Computer Science and Information Systems 19(3):1261–1282 https://doi.org/10.2298/CSIS210825029G

Federating Digital Contact Tracing using Structured Overlay Networks

Silvia Ghilezan[1,2], Simona Kašterović[2], Luigi Liquori[4], Bojan Marinković[3,1], Zoran Ognjanović[1], Tamara Stefanović[2]

[1] Mathematical Institute of the Serbian Academy of Sciences and Arts
Belgrade, Serbia
{bojanm, zorano}@mi.sanu.ac.rs

[2] Faculty of Technical Sciences, University of Novi Sad,
Novi Sad, Serbia
{gsilvia, simona.k, tstefanovic}@uns.ac.rs

[3] Clarivate, Serbia

[4] Inria & Université Côte d'Azur, France
Luigi.Liquori@inria.fr

Abstract. In this paper, we present a comprehensive, yet simple, extension to the existing systems used for Digital Contact Tracing in Covid-19 pandemic. The extension, called *BubbleAntiCovid19* (*BAC19*), enables those systems, regardless of their underlying protocol, to enhance their sets of traced contacts and to improve the global fight against pandemic during the phase of opening borders and enabling more traveling. *BAC19* is a *Structured Overlay Network*. Its protocol is inspired by the Chord and Synapse Structured Overlay Networks. We design the architecture of the Overlay Network Federation. We show that the federation can be used as a formal model of *Forward Contact Tracing*. *BAC19* provides a *fully exhaustive* retrieving procedure thanks to avoiding search during peer churn. Furthermore, we give simulation results for the *BAC19* system, the simulator written in Python.

Keywords: Covid-19, Digital Contact Tracing, Distributed Hash Tables, Structured Overlay Networks, Bluetooth, GPS.

1. Introduction

One of the biggest challenges the world is facing since the beginning of Covid-19 pandemic is to slow down the spreading of SARS-CoV-2 virus producing Covid-19 pandemic; *Prevention, Testing and Tracing* are the main pillars of the solution for controlling and slowing down the spread of the virus. Contact Tracing of an infected person is essential to control the spread of the disease.

Tracing. Contact tracing is the process of identifying, notifying, and monitoring people who came in close contact with an individual who was tested positive for an infectious disease, like Covid-19, while he/she was infectious. Contact tracing benefits the fight with the pandemic at multiple levels. Identifying and quarantining close contacts limits their ability to spread the disease. Therefore, in a period in which the disease and its effects are still being investigated, contact tracing plays a key role in preventing the further spread of the disease. In [20] the authors studied the spreading of H1N1 virus and showed that tracing close contacts can effectively slow down the rate of virus spreading. Furthermore, contact tracing data helps medical experts to find the origin of the virus and learn more about the nature of the virus.

Manual Contact Tracing. Contact tracing has mostly been done manually since many centuries ago just by taking note on a simple piece of paper the list of persons and goods you get in contacts with (see e.g. *La Peste* by A. Camus [8]). In the actual days, manual contact tracing could be exploited using simple telephone calls. Identifying contacts is done through an interview with the person infected with the virus. Each person is then contacted by phone. Health Authorities should quickly alert people who are close contacts that they may have been exposed to the virus. The sooner the contacts are notified, the lower the risk of the spreading further. However, due to the highly contagious nature of the SARS-CoV-2 virus and the fact that symptoms can manifest after many days (or even never, e.g. *asymptomatic cases*), manual contact tracing does not give satisfactory results. Health Departments and National Authorities do not have enough employees to do manual contact tracing. It must be further emphasized that the SARS-CoV-2 can be transmitted not only by direct contact, but also by indirect contact. The reason is that infected people can leave virus droplets on any physical object they touch. In this case, actual digital contact tracing solutions are ineffective. For the reasons stated above, digital contact tracing has been considered already at the beginning of the Covid-19 pandemic.

1.1. Problem

There is a plethora of Digital Contract Tracing (DGT) applications in use all over the world fighting the Covid-19 pandemic [14, 26]. They are developed on very different paradigms, centralized [10] vs. decentralized [35], GPS based (very few indeed because of a clear violation of privacy) vs. Bluetooth Low Energy based (the majority). The rush to make these applications work in the shortest time led to their great diversity. The most important open problem is their interoperability. There are many ongoing efforts to make a “federation” of these different systems. Herein, we address this problem and propose a solution based on mathematical models of overlay networks. We leave to the the reader to envisage the following scenario:

> Alice lives in the region which has centralized DCT *System A*, while Bob lives in the region which has centralized DCT *System B*. Bob has spent some time in the region A, and both of them are traveling together side by side with negative RT-PCR tests. However, Bob developed symptoms of Covid-19 after couple of days and was confirmed as positive.

1.2. Contributions

We develop a formal *Federation of Overlay Networks*, called *BubbleAntiCovid19* (*BAC19*), for connecting different digital contact tracing applications, which are currently in use all over the world. The model is based on the well-known model of Structured Overlay Network protocols like e.g. Chord [31, 32], Kademlia [27], and Synapse [21]. However, in these well-known cases nodes are volunteer to join/leave a network when ever it want. In *BAC19* we control all the nodes that participate in the system and it is not allowed that a node from the public domain joins the system. Also, *BAC19* is conceived as a topology composed by trusted overlay networks, in the sense that every node, before enter in one *BAC19* overlay, has been trusted by a *Health Authority* within the original application. There is one and only one *Health Authority per* original application included in the *BAC19* system. Original applications communicate with *Health Authorities* in order to validate the test result of each user. Even if it is possible to choose a different technique to solve this problem, e.g. distributed databases, our choice was Distributed Hash Tables (DHT) since we see it as a natural extension of the basic idea that all contacts of one person should be stored in one overlay network and the contact between persons could be seen as "the synapse nodes". Further, a distributed database is a federation governed by a "distributed server dictionary/jellow pages" (e.g. Microsoft Azure, Amazon elastic, etc...), whereas *BAC19* can be populated by nodes trusted by the *Health Authority*, so giving, in principle the capability to servers belonging to Municipalities, Departments, etc. to participate to the Overlay.

We prove that *BAC19* provides a fully exhaustive retrieving procedure of people that get in touch with other people having tested positive to the Covid-19 disease. Hence, *BAC19* is proven to be a simple yet powerful *interconnection* of already existing digital contact tracing applications that - by construction - do not communicate with each others as such providing their efficient interoperability.

BAC19 solves the problem presented in the scenario with Alice and Bob. Suppose System A and System B are part of *BAC19*. For Alice and Bob it will be sufficient that only one of them installs the system of the other region during the time of their travel, so that Alice gets informed that she is the first contact of an infected person, namely Bob during their joint travel.

We give simulation results by running a prototypical implementation of the *BAC19* system, written in Python. The simulator creates the *BAC19* topology, generates random requests for the search procedure and calculates the success rate. The simulator does not skip any node, so that the success rate of every iteration is 1, as expected.

As far as we know, the mathematical model and techniques presented in this paper have not been considered in other approaches.

