ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

TIK Institut für Technische Informatik und Kommunikationsnetze

Distributed Optimization in DTNs: Towards Understanding Greedy and Stochastic Algorithms

Andreea Picu, Thrasyvoulos Spyropoulos
TIK Report No. 326
July 2010

Abstract

Opportunistic or *Delay Tolerant Networks* is a new networking paradigm that is envisioned to complement existing wireless technologies (cellular, WiFi) by exploiting a "niche" performance-cost tradeoff. Wireless peers communicate when in contact, forming a network "on the fly" whose connectivity graph is highly dynamic and only partially connected. This gives rise to a number of challenging optimization problems, such as end-to-end routing, resource allocation, content placement etc. While globally optimal solutions are normally sought in network optimization, node actions and decisions in this context are inherently local. As a result, most solutions proposed rely on local heuristics without any guarantees about their convergence properties towards a desired global outcome. In this paper, we look deeper into the problem of Distributed Optimization in the context of Opportunistic Networks. We propose an analytical framework and use it to study deterministic (Greedy) and stochastic utility ascent (Markov Chain Monte Carlo) algorithms. Using this framework, we prove necessary and sufficient conditions for their correctness, and derive closed form solutions for their performance (convergence probability and delay), in generic mobility scenarios. We use real and synthetic mobility traces to validate our findings, and examine the impact of mobility properties in more depth.

1 Introduction

Communication over multiple wireless mobile hops often results in connectivity disruptions lasting from seconds (urban scenarios) to minutes or days (extreme environments). *Opportunistic Networks* (or *Delay Tolerant Networks* – DTNs) are envisioned to supplement existing wireless infrastructure-based services (e.g., offload cellular data traffic), and enable novel (e.g., social and location-based) applications. Nodes harness unused bandwidth by exchanging data whenever they are in proximity (*in contact*), with the goal to forward data (probabilistically closer) towards a (set of) destination(s). By introducing redundancy (e.g., coding [1] or replication [2, 3]) and intelligent mobility prediction algorithms, data of interest can be delivered or retrieved over a sequence of such contacts, despite the lack of end-to-end paths.

Finding, moving, and storing data over a *disconnected* and *highly dynamic* connectivity graph poses significant challenges. This has attracted a lot of researchers to study unicast and multicast routing problems [4], resource allocation problems [5], content placement and distributed caching [6], etc. These can often be formulated as an optimization problem over the connectivity graph. However, the volatile nature of this graph, the long delays to learn the state of nodes not currently in contact, and the cost of flooding state data over the network, imply that: (i) *centralized solutions are impractical*, and (ii) *distributed solutions requiring global knowledge are costly and suboptimal, as they may rely on obsolete data*.

As a result, the majority of algorithms proposed for these problems are intuitive heuristics, or greedy algorithms that deterministically choose the "best" available action locally [7, 8, 9]. No guarantee of convergence to the (globally) optimal configuration is usually given, resorting instead to simulations to evaluate performance. In optimization theory, it is well known that such algorithms, performing a gradient ascent over the solution space, converge if the problem is *convex*, but can easily fail in the presence of local maxima [10]. In the latter case, stochastic techniques like simulated annealing and evolutionary algorithms are often used.

In the context of Opportunistic Networking, the correctness and efficiency of such greedy, local algorithms is not straightforward to assess. *First, optimization problems in DTNs are by nature distributed*; the state of different nodes may change independently and without mutual knowledge. *Second*, in traditional optimization problems, the traversal of the solution space is under the control of the algorithm designer. Gradient-based algorithms usually define a local neighborhood of solutions and move to the best one among them. In contrast, *the solution space traversal in optimization problems over DTNs is dictated by node mobility*. *Third*, in traditional problems, local maxima are states whose utility is higher than any other state in its local neighborhood. In the context of DTN optimization, the "local" neighborhood of every state constantly changes over time; as a result *local maxima may be transient*.

