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

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

Listings

Chapter 1

New Proposed Scheme

1.1 Caching

ShareTrace requires an internet connection so that user actors can receive new symptom scores and contact details. It is possible, however, that a user may not have internet connection at the time of contact. From the perspective of the contact network, these "lagging contacts" may not be added for several days. To allow for this type of delay, we employ an interval-based checkpointing strategy that incurs low space complexity.

Let $\delta \in \mathbb{R}_{\leq 0}$ be the time between when the contact occurred and when the user actor receives the corresponding message. Assuming $B \leq L$, there are 3 cases to consider: $\delta \leq B$, $B < \delta \leq L$, and $\delta > L$.

Suppose $\delta \leq B$. Then there is no

Chapter 2

Evaluation

2.1 Experimental Design

Risk propagation requires a partitioning or clustering algorithm, as described in Algorithm **??**. We configured the METIS graph partitioning algorithm [**?**] 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.

2.1.1 Synthetic Graphs

We evaluate the scalability and efficiency of risk propagation on three types of graphs: a random geometric graph (RGG) [**?**], a benchmark graph (LFRG) [**?**], and a clustered scale-free graph (CSFG) [**?**]. Together, these graphs demonstrate some aspects of community structure [**?**] 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 [?] 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.

2.1.2 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) [?], a workplace (InVS15), and a scientific conference (SFHH) [?]. 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 $(t_{\text{now}} - 1 \text{ day})$ 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 2.1.1 with the exception that we only generate one score. Lastly, we perform 10 iterations over each data set to obtain an average performance.

2.2 Results

2.2.1 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. 2-1 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 **??**, send tolerance quantifies the trade-off between completeness and efficiency. Thus, $\gamma = 0.6$ optimizes for both criteria.

Unlike the update count, Fig. 2-1 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.

2.2.2 Message Reachability

To validate the accuracy of (**??**), we collected values of (**??**) and (**??**) 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 (**??**), let the *message reachability ratio* (MRR) be defined as

$$\text{mrr}(u) := \frac{m(u)}{\hat{m}(u)}. \tag{2.1}$$

Overall, (**??**) is a good estimator of (**??**). Across all synthetic graphs, (**??**) modestly underestimated (**??**) with quartiles $(Q_1, Q_2, Q_3) = (0.71, 0.84, 0.98)$ for the (2.1). For $\alpha = 0.8$ and $\gamma = 0.6$, the quartiles of (2.1) 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 2.1 provides mean values of (2.1) for both synthetic and real-world graphs. Fig. 2-2 indicates that moderate values of γ tend to result in a more stable MRR, with lower (higher) γ underestimating (resp. overestimating) (**??**). With regard to transmission rate, (2.1) tends to decrease with increasing α, but also exhibits larger interquartile ranges.

Because (**??**) does not account for the temporality constraints (**??**) and (**??**), it does not perfectly estimate (**??**). With lower γ and higher α, (**??**) suggests higher MR. However, because a message is only passed under certain conditions (see Algorithm **??**), this causes (**??**) to overestimate (**??**). While (**??**) theoretically is an upper bound on (**??**), it is possible for (**??**) to underestimate (**??**) if the specified value of $r_0(v)$ overestimates the true value of $r_0(v)$. When computing (2.1) for Fig. 2-2, we used the mean $r_0(v)$ across all users v, so $\text{mrr}(u) > 1$ in some cases.