1.3. Overlay networks in a Nutshell

Structured Overlay Networks [13] are suitable models of scalable and efficient organization of resources on the Internet. They represent logical organizations, independent on underlying network infrastructure that physically connects available assets. Structured Overlay Networks have been proven as very resilient tool (they can be reliable as central servers or distributed cloud solutions) in the situation when some parts of the underlying infrastructure fail or become overloaded or corrupted.

1.4. Organization of the Paper

The rest of the paper is organized as follows. Section 2 presents classifications of digital contact tracing applications. Section 3 reviews Chord and Synapse protocols of Structured Overlay Networks. It also briefly reviews basic notions of Abstract State Machines and some related work by the authors. Section 4 introduces the *BAC19* system and proves the completeness and full exhaustiveness of the retrieving procedure. Section 5 presents the simulation results for the *BAC19* system. Section 6 presents a discussion on other proposals for providing interoperable frameworks for digital contact tracing. Section 7 concludes the paper.

2. Digital Contact Tracing Applications

Advances in digital technology have enabled smartphones and other digital devices to be used for contract tracing. Particularly, more and more countries are showing interest in *Digital Contact Tracing* applications (DCT apps) implemented for smartphones. Despite the great variety among these applications, all contact tracing apps work on the principle of automatic data exchange with nearby devices. When a user of a particular contact tracing app is identified as infected, a special report is uploaded to the DCT app server. Based on that report, close contacts of the infected person (also DCT app users) are informed that they have been in a contact with a positive user and/or the app calculates their exposure risk. The identity of the infected persons is not disclosed in order to protect their privacy. Existing contact tracing apps can be classified based on two criteria:

– System architecture and topology;
– Contact-tracing technology.

More information about the classification of the existing contact tracing apps can be found in [30, 33].

2.1. Classification by System Architecture

Since the data is collected from users, their processing should be addressed. When it comes to DCT app data managing, the responsibility can be on a central authority or on each user individually. Therefore, contact tracing apps can be divided into three major categories depending on the architecture of the underlying system:

– Centralized apps - data are solely managed by a central server;
– Hybrid apps - multiple nodes can manage the data, but the control is centralized;
– Decentralized apps - each user is managing his/her data.

In centralized apps, the central server is responsible for ID generation, risk analysis and notifications. Centralized apps can, when using GPS and no encrypted IDs, raise privacy concerns and questions about massive surveillance. People often do not trust servers and as a consequence there is a low adoption rate of these applications. On the other hand, the advantage of centralized apps is the possibility for Health Authorities to make a transmission graph and learn more about the virus. Also, the possibility of false positive users is reduced. As we have already observed, privacy issues lead to low adoption rate and decrease the efficiency of the application. In order to solve the problem of distrust of the central server, decentralized apps were designed. In decentralized apps these functionalities are moved to the user devices and the central server is only an encounter point. However, decentralized methods also raise some privacy concerns, for example the identity of a patient can be easily revealed again when IDs are not encripted. The hybrid architecture proposes decentralized ID generation and centralized risk analysis. There are a few proposed hybrid contact tracing protocols, but their implementation in real contact tracing apps is still pending. More about these protocols can be found in [2]. For that reason, we will focus on apps with centralized and decentralized architecture, and we will provide an overview of the existing contact tracing apps based on the above classifications in Section 2.3. The main apps are summarized in Figure 1, which is motivated by [30].

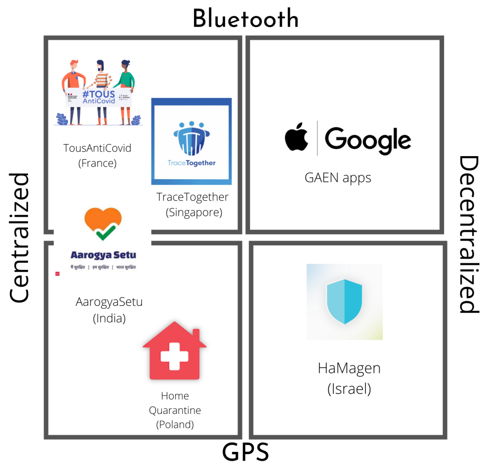

Fig. 1. Classification of analyzed applications

2.2. Classification by Contact-Tracing Technology

In order for two people to be in close contact, they need to stay in the same place, at a short distance, and for a long enough period of time. Therefore, the main data type used by the contact tracing apps is location data. There are various technologies for collecting and tracking location data, and contact tracing apps can be divided into two major categories depending on whether they track absolute or relative location:

– Absolute location apps – Apps that track the absolute location of their users are mostly based on GPS technology. Location data is stored in the form of geolocation coordinate pair. These apps are also known as Geolocation-based DCT apps.
– Relative location apps – Apps that track relative location of their users are mostly based on Bluetooth technology. These apps are also known as proximity DCT apps. A boarding pass or a ticket for a specific event can also be considered as relative location data. In order to use this kind of data, some contact tracing apps deploy QR code technology.

Geolocation-based DCT apps record past geo-trajectories of every user, and the calculation of exposure risk of a user is based on the intersection of its past trajectories and trajectories of patients. We give a brief review of the existing Geolocation-based apps in Section 2.3.

Two main advantages of geolocation-based DCT apps are the following:

1) Geolocation-based DCT apps are compatible with manual contact tracing. Compatibility of geolocation-based DCT apps and manual contact tracing has mutual benefits. On the one hand, past geo-trajectories of a patient can be added to an app by the contact tracer even if the patient did not use the app. This enables the app to warn more users. On the other hand, the app can give the information about places with higher exposure risk to a contact tracer, so that the contact tracer can identify high-risk service workers.
2) Another advantage is that geolocation-based DCT apps can recognize patterns of disease's spreading and locations with higher exposure risk, and they can inform health authorities about it.

Nevertheless, geolocation-based DCT apps have also disadvantages. The major challenges are privacy concerns, which cause low adoption rate of these applications. User's privacy can be violated in several ways. Recording all user's trajectories can result in revealing user's personal information such as identity, home address, work address, the identity of the patient and revealing user's exposure risk to other users. These problems have been elaborated in more details in [12].

Bluetooth-based DCT apps record direct contacts of the users. A device generates a unique, randomized identifier and assigns it to a user. There are two kinds of identifiers: static, identifiers do not change over time, and dynamic, identifiers change over time. During a direct contact devices exchange identifiers and save received identifiers. Once a user is identified as positive in the application, other users can calculate their exposure risk by checking whether they received a patient's identifier. We give a brief review of the existing Bluetooth-based apps in Section 2.3.

Depending on whether the exposure risk is calculated by the central server or the user's device, we have centralized and decentralized apps, respectively, see Figure 2, which is motivated by [18].

The major disadvantage is that Bluetooth-based DCT apps work only if both users have installed the *same* application.

In order to take advantage of both types of these apps, Bluetooth-GPS apps were designed, see Section 2.3. Given the different characteristics of DCT apps, the question arises whether it is possible to aggregate their data in order to track contacts more effectively. The answer can be found in Structured Overlay Networks.

2.3. DCT Apps - Overview

In this section we give a review of the existing DCT apps.

