MAPPING POTENTIAL ATTACKERS AGAINST NETWORK SECURITY USING LOCATION-AWARE REACHABILITY QUERIES ON GEO-SOCIAL DATA

Hafara Firdausi[1)], Bagus Jati Santoso[2)], Rohana Qudus[3)], Henning Titi Ciptaningtyas[4)]
[1, 2, 3, 4)] Department of Informatics, Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia 60111
e-mail: hafara.19051@mhs.its.ac.id[1)], bagus@if.its.ac.id[2)], rohanaq@gmail.com[3)], henning@if.its.ac.id[4)]

ABSTRACT

Attacks on network security can happen anywhere. Using Geo-Social Networks (GSN), i.e., a graph that combines social network data and spatial information, we can find the potential attackers based on the given location. In answering the graph-based problems, Reachability Queries are utilized. It verifies the reachability between two nodes in the graph. This paper addresses a problem defined as follows: Given a geo-social graph G and a location area q as a query point, we map potential attackers against network security using location-aware reachability queries. We employ the concepts of Reachability Minimum Bounding Rectangle (RMBR) and graph traversal algorithm, i.e., Depth-First Search (DFS), to answer the location-aware reachability queries. There are two kinds of the proposed solution, i.e., (1) RMBR-based solution map potential attackers by looking for intersecting RMBR values, and (2) Graph traversal-based solution map potential attackers by traversing the graph. We evaluate the performance of both proposed solutions using synthetic datasets. Based on the experimental result, the RMBR-based solution has much lower execution time and memory usage than the graph traversal-based solution.

***Kata Kunci**: Geo-Social Network, graph traversal, reachability query.*

PEMETAAN POTENSI PENYERANG TERHADAP KEAMANAN JARINGAN MENGGUNAKAN *LOCATION-AWARE REACHABILITY QUERIES* PADA DATA GEO-SOSIAL

Hafara Firdausi[1)], Bagus Jati Santoso[2)], Rohana Qudus[3)], Henning Titi Ciptaningtyas[4)]
[1, 2, 3, 4)] Departemen Teknik Informatika, Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia 60111
e-mail: hafara.19051@mhs.its.ac.id[1)], bagus@if.its.ac.id[2)], rohanaq@gmail.com[3)], henning@if.its.ac.id[4)]

ABSTRAK

Serangan terhadap keamanan jaringan dapat terjadi di mana saja. Dengan memanfaatkan Geo-Social Networks (GSN), yaitu grafik yang menggabungkan data jaringan sosial dan informasi lokasi, penyerang potensial pada suatu lokasi terjadinya serangan dapat ditemukan. Untuk menjawab permasalahan berbasis grafik, digunakanlah pendekatan Reachability Queries untuk menguji apakah terdapat jalur dari satu node ke node lainnya. Penelitian ini membahas masalah yang didefinisikan sebagai berikut: diberikan geo-social graph G dan area lokasi q sebagai titik kueri, penelitian ini memetakan penyerang potensial terhadap keamanan jaringan dengan memanfaatkan Location-aware Reachability Queries. Kami menggunakan konsep Reachability Minimum Bounding Rectangle (RMBR) dan algoritma graph traversal, yaitu, Depth-First Search (DFS), untuk menjawab Location-aware Reachability Queries. Terdapat dua jenis solusi yang diusulkan dalam penelitian ini, yaitu (1) solusi berbasis RMBR yang memetakan potensi penyerang dengan mencari daerah RMBR yang beririsan, dan (2) solusi berbasis graph traversal yang memetakan potensi penyerang dengan menelusuri grafik. Performa kedua solusi yang diusulkan dievaluasi menggunakan data sintetis. Berdasarkan hasil uji coba, solusi berbasis RMBR memiliki waktu eksekusi dan penggunaan memori jauh lebih rendah dibandingkan solusi berbasis graph traversal.

***Keywords**: Geo-Social Network, graph traversal, reachability query.*

I. INTRODUCTION

The utilization of Geo-Social Networks (GSN) has become a trend in recent years for several location-based services, such as a personalized point-of-interest (POI) recommendation, advertisement, and cyber-security [1]. Moreover, with the increasing availability of Wi-Fi and GPS-enabled mobile devices, users can provide their spatial information, e.g., current geographic location, to the existing social networks in various

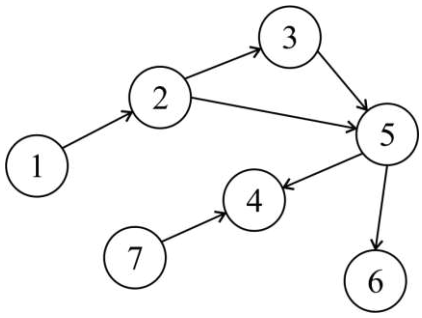

Fig. 1. The example of Reachability Query.

ways. For example, on Instagram, 1 billion monthly active users can upload location-tagged photos and add location information to their posts or stories. On the earlier social networks, such as Facebook and Twitter, users can do "check-in" by adding their current location when writing posts or tweets [1]. The social network data combined with spatial information can form what is usually known as a *Geo-Social Network* (GSN) [2], [1], [3]. Generally, it is a graph consisting of nodes representing entities, e.g., users or places, and edges representing relationships between the entities.

One of the applications of geo-social data is in the domain of network security. Consider the following scenario. Assume we are a network administrator who wants to look for potential attackers against network security in a particular area. Then, we collect geo-social data from several social networks and generate a geo-social graph from the given dataset. Using this geo-social graph, we can find the potential attackers based on the given location. One of the basic approaches to answer graph-based problems is by checking the reachability between two nodes in the graph. This approach is commonly known as *Reachability Query* [1], [4]–[10]. A reachability query, denoted by $R(n_1, n_2)$ verify whether a node $n_1 \in N$ is reachable from the other node $n_2 \in N$ by finding a path connecting them. For example, based on Fig. 1, the answer of a reachability query between node 1 to node 6, denoted by $R(1,6)$, is true. However, the result of a reachability query between node 1 to node 7, denoted by $R(1,7)$, is false because it is a directed graph.