This paper is an attempt to better understand the nature of distributed optimization algorithms in DTNs and to provide

an analytical framework to assess their performance. Our main contributions can be summarized as follows:

- We propose a Markov chain model that combines the mobility properties of a scenario and the actions of an algorithm into an appropriate transition matrix over a problem's solution space (Section 2).
- We propose and study two generic optimization algorithms, a deterministic utility ascent (greedy) one, and a stochastic utility ascent algorithm based on the Markov Chain Monte Carlo framework (Section 3).
- We prove necessary and sufficient conditions for the *correctness* of each algorithm, and use an extensive set of real and synthetic mobility traces to examine whether these conditions are met in practice (Section 3).
- We propose an analytical framework, based on transient analysis over the Markov chain corresponding to an algorithm, to derive closed form results for the convergence probability and convergence delay of each algorithm in a given mobility scenario (Section 4).

We believe that these initial results will shed some more light on the interplay between DTN algorithms and the complex mobility patterns observed in realistic scenarios, and will facilitate the design of better DTN algorithms.

2 Preliminaries

In this section, we describe some popular classes of distributed optimization problems in DTNs. We propose a Markov chain interpretation of such problems, and show how these classes can be mapped to the framework. Finally, we present the datasets of real and synthetic mobility scenarios that we will use in parallel with our analytical framework to examine the properties of different algorithms and to validate our analytical derivations.

2.1 Markov Chain Interpretation

Let $\mathcal{N}$ be the set of all nodes in our Opportunistic Network, $|\mathcal{N}| = N$. Each of the N nodes is identified by a unique ID. Node mobility is assumed to be driven (implicitly or explicitly) by social relations to other nodes as well as location preference (e.g., home locations and hotspots).

Each node chooses its state in $\mathcal{S}$ ($\mathcal{S}$ may be a subset of objects to store, or the choice to offer a service or not, etc.). A solution to many optimization problems in DTNs consists of configuring every node to an appropriate state. Hence, the solution space is $\Omega = \{\mathcal{S}\} \times \{\mathcal{S}\} \times \cdots = \{\mathcal{S}\}^N$. To describe transitions over Ω, we propose the use of a time-homogeneous discrete-time Markov chain $(\mathbf{X}_n)_{n \in \mathbb{N}_0}$, where $\mathbf{X}_n$ is an N-element vector, with each element taking values in $\mathcal{S}$.

Let us consider two potential configurations $\mathbf{x}, \mathbf{y} \in \Omega$, with $\mathbf{x} = (x_1, x_2, \ldots, x_N)$ and $\mathbf{y} = (y_1, y_2, \ldots, y_N)$. The values $x_i, y_i \in \mathcal{S}$ are the states of node i in configuration $\mathbf{x}$ and respectively $\mathbf{y}$. For example, $x_i = 1$ if node i stores an object in configuration $\mathbf{x}$ and $y_i = 0$ if the node no longer stores that object in configuration $\mathbf{y}$ (in this case $\mathcal{S} = \{0, 1\}$).

Definitions: *(i) Difference between two configurations*. First, let the indicator variable $\mathbf{I}_{\mathbf{xy};i} = \mathbb{1}\{x_i \neq y_j\}$. The variable is 1, when the state of node i is ***not*** the same in both configurations and 0 otherwise. Then, we define the distance between configurations $\mathbf{x}$ and $\mathbf{y}$: $d(\mathbf{x}, \mathbf{y}) = \sum_{1 \leqslant i \leqslant N} \mathbf{I}_{\mathbf{xy};i}$.

(ii) Contact Probability Matrix. Changes in the network configuration usually require a contact (e.g., a node drops a message to receive another one, or gives an object for storage at another node). As a result, transitions occur only between configurations $\mathbf{x}$, $\mathbf{y}$ differing in at most two node states i, j (i.e., the nodes in contact): $d(\mathbf{x}, \mathbf{y}) \leqslant 2$. Let p^c_{ij} be the probability that the next contact is between nodes i and j. These probabilities define a *Contact Probability Matrix* $\mathbf{P}^c$.

