure [16] which allows us to more accurately measure the performance of risk propagation. When constructing a RGG, we set the radius to $r(n) = \min\left(1, 0.25^{\log_{10}(n)-1}\right)$, where n is the number of users. This allows us to scale the size of the graph while maintaining reasonable density. We use the following parameter values to create LFRGs: mixing parameter $\mu = 0.1$, degree power-law exponent $\gamma = 3$, community size power-law exponent $\beta = 2$, degree bounds $(k_{\min}, k_{\max}) = (3, 50)$, and community size bounds $(s_{\min}, s_{\max}) = (10, 100)$. Our choices align with the suggestions by [32] in that $\gamma \in \mathbb{R}_{[2,3]}$, $\beta \in \mathbb{R}_{[1,2]}$, $k_{\min} < s_{\min}$, and $k_{\max} < s_{\max}$. To build CSFGs, we add $m = 2$ edges for each new user and use a triad formulation probability of $P_t = 0.95$. For all graphs, we remove self-loops and isolated users.

The following defines our data generation process. Let p be the probability of a user being "high risk" (i.e., $r \geq 0.5$) Then, with probability $p = 0.2$, we sample $L + 1$ values from the uniform distribution $\mathbb{U}_{[0.5,1)}$. Otherwise, we sample from $\mathbb{U}_{[0,0.5)}$. This assumes symptom scores and exposure scores are computed daily and includes the present day. We generate the times of these risk scores by sampling a time offset $t_{\text{off}} \sim \mathbb{U}_{[0\text{s};86{,}400\text{s}]}$ for each user such that $t_d = t_{\text{now}} + t_{\text{off}} - d$ days, where $d \in \mathbb{N}_{[0,L]}$. To generate a contact times, we follow the same procedure for risk scores, except that we randomly sample one of the $L + 1$ times and use that as the contact time.

We evaluate various transmission rates and send tolerances:

$$(\gamma, \alpha) \in \{0.1, 0.2, \ldots, 1\} \times \{0.1, 0.2, \ldots, 0.9\}.$$

For all γ, α, we set $n = 5{,}000$ and $K = 2$.

To measure the scalability of risk propagation, we consider $n \in \mathbb{N}_{[10^2,10^4]}$ users in increments of 100 and collect 10 iterations for each n. The number of actors we use depends on n such that $K(n) = 1$ if $n < 10^3$ and $K(n) = 2$ otherwise. Increasing K for our choice of n did not offer improved performance due to the communication overhead.

4.1.2 Real-World Graphs

We analyze the efficiency of risk propagation on three real-world contact networks that were collected through the SocioPatterns collaboration. Specifically, we use contact data in the setting of a high school (Thiers13) [17], a workplace (InVS15), and a scientific conference (SFHH) [18]. Because of limited availability of large-scale contact networks, we do not use real-world contact networks to measure the scalability of risk propagation.

To ensure that all risk scores are initially propagated, we shift all contact times forward by t_{now} and use ($t_{\text{now}} - 1$ day) when generating risk scores times. In this way, we ensure the most recent risk score is still older than the first contact time. Risk score values are generated in the same manner as described in Section 4.1.1 with the exception that we only generate one score. Lastly, we perform 10 iterations over each data set to obtain an average performance.

4.2 Results

4.2.1 Efficiency

Prior to measuring scalability and real-world performance, we observed the effects of send tolerance and transmission rate on the efficiency of risk propagation. As ground truth, we used the maximum update count for a given transmission rate. Fig. 4-1 indicates that a send tolerance of $\gamma = 0.6$ permits 99% of the possible updates. Beyond $\gamma = 0.6$, however, the transmission rate has considerable impact, regardless of the graph. As noted in Section 3.3, send tolerance quantifies the trade-off between completeness and efficiency. Thus, $\gamma = 0.6$ optimizes for both criteria.