Geolocation-based DCT apps. *Home Quarantine.* At the beginning of the Covid-19 pandemic, Ministry of Digital Affairs of Poland developed the Home Quarantine app. More details about this app can be found in [34]. This is a typical example of a centralized app which deploys GPS technology. It is developed to support the authorities,

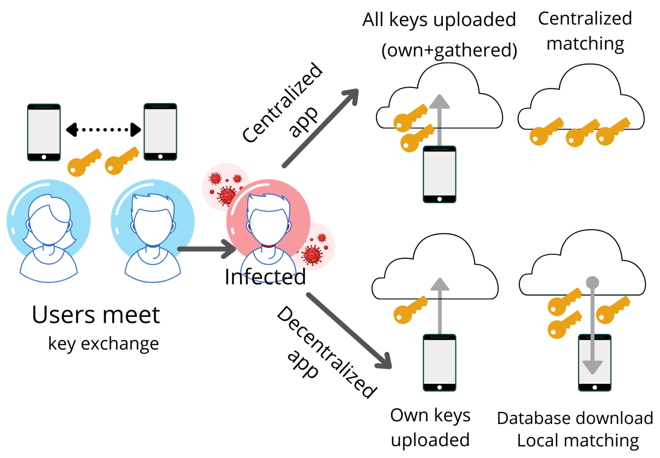

Fig. 2. Centralized vs Decentralized Bluetooth-based apps

especially the police and social services, with adequate information about people undergoing mandatory home quarantine. Users are also required to upload their digital photos. So, aside the GPS technology the app also uses face recognition. The app is mandatory for anyone who has developed coronavirus symptoms. It should be emphasized that Poland also developed the ProteGO Safe app for alerting users of close contact with an infected person based on The (Google/Apple) Exposure Notification (GAEN) system [4].

The Shield (HaMagen). In March 2020, Israeli Ministry of Health developed The Shield app [3]. This is a typical example of a decentralized app which deploys GPS technology. Location data is stored in the phone. If a user tests positive, he/she can upload his/her location history to the central server. Once the user uploaded his/her location history, it is added into a JSON file that is updated with new data on an hourly basis. Matching the locations happens on the phone. If the match is found, the app shows you the exact time and location. The app is later updated to work with Bluetooth technology but on a voluntary basis, every user can choose whether to use the proximity data or not.

Bluetooth-based DCT apps. *Blue-Trace protocol apps.* Singapore's Government Technology Agency in collaboration with Ministry of Health in March 2020 released the TraceTogether app that allows digital contact tracing using the custom Blue-Trace protocol. Australia has later adopted the protocol and released the CovidSafe app. More details about these apps can be found in [1]. Contact tracing is done using Bluetooth Low Energy and proximity data is encrypted and stored only on the users phone. Users in the contact log are identified using anonymous time-shifting "temporary IDs". If a user tests positive for the infection, the Ministry of Health requests his/her contact log. The user has the right to choose whether to share the contact log or not. If the user chooses to share the log, the contact log is uploaded to a central server and the Health Authority is then responsible for matching the log to contact detail and informing close contacts of the infected user. These apps are examples of Bluetooth-based centralized apps. It should also be noted that

Singapore solved the problem of tracing people who don't use smartphones by enabling the app to work with Token - a physical Bluetooth-based device.

ROBERT protocol app. Inria (France) and Fraunhofer AIESEC (Germany) released the ROBERT protocol (ROBust and privacy-presERving proximity Tracing) [9]. The French Government released the StopCovid (later renamed to TousAntiCovid) app in May 2020. It also deploys Bluetooth technology and belongs to the category of centralized apps. The difference between this app and apps based on the BlueTrace protocol relates to confirmation of positive users. More precisely, in France when a person is confirmed to be positive, the lab gives a patient a QR code and the scanned code is the proof for the app that you are infected. It is up to you to share this information with the app, and if you choose to share this information with a central server, the server is responsible for alerting your close contacts.

Google/Apple exposure notification apps. In April 2020, Google and Apple announced the joint work on decentralized Bluetooth-based protocol named The Google/Apple Exposure Notification (GAEN) system [4]. Many states then developed different apps using the Google/Apple Exposure Notification framework including Austria (Stopp Corona app), Germany (Corona-Warn-App), Italy (Immuni), Canada (COVID Alert) etc. The principle by which applications work is as follows. During a close contact, user's phones exchange random Bluetooth identifiers. These identifiers change frequently and the information about exchanged ID's is stored on the user's phone. When a user gets infected, he/she can decide to upload ID's he/she was using the last 14 days to the server. Phones of all users periodically download the list of ID's which belong to the infected users and does the matching locally.

Bluetooth-GPS apps. Apps that deploy both Bluetooth and GPS technology are rare. One app of this kind is the Aarogya Setu app [6], developed by National Informatics Centre that comes under the Ministry of Electronics and Information Technology, Government of India. Aarogya Setu is following the centralized approach, and is one of the world's fastest growing applications. The app mainly uses proximity data and GPS data are recorded only once in 30 minutes. The location data is mainly used to identify the locations where you might have caught the infection and identify potential hotspots that may be developing when multiple infected people visit the same place. Interaction between users is recorded by exchange of Device Identification Numbers (DID's) which are static. Contact tracing data is kept on the phone. Council of Medical Research (ICMR) shares the list of Covid-19 positive persons with the Aarogya Setu server, and information about contact tracing is uploaded to the server only if you are tested positive. The central server is then responsible for alerting your close contacts.

3. Overlay Networks

In this section, we briefly review basic notions of Structured Overlay Networks for the purpose of this work. Some Structured Overlay Networks are implemented in a form of Distributed Hash Tables (DHTs). One of DHT protocols is the Chord protocol. We also provide a brief review of Abstract State Machines, since they are used for proving properties of Chord. For more details, please see [7, 16, 21, 23, 25, 31, 32].

3.1. Chord and Synapse

Chord was introduced in [31,32]. Nodes that are part of a Chord system form a ring shaped network. The basic operations of a Chord node are entering and leaving the system and the mapping given key onto the corresponding node of the system using consistent hashing.

The correctness and efficiency of the Chord's protocol lookup procedure was in the focus of several papers, e.g. [23, 25, 31, 32]. In [11] the authors exploited the benefits of Chord, such as accelerating the lookup, and developed an energy efficient routing protocol for Wireless Sensor Networks, called *CHEARP: Chord-based Hierarchical Energy-Aware Routing Protocol for Wireless Sensor Networks*. However, these properties will not be in the focus of this paper. Our goal is to deliver information of every affected node, so we will not use any presented improvements to speed up the process of getting results, but to linearly pass every node in a Chord network, to be sure that no information is missed.

Interconnection of several Structured Overlay Networks is a very hard problem since different networks may use different protocols, and even in the case of several DHT networks that use the same protocol (e.g. Chord) it is enough that every overlay network uses *its own hash function* and information between two of them cannot be exchanged. A proposal to solve this issue was given by defining the Synapse protocol in [21]. Its performances were analyzed in [22], whereas one real-life proof of concept was developed in [24].