(ii) Acceptance Probability Matrix. When nodes i and j meet, the optimization algorithm may change the state of none, one, or both nodes (e.g., in routing: *none* – no message exchange, *one* – node j receives a message, or *both* – node j receives a message and node i deletes its copy of that message). All the other nodes keep their state. We denote the possible change $\mathbf{x} \to \mathbf{y}$, where $x_i \neq y_i$ or $x_j \neq y_j$ or both, and $x_k = y_k$, $\forall\ k \neq i, j$. We define the *acceptance probability* $A_{\mathbf{xy}}$ or A_{ij}[1] as the probability that the optimization algorithm performs a change. This is usually determined by some utility function. For example, for a greedy algorithm $A_{\mathbf{xy}} = 0$ if configuration $\mathbf{y}$ is worse than $\mathbf{x}$ and $A_{\mathbf{xy}} = 1$ otherwise. For the algorithm to be fully distributed, the utility function must be decomposable, such that it can be evaluated locally by each node (e.g., degree, power, storage, for which $U(\mathbf{x}) = \sum_{1 \leqslant i \leqslant N} U(x_i)$).

Given these definitions we can now write the transition probabilities for our Markov chain $\mathbf{X}_n$ as:

$$p_{\mathbf{xy}} = \mathbb{P}\left[\mathbf{X}_{n+1} = \mathbf{y} \,|\, \mathbf{X}_n = \mathbf{x}\right] = \begin{cases} 0, & d(\mathbf{x}, \mathbf{y}) > 2 \\ p^c_{ij} \cdot A_{\mathbf{xy}}, & 1 \leq d(\mathbf{x}, \mathbf{y}) \leq 2 \\ 1 - \sum\limits_{\substack{1 \leqslant \mathbf{z} \leqslant |\Omega| \\ \mathbf{z} \neq \mathbf{x}}} p_{\mathbf{xz}}, & d(\mathbf{x}, \mathbf{y}) = 0 \text{ or } \mathbf{x} = \mathbf{y}. \end{cases} \quad (1)$$

where, i and j are the two nodes in which states $\mathbf{x}$ and $\mathbf{y}$ may differ. p^c_{ij} is the ***mobility component*** of the transition probability and $A_{\mathbf{xy}}$ is the ***algorithm component*** of the transition probability.

We now look at how specific classes of problems could be mapped into this framework.

2.1.1 Content Placement or Distributed Caching

The problem could be formally defined as follows: *given a finite budget of replicas L for some content, find an optimal*

[1] Since state transitions in the Markov chain occur solely upon contact between two nodes, any transition related quantities ($p_{\mathbf{xy}}$, $A_{\mathbf{xy}}$ etc) can be more simply indexed by the two nodes meeting (p_{ij}, A_{ij} etc). We favor this simpler notation.

subset of nodes $\mathcal{L}^\star \subset \mathcal{N}$ who will store the content, so as to minimize some accessibility cost. This may be a popular video, for example, that is expected to be heavily requested in the future. For this problem:

- The node state space is $\mathcal{S} = \{0, 1\}$, 1 = store replica.
- Ω consists of the $\binom{N}{L}$ possible choices of L relays out of the total N nodes.
- Let $\mathbf{x}$ be the current configuration. With probability p_{ij}^c, the transition $\mathbf{x} \to \mathbf{y}$ *may* occur, if $x_i = 1$, $x_j = 0$ and $y_i = 1$, $y_j = 0$:

$$\begin{array}{cccccccc} & & & i & & j & & \\ \mathbf{x} = (x_1, & \ldots, & 1, & \ldots, & 0, & \ldots, & x_N) \\ \mathbf{y} = (y_1, & \ldots, & 0, & \ldots, & 1, & \ldots, & y_N) \end{array}$$

- $A_{\mathbf{xy}}$ for this algorithm decides whether the transition *does* indeed occur.