This paper addresses a problem defined as follows: given a geo-social graph G and a location area q as query point, we map potential attackers against network security using location-aware reachability queries. We employ the concepts of *Reachability Minimum Bounding Rectangle* (RMBR) [1] and graph traversal algorithm, i.e., *Depth-First Search* (DFS) [11] to answer the location-aware reachability queries. There are two kinds of the proposed solution to map potential attackers, i.e., *RMBR-based solution* and *Graph traversal-based solution*. The RMBR-based solution map potential attackers by looking for intersecting RMBR values, while the Graph traversal-based solution map potential attackers by traversing the graph. We evaluate the performance of the proposed approaches, in terms of execution time and memory usage, using synthetic datasets.

The rest of the paper is organized as follows. We present the related works of the Geo-Social Network (GSN) in Section II. Section III explains the detail of our proposed method, including the algorithm and data structure. Then, we evaluate and discuss the performance of our proposed method in Section IV. Finally, Section VI concludes the paper.

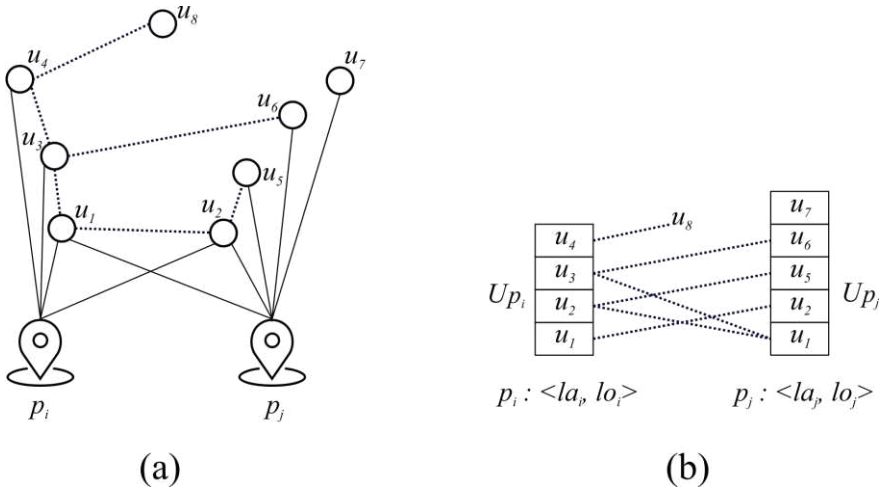

Fig. 2. Geo-Social Network (GSN).

II. Related Work

There are several previous works related to Geo-Social Network (GSN) processing applied in various fields. Firstly, this work [2] focused on extracting useful information by combining both the social relationship and users' current location. Armenatzoglou et al. formulated a novel Geo-Social Network query, called *Nearest Star Group* (NSG), to find a user group with an m-member that forms a star subgraph and has a minimum total Euclidean distance of its member to the given query location. The NSG query utilized two basic GSN queries, i.e., *Range Friends* (RF) to finds a set of friends of a user within a given range, and *Nearest Friends* (NF) to finds a set of nearest friends of a user to a given location.

Later, several studies focus on answering reachability queries using two approaches, i.e., indexing [4], [7] and labelling [5], [6], [8]–[10], [12]. Liang et al. [7] proposed an improved hop-based reachability indexing scheme 3-Hop* and a two-stage node filtering algorithm based on 3-Hop* for efficiently answering graph pattern queries. The experimental result demonstrated that the proposed approaches achieve faster reachability query evaluation and less indexing costs than state-of-the-art methods. In anoth er work, Veloso et al. [4] proposed a novel indexing method named FELINE (*Fast rEfined online search*) to provide reachability information in constant time for a significant portion of queries. The FELINE approach utilized the concepts of Weak Dominance Drawing to indexing an extensive graph. Based on the conducted experiments, FELINE outperforms its state of the arts in terms of query and construction time and index size.

In [5], Du et al. proposed a new labelling scheme, named *HT*, to accelerate k-hop reachability query answering. The HT approach consists of two kinds of the label. First is a constrained 2-hop distance label to capture a certain percentage of reachability information and the shortest paths between a set of hop nodes and other nodes. Second is two complementary topological levels to determine whether the given query point is unreachable. Another work by Wei et al. [10] proposed a new labelling approach to answer reachability queries. The proposed approach has employed the randomness of Independent Permutation (IP). Then, Su et al. [8] improved the performance of IP developed in [10] by proposing a novel *Bloom Filter Labelling* (BFL) that effectively pruned data to answer more reachability queries directly.

Duan et al. [6] proposed a new labelling index method based on graph stratification (GSL). The GSL method utilized the properties of a bipartite graph to link all nodes into a mutually disjoint chain structure, i.e., chain-in and chain-out structure. Another work by Zhou et al. [9] focused on answering reachability queries in a high dimension graph by proposing an efficient labelling approach named MGTag. The MGTag transforms a graph into several partitions, then utilized a four-dimensional labelling scheme to generate a graph labelling index.

In addition to the two widely studied approaches, Sarwat et al. [1] proposed an approach to answer graph reachability queries with spatial range predicates (*RangeReach*) on a Geo-Social Network, called *GeoReach*. The authors also proposed a data structure for indexing graphs with spatial indexing entries called *Spatially-Augmented Graph*. Moreover, the GeoReach approach employed a pruned-graph traversal algorithm.