In the Synapse protocol, the interconnection of intra-overlay networks is achieved by co-located nodes taking part in several of these intra-overlays, called *Synapses*. Each node acts in accordance with the policies of each of its intra-overlays, but it also has the extra-role of forwarding the requests to some other intra-overlay it belongs to. Every node comes with a proper logical name in each intra-overlay; in particular, synapses have as many logical names as the number of networks they belong to. Each node is responsible for a set of resources it hosts, (key,value) pairs. Every key also comes with a proper logical name peculiar to each intra-overlay.

For the purpose of this paper we will consider the so-called, *white-box* version of the Synapse protocol that, in short, allows to consider all the keys as they were using the same hash table (see [21] for details). Again, since we will use the linear search procedure in one Chord network we can be sure that information will be retrieved if it exists in the system.

3.2. Abstract State Machines

In this subsection, we briefly review basic notions of Abstract State Machines for the purpose of this work. For more details, please look at [7, 16].

Abstract State Machine (ASM) [7, 16] is a formalization method for algorithms at the appropriate abstraction level. An ASM $\mathcal{A}$ is defined as a program Prog which consists of:

– an at most countable set of states, its subset of initial states, and
– a finite number of transition rules.

States are first order structures over a fixed signature, whereas transitional rules:

– update (:=),
– sequential (seq ... endseq),

– conditional (if . . . then . . . else . . . endif),
– parallel (par . . . endpar),
– nondeterministic (choose $v \in U$ satisfying $g(v)$ . . . endchoose) and
– universal (forall v with g . . . endforall)

represent next-state functions. An execution of one of the last two types of rules introduces a variable v. In the case of nondeterministic rule, the transition is executed with a value of v which satisfies a guard g, while in the case of universal rule, the transition is executed simultaneously for all values v which satisfy a guard g. An ASM can interact with its environment using external functions (oracles) by providing arguments to oracles and receiving the corresponding results.

In a distributed case with many agents, every agent executes its own program and has its own partial view of a global state. The nullary function Me allows an agent to identify itself among other agents. The global program is the union of all agents' programs, whereas a transition between two states is obtained by an evaluation of transition functions of all agents.

An ASM $\mathcal{A}$ models a real system $\mathcal{S}$ in terms of evolution of states described by runs. A run of $\mathcal{A}$ is a (possible infinite) sequence of $S_0, S_1, S_2, \ldots$, where S_0 is an initial state, and every S_{i+1} is obtained from S_i by executing a transitional rule. In this paper we consider only runs in which states are global and agents' moves are atomic (instantaneous). The most general kind of runs for a distributed ASM are partially ordered runs. To prove properties of partially ordered runs, thanks to the results proved in [16], the attention may be restricted to their linearizations that are sequential runs and satisfy the following fundamental properties:

– All linearizations of the same finite initial segment of a run have the same final state.
– A property holds in every reachable state of a run iff it holds in every reachable state of any of its linearization.

Note that this implies that it is enough to find only one sequence of transitions and the runs that are considered here and start from the same initial state will have the same final state.

The main notions introduced in [23, Definition 5.1] are:

– *stable states* in a Chord network, where a state of a network is stable if the successor (predecessor) pointers of all nodes form an ordered ring, and
– *regular runs*, where a run is regular if it is a linearization such that nodes leave and enter the network only in stable states.

Having these notions, the paper [23] proves that the presented formalization of Chord consistently maintains the topological structure of rings and manipulates with distributes keys. In Section 4 we will explain that our model of *BAC19* satisfies the mentioned constraints and that results from [23] hold also for *BAC19*.

On the other hand, the papers [21, 22] recognize the fact that the search procedure of the Synapse protocol is not complete nor fully exhaustive. This is due to the fact that the lookup procedure of the Chord network can "skip" in routing some of the Synapse nodes, and thus not to spread the query to all networks that are reachable. To avoid this situation we redefine the lookup procedure with Algorithm 1, not to skip any node. This allows to achieve exhaustiveness and keep the routing complexity still be acceptable due to the relatively small number of contacts for each device.

4. System *BAC19*

In this section, we propose the design of the system *BubbleAntiCovid19* (*BAC19*), which is a formal Federation of Structural Overlay Networks for connecting different Digital Contact Tracing applications that are currently in use all over the world.

BAC19 federation (Figure 3) consists of several Chord networks:

– a network for each person/device of his/her first contacts (black circles in Figure 3),
– dedicated red network to connect all infected persons (red circle in Figure 3),
– dedicated $amber$ network to connect all the first contacts of infected persons (amber circle in Figure 3),

and

– *Gateways* (black rectangles in Figure 3) as the connections to the existing Digital Contact Tracing systems (black rounded corners rectangles in Figure 3).

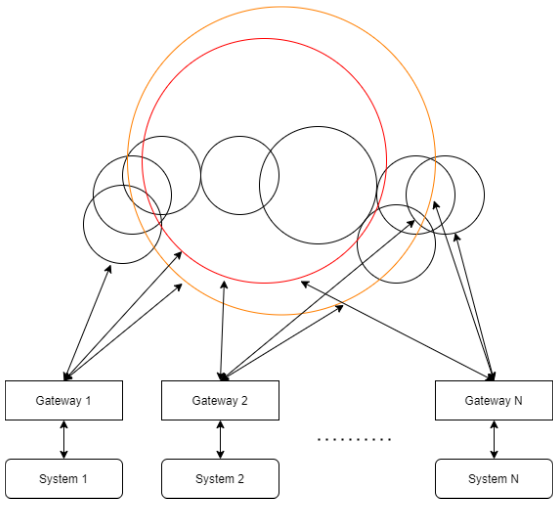

Fig. 3. *BAC19* Federation of Structural Overlay Networks

The first connection between the proposed extension and an existing system for contact tracing is called *Gateway*. The purpose of a *Gateway* is to maintain communication between two parts and to transform messages in a way that both sides can communicate efficiently.

The most important thing is to maintain the mappings between identifiers (IDs) used on both sides of a *Gateway*. As we could see in Section 2, some systems periodically change IDs, so the possibility to trace those changes is vital for functioning of *BAC19*. Regarding IDs, our goal is to have one identifier per one person/device regardless of how many systems it appears in. We argue that this is possible to achieve because of the scalability of Structured Overlay Networks. First, it is possible to use sufficiently large

Algorithm 1: FindSuccessor

```
FINDSUCCESSOR (KEY) =
For Given $key$
//$successor(id(Me))$ is responsible for key
if $member\_of(key, id(Me), successor(id(Me)))$ then
    Respond With $successor(id(Me))$
else
    //Me forwards query to its successor
    Forward Query To $successor(id(Me))$
```

codomain of the hash function (e.g. 2^{128} or greater). Also, it is possible to select enough parameters of a person/device so that it can be uniquely identified. We are not storing any other attribute of a person/device except a newly introduced identifier in our extension.

More precisely, with respect to the specifications that are provided in [22,23], we need to introduce the following changes:

– the set
$$Network = \{red, amber, net_1, \ldots, net_N\}, \qquad N \in \mathbb{N}$$
to denote all possible networks, where N is the number of possible persons/devices in the proposed extension;
– the set $Time$ and the function
$$contact\_time() : (Chord \cup \{amber\}) \times Chord \rightarrow Time$$
to denote the time of the contact between two persons;
– the external function $current\_date()$ to get the current date.