We note here that the formulation could be applied to relay selection problems for *optimal multicast* [11, 12], network coverage problems, etc.

2.1.2 Resource allocation:

Let us assume that there are K possible objects in the network, yet each node can store only $B < K$ of them. This description corresponds, for example, to the problem of PodNet channels assignment [13, 14]. It is also relevant to the DTN buffer management problem [15, 5]. While this problem is slightly more complicated, it can still be mapped to the above framework.

The state space for each node is larger, consisting of $\binom{K}{B}$ possible states. This makes the problem solution space explode to $\left[\binom{K}{B}\right]^N$ states. Moreover, when nodes i and j meet, there are multiple ways to re-arrange their respective states (as opposed to the binary choice faced before). The union of the contents in the two nodes' buffers may contain up to $2B$ unique objects. Out of these, there are up to $\binom{2B}{B} \times \binom{2B}{B}$ ways to re-assign the objects to i and j (i and j can decide to store the same object(s)). Various algorithms can be defined. For example, an algorithm may pick a random element from j's buffer and replace one of its own [14], or pick the configuration most improving the utility (among the locally available ones) [15].

2.2 Datasets

In order to cover a broad range of mobility scenarios, we use five real contact traces and two synthetic mobility models with their characteristics summarized in Table 1. The real contact traces we use are the following: (i) the MIT *Reality Mining* trace (MIT) [16], (ii) the iMotes Infocom 2005 trace (INFO) [17], (iii) the ETH trace (ETH) [18], (iv) a trace from an outdoor training scenario in the Swiss military (ARMA) and (v) the San Francisco taxis trace (SFTAXI).

The two mobility models we use are the following:

(i) TVCM: In the Time-variant Community Model [19], each node is randomly assigned one or more home location areas ("communities") on the plane. Nodes perform random waypoint trips inside and outside home locations, with a probability $1 - p$ of roaming outside the community (in the next trip) and a probability p of staying or getting back in the home location. By choosing different travelling probabilities p for each node, a large range of heterogeneous node behaviors can be reproduced.
We use four versions of a simple TVCM scenario throughout our analysis. Our scenario has two types of nodes: "normal" nodes moving either exclusively or mostly in their home locations, and more "gregarious" nodes covering the entire simulation area. In the four versions we vary the number of "normal" nodes and communities, the size of the simulation area and the value of p (1 – nodes are confined to their home locations, communities are *disjoint* and can only communicate through bridges; 0.7 – nodes occasionally travel outside their home locations). In all four versions there are 4 "gregarious" nodes and the transmission range is 30 m. We denote these four versions of our TVCM scenario:
  - TVCM24: 24 nodes in 200 × 200 m, 2 communities (12 and 8 nodes), with $p = 0.7$;
  - TVCM104: 104 nodes in 300 × 300 m, 4 communities (50, 2 × 20 and 10 nodes), with $p = 0.7$;
  - TVCM24**d**: 24 nodes in 200 × 200 m, 2 communities (12 and 8 nodes), with $p = 1$ ("d" for disjoint communities);
  - TVCM104**d** 104 nodes in 300 × 300 m, 4 communities (50, 2 × 20 and 10 nodes), with $p = 1$ ("d" for disjoint communities), see Figure 1.

The TVCM104**d** scenario is illustrated in Figure 1.

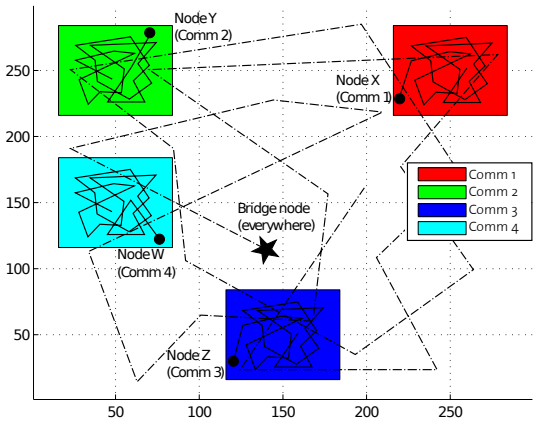


Figure 1: *TVCM104**d** Mobility Scenario*

(ii) HCMM: The Home-cell Community-based Mobility Model (HCMM) [20] is an extension of the Community-based Mobility Model (CMM) [21]. CMM was the first mobility model directly driven by a social network. The Caveman model [22] is used to define a network with social communities and each community is assigned to a home location. In contrast to TVCM,

