Plausible Mobility Inference from Wireless Contacts Using Optimizationo

Alirea k.Monfared Mostafa Ammar Ellen Zegura

***Abstract*—Recording locations of mobile nodes and connectivity between them is essential in the study of various types of wireless networks, including but not limited to opportunistic networks. Working with mobility information gives an extra degree of freedom in the study of such networks, while in a contact log, a lot of information like the locations of the nodes and their velocities is irretrievably lost. On the other hand, contact information can be measured more easily compared to detailed locations of the nodes. In this paper, we present a solution for inferring mobility from contact information. This technique uses a simple non-linear optimization approach to suppress the problem as a feasibility problem with fundamental constraints on the mobility of the nodes. In this paper, we introduce our technique and its underlying assumptions, and discuss applicability of more than one input to enhance the performance of inferred mobility.**

***Index Terms*—mobility trace, contact trace, plausible mobility**

I. Introduction

STUDIES of mobile wireless protocols benefit from real-world traces that measure and record node locations over time. These *mobility traces* can be combined with radio models to produce network connectivity for simulation studies. Measuring node locations with fine-grained time and space granularity is challenging, however. Outdoor positioning systems are sensitive to weather and obstacles, while indoor positioning systems generally rely on availability of WiFi or other signal infrastructure that may not be available.

As an alternative, *contact traces* measure and record pair-wise node connectivity over time and do not require accurate positioning systems. Because they come from radio transmissions, contact traces are able to capture environmental effects on communication capability, such as obstacle effects, that are missing from mobility traces. Contact traces can record on-off timing information about links, hence they can be more compact than mobility traces that must record data at every time instant.

Contacts are typically recorded by using Bluetooth or WLAN in an infra-structureless mode. All other devices in range are seen as a contact [**?**]. Collecting contact traces is challenging, however, as it requires instrumenting nodes with collection capability and conducting a well-planned and coordinated collection "run".

Alireza K.Monfared is with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, e-mail: alireza@gatech.edu.

Mostafa Ammar is with the School of Computer Science, Georgia Institute of Technology, Atlanta, GA, e-mail:ammar@gatech.edu.

Ellen Zegura is with the School of Computer Science, Georgia Institute of Technology, Atlanta, GA, e-mail: ewz@cc.gatech.edu.

While recording contact opportunities, a lot of information is irretrievably lost. No information on exact location an velocity of the nodes can be retrieved from a contact trace. As time elapses, the amount of this information decreases and it becomes more and more difficult to even relatively locate the nodes. Such traces are only representative of the particular context in which they are measured. For example, a WiFi radio contact trace seems not to be appropriate to study a scenario possessing the same characteristics in every aspect, except a Bluetooth radio without running the entire trace collection experiment again which carries a significant organizational overhead in preparing, distributing and collecting contact loggers, not to mention non-technical issues related to legality and participant consent. In a similar fashion, every small change in the contact scenario necessitates repetition of the whole collection process.

Contact trace collection and analysis have been reported in the literature and have been used extensively to study mobile networks for various settings ranging from a campus scenario [**?**][**?**] to a conference [**?**] to urban scenarios [**?**] or a disaster area. Aschenbruck et al.[**?**] has summarized these types of datasets.

We are interested in the problem of inferring a plausible mobility trace from a contact trace. By "plausible", we mean a trace that is consistent with the contacts, and not originating from a given radio model. In general, there may be multiple mobility traces that are consistent with a given trace. We do not attempt to enumerate them. Although this plausible mobility might not match the original locations, it matches to a mobility trace that could have generated the same contact trace, or a trace very similar to it. It can be discussed that the exact locations of the nodes is not required in most the experiments that deal with studying mobility or modifying contacts , unless the researcher has access to a "valid" mobility trace presenting the same characteristics.

Coming from the contact domain to the mobility domain, may provide researchers with opportunities not present with the contacts alone, such as transforming the settings of the original data collection, one that assumes a different radio, for example.