The algorithm FINDSUCCESSOR finds a responsible node for a given ID and it is originally defined in [31]. The lookup procedure of the Chord network can skip some of the synapse nodes, and thus not to spread the query to all networks that are reachable. To avoid this situation and to obtain *full exhaustiveness* of the retrieval procedure, the retrieval procedure is redefined with **Algorithm 1**, not to skip any node.

With this proposal we are not compromising performances of the extension by much. Since the number of contacts of a given person using the system is *relatively small*, it is manageable to allow increasing the complexity of the worst case retrieval from $O(logN)$ to $O(N)$. In the predefined time-slots, our extension will receive the following information from a system:

– all identified infected cases since the last import (**Algorithm 2**),
– all confirmed cases that are not infected anymore since the last import (**Algorithm 3**),
– all identified contacts since the last import in the form of the tuple $\langle id_i, id_j, t\rangle$ $(i \neq j)$ with the meaning that persons id_i and id_j had a risk contact at t timestamp. For the purpose of providing privacy protection timestamp should be kept at the precision of days. Unfortunately, this type of communication is not possible with the systems that are categorized as decentralized Bluetooth systems, since the fact that contact tracing computation is performed at users' devices and not shared with the central storage.

Algorithm 2: Put

```
PUT =
For all $inf \in NewCases$
  Invoke PUT Of Network $red$ To Store $inf$
For all $id \in net_{inf}$
  if $contacttime(amber, id) < t$ or $contacttime(amber, id) = undef$ then
    Set $contacttime(amber, id) = t$
```

Algorithm 3: Leave

```
LEAVE =
For all $inf \in Healed$
  Invoke LEAVE Of Network $red$ for $inf$
```

These systems can only share newly identified cases and their time of recovery (**Algorithm 4**).

Also, if needed, it is possible to introduce the new *Gateway* with the purpose to enter manually recognized contacts to the system.

When information is received from the origin systems, as the first step *BAC19* will connect all newly recognized infected cases to the red network, as well as to remove all cured. A node will remain in the red network until its recovering is confirmed. All IDs that are recognized as the risk contacts of a person/device (e.g. id_i) will be added to its bubble. They will stay there until $t + 14\ days$, where t is the time of their contact and 14 days is a widely used time frame in Digital Contact Tracing implementations. If the id_i is the member of the red network all the members of its network will be added to the $amber$ network and stay there during the same time frame $t + 14\ days$. If a contact is already in the $amber$ network, the timestamp will be updated to the higher value (**Algorithm 5**).

During the opposite way of communication, *BAC19* will pass on information to all nodes in the $amber$ network to *Gateways*. If an identifier is recognized in the set of mappings for the particular origin system, the corresponding information is transferred to the origin system to alert (if not already) the person/device that he/she had risk contact with an infected person at stored timestamp. Also, *BAC19* is capable to send information on the second level contacts (the result is stored in the set $Result$, **Algorithm 6**).

Namely, for all nodes of the $amber$ network it is possible to go through every origin bubble and pass those identifiers to the *Gateways*. Then the origin systems can inform those persons that they should increase their awareness since they are second level contacts.

Algorithm 4: Set contact time

```
SETCONTACTTIME =
par
  Set $contact\_time(id_i, id_j) = t$
  Set $contact\_time(id_j, id_i) = t$
endpar
```

Algorithm 5: Leave of network

```
LEAVE =
For all $net_{id_i} \in Network \setminus \{red\}$
  For all $id_j \in net_{id_i}$
    if $contacttime(id_i, id_j) + 14\,days\ > currentdate()$ then
      Invoke LEAVE Of Network $id_j$ for $id_i$
```

Algorithm 6: Get all nodes

```
GET =
seq
  Invoke GET all nodes from $amber$ and store the result in $amber$
  For all $id \in amber$
    Invoke GET all nodes from $net_{id}$ and append the result to $Result$
endseq
```

The “consistency” of the whole *BAC19* system is naturally related to the consistency of the overlay network it employs, namely Chord [32]. Informally, consistency refers to the fact that if a (key-value) is stored in *BAC19* than every query having a subject a stored key will route to the stored value; in other words if a (key-value) is memorised in some Chord node, then FINDSUCCESSOR(KEY) will succeed to route the query till the value. More formally, and according to Theorem IV.3 of [32], consistency is defined as follows: “if any sequence of JOIN operations are executed interleaved with stabilization” (performed by the fundamental Chord function STABILIZE that ran periodically in every node, checking and correcting that every node does not have *dangling* successor pointers), “then at some time after the last JOIN the successor pointers will form a *cycle* on all the nodes in the network”. This property, cited by Stoica *et al.* as *Inconsistency is a “transient” state* apply also, by construction, to our *BAC19* Federation network.

Using the results from [22, 23] we can prove the following statement:

Theorem 1 *The proposed extension stores and retrieves only consistent information on Covid-19 positive cases (identified by the origin systems) and their contacts and makes it available to all origin systems.*

Proof. It is shown in the paper [23] that it might happen that stable states in a Chord network cannot be achieved if Leave and/or Put algorithms are executed in the unstable states. Thus, to avoid that in *BAC19*, it is necessary to ensure that executions of each of Algorithm 2 to Algorithm 6 do not intertwine. Since all the nodes are under the control of our system *BAC19* we can assure that this issue won’t happen.

Executions of the proposed extension are performed in the controlled environment. Due to the scheduled time intervals for running different tasks, the nodes’ leaving from the bubbles will not happen during the unstable states, i.e., there will be only runs compatible with conditions of [23, Theorems 5.3, 5.4, and 5.7]. Also, the fact that the algorithm FINDSUCCESSOR is changed guarantees that all nodes will be contacted during the search procedure, and that the retrieving procedure of the Synapse protocol [22] is fully exhaustive.

As a consequence of the mentioned adaptations of the proposed Chord model, all possible execution of *BAC19* fulfill conditions from [23, Theorems 5.3, 5.4, and 5.7]. So, with these modifications starting from the given state *BAC19* will always reach the stable state, and the retrieved information will be consistent. □

Consequently, *BAC19* solves the problem presented in the scenario with Alice and Bob, that is raised in the introduction in Problem 1.1. Let both System A and System B be part of *BAC19*. For Alice and Bob it will be sufficient that only one of them installed the system of the other region during the time of their travel, so that Alice gets informed that she is the first contact of an infected person, namely Bob during their joint travel.

5. Simulation Results

To better capture the relevance of our results, we have conducted simulations of *BAC19*. The purpose of conducted simulations is to show that the retrieving procedure of *BAC19* is fully exhaustive, like we stated in Theorem 1. The simulator is written in Python. It creates the *BAC19* topology, generates random requests for the search procedure and calculates the success rate. We expect that the simulator will not skip any node, so that the success rate of every iteration will be obviously 1.