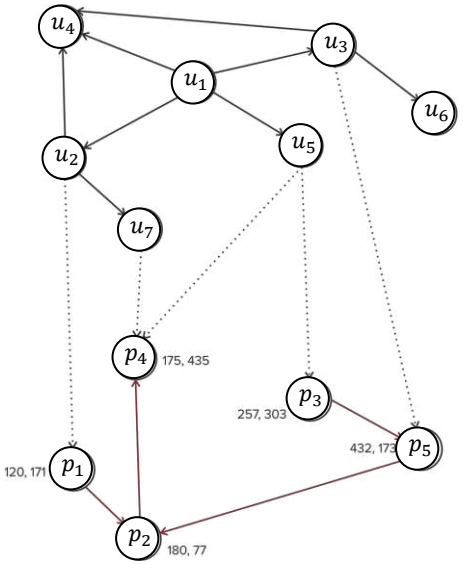

Fig. 3. Graph initialization.

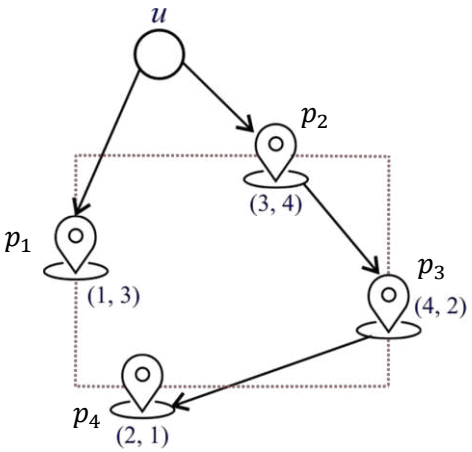

Fig. 4. Reachability Minimum bounding Rectangle (RMBR) of u.

Inspired by work in [1], this paper study the application of graph reachability query in the network security area. We propose an approach to map potential attackers against network security using location-aware reachability queries on geo-social data. Firstly, we define the Location-aware reachability query. Then we utilize the concepts of Reachability Minimum Bounding Rectangle (RMBR) and graph traversal algorithm proposed in [1] to answer the reachability query.

III. Problem and Proposed Method

This section consists of two parts. Firstly, we define the problem answered in this paper and briefly explain several concepts we use. Secondly, we describe the proposed method to answer the defined problem.

A. Problem Definition

Given geo-social data consisting of a set of users U and a set of places P. We generate a geo-social network from the given dataset as a directed graph, denoted by $G = (N, E)$, where N denotes the nodes and E denotes the edges. Each node $n_i \in N$ can represent two kinds of objects, i.e., users and places. Specifically, we denote a user and a place as $u_i \in N$ and $p_i \in N$, respectively, where $i < |N|$. Each place is defined by its spatial coordinates, i.e., latitude as la and longitude as lo. Hence, the location of a place is denoted by $p_i{:} < la_i, lo_i >$. As illustrated in Fig. 2 (a), there are two kinds of edge $e_i \in E$, i.e., the dashed lines and the solid lines, each representing a different type of relationship. The dashed line between two users, e.g., u_1 and u_2, represents the friendship relation. At the same time, the solid line between a user and a place, e.g., u_1 and p_1, respectively, indicates that the user u_1 has checked in at the place p_1. A group of users who have checked in at a place p_i are denoted by Up_i as shown in Fig. 2 (b). The problem answered in this paper defined as follows:

Definition 1 (*Location-aware Reachability Query*). Given a geo-social graph G and a location area q as query point that defined by a minimum bounding rectangle (MBR), we find potential attackers against network security on the given location.

$$LRQ(G, q) = \sum_{i \in PA} u_i \in N \tag{1}$$

B. Proposed Method

There are two main parts of the proposed method, i.e., initialization and mapping. In the first part, we initialize the graph with the given dataset and calculate the *Reachability Minimum Bounding Rectangle* (RMBR) value for each node. In the second part, we map the potential attackers using two kinds of approach, i.e., RMBR-based solution and graph traversal-based solution.

TABLE I
ATTRIBUTES OF (A) USER NODE, (B) PLACE NODE, AND (C) EDGE.

(A)

Attribute	Description
ID	User ID
Name	User's name
Friends	A list of user nodes with friendship relation
Check-Ins	A list of check-in histories
RMBR	Reachability Minimum Bounding Rectangle value

(B)

Attribute	Description
ID	Place ID
Name	Place's name
Lat	Latitude of location
Long	Longitude of location
Follow	A list of places followed
RMBR	Reachability Minimum Bounding Rectangle value

(C)

Attribute	Description
ID_1	ID of the source node
ID_2	ID of the destination node

TABLE II
THE DATASET OF (A) USERS, (B) PLACES, (C) RELATIONS BETWEEN USERS, (D) RELATIONS BETWEEN PLACES, AND (E) RELATIONS BETWEEN USERS AND PLACES.

(A)

ID	Name
u_1	Budi
u_2	Lia
u_3	Agung
u_4	Rara
u_5	Satria
u_6	Ilham
u_7	Lestari

(B)

ID	Name	Lat	Long
p_1	Café A	120	171
p_2	Café B	180	77
p_3	Town Square	257	303
p_4	Library	175	435
p_5	Park	432	173

(C)

ID_1	ID_2
u_1	u_2
u_2	u_7
u_1	u_3
u_2	u_4
u_3	u_4
u_1	u_4
u_3	u_6
u_3	u_3
u_1	u_5

(D)

ID_1	ID_2
p_1	p_2
p_3	p_5
p_5	p_2
p_2	p_4

(E)

ID_1	ID_2
u_3	p_5
u_5	p_4
u_7	p_4
u_5	p_3
u_2	p_1

1) Initialization

Consider the given dataset in Table II. There are five kinds of dataset given, i.e., users, places, relations between users, relations between places, and relations between users and places. All data has attributes defined in Table I. Each user node stores its identity (ID), name, a list of other user nodes with friendship relations, a list of check-in histories in several locations, and RMBR value. Table I (a) shows the attributes stored by each user node. At the same time, each place node stores its identity (ID), name, geographic coordinate consisting of latitude and longitude, a list of other places followed by the node, and RMBR value, as shown in Table I (b). Table I (c) describes the attributes stored by each edge, i.e., the identity of the source node and destination node.