Our proposed algorithm is based on optimizing a mobility trace using the limited information from the contacts. It operates by solving a feasibility problem for a small set of fundamental constraints on mobility of the nodes. We will describe the details of the algorithm in this report and we have made the code publicly available [CITATION NEEDED]

Contributions of this work are summarized as follows:

- An algorithm to infer mobility information based on

contact traces that enables richer understanding of mobile networks.
- Application of multiple inputs for the proposed algorithm to enhance re-usability of the output trace.

The rest of this paper is organized as follows [TO BE COMPLETED]

II. Background

In this section, we define the basic concepts of mobility and contacts as they have been used in our proposed algorithm, and provide the background knowledge required for understanding the context and technical details of our mobility inversion algorithm. We start this section by a few definitions.

Contact Trace: A contact trace is a log of contact-start and contact-end events, each labeled with a time stamp and with the identifiers of the two nodes involved in the contact event. Mathematically, contact traces can be represented using a sequence of graphs showing the connections over time, which are called *evolving graphs* [**?**]. We are interested in contact traces collected in the course of an experiment. In such experiments, mobile devices equipped with a wireless interface are attached to moving entities (e.g., people or vehicles). These mobile devices sample their environments periodically and log the presence of neighboring nodes. At the conclusion of the experiment these logs are collected and a contact trace is constructed from them.

Contact-start events occur when one of the devices receives information from the other and contact-end events occur when transmission stops. Many aspects of the radio environment can affect these events. In general, a contact trace becomes meaningful when studied along with a *radio model.* A radio model specifies the circumstances under which two nodes are defined to be in contact. In the simplest model, two nodes are defined to be in contact whenever their distance is less than a given transmission range, and disconnected otherwise. This circular transmission range model is considered very simplistic due to many reasons, including the existence of obstacles, lack of symmetry in real contacts, inequality of signal strength inside the assumed transmission circle, random effects like fading, shadowing, etc. [**?**]. A more complex model can combine these effects, in a so-called *pathloss model*, that can be used to determine the signal strength at the receiver end which can be fed to a *decider* accepting signals above a known threshold. These information can lead to a more realistic contact model. Despite these, the circular transmission range model has been extensively used due to its simplicity and the fact that it allows to focus on other, perhaps more significant factors in the experiment.

Mobility Trace*:* A mobility trace is a log of locations of nodes over time. Similar to a contact trace, a mobility trace can also be asynchronized or synchronized, synthetically modeled or originating from real data collected using GPS equipped devices or association to the base stations of a 3G/WiFi network.

The mobility inversion problem combines a radio model, and a contact trace to obtain a consistent mobility trace showing their locations using no further information[1].

Few solutions have been proposed for this specific problem. Wang et al.[**?**] solves the same problem by dividing it into smaller sub-problems, and relying on decreasing number of possibilities. Our analysis of such approach has proven that the complexity of such solution grows exponentially with increasing number of nodes, and hence is only applicable to small scenarios. Ristanovic et al.[**?**] suggests a solution that relies on knowledge of some anchor node with fixed locations which can give a rough estimate of location of mobile Bluetooth devices that have a low range. The immediate downside of this approach is its dependence on the location of devices and the underlying technologies used.

Whitbeck et al. [**?**] has proposed a complete algorithm for the mobility inversion problem inspired by the work in dynamic graph drawing. They define three fundamental constraints on the mobility of the nodes as follows:

$$||x_i(t+\Delta t) - x_i(t)||_2^2 \leq v_{max}\Delta t$$

$$r - 2v_{max}min\{t - p_{ij}^{\uparrow}, n_{ij}^{\downarrow} - t\} \leq d_{ij}(t) \leq r$$

$$r \leq d_{ij}(t) \leq r + 2v_{max}min\{t - p_{ij}^{\downarrow}, n_{ik}^{\uparrow} - t\}$$

Where $x_i(t)$ and $d_{ij}(t)$ are the location of node i and distance between nodes i and j at time t respectively, $p_{ij}^{\uparrow}$and $n_{ij}^{\uparrow}$are the previous and the next time that node i and j will be in contact , $p_{ij}^{\downarrow}$ and $n_{ij}^{\downarrow}$are the previous and the next time that node i and j will be lose their contact, all relative to the current time t, and r denotes the transmission range of the nodes. Note that these definitions rely on the assumption of circular transmission range model.