The *BAC19* topology is created in three phases (see Figure 4). The simulator first randomly chooses which nodes will be in contact. When creating the network of each individual, special care was taken to ensure that "being in contact" is a symmetrical relation. Therefore, if persons P1 and P2 were in contact, then id of person P1 will be placed in the network of person P2, and id of person P2 will be placed in the network of person P1. According to [19] and [28], the number of close contacts of one person in a period of 14 days is between 30 and 40. For that reason, we have set in *BAC19* simulator the lower and the upper bound for the number of close contacts to be 30 and 40. Further, a *red* network is created by randomly choosing a certain percentage of nodes that will be infected. Finally, by going through the networks of all infected nodes and placing the results in *amber*, the *amber* network is created. For more details on *red* and *amber* network, please look at Figure 3 and Section 4.

The search requests were also generated completely at random. To achieve statistical significance for the performed simulations, for each configuration of the simulation we have generated 20 random networks and conducted 50 searches on every of them.

We have analyzed the results for two types of simulations:

- fixed network size (number of nodes) and variable percentage of infected nodes;
- variable network size and fixed percentage of infected nodes.

For the network size we chose among 1000, 2000 and 5000 nodes. For the percentage of infected nodes we chose among 1%, 5% and 10%. According to [5], only 13% of the world's population had been infected by SARS-CoV-2 by the end of 2020, and the same trend seems to be confirmed in 2021. That is why we assume that percentage of infected nodes at a certain point of time can not be more than 10%.

The results from *BAC19* simulator show that the success rate of the retrieving procedures is constantly 1, and it does not depend on the network size or the percentage of infected nodes. The results for specific simulations can be seen in Figure 5. The first part

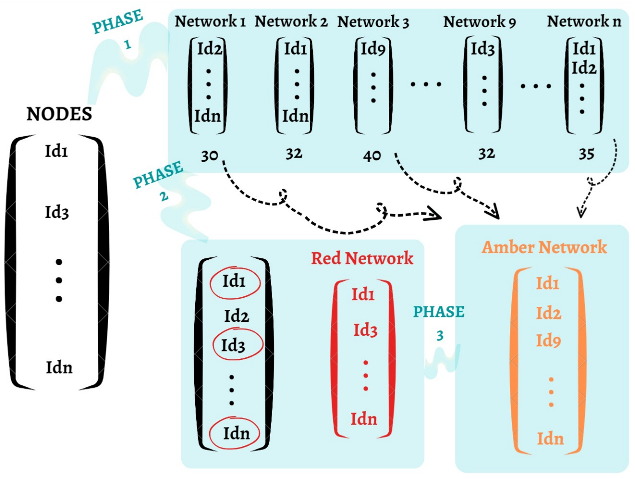

Fig. 4. Creation of *BAC19* topology

of the figure presents the results for a network with 5000 nodes. Although the percentage of infected nodes changes, the success rate remains the same. The second part of the figure presents the results for networks with 5% infected nodes. Although the size of the network changes, the success rate remains the same.

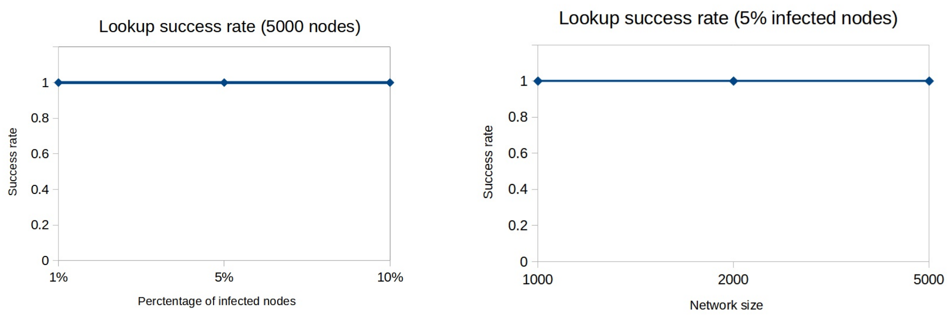

Fig. 5. Simulation results

By changing the original algorithm FINDSUCCESSOR from [31] in the way as it is presented in Algorithm 1, the search procedure slows down but the robustness of the model significantly increases. The simulation results confirm our expectations regarding the success of the search procedure, showing that the search procedure of *BAC19* is fully exhaustive.

6. Discussion

The paper [36] proposes building a common API. This approach is rather similar to the extension proposed in this paper. However, these approaches have also two significant differences:

- while [36] is building API connection points between each of two different origin systems that are connected, our extension proposes a version to common bus where each of the origin systems communicates with the proposed extension and in this way reduces and simplifies the number of connection points that needs to be maintained when several origin systems are connected;
- with *BAC19* we are simplifying also information that is being exchanged, and we do not violate privacy in the origin systems (since our extension does not collect information of an origin DCT system).

ETSI GS-E4P presents in [15] an interoperability framework for pandemic contact tracing systems which allows the centralized and decentralized modes of operation to fully interoperate.

European Community presents in [29] a guideline on interoperability specifications for cross-border transmission chains between approved apps, by using a Federation Gateway Service for synchronizing the diagnosis keys (keys of infected users) across backend servers of each national app. However, this approach focuses only on Google/Apple exposure notification apps because the majority of European countries have developed this kind of apps, and also because one Google/Apple exposure notification app can detect the contact with a user of another Google/Apple exposure notification app. In this paper we do not focus on a certain type of DCT apps, because we want to achieve the connection between them regardless the digital contact tracing technology employed and their system architecture.

However, [15, 29, 36] do not completely solve the problem presented in the scenario with Alice and Bob. Theorem 1 and simulation results show that the *BAC19* represents a much more convenient and efficient solution to this problem.

7. Conclusion and Further Work

In this paper we have presented *BAC19* a new and efficient Structured Overlay Network connecting existing systems for digital contact tracing. The advantages of *BAC19* (its usage) are:

- a person does not install anything new on his/her mobile device (except a new application which is used in the region that this person is visiting;
- the overlay does not store any personal sensitive information;
- the overlay is independent regarding how the origin system calculated contacts or is it based on Bluetooth or GPS technology;
- the overlay supports manual entry of recognized contacts;
- there are no new highly complicated calculations of possible contacts beside those that are performed by the original digital contact tracing systems.

The presented extension *BAC19* is the so-called digital forward tracing system, i.e. finding all direct contacts of an infected person. We plan in the future to explore the possibilities to adapt *BAC19* to also enable digital backward tracing system, i.e. finding the source of infection using contacts.

Furthermore, the development from a simulator to a real working system prototype should encompass the following steps:

(i) to choose one of the multiple implementations of Chord, existing, e.g., in github,
(ii) to transform the pseudocode and the simulation code into some real code,
(iii) to embed this code into an Android app: to do this, Chord implementations in Java or Kotlin are worth to be chosen and "enriched" with our pseudocode, and
(iv) to try to build a *proof of principle*, a.k.a. *proof of concept* of the whole BAC19 system.

The proof of concept results should at least overlap our simulation results and so validate our BAC19 system. The results could be presented in some standardization institute groups, e.g. ETSI ISG E4P[5]. Another potential future work is to make a systematic comparison/integration of our BAC19 solution using/extending the GAEN, Apple-Google exposure notification solution [17], which, by construction, is fully decentralized. Scientific community should be aware of the fact that Covid-19 pandemics is not finished yet, and our BAC19 system is fully compatible/parametric/polymorphic with any other kind of future virus and pandemics (e.g. Monkeypox[6]).