The initialization process consists of three steps: (1) graph initialization, (2) RMBR value initialization, and (3) RMBR value update. Firstly, we create a directed graph using the given dataset, as illustrated in Fig. 3. The directed graph is one-way, based on the relationship given in Table II. Algorithm 1 presents graph initialization steps. There are two graphs initialized, i.e., (1) *loc_graph* that used for updating the RMBR value of place nodes and (2) *full_graph* that used in graph traversal-based solution.

Secondly, we initialize the Reachability Minimum Bounding Rectangle (RMBR) value of each node, defined as follows:

Definition 2 (*Minimum Bounding Rectangle* [13]). Minimum Bounding Rectangle (MBR), also known as "bounding box", is a two-dimensional area extensively used to approximate a more complex object in the spatial data structures. We denote an MBR as follows:

$$MBR = < \min(x), \min(y), \max(x), \max(y) > \tag{2}$$

Definition 3 (*Reachability Query*[1] [4]). Given a node $n_1 \in N$ and other node $n_2 \in N$ in a directed graph G, the reachability query, denoted by $R(n_1, n_2)$, verify whether node n_1 is reachable from the other node n_2 by finding a path connecting them. Mathematically, the reachability query can be defined as follows:

$$R(n_1, n_2) = \begin{cases} true & if\ n_1 = n_2\ \ or\ \ \exists (n_1, n_3) \in E \wedge R(n_3, n_2) \\ false & otherwise \end{cases} \tag{3}$$

Algorithm 1 Graph Initialization

```
for l in location do
    for i in follow_list do
        loc_graph ← i
        full_graph ← i
    end
end
for p in user do
    for i in friend_list do
        full_graph ← i
    end
    for j in checkin_list do
        full_graph ← j
    end
end
```

Fig. 5. Graph initialization algorithm.

Algorithm 2 Graph Traversal

```
let Q be list
while Q do
    v = Q.pop()
    if v not in path then
        path[v] = path + [v]
        Q = graph[v] + Q
    end
end
```

Fig. 6. Graph traversal algorithm.

Definition 4 (*Reachability Minimum Bounding Rectangle* [1]). Given a node $n \in N$, the Reachability Minimum Bounding Rectangle of node n_1, denoted by $RMBR(n_1)$, is an MBR consisting of all nodes $n_2 \in N$ that reachable from the given node n_1, denoted by $RN(n_1)$. Specifically, the Reachability Minimum Bounding Rectangle can be defined as follows:

$$RMBR(n_1) = < \min(X), \min(Y), \max(X), \min(Y) >$$
$$\text{where } X_{n_1} = \sum_{i \in RN(n_1)} \min(x) \wedge \max(x) \in n_i, \quad Y_{n_1} = \sum_{i \in RN(n_1)} \min(y) \wedge \max(y) \in n_i \tag{3}$$

For example, as illustrated in Fig. 4, the reachable nodes of node u is $RN(u) = \{p_1, p_2, p_3, p_4\}$, where $X = \{1, 2, 4, 2\}$ and $Y = \{3, 4, 2, 1\}$. The Reachability Minimum Bounding Rectangle of node u is calculated as follows: $min(X) = 1$, $min(Y) = 1$, $max(X) = 4$, and $max(Y) = 4$. Hence, $RMBR(u) =< 1, 1, 4, 4 >$.

Based on the above definitions, we initialize the RMBR value of each node on the formed graph, both place and user nodes. The RMBR of place nodes is defined by their latitude and longitude coordinate. For example, using the dataset given in Table II (b), we set $\min(x)$ and $\max(x)$ with the latitude (Lat) coordinate and $\min(y)$ and $\max(y)$ with the longitude ($Long$) coordinate. Table III (a) demonstrate the initial value of RMBR for each place node. At the same time, we set the RMBR value of user nodes as empty.

Thirdly, we update the RMBR value of each node, divided into two steps: (1) update the place nodes, and (2) update the user nodes. We update the RMBR value of place node p_1 by identifying the other place nodes P' that the node p_1 follows, denoted by $RN(p_1) = P'$. For example, we want to update the RMBR value of place node p_1 with initial RMBR value $RMBR(p_1) =< 120, 171, 120, 171 >$. Based on the Table II (d), we know that the node p_1 follows p_2 and the node p_2 follows p_4. Hence, the followed place nodes of p_1 is $RN(p_1) = \{p_1, p_2, p_4\}$, with $X_{p_1} = \{120, 180, 175\}$ and $Y_{p_1} = \{171, 77, 435\}$. The updated RMBR value of p_1 is calculated as follows: $\min(X_{p_1}) = 120$, $\min(Y_{p_1}) = 77$, $\max(X_{p_1}) = 180$, and $\max(Y_{p_1}) = 435$, so, $RMBR(p_1) =< 120, 77, 180, 435 >$. The RMBR update process of place nodes is pseudo-coded in lines 1-10 of Algorithm 3.

TABLE III
THE RMBR VALUE OF (A) PLACE NODES BEFORE UPDATED, (B) PLACE NODES AFTER UPDATED, AND (C) USER NODES AFTER UPDATED.