The first constraint states that no node can move faster than a maximum speed v_{max} and the other two confine the distance between two nodes based on their previous and future contact opportunities. The authors also define, an attractive force, similar to the classical "spring" force on each node from all of its contacts, as well as a repulsive force, similar to the classical "Coulomb" force from all other present nodes, and a "drag" force to prevent excessive speeds that may make nodes go far away from each other when borders are not obvious. The equilibrium of these forces on each node is then calculated and used to determines its location. Together, this gives a relative set of locations satisfying the given contacts. A major disadvantage of this approach is the necessity of a smart choice of many constants involved in the calculation of the forces. Since there is no rule of thumb for many of these parameters, the performance of the application becomes dependent of the specific scenario under study, and it may become a series of trial and errors when such information is not available. For a more detailed analysis of this drawback, refer to Section III-B.

[1]These concepts will be more formally defined in the next section

III. Mobility Inversion Algorithm

In this section, we define the problem and its solution in a mathematical framework. Let N be the number of nodes that reside in a field of size $L \times L$ [2]. We assume each node has a location in d dimensions, with $d = 2$ in our simulations. A contact trace in this setting, corresponds to an $N \times N \times T_m$ matrix of adjacencies of its corresponding evolving graph [?] where T_m is the number of time steps. Note that this representation assumes that we have access to snapshots of connectivity structure of nodes at some discrete times $t = t_o, ..., T_m$. A mobility trace is a $d \times N \times T_m$ matrix, where its element $(:, i, t)$[3] is the location of node i at time t in both dimensions. Table I describes the parameters used in the above description.

Parameter Name	Description
N	Number of nodes
R_i	Transmission Range of i^{th} Input
v_m	Maximum Speed
T_m	Trace Length in Time
d	Dimensions of the problem (2 in our case)
L	Size of the Simulation Field (square) Side

Table I
Summary of basic parameters used in the algorithm

Our algorithm does not use any information on initial locations or assumption about the mobility model of the nodes. The only assumptions made are the following:

- At any moment in time, two nodes x_i and x_j are in contact once their distance, $||x_i^t - x_j^t||_2^2$ is less than a given radio range, R, and are out of contact if this distance is larger than R.
- Nodes move slower than a given maximum speed v_m, i.e. , the distance between the location of a node i at time t and $t-1$, $||x_i^t - x_i^{t-1}||_2^2$ is smaller than the maximum possible distance than be traveled, namely $v_m \times \Delta t$, where Δt is the difference between snapshots.
- Nodes are always confined inside the mobility field.

We will describe the technical details of the algorithm and its variants in the next section.

A. Single Input Mobility Inversion

In this section, we describe the algorithm for a single input contact trace, and then we generalize this to an arbitrary number of inputs in section **??**. This front end is depicted in figure 1, where we use the notation $G = (V, E)$ for the evolving graph corresponding to the input contact, which is a time-series of graphs $G^t = (V^t, E^t)$. In our notation, we associate graph G, with matrix C with the notation $G \Longleftrightarrow C$. This means that graphs G^t have adjacency matrices $C(:, :, t)$, or in our notation $G^t \Longleftrightarrow C(:, :, t)$.

Our algorithm starts by finding an initial set of locations for nodes, consistent with the initial state of the contact trace, namely $G^1 = C(:, :, t_0)$. After this initial step, we solve a feasibility problem to find a set of locations for each of the snapshots in the contact trace that is consistent with the three fundamental constraints mentioned in section III. We the location of nodes at the previous step as the initial point for the problem, and solve this problem iteratively to find the location of the nodes at each of the snapshots. Together, these yield a mobility trace consistent with input contacts. This process is describe in Algorithm 1.

Algorithm 1 Basic mobility inversion algorithm with one input contact trace

```
Input: Contact Trace $C_{N \times N \times T_m}$, Transmission range $R$, Maximum speed $v_m$, number of dimensions $d = 2$
$X_{final} \leftarrow \mathbf{0}_{d \times N \times T_m}$
Find a set of initial locations $X_0$.
$X_{final}(:, :, 0) \leftarrow X_0$
for $t := t_0$ to $T_m$ do
    solve the nonlinear optimization problem in figure 1 with $G^t \Longleftrightarrow C(:, :, t)$ , $X_0$, $R$, $v_m$
    $X_{final}(:, :, t) \leftarrow X$
    $X_0 \leftarrow X$
end for
Output: $X_{final}$
```

Figure 1. Optimization algorithm takes a contact traces along with its corresponding radio range and a value of maximum speed to infer a mobility in an iterative manner. Inferred locations for each time step are given as an initial point of the solver for the next step.