Acknowledgments. Authors would like to thanks anonymous referees for their useful comments that help to greatly improve the paper. This work was partly supported by the *Science Fund Republic of Serbia #6526707 AI4TrustBC*.

References

1. Abbas, R., Michael, K.: COVID-19 Contact Trace App Deployments: Learnings From Australia and Singapore. IEEE Consumer Electronics Magazine 9(5), 65–70 (2020)
2. Ahmed, N., Michelin, R.A., Xue, W., Ruj, S., Malaney, R., Kanhere, S.S., Seneviratne, A., Hu, W., Janicke, H., Jha, S.K.: A Survey of COVID-19 Contact Tracing Apps. IEEE Access 8, 134577–134601 (2020)
3. Altshuler, T.S., , Hershkovitz, R.A.: Digital Contact Tracing and the Coronavirus: Israeli and Comparative Perspectives. In: Foreign Policy at Brookings (2020)
4. Apple, Google: (google/apple) exposure notification, aka, privacy-preserving contact tracing (2020), `https://www.google.com/covid19/exposurenotifications/`; `https://www.apple.com/covid19/contacttracing/`
5. Ayoub, H.H., Mumtaz, G.R., Seedat, S., Makhoul, M., Chemaitelly, H., Abu-Raddad, L.J.: Estimates of global sars-cov-2 infection exposure, infection morbidity, and infection mortality rates in 2020. Global Epidemiology 3, 100068 (2021), `https://www.sciencedirect.com/science/article/pii/S2590113321000225`
6. Basu, S.: Effective Contact Tracing for COVID-19 Using Mobile Phones: An Ethical Analysis of the Mandatory Use of the Aarogya Setu Application in India. Cambridge Quarterly of Healthcare Ethics 30(2), 262–271 (2021)

[5] See ETSI Industry specification group "Europe for Privacy-Preserving Pandemic Protection" (ISG E4P) `https://www.etsi.org/committee-activity/activity-report-e4p`.

[6] See `https://www.cdc.gov/poxvirus/monkeypox/about.html`.

7. Börger, E., Stärk, R.F.: Abstract State Machines. A Method for High-Level System Design and Analysis. Springer (2003), http://www.springer.com/computer/swe/book/978-3-540-00702-9
8. Camus, A.: La peste. Gallimard (1947)
9. Castelluccia, C., Bielova, N., Boutet, A., Cunche, M., Lauradoux, C., Le Métayer, D., Roca, V.: ROBERT: ROBust and privacy-presERving proximity Tracing (May 2020), https://hal.inria.fr/hal-02611265, working paper or preprint
10. Castelluccia, C., Bielova, N., Boutet, A., Cunche, M., Lauradoux, C., Métayer, D.L., Roca, V.: ROBERT (ROBust and privacy-presERving proximity Tracing protocol). Tech. rep., Inria and Fraunhofer AISEC (2020), https://hal.inria.fr/hal-02611265
11. Cheklat, L., Amad, M., Omar, M., Boukerram, A.: CHEARP: chord-based hierarchical energy-aware routing protocol for wireless sensor networks. Comput. Sci. Inf. Syst. 18(3), 813–834 (2021), https://doi.org/10.2298/csis200308043c
12. Cheng, X., Yang, H., Krishnan, A.S., Schaumont, P., Yang, Y.: KHOVID: interoperable privacy preserving digital contact tracing. CoRR abs/2012.09375 (2020), https://arxiv.org/abs/2012.09375
13. El-Ansary, S., Haridi, S.: An overview of structured p2p overlay networks. Handbook on Theoretical and Algorithmic Aspects of Sensor, Ad Hoc Wireless, and Peer-to-Peer Networks (08 2005)
14. ETSI: Comparison of existing pandemic contact tracing systems. Tech. Rep. DGS E4P-002, ETSI (2021), https://www.etsi.org/deliver/etsi_gr/E4P/001_099/002/01.01.01_60/gr_E4P002v010101p.pdf
15. ETSI: Pandemic proximity tracing systems: Interoperability framework. Tech. Rep. DGS E4P-007, ETSI (2021), https://www.etsi.org/deliver/etsi_gs/E4P/001_099/007/01.01.01_60/gs_e4p007v010101p.pdf
16. Gurevich, Y.: Evolving algebras 1993: Lipari guide. In: Börger, E. (ed.) Specification and validation methods, pp. 9–36. Oxford University Press (1993)
17. Hoepman, J.H.: A Critique of the Google Apple Exposure Notification (GAEN) Framework. ArXiv abs/2012.05097 (2020)
18. Huang, J., Yegneswaran, V., Porras, P., Gu, G.: On the privacy and integrity risks of contact-tracing applications (2020)
19. Keeling, M., Hollingsworth, T., Read, J.: The Efficacy of Contact Tracing for the Containment of the 2019 Novel Coronavirus (COVID-19). J Epidemiol Community Health (10 2020)
20. Kong, X., Qi, Y., Song, X., Shen, G.: Modeling disease spreading on complex networks. Comput. Sci. Inf. Syst. 8(4), 1129–1141 (2011), https://doi.org/10.2298/CSIS110312061K
21. Liquori, L., Tedeschi, C., Vanni, L., Bongiovanni, F., Ciancaglini, V., Marinkovic, B.: Synapse: A scalable protocol for interconnecting heterogeneous overlay networks. In: Crovella, M., Feeney, L.M., Rubenstein, D., Raghavan, S.V. (eds.) NETWORKING 2010, 9th International IFIP TC 6 Networking Conference, Chennai, India, May 11-15, 2010. Proceedings. Lecture Notes in Computer Science, vol. 6091, pp. 67–82. Springer (2010), https://doi.org/10.1007/978-3-642-12963-6_6
22. Marinković, B., Ciancaglini, V., Ognjanović, Z., Glavan, P., Liquori, L., Maksimović, P.: Analyzing the exhaustiveness of the synapse protocol. Peer Peer Netw. Appl. 8(5), 793–806 (2015), https://doi.org/10.1007/s12083-014-0293-z
23. Marinković, B., Glavan, P., Ognjanović, Z.: Proving properties of the chord protocol using the ASM formalism. Theor. Comput. Sci. 756, 64–93 (2019), https://doi.org/10.1016/j.tcs.2018.10.025
24. Marinković, B., Liquori, L., Ciancaglini, V., Ognjanović, Z.: A distributed catalog for digitized cultural heritage. In: Gusev, M., Mitrevski, P. (eds.) ICT Innovations 2010 - Second International Conference, ICT Innovations 2010, Ohrid, Macedonia, September 12-15, 2010. Revised