(A)

ID	Min x	Min y	Max x	Max y
p_1	120	171	120	171
p_2	180	77	180	77
p_3	257	303	257	303
p_4	175	435	175	435
p_5	432	173	432	173

(B)

ID	Min x	Min y	Max x	Max y
p_1	120	77	180	435
p_2	175	77	180	435
p_3	175	77	432	435
p_4	175	435	175	435
p_5	175	77	432	435

(C)

ID	Min x	Min y	Max x	Max y
u_1	120	77	432	435
u_2	120	77	180	435
u_3	175	77	432	435
u_4	-	-	-	-
u_5	175	77	432	435
u_6	-	-	-	-
u_7	175	435	175	435

Algorithm 3: RMBR Update

```
for l in location do
    for i in follow_list do
        if l.rmbr[min] > i.rmbr[min] then
            l.rmbr[min] = i.rmbr[min];
        end
        if l.rmbr[max] < i.rmbr[max] then
            l.rmbr[max] = i.rmbr[max];
        end
    end
end
for p in user do
    if p.checkin > 0 and p.friends = 0 then
        for i in checkin_list do
            p.rmbr ← i.rmbr;
        end
    end
    if p.friends > 0 and p.checkin = 0 then
        for j in friend_list do
            p.rmbr ← j.rmbr;
        end
    end
    if p.rmbr ≠ ∅ and p.friends > 0 then
        for k in friend_list do
            if p.rmbr[min] > k.rmbr[min] then
                p.rmbr[min] = k.rmbr[min];
            end
            if p.rmbr[max] < k.rmbr[max] then
                p.rmbr[max] = k.rmbr[max];
            end
        end
    end
end
```

Fig. 7. RMBR update algorithm.

The process continues with updating the RMBR value of user nodes. There are three kinds of condition to update the RMBR of a user u_1: (1) if the user node u_1 has no relationship with other users but has checked-in at some places P', then $RN(u_1) = P'$; (2) if the user node u_1 has never checked-in at some places, but has a relationship with other user nodes U', then $RN(u_1) = U'$; and (3) if the user node u_1 has checked-in at some places P' and has a relationship with some users U', then $RN(u_1) = P' \cup U'$. Lines 11-32 of Algorithm 3 show the RMBR value update process of user nodes.

For example, we want to update the RMBR value of a user node u_1. Based on the Table II (d), user $u_1$has relationship with u_2, u_3, u_4, u_5, so, $RN(u_1) = \{u_1, u_2, u_3, u_4, u_5\}$. Firstly, we need to calculate the RMBR of each friend node of u_1, e.g., u_2. User u_2 has relationship with user u_4 and u_7, and has checked-in at place p_1, denoted by $RN(u_2) = \{u_4, u_7, p_1\}$. The user u_4 has no relationship and never checked-in anywhere, so, $RN(u_4) = \{\emptyset\}$

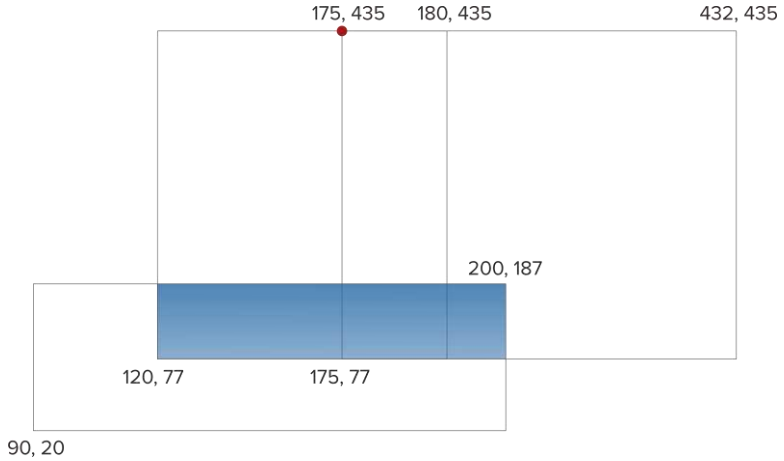

Fig. 8. Reachability Minimum Bounding Rectangle-based solution.

Algorithm 4: RMBR-based Solution

```
for p in user do
    if p.rmbr ≠ ∅ then
        if p.rmbr[min] > input[max] or p.rmbr[max] < input[min] then
            continue
        end
        suspect ← p
    end
end
return suspect
```

Fig. 9. RMBR-based solution algorithm.

Algorithm 5: First step of Graph traversal-based Solution

```
for l in location do
    if l.rmbr[min] > input[max] or l.rmbr[max] < input[min] then
        continue
    end
    sites ← l
end
```

Fig. 10. First step of graph traversal-based solution algorithm.

Algorithm 6: Second step of Graph traversal-based Solution

```
let path and Q be dict
Q[start] = 0
while Q do
    v, hop = Q.popitem()
    if not v in path then
        path[v] = hop
        for neighbor in graph[v] do
            if neighbor not in path, q then
                q[neighbor] = hop + 1
            end
        end
    end
end
```

Fig. 11. Second step of graph traversal-based solution algorithm.

and $RMBR(u_4) = <>$. While the user u_7 has no relationship, but has check-in at place p_4, so, $RN(u_7) = \{p_4\}$ and $RMBR(u_7) = < 175, 435, 175, 435 >$. From the Table III (b), we know that $RMBR(p_1) = < 120, 77, 180, 435 >$. Hence, given $X_{u_2} = \{175, 175, 120, 180\}$ and $Y_{u_2} = \{435, 435, 77, 435\}$, the RMBR of u_2 is calculated as follows: $\min(X_{u_2}) = 120$, $\min(Y_{u_2}) = 77$, $\max(X_{u_2}) = 180$, $\max(Y_{u_2}) = 435$. The updated RMBR value of user node u_2 is $RMBR(u_2) = < 120, 77, 180, 435 >$. Finally, after calculating the RMBR of each friend node u_1, we get $RMBR(u_1) = RMBR(u_2, u_3, u_4, u_5) = < 120, 77, 432, 435 >$. Table III (c) demonstrate the updated RMBR value of user nodes.