To complete our description of Algorithm1, we need to clarify how we find the initial value of X_0 and how we solve the nonlinear optimization of figure 1 for each snapshot.

For the latter, we have used the Limited-memory Broyden–Fletcher–Goldfarb–Shanno (l-BFGS) algorithm [?]. The l-BFGS is a variant of the Broyden–Fletcher–Goldfarb–Shann (BFGS), an approximation of Newton's method that seeks the stationary point of a function by finding the zero of its derivative. The advantage of the BFGS to its alternative methods is that it does not need to evaluate the Hessian matrix of the second derivatives. Instead, the Hessian is approximated using rank-one updates of gradient evaluations. The basic operation of the BFGS is described in Algorithm 2.

In Algorithm 1, we have exploited Interior Point OPTimizer, pronounced eye-pea-Opt (Ipopt) [?] as the underlying solver, to perform the iterative task in the first line of the for loop. Ipopt is an open source software package for large-scale nonlinear optimization, implemented in C++ which uses the Harwell Subroutine Library (HSL) [?] as a third-party solver.

To address the issue of choosing a suitable initial point, for the first iteration, we use the same method as above, except that only constraints corresponding to ranges, and residence of nodes inside the simulation field will be considered and the maximum speed constraint, which necessitates a reasonable set of "previous" locations, will be ignored. Since l-BFGS still needs an initial point to seek the stationary point of the

[2]The filed can be defined as an $L_1 \times L_2$rectangle. This assumption is made for simplifying our descriptions. If the field has an irregular shape, the size of its smallest bounding rectangle should be used.

[3]Throughout this document, we use the notation of colons,(..., :, ...), to represent all possible numbers that can come in a specific vector location

Algorithm 2 Basic BFGS algorithm.

```
Input: Objective function $f(X)$, approximate initial Hessian $B_0$(can be identity matrix), maximum number of iterations $max - iter$
for    $k$ :=1 to $max - iter$ do
       Solve      for direction,$p_k$ : $B_k p_k = \Delta f(X_k)$
       Perform    line search with acceptable step size $\alpha_k$ to perform the update: $X_{k+1} = X_k + \alpha_k p_k$
       Set        : $s_k := X_k + \alpha_k p_k$
       $Y_{k+1} = \Delta f(X_{k+1}) - \Delta f(X_k)$
       $B_{k+1} = B_k + \frac{Y_k Y_k^T}{Y_k^T s_k} - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k}$
end    for
Output: $Y_k$
```

function, we construct a distance matrix by finding the shortest path among all node pairs in terms number of hops using the initial connectivity structure, $G^1 \iff C(:,:,t_0)$. Next, we multiply this hop-count matrix by the transmission range R, i.e. we assume two nodes m hops from each other are at a distance $m \times R$, and replace the entries with a value of infinity, i.e unreachable nodeparis, with a reasonably large distance. Finally, we apply the classical Multidimensional Scaling (MDS) [**?**] to yield a decent initial point. MDS achieves this by double-centering the given distance matrix, i.e. subtracting row and column means from it and adding back the grand mean, and then performing a singular value decomposition (SVD) on the double-centered distance matrix. Using basic definition of Euclidean distance, it can be shown that if SVD is of the form $D = UVU'$, which is possible due to the symmetry of the distance matrix, then $X = UV^{1/2}$. To attain the desired dimension of 2 for the locations, we define X as $X = U_2 V_2^{1/2}$,where $U_2$and V_2 contain only the first two largest eigenvalues of D, and their corresponding eigenvectors. We finally perform a proper translation to map the center of mass of X to the center of the simulation field. This value of X provided by MDS can be then passed to the version of the optimization problem that ignores the speed constraint to find the first value of X_0.This is also used as initial location of nodes as well, i.e. $X_{final}(:,:,0)$.