transition probabilities are directly linked to the weights on the overlay social network. HCMM adds location-driven mobility to CMM. The travelling probability to a location α no longer depends on nodes currently at that location but on the total weight of nodes assigned to α as their home location (i.e., irrespective of their current position). We use this model to create two synthetic scenarios:

- HCMM20: 20 nodes in 200 × 200 m, 2 communities;
- HCMM104: 100 nodes in 300 × 300 m, 10 communities;

3 Distributed Optimization Algorithms

In this section, we discuss two distributed algorithms that can traverse the solution space defined earlier, in search of the optimal solution: a *greedy* and a *stochastic* utility-ascent algorithm. Our goal is to prove necessary and sufficient conditions for the correctness of the algorithms (i.e., guaranteed discovery of optimal solution). For illustration, we focus here on the simpler content placement problem. However, the results presented in this section (Thms. 2 and 3) and in the next one, apply to other problems as well, e.g., instances of the resource allocation problem. We defer the detailed treatment of this class of problems to future work.

Before moving to the core of the subject, we consider a toy algorithm, *Direct Placement*. This is a direct extension of the Spray and Wait routing protocol [3], and will be useful to introduce some first aspects of the optimization problem in hand, and to motivate utility ascent schemes.

3.1 Toy Algorithm: Direct Placement

Suppose $\mathcal{L}^\star$ is known, i.e., we know the IDs of the L highest utility nodes. A simple algorithm would be for the source node to distribute one of L copies to the first nodes it encounters, e.g., using binary spraying [3]. Then the nodes holding a copy of the content only forward it directly to nodes in $\mathcal{L}^\star$. Translated into the framework in § 2, the acceptance probability (algorithm component) is of the form:

$$A_{\mathbf{xy}} = A_{ij} = \mathbb{1}\{i \in \mathcal{N} \setminus \mathcal{L}^\star, j \in \mathcal{L}^\star\},$$

where $x_i = 1$, $x_j = 0$ (i has a copy, j does not) and $y_i = 0$, $y_j = 1$ (the copy moved to j, if j had high utility).

Theorem 1 states that the L copies of the content will eventually be absorbed by the nodes in $\mathcal{L}^\star$ if and only if there is a non-zero contact probability between any node in $\mathcal{L}^\star$ and any other node in the rest of the network. For $L = 1$, this means that every node must have a direct tie to the highest utility node (e.g., highest degree node).

Definition 1 (Visit probability)**.** *The random variable* $T_{\mathbf{xy}} := \min\{n \geqslant 1 | \mathbf{X}_n = \mathbf{y},\ \mathbf{X}_0 = \mathbf{x}\}$ *counts the number of steps needed by the Markov chain to get from configuration* $\mathbf{x}$ *to configuration* $\mathbf{y}$*.* $T_{\mathbf{xy}}$ *is called* hitting time *from state* $\mathbf{x}$ *to state* $\mathbf{y}$*. If* $\mathbf{y}$ *is never reached, we set* $T_{\mathbf{xy}} = \infty$*. The probability to get from state* $\mathbf{x}$ *to state* $\mathbf{y}$ *after arbitrarily many steps is called* visit probability $v_{\mathbf{xy}}$*:*

$$v_{\mathbf{xy}} := \mathbb{P}[T_{\mathbf{xy}} < \infty]. \tag{2}$$

Theorem 1. *For all source nodes* i *and all initial copy allocations* $\mathcal{L}$*,* Direct Placement *is correct if and only if* $p^c_{ij} > 0$ *for all* $i \in \mathcal{N} \setminus \mathcal{L}^\star$ *and all* $j \in \mathcal{L}^\star$*.*