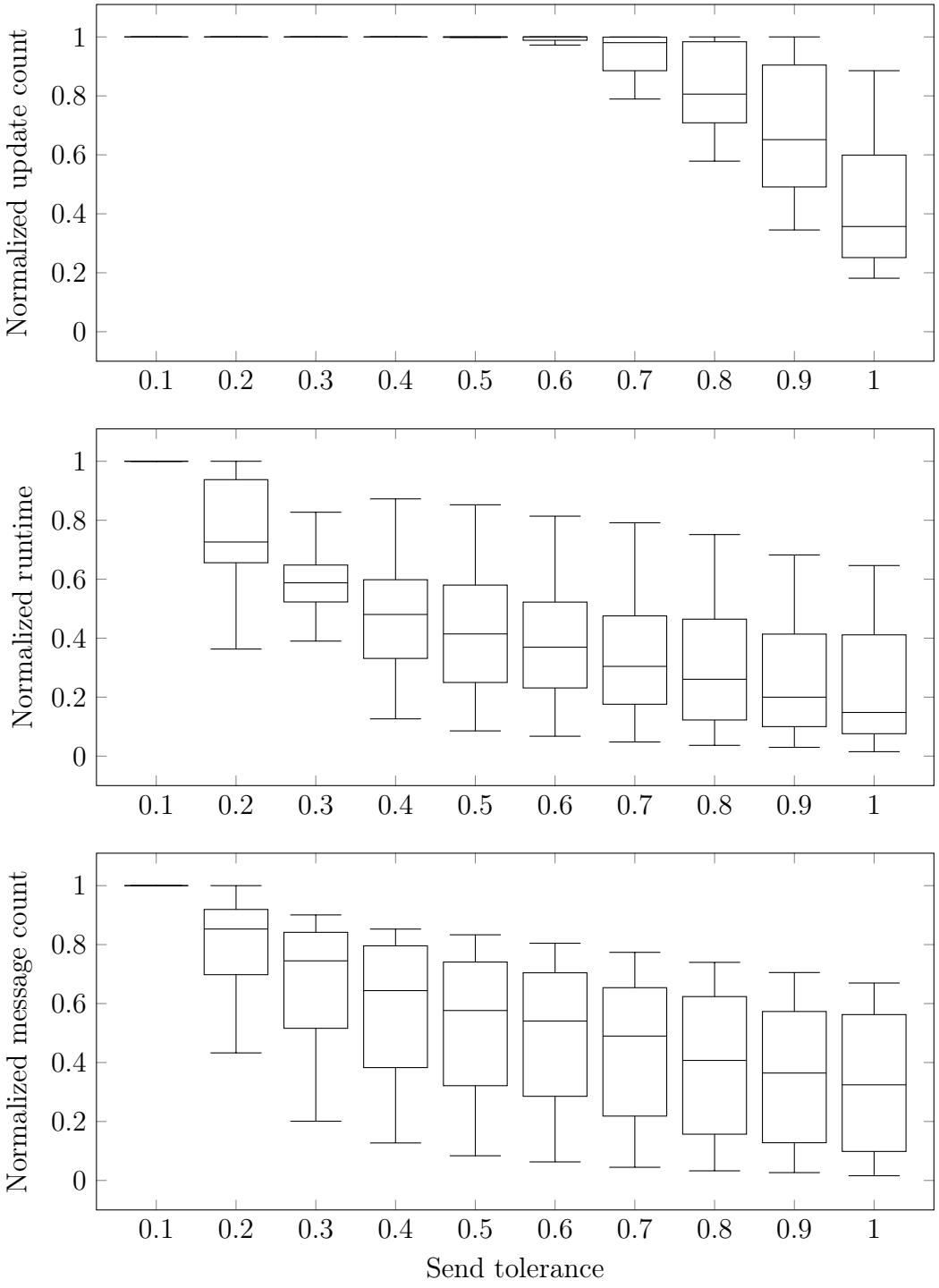

Figure 2-1: Effects of send tolerance on efficiency. All dependent variables are normalized across graphs and transmission rates.