Implementation of Algorithm 1 is performed in MATLAB interfacing Ipopt, as described above. The code is publicly available for interested readers [CITATION NEEDED]

B. Discussion

In this section, we briefly review the main advantages of our mobility inversion algorithm, compared to the force based algorithm of Whitbeck et al.[**?**]. We will also point out the drawbacks of the algorithm. We will discuss the performance metrics in the next section.

Simplicity of the front-end is the first advantage of our algorithm. The force-based algorithm not only relies on the input contact trace, but also many parameters that need to be set. These include the rigidity constant K and the damping factor τ in the attractive force, the intensity constant G , cutoff distance d_{max},and parameters α and ϵ_0 that affect the intensity of the repulsive force with distance, as well as the strength parameter D of the drag force. There is no general rule for setting these parameters, and for the algorithm to perform well, these need to be set in a trial and error fashion. On the other hand, the optimization algorithm does not need any external parameters other than a maximum speed and transmission range, which are natural to consider for every scenario and are present in the force-based algorithm nevertheless. Detailed parameters for the solver are chosen internally and are the same for every instance of the problem.

Optimization algorithm can take one or more contact traces, their corresponding radio ranges, and an upper bound on speed of movements as input and infer a mobility trace that can generate these contacts. This has not been foreseen in the force-based algorithm of Whitbeck[**?**], but can be possibly incorporated with some modifications.

A drawback, present in both algorithms is the radio model. It has been discussed in the literature, many other factors including but not limited to presence of obstacles, fading, path-loss, etc affect the radio model [**?**]. Despite this, the optimization mobility inversion algorithm is designed in a modular manner so that any radio model can be integrated into it provided that the an well-behave penalty, possibly in the form of a non-linear inequality, corresponding the connectivity status of the nodes is suggested[4]. Since the solver at the heart of our algorithm, BFGS, can solve any nonlinear optimization problem, a new contact model that can be described mathematically, and can be derived from a contact trace, possibly with additional information, can replace the circular model, without affecting the other modules. We have addressed this drawback in our future work with minimal modifications to problem definition which does not affect the solver.

IV. Performance Evaluation

In this section, we evaluate the performance of the mobility inversion algorithm proposed.

A. Metrics

Before presenting the evaluations, we need to define a performance metric to measure how "good" is the performance of the algorithm on a trace. To do this, in the very general framework ,we need to derive measures of comparison from the inputs and the outputs of the algorithm. Such comparison can be done in three levels as follows.

Mobility-level Comparison: The lowest level of evaluation is done with mobility to mobility comparison. Assuming that we have access to the mobility trace from which the input contact trace originated, which is only viable for test purposes, we can compare the original input mobility trace (original trace) and the output mobility trace (inferred trace) for each location. To do this, we find the distance between each nodepair at each time step for both traces. Then we plot a two-dimensioanl histogram of these distances. In the ideal case,

[4]In the circular model, for example, this penalty is that the distance between two connected nodes is smaller than a transmission range, and the distance between two disconnected nodes is larger than this range.

one would expect that the histogram would show values on its diagonal and zeros everywhere else, i.e., the distances between each nodepair at each time step is the same for the original and inferred traces. We do not expect to see this behavior in our algorithm though. The reason is that constraints in Algorithm 1, are all of the form that pair-wise distance is either larger or smaller than a given transmission range. This does not give any ability to the algorithm to predict the exact distances, unless the problem has a unique solution, which is not the case.