Proof. ($L = 1$ *case)* Let us first prove Theorem 1 for the simple case of $L = 1$, when the solution space Ω corresponds to the set of nodes $\mathcal{N}$, and the Markov Chain $\mathbf{P}$ is one-dimensional. In this case, a potential solution $\mathbf{x} \in \Omega$ of the optimization problem is still an N-element vector, but this vector has a single non-zero element $\mathbf{x}(i)$, where i is the node holding the one existing copy of content. Given this fact, we can use the more intuitive notation $\mathbf{x} = i$.

Denote by l the single node in $\mathcal{L}^\star$. l is an absorbing state in the chain, and we are interested in v_{il} for all $i \neq l$. We will use *first step analysis* [23], Eq. (1) and the fact that $A_{ik} = 0$, for any $k \neq l$.

$$\begin{aligned} v_{il} &= \mathbb{P}[T_{il} < \infty] = \sum_{1 \leqslant k \leqslant N} \mathbb{P}[T_{il} < \infty | X_1 = k] \cdot p_{ik} \\ &= p_{il} + v_{il} \cdot p_{ii} + \sum_{\substack{1 \leqslant k \leqslant N \\ k \neq l,i}} v_{kl} \cdot p^c_{ik} \cdot A_{ik} \\ &= p_{il} + v_{il} \cdot p_{ii} \qquad (3) \\ &= p^c_{il} \cdot A_{il} + v_{il} \cdot \left(1 - \sum_{\substack{1 \leqslant k \leqslant N \\ k \neq i}} p^c_{ik} \cdot A_{ik}\right) \\ &= p^c_{il} + v_{il} \cdot (1 - p^c_{il}) \\ &\Rightarrow v_{il} \cdot p^c_{il} = p^c_{il}. \qquad (4) \end{aligned}$$

"Sufficient": Eq. (4) implies that if $p^c_{il} > 0$, then $v_{il} = 1$ and hence *Direct Placement* is correct.

"Necessary": Let us assume that *Direct Placement* is correct, that is, $v_{il} = 1$. From Eq. (3), we get $p_{il} + p_{ii} = 1$ for all $i \neq l$. Assuming $p^c_{il} = 0$, and hence $p_{il} = 0$, would mean that $p_{ii} = 1$. From this, we can infer that $\mathbb{P}[T_{ij} < \infty] = 0$ for all $j \neq i$, which contradicts our hypothesis that $v_{il} = 1$. Therefore, we must have $p^c_{il} > 0$ for all $i \neq l$.

($L > 1$ *case)* This can be seen as a system of L particles on the same chain. Since these particles are not interacting in the case of *Direct Placement*, we can treat this case as L *appropriate* one-dimensional chains (instead of an L-dimensional chain). For the sake of presentation, we treat the case of $L = 2$ (The proof is similar for $L > 2$). Denote by l_1 and l_2 the two nodes in $\mathcal{L}^\star$. We define two chains that are exactly as in the $L = 1$ case, with the only difference being that in the first chain l_2 and in the second chain l_1 are removed[2].

[2]Note that this is not a coupling, since the two chains are not identical. Furthermore, the choice of which chain should converge to which state of the two, l_1 and l_2, does not make any difference, because we need to prove convergence for *all possible initial configurations* of the 2 copies

	MIT	INFO	ETH	ARMA	SFTAXI	TVCM
Scale and context	92 campus students & staff	41 conference attendees	20 lab students & staff	152 people	536 taxis	24/104 nodes
Period	9 months	3 days	5 days	9 hours	1 month	11 days
Scanning Interval	300s (Bluetooth)	120s (Bluetooth)	0.5s (Ad Hoc WiFi)	30s (GPS)	30s (GPS)	N/A
# Contacts total	81 961	22 459	22 968	12 875	1 339 274	1 000 000