In updating the RMBR value, we utilize a graph traversal algorithm named *Depth-First Search* (DFS) to explore each branch as far as possible until it gets the deepest node of each branch. This algorithm will first visit the deepest node to update the RMBR value and then move to the upper level until the root node. Algorithm 2 demonstrate the DFS algorithm used in the initialization step.

2) Reachability Minimum Bounding Rectangle-based Solution

We propose a method based on Reachability Minimum Bounding Rectangle to map potential attackers in the given location. Using the RMBR value of each node calculated in the first step, we search the RMBR area of nodes that intersects with the given location. For example, given a location area $q = < 90, 20, 200, 187 >$ as query point and a directed graph G, we find potential attackers on the given location $LRQ(G, q) = \{p_1, p_2, p_3, p_5\}$. Fig. 8 illustrates the area of intersection with the given location, denoted by blue area. However, this approach

TABLE IV
MAPPING RESULTS USING GRAPH TRAVERSAL-BASED SOLUTION.

ID Place	ID User	Hop
p_1	u_2	1
p_1	u_1	2
p_2	u_2	2
p_2	u_3	2
p_2	u_1	3
p_2	u_5	3
p_3	u_5	1
p_3	u_1	2
p_5	u_3	1
p_5	u_1	2
p_5	u_5	2

TABLE V
VALUE OF PARAMETERS USED IN EXPERIMENTS.

Parameter	Values
Number of nodes	1000, 2000, **5000**, 10000
Number of edges	5, 10, 20, 30, 40
User and place data ratios	25:75, **50:50**, 75:25

only maps a list of potential attackers on the given location without calculating each user's proximity to the given location or the probability of each user to be a potential attacker. The Reachable Minimum Bounding Rectangle-based solution is pseudo-coded in Algorithm 4.

3) Graph Traversal-based Solution

As a comparison, we also propose a method based on the graph traversal to map potential attackers in the given location. The graph traversal-based solution consists of two steps. Firstly, we search a list of places located at the query location area by looking for the intersecting RMBR values with the query location. This first step is pseudo-coded in Algorithm 5. Secondly, using a retrieved list of places, we trace each place one by one to find a list of nodes that can reach that place. We store each node's score, represented by the number of hops it takes to get to that place. Algorithm 6 presents the second step of graph traversal-based solution for potential attackers mapping.

For example, based on Table III (b), there are four places with RMBR values that intersect with the query location, i.e., p_1, p_2, p_3, and p_5. These place nodes will be used as starting points of the graph transversal process. Table IV demonstrate the mapping result of the graph traversal-based solution. Each user has a proximity value to the query location, represented as the number of hops. The smaller the hop value, the more likely the user is to attack.

IV. EXPERIMENTAL RESULT

In this section, we evaluate the performance of two proposed solutions, i.e., (1) RMBR-based and (2) Graph transversal-based solution, to map potential attackers against network security using location-aware reachability queries on geo-social data. The proposed solutions are evaluated based on the computation costs used, i.e., query execution time and memory usage.

Environmental setting and Dataset. We perform the experiments using a synthetic dataset by varying the number of several parameters defined in Table V. The synthetic dataset is generated using Python, as well as the implementation of the proposed methods. All the experiments run on a computer with an Intel (R) Core (TM) Processor i7-5500U CPU @ 2.40GHz x 4 and 12GB RAM.

A. Query Execution Time

There are three parameters we study in the experiments, i.e., the number of nodes, the number of edges, and the ratio of user and place nodes. First, we study the effect of these three parameters on the query execution time.

Effect of the number of nodes. We vary the number of nodes as follows: 1 x 10^3, 2 x 10^3, 5 x 10^3, 10 x 10^3, while the number of edges is generated randomly according to the number of nodes. Fig. 12 represents the query execution time of the two proposed solutions. The RMBR-based solution takes an average query execution time of less than one second for each scenario. In contrast, the query execution time of the Graph traversal-based approach is always above one second. It implies that utilizing the *Reachability Minimum Bounding Box* concept decreases the query execution time by 99%.

The RMBR-based solution has complexity $O(n)$, representing that this algorithm's complexity increases linearly and is directly proportional to the number of nodes. As shown in Fig. 12, the number of nodes less affects query execution time in RMBR-based solutions. In contrast, the increasing number of nodes significantly affects the query execution time of the Graph traversal-based solution. It appears because the Graph traversal-based solution has complexity $O(n(N + E))$, where N is the number of nodes and E is the number of edges being traced.

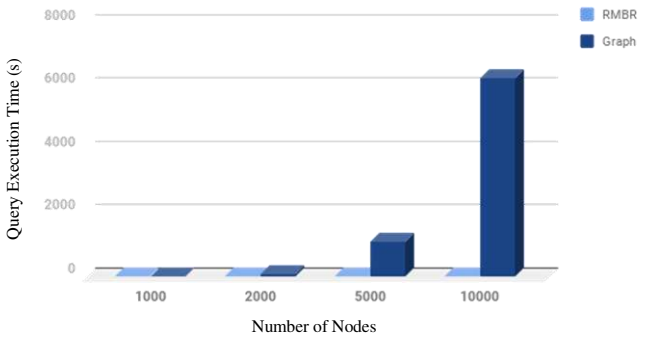

Fig. 12. Effect of the number of nodes on query execution time.

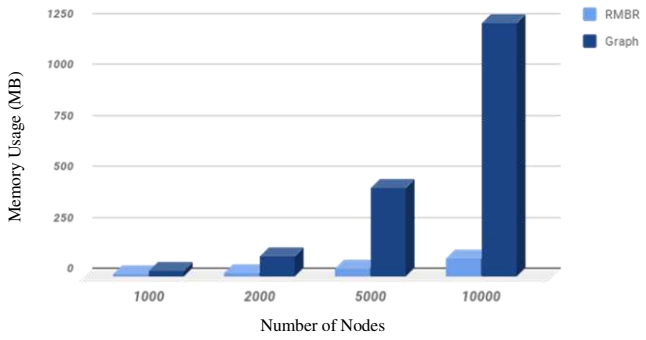

Fig. 13. Effect of the number of nodes on memory usage.