Contact-level Comparison: In the next level, we compare the contact traces. While the input to our algorithm is a contact trace (original contact trace), we can infer an output contact trace (inferred contact trace) with the proper transmission range that the algorithm has used at the output and a proper radio model We can then compare the two corresponding evolving graphs over time based on the number of missing links, i.e. present in the input contact trace and missing in the inferred contact trace, extra links, i.e. only seen in the inferred contact trace, and the sum of these two numbers. While a low number of link errors in this level is very desired to be seen and is sufficient to have a successful mobility inference, it is not necessary for our purposes. These errors can be defined as a function of time as follows:

$$E_{missing\ link}(t) = \frac{\sum_{i,j}(C_R(i,j,t) \wedge (\neg C'_R(i,j,t)))}{\sum_{i,j}(C_R(i,j,t) \wedge \mathbf{1})} \times 100$$

$$E_{extra\ link}(t) = \frac{\sum_{i,j}((\neg C_R(i,j,t)) \wedge (C'_R(i,j,t))}{\sum_{i,j}(C_R(i,j,t) \wedge \mathbf{1})} \times 100$$

$$E_{total\ link}(t) = \frac{\sum_{i,j}(C_R(i,j,t) \oplus C'_R(i,j,t))}{\sum_{i,j}(C_R(i,j,t) \wedge \mathbf{1})} \times 100$$

Where C_R and C'_R are contact traces corresponding evolving graph of ground truth, aka the input to the algorithm, and derived contact which comes from the inferred mobility, and $\mathbf{1}$ is a matrix of all ones of the size $N \times N \times T_m$.$\neg$, $\oplus$ and $\wedge$ represent element-wise logical NOT, XOR and AND respectively. Note that for such comparison to be reasonable, we need both contact traces to correspond to the same transmission range R.

Connectivity Characteristics Comparison: In the third level of comparisons, we use the definitions Bui et al.[?] to compare journeys, i.e. space-time paths between node pairs over time. To do this, we compute the transition time of journeys among each node pair at each time step for both original and inferred contact traces , i.e., the time duration of a journey starting from the first node at a specific time instant and destined for the second. Similar to the mobility-level comparisons, we we plot a two-dimensioanl histogram of these journeys. It is expected that the journeys in the original and inferred contact traces show correlation.

Simulation: The last method of comparison that we use involves running a real simulation on original contact trace and the inferred trace. We perform our simulations using the Opportunistic Network Environment simulator (the ONE) [?]. In these experiments, we send random packets with a specific time to live (TTL) from a randomly chosen node destined for another randomly chosen node at every time step. Then, we compare the cumulative number of messages that reach their destinations before the TTL to total number of messages (packet delivery ratio). For similar settings of such simulation, we expect to observe a similar behavior for the packet delivery ratio over time for both traces.

B. Simulation Results

In our experiments, we use two mobility models:

- Random Waypoint (RWP)[?], which is often used for simulation purposes in mobility literature despite its known limitations [?]. In this model speed, direction, and destination of each mobile node is chosen randomly and independently of other nodes.
- Reference Point Group Mobility (RPGM) [?] which organizes mobile hosts into groups according to their logical relationships with each group having a logical center known as reference center whose movement is followed by all nodes in the group. This model allows independent random motion behavior for each node, in addition to the group motion

The experiments setting consists of 50 nodes and a duration of 1000 time steps, spanning over a $1000 \times 1000\ m^2$ field, with a maximum speed of $10m/s$.

To get better results, especially in the case of contact-level comparisons, we make a slight modification to the iterative part of Algorithm 1 depicted in Figure 1 as follows. If two nodes are connected, their distance should be smaller than $R(1-\epsilon_{in})$ and if their are not connected, their distance should be larger than $R(1+\epsilon_{out})$. Note that this does not violate any of the assumptions in the algorithm. While connected nodes are still nearer than R and disconnected nodes are still farther than R, this modification allows to remedy for mistakes that the optimizer can make. In all the simulations, we have set $\epsilon_{in} = 0.2$ and $\epsilon_{out} = 1$. Fore more detailed discussion of effect of these choices for ϵ_{in} and ϵ_{out} on the results, see section REFER.

Figure 2. Pairwise distance histogram for RWP model

Figure 3. Pairwise distance histogram for RPGM model

Figures 2 and 3 show the histogram for pairwise distances. As it has been previously stated, we do not expect to see a perfect correlation between the original and inferred distances. Instead, due to the splitting effect of the transmission range constraints, we generally expect to see large distances versus large distances and small distances versus small distances. It is also worthy to note that many of the distances in the original traces are mapped into a distance of $2R$ in the inferred traces. The reason for this behavior lies in the fact that we have set $\epsilon_{out} = 2$ in our simulation, this means that every disconnected nodepair is placed such that their distance is larger than $R(1+\epsilon_{out}) = 2R$, which is mapped to a number very near to $2R$

by the solver. This also describes why we avoided to use R itself in the range constraints.