Table 1: *Mobility traces characteristics.*

In order for *Direct Placement* to work, we need that $v_{il_1} = 1$ for all $i \neq l_1, l_2$ for the first chain and $v_{il_2} = 1$ for all $i \neq l_1, l_2$, in the second one. Applying the proof for the case $L = 1$ to these two chains (the state missing in each, does not affect the proof), we obtain that $p^c_{il_1} > 0$ and $p^c_{il_2} > 0$, for all $i \neq l_1, l_2$, concluding the proof for $L = 2$. ■

3.2 Single-Hop Paths in Mobility Traces

Our toy algorithm, *Direct Placement*, is a simple optimization algorithm that has the advantage of making few transmissions ($\approx 2L$) to reach its goal. Theorem 1 states that the applicability of it heavily depends on the mixing properties of the nodes in the mobility scenario considered. We examine here whether and to what extent the conditions of Theorem 1 hold in practice.

For our evaluation, we consider a number of real mobility traces, introduced in Section 2.2. If there is a contact between i and j during the trace, then $p^c_{ij} > 0$, and the two nodes are reachable in the sense of Theorem 1. While it is not entirely unrealistic to expect e.g., the highest degree node in a network to be directly related (i.e., have at least one contact) to every other node in the network, it is unlikely. This is confirmed by Figure 2, which shows the percentage of node pairs that are reachable by a single hop over time.

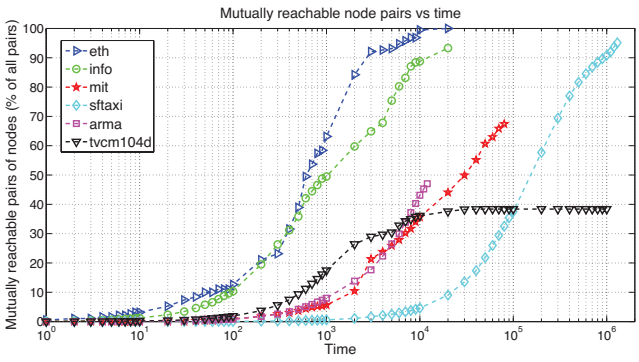


Figure 2: *Single hop paths in various traces*

In the ETH and Infocom traces, over 90% of the node pairs are directly reachable relatively early in the trace. For these traces, the *Direct Placement* algorithm has a chance of performing decently, regardless of the utility function used. The MIT and Armasuisse traces and the TVCM model, on the other hand, barely achieve 40 – 70% of directly reachable node pairs by the end of the trace.

In a nutshell, in realistic mobility settings, the mobility-related conditions required by the toy algorithm are often not fulfilled. What is more, knowledge of the IDs of the highest utility nodes is not available for many problems. We now move on to analyzing more flexible and practical algorithms.

3.3 Algorithm 1: Greedy Forwarding

Suppose now we no longer know the IDs of nodes in $\mathcal{L}^\star$. Without any other knowledge, a reasonable action is to locally try to increase the utility of the current configuration, at every opportunity presented, until the copies arrive at the highest utility nodes, $\mathcal{L}^\star$, or no further improvement is possible. This is a *greedy* or deterministic *gradient ascent* algorithm, whose performance depends on the utility function and the existence of local maxima.

Definition 2 (Greedy Forwarding)**.** *After an initial configuration is achieved (e.g., with spraying), each node with a copy hands it over to a node it encounters if and only if this increases the utility function. In the framework of Section 2, Greedy Forwarding corresponds to:*

$$A_{\mathbf{xy}} = \mathbb{1}_{\{U_\mathbf{x} < U_\mathbf{y}\}}. \tag{5}$$

The state corresponding to one copy at each node in $\mathcal{L}^\star$ is an absorbing state. However, there may be additional absorbing states (i.e., local maxima of U), in which case, the algorithm is not always "correct" (i.e., not guaranteed to converge to the optimal solution from every initial copy assignment).