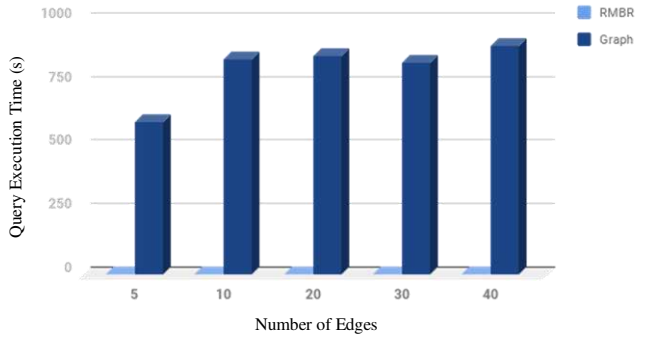

Fig. 14. Effect of the number of edges on query execution time.

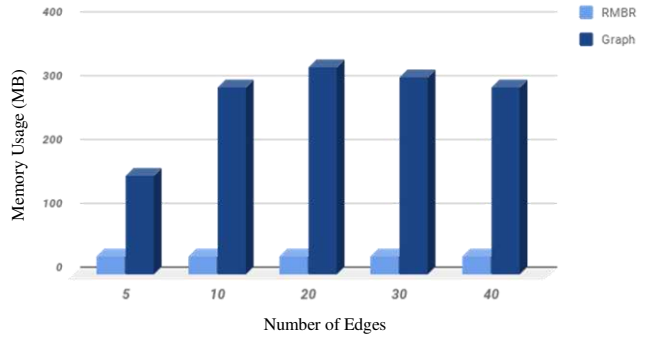

Fig. 15. Effect of the number of edges on memory usage.

The RMBR-based solution performs well on a graph with many nodes, up to 10 x 10^3 nodes in this experiment. The query execution time of this solution is reasonably stable in all scenarios. However, the query results of this solution are less informative; it only presents a list of users that can be potential attackers without providing the probabilities. On the other hand, the Graph traversal-based solution presents the probability a user can be a potential attacker by tracing and calculating their proximity to the place node in the query area. Consequently, this approach performs poorly in query execution time, especially on a graph with a large number of nodes.

Effect of the number of edges. We use variations in the number of edges as follows: 5, 10, 20, 30, 40, with the number of nodes for users and places, each of which is 5000 nodes. Based on Fig. 14, we recognize that the execution time of the RMBR-based solution remains stable even as the number of edges increases. In comparison, the execution time of the Graph traversal-based solution is reasonably unstable as the number of edges increases.

As previously discussed, the RMBR-based algorithm has complexity $O(n)$, while the Graph traversal-based solution has complexity $O(n(N+E))$. Hence, the number of edges does not affect the query time of the RMBR-based solution. In contrast, the Graph traversal-based solution is affected because this approach needs to trace the graph via edges. The experimental results show that the RMBR-based solution has about 99% faster query execution time than the Graph traversal-based solution.

Effect of the ratio of user and place nodes. The variation of the ratio between user and place nodes used is 25:75, 50:50, and 75:25, with the total number of nodes is 5000. Thus, the scenarios in this experiment are defined as 1250 users and 3750 places, 2500 users and 2500 places, and 3750 users and 1250 places. The experimental result in Fig. 16 demonstrates that the query execution time of the Graph traversal-based solution decreases as the number of place nodes decreases. It appears because the Graph traversal-based solution uses place nodes inside the query area as the starting points for traversing the graph. Decreasing the number of place nodes can make the mapping process faster. In contrast, the query execution time of the RMBR-based solution increases as the number of place nodes decreases and the number of user nodes increases. It happens because the RMBR-based solution maps potential attackers by checking the RMBR areas of all users. A large number of users will make the mapping process even longer.

From this experiment, we know that the performance of the RMBR-based solution is less effective in a graph with a large number of user nodes. Contrarily, the performance of the Graph traversal-based solution is less effective in a graph with a large number of place nodes. The RMBR-based solution has a 99% faster query execution time than the Graph traversal-based solution.

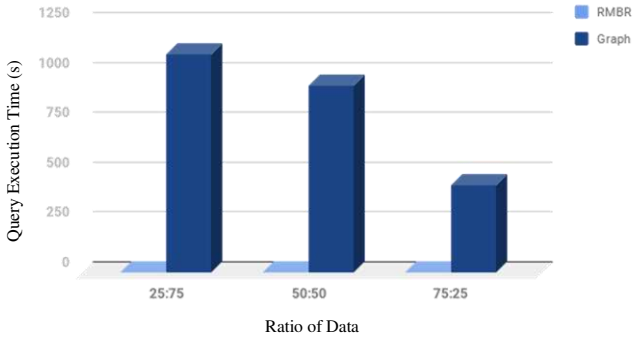

Fig. 16. Effect of the data ratio on query execution time.

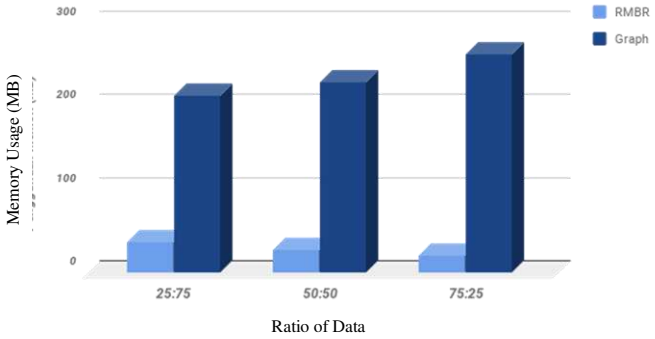

Fig. 17. Effect of the data ratio on memory usage.