Figure 4. Contact-level errors for RWP over time. Extra links are missing in the input, but present in the contacts inferred from the output mobility. Missing links are present in the input, but missing in the contacts inferred from the output mobility. Total link errors is the sum of these.

Figure 5. Contact-level errors for RPGM over time. Extra links are missing in the input, but present in the contacts inferred from the output mobility. Missing links are present in the input, but missing in the contacts inferred from the output mobility. Total link errors is the sum of these.

Figures 4, 5, and show the contact-level errors. Note that in all our simulations, most of the error comes from the missing links which are present in the original contact trace but absent in the inferred contact trace. Part of this is due to the fact that a value of $\epsilon_{out} = 1$ gives a safety margin of $2R - R = R$ to the solver for extra links error while a value of $\epsilon_{in} = 0.5$ gives it a safety margin of $R - 0.5R = 0.5R$ for missing link errors. Nevertheless, in all our simulations the total number of link errors is less than 100 out of the 1225 possible links, which is a reasonably good performance compared to the existing algorithm of Whitbeck et. al [**?**], and given the fact that the optimization algorithm has not been given any location information.

Figure 6. Journey duration histogram for RWP.

Figure 7. Journey duration histogram for RPGM.

Figures 6, 7, and show the histogram for journey durations. As the figures suggest, a strong correlation is observed for RPGM.

Figures 8, 9, and show the packet delivery ratio versus time for the three mobility models. As stated earlier, all simulations are performed using the ONE simulator for Delay Tolerant Networks. We send a 1KB packet from a randomly chosen node to another randomly chosen at every second. The value of TTL for messages are set to 3 minutes, while the traces are 1000 seconds long. We have used the Probabilistic ROuting Protocol using History of Encounters and Transitivity (PROPHET) [**?**] as the routing protocol for the simulator.

- Self-similar Least Action Walk (SLAW) [**?**] which is a fairly new model that tries to capture statistical patterns of human mobility including truncated power-law distribution of flights, pause-times and inter-contact times, and heterogeneously defined areas of individual mobility. One of the main features of this model is that it captures social contexts.

V. Conclusion

We have presented an algorithm to infer mobility from contacts. Our algorithm accepts a contact trace that indicated the connections among nodes over time. Without any extra

Figure 8. Journey duration histogram for RWP.

Figure 9. Journey duration histogram for RPGM.

information about the locations of these nodes, we infer a set of locations, aka a mobility trace, that could have generated these contacts. Through performance evaluations, we have shown that this inferred mobility shows comparable connectivity characteristics compared to the original locations from which the input contact trace has come. This opens up the opportunity to use these locations in any experiment, and avoid the expensive process of collecting location information. It also gives rise to an ongoing research to use these locations to infer contact traces for other scenarios, which results in transformation of a contact trace.

Acknowledgment

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. 1117190.

Appendix

In this appendix, we survey other techniques to infer the locations of wireless nodes, given various amount of information.

A closely related problem to our mobility inversion, localization has been a well-studied problem in various disciplines of networking. In wireless sensor networks, the problem also called the "graph realization" problem, is generally stated as follows:

Given a graph $G = (V, E)$ such that V is the set of n vertices (nodes), and D is an $n \times n$ matrix of pairwise distances between nodes, find a $d-dimensional$ embedding that preserves D.

Note that in our problem, we deal with an evolving graph $G = \{G^t(V^t, E^t) \quad \forall t\}$, which is a series of graphs whose information are correlated. Also, we do not assume access to the distance matrix, D, in our algorithm.

Next, we will briefly review three widely used proposed solutions for graph realization problem, which is proved to be NP-Hard in the general case[**?**].