Unlike the update count, Fig. 4-1 shows a more variable relationship with respect to runtime and message count. While, in general, transmission rate (send tolerance) has a direct (resp. inverse) relationship with runtime and message count, the graph topology seems to have an impact on this fact. Namely, the LFRG displayed less variability across send tolerance and transmission rate than the RGG and CSFG, which is the cause for the large interquartile ranges. Therefore, it is useful to consider the lower quartile Q_1, the median Q_2, and the upper quartile Q_3. For $\alpha = 0.8$ and $\gamma = 0.6$, risk propagation is more efficient with $(Q_1, Q_2, Q_3) = (0.13, 0.13, 0.46)$ normalized runtime and $(Q_1, Q_2, Q_3) = (0.13, 0.15, 0.44)$ normalized message count.

4.2.2 Message Reachability

To validate the accuracy of (3.5), we collected values of (3.1) and (3.5) for real-world and synthetic graphs. For the latter set of graphs, we observed reachability while sweeping across values of γ and α.

To measure the accuracy of (3.5), let the *message reachability ratio* (MRR) be defined as

$$\mathrm{mrr}(u) := \frac{m(u)}{\hat{m}(u)}. \tag{4.1}$$

Overall, (3.5) is a good estimator of (3.1). Across all synthetic graphs, (3.5) modestly underestimated (3.1) with quartiles $(Q_1, Q_2, Q_3) = (0.71, 0.84, 0.98)$ for the (4.1). For $\alpha = 0.8$ and $\gamma = 0.6$, the quartiles of (4.1) were $(Q_1, Q_2, Q_3) = (0.52, 0.77, 1.12)$ and $(Q_1, Q_2, Q_3) = (0.79, 0.84, 0.93)$, respectively. Table 4.1 provides mean values of (4.1) for both synthetic and real-world graphs. Fig. 4-2 indicates that moderate values of γ tend to result in a more stable MRR, with lower (higher) γ underestimating (resp. overestimating) (3.1). With regard to transmission rate, (4.1) tends to decrease with increasing α, but also exhibits larger interquartile ranges.

Because (3.5) does not account for the temporality constraints (3.2) and (3.4), it does not perfectly estimate (3.1). With lower γ and higher α, (3.5) suggests higher MR. However, because a message is only passed under certain conditions (see Algorithm 3), this causes (3.5) to overestimate (3.1). While (3.5) theoretically is an upper bound on (3.1), it is possible for (3.5) to underestimate (3.1) if the specified value of $r_0(v)$ overestimates the true value of $r_0(v)$. When computing (4.1) for Fig. 4-2, we used the mean $r_0(v)$ across all users v, so $\mathrm{mrr}(u) > 1$ in some cases.