Selected Papers. Communications in Computer and Information Science, vol. 83, pp. 176–186 (2010), https://doi.org/10.1007/978-3-642-19325-5_18
25. Marinković, B., Ognjanović, Z., Glavan, P., Kos, A., Umek, A.: Correctness of the chord protocol. Comput. Sci. Inf. Syst. 17(1), 141–160 (2020), https://doi.org/10.2298/CSIS181115017M
26. Martin, T., Karopoulos, G., Hernández-Ramos, J.L., Kambourakis, G., Fovino, I.N.: Demystifying COVID-19 Digital Contact Tracing: A Survey on Frameworks and Mobile Apps. Wireless Communications and Mobile Computing 2020(8851429), 29 (2020), https://www.hindawi.com/journals/wcmc/2020/8851429/
27. Maymounkov, P., Mazières, D.: Kademlia: A peer-to-peer information system based on the XOR metric. In: Druschel, P., Kaashoek, M.F., Rowstron, A.I.T. (eds.) Peer-to-Peer Systems, First International Workshop, IPTPS 2002, Cambridge, MA, USA, March 7-8, 2002, Revised Papers. Lecture Notes in Computer Science, vol. 2429, pp. 53–65. Springer (2002), https://doi.org/10.1007/3-540-45748-8_5
28. Mcaloon, C., Wall, P., Butler, F., Codd, M., Gormley, E., Walsh, C., Duggan, J., Murphy, T., Nolan, P., Smyth, B., O'Brien, K., Teljeur, C., Green, M., O'Grady, L., Culhane, K., Buckley, C., Carroll, C., Doyle, S., Martin, J., More, S.: Numbers of close contacts of individuals infected with SARS-CoV-2 and their association with government intervention strategies. BMC Public Health 21 (12 2021)
29. eHealth Network: Interoperability specifications for cross-border transmission chains between approved apps (2020)
30. Ocheja, P., Cao, Y., Ding, S., Yoshikawa, M.: Quantifying the privacy-utility trade-offs in COVID-19 contact tracing apps (2020)
31. Stoica, I., Morris, R.T., Karger, D.R., Kaashoek, M.F., Balakrishnan, H.: Chord: A scalable peer-to-peer lookup service for internet applications. In: Cruz, R.L., Varghese, G. (eds.) Proceedings of the ACM SIGCOMM 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication, August 27-31, 2001, San Diego, CA, USA. pp. 149–160. ACM (2001), https://doi.org/10.1145/383059.383071
32. Stoica, I., Morris, R.T., Liben-Nowell, D., Karger, D.R., Kaashoek, M.F., Dabek, F., Balakrishnan, H.: Chord: a scalable peer-to-peer lookup protocol for internet applications. IEEE/ACM Trans. Netw. 11(1), 17–32 (2003), https://doi.org/10.1109/TNET.2002.808407
33. Tang, Q.: Privacy-preserving contact tracing: current solutions and open questions (2020)
34. Taylor, L., Sharma, G., Martin, A., Jameson, S. (eds.): Data justice and COVID-19: Global perspectives. Meatspace Press (aug 2020)
35. Troncoso, C., et al.: Decentralized Privacy-Preserving Proximity Tracing. Tech. rep., École Polytechnique Fédérale de Lausanne, ETH Zurich, KU Leuven, Delft University of Technology, University College London, Helmholtz Centre for Information Security, University of Torino, ISI Foundation (2020), https://github.com/DP-3T/documents/blob/master/DP3T%20White%20Paper.pdf
36. Vukolic, M.: On the interoperability of decentralized exposure notification systems. CoRR abs/2006.13087 (2020), https://arxiv.org/abs/2006.13087

Silvia Ghilezan is a Professor of mathematics with the University of Novi Sad and Mathematical Institute of the Serbian Academy of Sciences and Arts. On several occasions she has held visiting positions at University of Oregon, École Normale Supérieure de Lyon, Université Paris Diderot - Paris 7, University of Turin, Radboud University and McGill University. The major lines of her research are in mathematical logic with application to programming languages, concurrency and mathematical linguistics. Her current research interests include formal methods for new challenges in privacy protection and artificial intelligence. She has collaborated with over seventy co-authors on publications in leading scientific journals and conferences (POPL, LPAR, TLCA, PPDP), books and editorials. She acts as a SC member of FSCD, an advisor for ARVR enterprises and industry, a popularizer of science and a promoter of gender balance in science. She was awarded the distinction Chevalier (2013) and Officier (2021) de l'Ordre des Palmes Académiques of the French Republic.

Simona Kašterović is a teaching assistant at the Faculty of Technical Sciences, University of Novi Sad. She received her B.Sc. degree at the Faculty of Sciences, University of Novi Sad in 2015. In 2017 she received her M.Sc. degree at the Faculty of Technical Sciences, University of Novi Sad. Currently, she is a Ph.D. student in applied mathematics at the same faculty. In 2018 she spent three months as visiting researcher at University Paris Diderot (Paris 7) in Paris, France. Her research interests include mathematical logic and its application in computer science: proof theory, lambda calculus, type theory; uncertain reasoning; probabilistic logic; computer assisted mathematical reasoning, formal methods for artificial intelligence.

Luigi Liquori got his MS in 1990 at Udine University, Italy. He got his Ph.D. in 1996 at University of Turin, Italy, and his H.d.R. in 2007 at Institut National Polytechnique de Lorraine, France. He served as Lecturer at the Ecole Nationale des Mines de Nancy from 1999. Since 2001, he is a senior researcher at French Institute for Research in Computer Science and Automation. His research's fields range from logics, type theory, lambda calculus, foundations of interactive proof assistants, to semantics of object oriented programming languages, until foundations of overlay networks, IoT protocols, and recently digital contact tracing against Covid-19.

Bojan Marinković got his PhD during 2014 at The Faculty of Technical Sciences University of Novi Sad, Serbia. Currently, he is Data Architect at Clarivate, Serbia. Until October 2018, he has been Research Assistant Professor at Mathematical Institute of the Serbian Academy of Sciences and Arts, however still interested in research in the following domains: distributed systems, applications of non-classical mathematical logic in computer science, and digitization of cultural and scientific heritage. During 2009, he spent three months as visiting researcher at Inria Sophia Antipolis, France.

Zoran Ognjanović is a research professor at the Mathematical Institute of the Serbian Academy of Sciences and Arts. He received his PhD degree in mathematical logic from University of Kragujevac, Serbia, in 1999. He has authored or coauthored two monographs, and a number of technical papers in major international journals and conferences. His research interests concern: applications of mathematical logic in computer science, artificial intelligence and uncertain reasoning, automated theorem proving, applications of heuristics to satisfiability problem, and digitization of cultural and scientific heritage. He is a recipient of the Serbian Academy of Sciences and Arts Award in the field of mathematics and related sciences for 2013 and the annual award of Serbian Ministry of Science for results in fundamental research in 2004.

Tamara Stefanović is a teaching assistant at the Faculty of Technical Sciences, University of Novi Sad, Serbia. She received her B.Sc. degree at the Faculty of Sciences, University of Novi Sad in 2016. In 2019 she received her M.Sc. degree at the same faculty. Currently, she is a Ph.D. student in applied mathematics at the Faculty of Technical Sciences, University of Novi Sad. Her current scientific focus is on mathematical logic and its application in computer science, especially mathematical models for data privacy.

Received: August 25, 2021; Accepted: July 30, 2022.