Multidimensional Scaling (MDS) is a statistical technique used for exploring the similarities or dissimilarities in data. To find an embedding $X = [x_1 ... x_m]^T$[5]of n vertices in $\mathbb{R}$, and given a known distance matrix D, multidimensional scaling, seeks to minimize a stress function as follows:

$$\min_{x_1 \dots x_m} \sum_{i<j} \quad (||x_i - x_j|| - d_{ij})^2$$

Classical MDS requires distances between every pair of nodes. Shang et al. [**?**] have proposed to use shortest paths computed from a connectivity matrix instead of the distance matrix in their MDS-MAP[6]. MDS yields a relative map of

[5]$[]^T$ represents transpose operation on a vector.

[6]Connectivity matrix is defined similar to an adjacency matrix. For a graph $G = (V, E)$ the $n \times n$ connectivity matrix A is defined such that $A_{ij}$is 1 whenever nodes i and j are within transmission range of each other and 0 otherwise.

locations of the nodes, and given d anchor nodes (nodes with known locations), this can be turned into an absolute map with the target embedding, X. MDS has the disadvantage of pulling nodes towards the center of the area, and thus performs poorly on irregularly-shaped graphs. To overcome this problem, Shang et al. [**?**], have produced a version of the algorithm, MDS-MAP(p) in which MDS is applied for the subgraph of a given number of hops, p, around each node to produce a local map for that node, and then these local maps are merged to produce a global relative map.

Spectral Graph Drawing (SGD) is another solution proposed for the graph realization problem. SGD also defines a minimization problem based on the Laplacian matrix, L. This matrix is defined as follows:

$$L = K - W$$

$$W = [w_{ij}] = \frac{1}{1+d_{ij}} \quad or \quad e^{-d_{ij}}$$

$$K = diag(k_{ii}) = \sum_{j \epsilon N(i)} w_{ij}$$

W is a weighting matrix, inversely proportional to the pairwise distances D, and K is the diagonal matrix of node degrees. SGD solves the following optimization problem to find X:

$$X^* = \underset{X}{argmin}[X^T L X]$$

$$s.t. \quad X^T X = 1$$

$$X^T \mathbf{1}_n = 0$$

The first constraint in the above equation ensures that the trivial solution of all nodes in the same place is not chosen in favor of the others. The selection of variance, $X^T X$, to be equal to 1 in this constraint is arbitrary. The second constraint, ensures the invariance of the solution under transitions by setting a mean of zero. Koren [**?**] argues that like classical MDS, SGD tends to draw the nodes to the center and hence has a poor performance on irregularly-shaped graphs. A variant of SGD, called degree normalized spectral graph drawing (DN-SGD), updates the second constraint of original SGD with weights proportional to degrees, i.e. $X^T K X = 1$. SGD and DN-SGD have been shown to be solvable by finding the generalized eigenvalues of the matrices L and K. Broxten et al. [**?**] have shown that for DN-SGD to perform reasonably on static sensor network localization, a two-step localization, followed by a post-processing to filter outliers is required.

Finally, the localization problem has been formulated as a Semidefinite program (SDP), a specific type of convex optimization which can be solved by interior methods. An SDP feasibility is as follows:

$$find\ X \in \mathbb{R}^{(n+2)\times(n+2)}$$

$$\begin{aligned} s.t.&(1;0;\mathbf{0})^T X (1;0;\mathbf{0}) = 1 \\ &(0;1;\mathbf{0})^T X (0;1;\mathbf{0}) = 1 \\ &(1;1;\mathbf{0})^T X (1;1;\mathbf{0}) = 2 \\ &(0;e_i - e_j)^T X (0;e_i - e_j) = d_{ji}^2 \ \forall (j,i) \in V_x \\ &(a_k - e_j)^T X (a_k - e_j) = d_{jk}^2 \ \forall (j,k) \in V_a \\ &X \succcurlyeq 0 \end{aligned}$$

$V_x \subseteq V$ is the set of nodes with unknown locations, and $V_a \subseteq V$ is the set of nodes with known locations (anchors). and e_i is a vector of all zero entries except i^{th}-entry, and $a_k \quad \forall k = 1...m$ is the locations of the anchor nodes. The last constraint requires X to be a positive semidefinite matrix[7].Biswas et al. [**?**], [**?**] have discussed details of using semidefinite programming for solving the localization problem.

[7]A matrix, M, is positive semidefinite, if $z^T M z \geq 0$ for all non-zero column vectors z