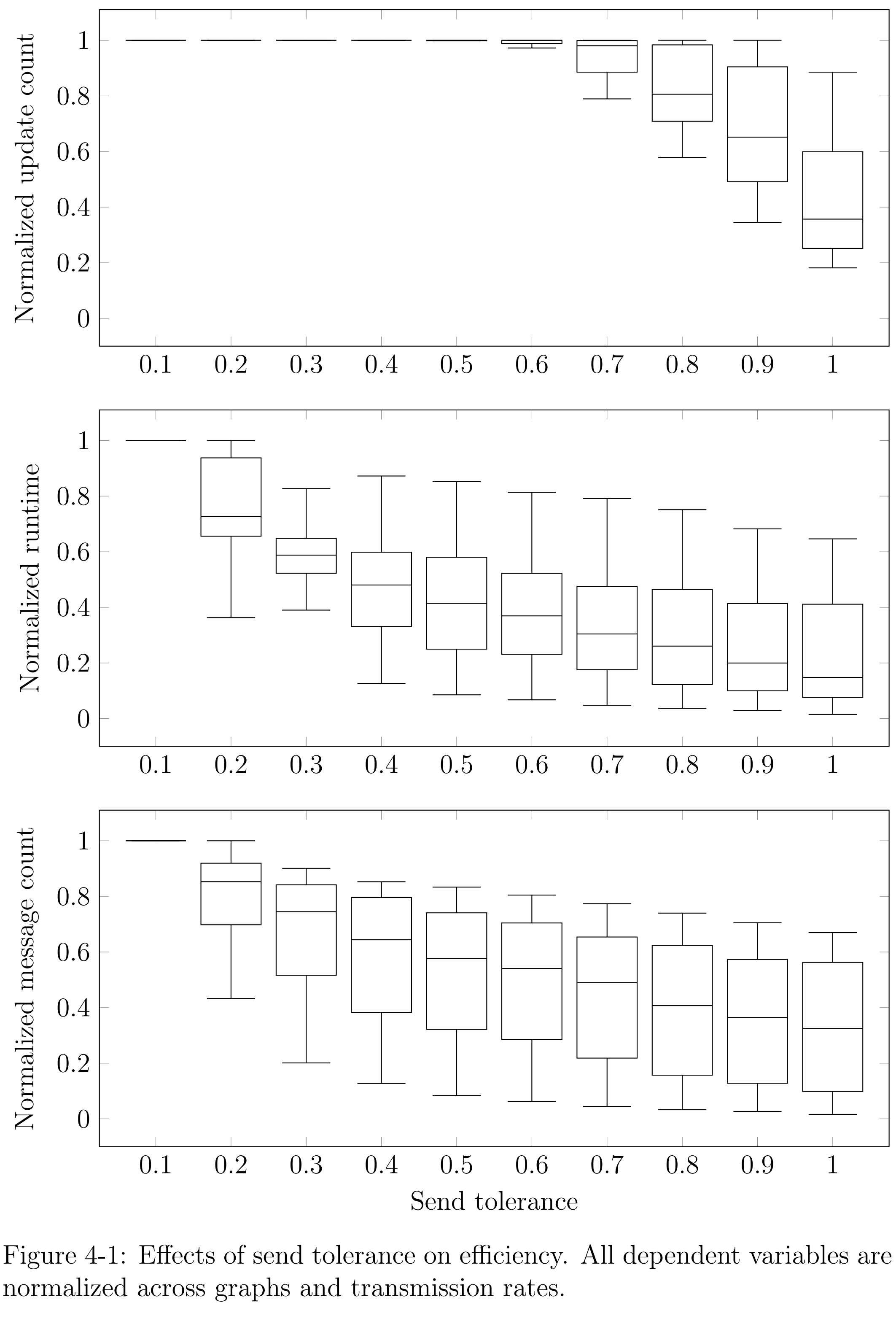

Figure 4-1: Effects of send tolerance on efficiency. All dependent variables are normalized across graphs and transmission rates.