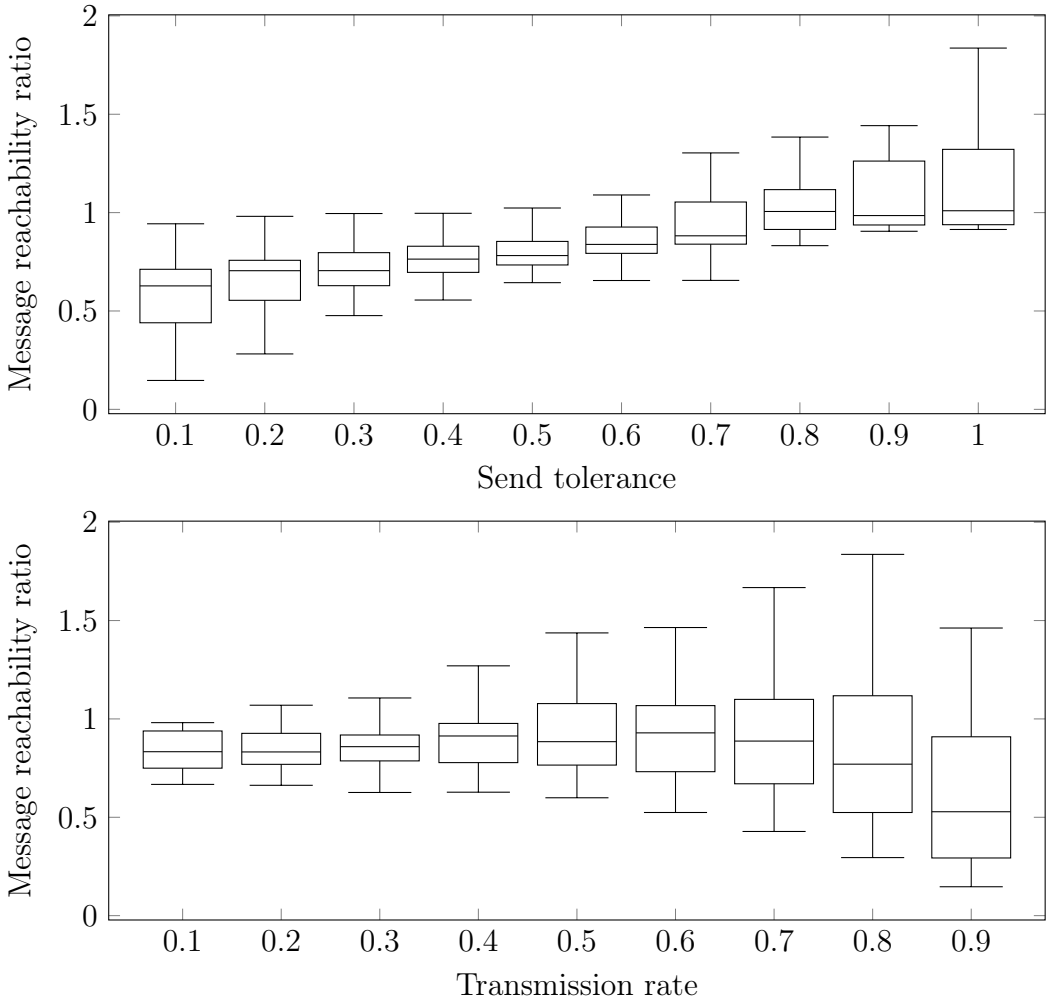

Figure 2-2: Effects of send tolerance and transmission rate on the message reachability ratio. Independent variables are grouped across graphs.

Setting	$\text{mrr}(u) \pm 1.96 \cdot \text{SE}$
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 2.1: 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).

2.2.3 Scalability

Fig. 2-3 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 ($\pm 1.96 \cdot \text{SE}$).

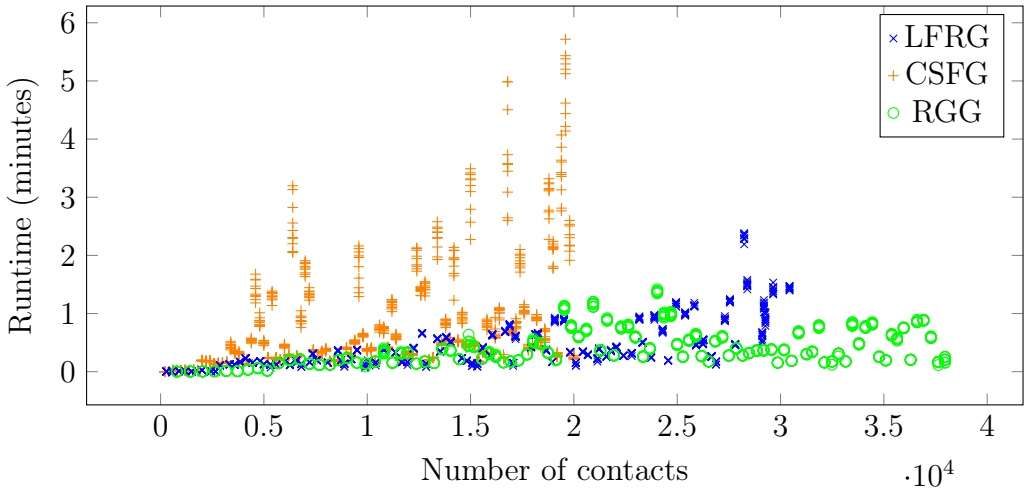

Figure 2-3: Runtime of risk propagation on synthetic graphs containing 100–10,000 users and approximately 200–38,000 contacts.