B. *Memory Usage*

In this section, we study the effect of three parameters on memory usage, i.e., the number of nodes, the number of edges, and the ratio of user and place nodes.

Effect of the number of nodes. The experimental result in Fig. 13 depicts the memory usage of two proposed solutions that increases as the number of nodes increases. Still, the memory usage of the RMBR-based solution is about 90% less than the Graph traversal-based solution. It appears because the Graph traversal-based solution needs to traversing each place nodes inside the query area to maps the potential attackers and calculate the proximity between each user and place node. A large number of nodes significantly affects the memory usage of this approach. As a replacement, this solution provides additional information about the probabilities that users can be potential attackers. On the contrary, the RMBR-based solution only needs to check the RMBR areas of all user nodes to maps a list of potential attackers without calculating the proximities between each user and place node. Hence, this solution does not require significant memory usage.

Effect of the number of edges. Same as the effect of the number of edges on query execution time, Fig. 15 shows that the memory usage of the RMBR -based solution is reasonably stable, while the memory usage of the Graph traversal-based solution is unstable as the number of edges increases. The RMBR-based solution has about 89% less memory usage than the Graph traversal-based solution.

Effect of the ratio of user and place nodes. Contrary to the effect on query execution time, Fig. 17 shows that the memory usage of the Graph traversal-based solution increases as the number of place nodes decreases and the number of user nodes increases. It happens because the large number of user nodes affects the memory consumption to store potential attackers. In contrast, the memory usage of the RMBR-based solution decreases as the number of place nodes decreases and the number of user nodes increases. The RMBR area of each user node is updated based on the RMBR value of place nodes. Hence, the decreasing of place nodes can slightly reduce memory usage. The RMBR-based solution has 88% less memory usage than the Graph traversal-based solution.

V. Conclusion

This paper proposes two kinds of approaches to map potential attackers against network security using location-aware reachability queries on geosocial data, i.e., RMBR-based and Graph traversal-based solutions. The RMBR-based solution maps potential attackers by finding the intersection of the Reachability Minimum Bounding Rectangle of users on the given query location. While the Graph traversal-based solution map potential attackers by finding a list of places included in the given query location and tracing each place node using graph traversal algorithms, i.e., Depth-First Search, to find the users related to that place. The experimental results demonstrate that the RMBR-based solution is more effective than the Graph traversal-based solution. Utilizing the Reachability Minimum Bounding Rectangle (RMBR) concept can decrease the execution time to 99% and the memory usage to 89%. However, the RMBR-based solution only finds a list of users mapped to be potential attackers without calculating their probability of becoming attackers. In contrast, the Graph traversal-based solution calculates the proximity of each user and the query location. Hence, it provides information about the possibility that a user could be a potential attacker. We can combine the effectiveness of the RMBR-based approach and a proximity calculation method as a future improvement of this work.

REFERENCES

[1] M. Sarwat and Y. Sun, "Answering location-Aware graph reachability queries on geosocial data," *Proc. - Int. Conf. Data Eng.*, 2017, pp. 207–210.
[2] N. Armenatzoglou, S. Papadopoulos, and D. Papadias, "A general framework for geoSocial query processing," *Proc. VLDB Endow.*, vol. 6, no. 10, 2013, pp. 913–924.
[3] D. Wu, J. Shi, and N. Mamoulis, "Density-Based Place Clustering Using Geo-Social Network Data," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 5, pp. 838–851, 2018.
[4] R. R. Veloso, L. Cerf, W. Meira, and M. J. Zaki, "Reachability queries in very large graphs: A fast refined online search approach," *Adv. Database Technol. - EDBT 2014 17th Int. Conf. Extending Database Technol. Proc.*, vol. 1, no. c, pp. 511–522, 2014.
[5] M. Du, A. Yang, J. Zhou, X. Tang, Z. Chen, and Y. Zuo, "HT: A Novel Labeling Scheme for k-Hop Reachability Queries on DAGs," *IEEE Access*, vol. 7, pp. 172110–172122, 2019.
[6] Y. Duan, X. Li, and L. Ding, "A reachability query method based on labeling index on large-scale graphs," *Proc. - 2015 Int. Conf. Comput. Sci. Comput. Intell. CSCI 2015*, 2016, pp. 77–82.
[7] R. Liang, H. Zhuge, X. Jiang, Q. Zeng, and X. He, "Scaling hop-based reachability indexing for fast graph pattern query processing," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 11, pp. 2803–2817, 2014.
[8] J. Su, Q. Zhu, H. Wei, and J. X. Yu, "Reachability Querying: Can It Be even Faster?," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 3, pp. 683–697, 2017.
[9] S. Zhou, P. Yuan, L. Liu, and H. Jin, "Mgtag: A multi-dimensional graph labeling scheme for fast reachability queries," *Proc. - IEEE 34th Int. Conf. Data Eng. ICDE*, 2018, pp. 1376–1379.
[10] H. Wei, J. X. Yu, C. Lu, and R. Jin, "Reachability querying: an independent permutation labeling approach," *VLDB J.*, vol. 27, no. 1, pp. 1191–1202, 2014.
[11] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms, Second Edition*. 2001.
[12] H. Yildirim and M. J. Zaki, "Graph indexing for reachability queries," *Proc. - Int. Conf. Data Eng.*, 2010, pp. 321–324.
[13] J. Wood, "Minimum Bounding Rectangle," in *Encyclopedia of GIS*, S. Shekhar and H. Xiong, Eds. Boston, MA: Springer US, 2008, pp. 660–661.