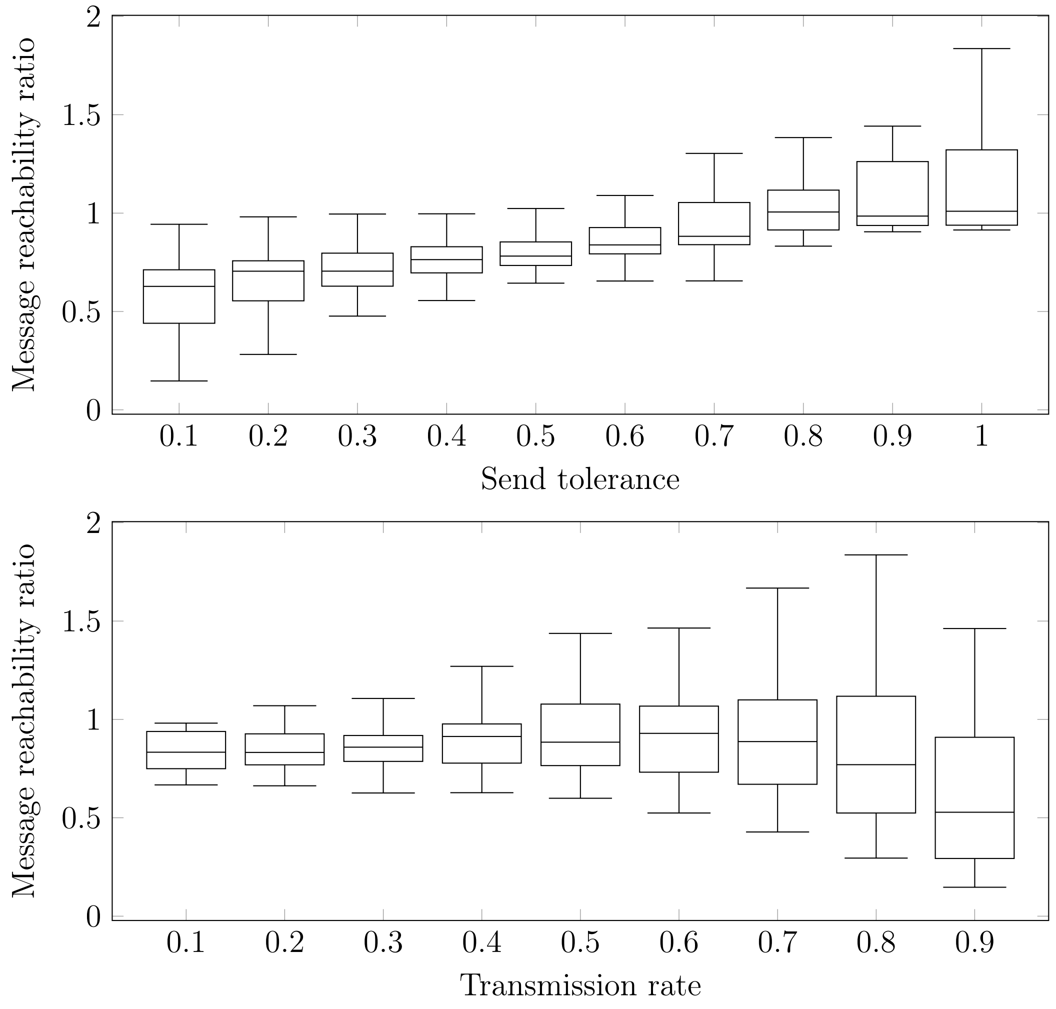

Figure 4-2: Effects of send tolerance and transmission rate on the MRR. Independent variables are grouped across graphs.

Setting	mrr(u) $\pm 1.96 \cdot$ SE
Synthetic	
LFR	0.88 ± 0.14
RGG	0.74 ± 0.12
CSFG	0.90 ± 0.14
	$\mathbf{0.85 \pm 0.08}$
Real-world	
Thiers13	0.58 ± 0.01
InVS15	0.63 ± 0.01
SFHH	0.60 ± 0.01
	$\mathbf{0.60 \pm 0.01}$

Table 4.1: Message reachability ratio for synthetic and real-world graphs ($\alpha = 0.8$, $\gamma = 0.6$). Synthetic (real-world) ratios are averaged across parameter combinations (resp. runs).

4.2.3 Scalability

Fig. 4-3 describes the runtime behavior of risk propagation. The runtime of CSFGs requires further investigation. A linear regression fit explains ($R^2 = 0.52$) the runtime of LFRGs and RGGs with a slope $m = (1.1 \pm 0.1) \cdot 10^{-3}$ s/contact and intercept $b = 4.3 \pm 1.6$s ($\pm 1.96 \cdot$ SE).

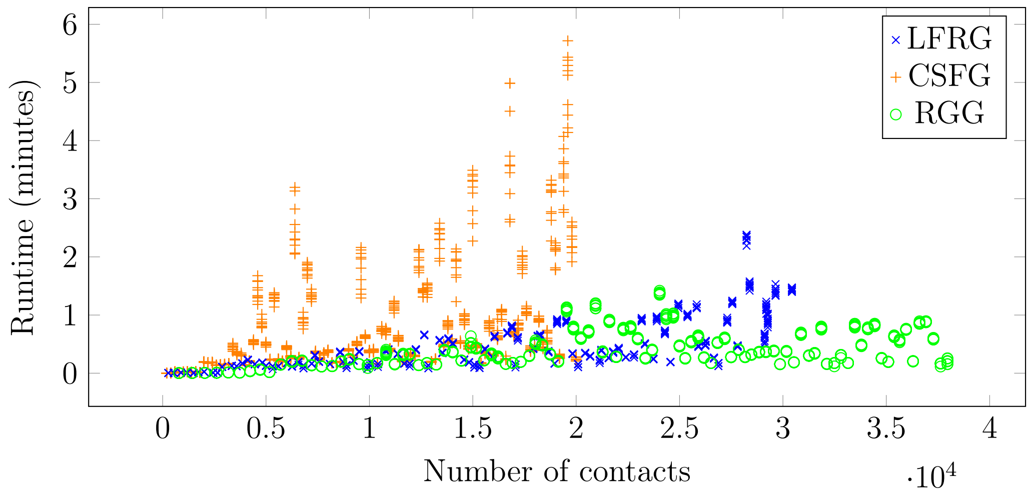

Figure 4-3: Runtime of risk propagation on synthetic graphs containing 100–10,000 users and approximately 200–38,000 contacts.

Chapter 5

Conclusions

Applications like ShareTrace are fundamentally collaborative in that users exchange data amongst each other to achieve an objective or gain personal utility. Maintaining personal data ownership and privacy in this collaborative setting while ensuring architectural scalability and security is an ongoing challenge in the fields of machine learning and cloud computing [10, 24, 26, 43]. Our formulation of risk propagation offers scalability and efficiency and is thus a viable candidate for real-world usage to estimate the spread of infectious diseases. Moreover, message reachability provides researchers and system designers the ability to quantify both the risk of an individual and the effects parameter values have on the efficiency and accuracy of risk propagation.

Bibliography

[1] Charu C. Aggarwal and Haixun Wang. A survey of clustering algorithms for graph data. In Ahmed K. Elmagarmid and Amit P. Sheth, editors, *Manag. Min. Graph Data*, volume 40 of *Adv. Database Syst.* Springer, 2010.

[2] Gul Abdulnabi Agha. *Actors: A Model of Concurrent Computation in Distributed Systems.* MIT Press, 1986.

[3] Nadeem Ahmed, Regio A. Michelin, Wanli Xue, Sushmita Ruj, Robert Malaney, Salil S. Kanhere, Aruna Seneviratne, Wen Hu, Helge Janicke, and Sanjay Jha. A survey of COVID-19 contact tracing apps. *IEEE Access*, 8, 2020.

[4] Apple Inc. and Google LLC. Privacy-preserving contact tracing, 2021.

[5] Erman Ayday, Youngjin Yoo, and Anisa Halimi. ShareTrace: An iterative message passing algorithm for efficient and effective disease risk assessment on an interaction graph. In *Proc. 12th ACM Con. Bioinformatics, Comput. Biology, Health Inform.*, BCB 2021, 2021.

[6] Henry Baker and Carl Hewitt. Laws for communicating parallel processes. Technical report, Massachusetts Institute of Technology, 1977.

[7] Alain Barrat and Ciro Cattuto. Temporal networks of face-to-face human interactions. In Petter Holme and Jari Saramäki, editors, *Temporal Netw.*, Underst. Complex Syst. Springer, 2013.

[8] Christopher M. Bishop. Pattern recognition and machine learning. In M. I. Jordan, Robert Nowak, and Bernhard Schoelkopf, editors, *Inf. Sci. Stat.* Springer, 2006.

[9] Aydın Buluç, Henning Meyerhenke, Ilya Safro, Peter Sanders, and Christian Schulz. Recent advances in graph partitioning. In Lasse Kliemann and Peter Sanders, editors, *Algorithm Eng.: Sel. Results Surveys.* Springer, 2016.

[10] Ignacio Cano, Dhruv Mahajan, Giovanni Matteo Fumarola, Arvind Krishnamurthy, Markus Weimer, and Carlo Curino. Towards geo-distributed machine learning. *IEEE Database Eng. Bull.*, 40, 2015.

[11] Hyunghoon Cho, Daphne Ippolito, and Yun William Yu. Contact tracing mobile apps for COVID-19: Privacy considerations and related trade-offs. *arXiv*, 2020.

[12] Meggan E. Craft. Infectious disease transmission and contact networks in wildlife and livestock. *Phil. Trans. R. Soc. B*, 370, 2015.

[13] Jesper Dall and Michael Christensen. Random geometric graphs. *Phys. Rev. E*, 66, 2002.

[14] Leon Danon, Ashley P. Ford, Thomas House, Chris P. Jewell, Gareth O. Roberts, Joshua V. Ross, and Matthew C. Vernon. Networks and the epidemiology of infectious disease. *Interdiscip. Perspect. Infect. Dis.*, 2011, 2011.

[15] Aaqib Bashir Dar, Auqib Hamid Lone, Saniya Zahoor, Afshan Amin Khan, and Roohie Naaz. Applicability of mobile contact tracing in fighting pandemic (COVID-19): Issues, challenges and solutions. *Comput. Sci. Rev.*, 38, 2020.

[16] Santo Fortunato. Community detection in graphs. *Phys. Rep.*, 486, 2010.

[17] Julie Fournet and Alain Barrat. Contact patterns among high school students. *PLoS ONE*, 9, 2014.

[18] Mathieu G'enois and Alain Barrat. Can co-location be used as a proxy for face-to-face contacts? *EPJ Data Sci.*, 7, 2018.

[19] Lea Hamner, Polly Dubbel, Ian Capron, Andy Ross, Amber Jordan, Jaxon Lee, Joanne Lynn, Amelia Ball, Simranjit Narwal, Sam Russell, Dale Patrick, and Howard Leibrand. High SARS-CoV-2 attack rate following exposure at a choir practice – Skagit County, Washington, March 2020. *MMWR Surveill. Summ.*, 69, 2020.

[20] Minyang Han and Khuzaima Daudjee. Giraph unchained: Barrierless asynchronous parallel execution in pregel-like graph processing systems. *Proc. VLDB Endow.*, 8, 2015.

[21] Petter Holme. Modern temporal network theory: a colloquium. *Eur. Phys. J. B*, 88, 2015.

[22] Petter Holme and Beom Jun Kim. Growing scale-free networks with tunable clustering. *Phys. Rev. E*, 65, 2002.

[23] Petter Holme and Jari Saramäki. Temporal networks. *Phys. Rep.*, 519, 2012.

[24] Kevin Hsieh, Aaron Harlap, Nandita Vijaykumar, Dimitris Konomis, Gregory R. Ganger, Phillip B. Gibbons, and Onur Mutlu. Gaia: Geo-distributed machine learning approaching LAN speeds. In *14th USENIX Symp. Networked Syst. Des. Implement. (NSDI 17)*, 2017.

[25] Donald B. Johnson. Efficient algorithms for shortest paths in sparse networks. *J. ACM*, 24, 1977.

[26] Eric Jonas, Qifan Pu, Shivaram Venkataraman, Ion Stoica, and Benjamin Recht. Occupy the cloud: Distributed computing for the 99%. In *Proc. 2017 Symp. Cloud Comput.*, SoCC '17, 2017.

[27] Brian Karrer and M. E. J. Newman. Message passing approach for general epidemic models. *Phys. Rev. E*, 82, 2010.

[28] George Karypis and Vipin Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM J. Sci. Comput.*, 20, 1998.

[29] Andreas Koher, Hartmut H. K. Lentz, James P. Gleeson, and Philipp Hövel. Contact-based model for epidemic spreading on temporal networks. *Phys. Rev. X*, 9, 2019.

[30] Frank R. Kschischang, Brendan J. Frey, and Hans A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Trans. Inf. Theory*, 47, 2001.

[31] Christiane Kuhn, Martin Beck, and Thorsten Strufe. Covid notions: Towards formal definitions – and documented understanding – of privacy goals and claimed protection in proximity-tracing services. *Online Soc. Netw. Media*, 22, 2021.

[32] Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi. Benchmark graphs for testing community detection algorithms. *Phys. Rev. E*, 78, 2008.

[33] Bo Li and David Saad. Impact of presymptomatic transmission on epidemic spreading in contact networks: A dynamic message-passing analysis. *Phys. Rev. E*, 103, 2021.

[34] Andrey Y. Lokhov, Marc Mézard, Hiroki Ohta, and Lenka Zdeborová. Inferring the origin of an epidemic with a dynamic message-passing algorithm. *Phys. Rev. E*, 90, 2014.

[35] Federica Lucivero, Nina Hallowell, Stephanie Johnson, Barbara Prainsack, Gabrielle Samuel, and Tamar Sharon. COVID-19 and contact tracing apps: Ethical challenges for a social experiment on a global scale. *J. Bioeth. Inq.*, 17, 2020.

[36] Tania Martin, Georgios Karopoulos, José Hernández-Ramos, Georgios Kambourakis, and Igor Nai Fovino. Demystifying COVID-19 digital contact tracing: A survey on frameworks and mobile apps. *Wirel. Commun. Mob. Comput.*, 2020, 2020.

[37] Robert McCune, Tim Weninger, and Greg Madey. Thinking like a vertex: A survey of vertex-centric frameworks for large-scale distributed graph processing. *ACM Comput. Surveys*, 48, 2015.

[38] Cristina Menni, Ana M Valdes, Maxim B Freidin, Carole H Sudre, Long H Nguyen, David A Drew, Sajaysurya Ganesh, Thomas Varsavsky, M Jorge Cardoso, Julia S El-Sayed Moustafa, Alessia Visconti, Pirro Hysi, Ruth C E Bowyer, Massimo Mangino, Mario Falchi, Jonathan Wolf, Sebastien Ourselin, Andrew T Chan, Claire J Steves, and Tim D Spector. Real-time tracking of self-reported symptoms to predict potential COVID-19. *Nat. Med.*, 26, 2020.

[39] James Moody. The importance of relationship timing for diffusion. *Soc. Forces.*, 81, 2002.

[40] Romualdo Pastor-Satorras, Claudio Castellano, Piet Van Mieghem, and Alessandro Vespignani. Epidemic processes in complex networks. *Rev. of Mod. Phys.*, 87, 2015.

[41] Ramesh Raskar, Isabel Schunemann, Rachel Barbar, Kristen Vilcans, Jim Gray, Praneeth Vepakomma, Suraj Kapa, Andrea Nuzzo, Rajiv Gupta, Alex Berke, Dazza Greenwood, Christian Keegan, Shriank Kanaparti, Robson Beaudry, David Stansbury, Beatriz Botero Arcila, Rishank Kanaparti, Vitor Pamplona, Francesco M Benedetti, Alina Clough, Riddhiman Das, Kaushal Jain, Khahlil Louisy, Greg Nadeau, Vitor Pamplona, Steve Penrod, Yasaman Rajaee, Abhishek Singh, Greg Storm, and John Werner. Apps gone rogue: Maintaining personal privacy in an epidemic. *arXiv*, 2020.

[42] Christopher S. Riolo, James S. Koopman, and Stephen E. Chick. Methods and measures for the description of epidemiologic contact networks. *J. Urban Health*, 78, 2001.

[43] Arjun Singhvi, Sujata Banerjee, Yotam Harchol, Aditya Akella, Mark Peek, and Pontus Rydin. Granular computing and network intensive applications: Friends or foes? In *Proc. 16th ACM Workshop Hot Top. Netw.*, HotNets-XVI, 2017.

[44] Haohuang Wen, Qingchuan Zhao, Zhiqiang Lin, Dong Xuan, and Ness Shroff. A study of the privacy of COVID-19 contact tracing apps. In Noseong Park, Kun Sun, Sara Foresti, Kevin Butler, and Nitesh Saxena, editors, *Secur. Priv. Commun. Netw.*, volume 335 of *Lect. Notes Inst. Comput. Sci., Soc. Inform. Telecomm. Eng.* Springer Int. Publ., 2020.

[45] Tao Zhou, Jie Ren, Matú š Medo, and Yi-Cheng Zhang. Bipartite network projection and personal recommendation. *Phys. Rev. E*, 76, 2007.

[46] Lorenzo Zino and Ming Cao. Analysis, prediction, and control of epidemics: A survey from scalar to dynamic network models. *IEEE Circuits Syst. Mag.*, 21, 2021.