An important question here is the following: *When can the solution space be efficiently navigated with the* Greedy Forwarding *algorithm? What are the properties of the mobility model or the utility function that make simple utility ascent algorithms applicable?* Theorem 2 derives necessary and sufficient conditions on the *contact matrix* $\mathbf{P}^c$ for an increasing utility path to exist between any node in $\mathcal{N} \setminus \mathcal{L}^\star$ and any node in $\mathcal{L}^\star$.

Theorem 2. *For all source nodes i and all initial copy allocations $\mathcal{L}$,* Greedy Forwarding *is correct if and only if for all $i \in \mathcal{N} \setminus \mathcal{L}^\star$ there exist at least L nodes $j_1, \ldots, j_L \in \mathcal{N}$ with $U_i < U_{j_{(.)}}$, such that $p^c_{ij_{(.)}} > 0$.*

Proof. We will again use the visit probability from Def. 1. We start with $L = 1$ and $\mathcal{L}^\star = \{l\}$. This allows us once more to use the more intuitive simplified notation $\mathbf{x} = i$, where $\mathbf{x}(i) = 1$ and $\mathbf{x}(j) = 0$ for all $j \neq i$.

"Sufficient": If $\forall\, i \neq l$, $\exists\, j \in \mathcal{N}$ with $U_i < U_j$, such that $p^c_{ij} > 0$, then we must prove that, for all $i \neq l$:

$$v_{il} = p^c_{il} \cdot A_{il} + v_{il} \cdot p_{ii} + \sum_{\substack{1 \leqslant k \leqslant N \\ k \neq l,i}} v_{kl} \cdot p^c_{ik} \cdot A_{ik} = 1. \quad (6)$$

Let us further develop Eq. (6), using Eq. (1) and Eq. (5):

$$\begin{aligned} v_{il} &= p^c_{il} + v_{il} \cdot \left(1 - \sum_{\substack{1 \leqslant k \leqslant N \\ k \neq i}} p^c_{ik} \cdot A_{ik}\right) + \sum_{\substack{k \neq l,i \\ U_k > U_i}} v_{kl} \cdot p^c_{ik} \\ &= p^c_{il} + v_{il} - v_{il} \cdot \sum_{\substack{k \neq i \\ U_k > U_i}} p^c_{ik} + \sum_{\substack{k \neq l,i \\ U_k > U_i}} v_{kl} \cdot p^c_{ik} \\ 0 &= p^c_{il} - v_{il} \cdot \sum_{\substack{k \neq i \\ U_k > U_i}} p^c_{ik} + \sum_{\substack{k \neq l,i \\ U_k > U_i}} v_{kl} \cdot p^c_{ik} \\ v_{il} &= \frac{p^c_{il} + \sum\limits_{\substack{k \neq l,i \\ U_k > U_i}} v_{kl} \cdot p^c_{ik}}{\sum\limits_{\substack{k \neq i \\ U_k > U_i}} p^c_{ik}} = \frac{\sum\limits_{\substack{k \neq i \\ U_k > U_i}} v_{kl} \cdot p^c_{ik}}{\sum\limits_{\substack{k \neq i \\ U_k > U_i}} p^c_{ik}}. \end{aligned} \quad (7)$$

From Eq. (7), we see that it would suffice that $v_{kl} = 1$ $(k \neq i)$, for $v_{il} = 1$ $(i \neq l)$ to be true. This is perfectly sensible since v_{kl} and v_{il} are almost the same. We already know $v_{ll} = 1$, since $A_{li} = 0$ for all $i \in \mathcal{N}$, hence we have to prove it for $k \neq i, l$. We will use *complete induction*.

Let us take k_1 such that $U_l > U_{k_1} > U_j$ for all $j \neq l, k_1$; in other words, k_1 is the node with the second highest utility in the network. Hence, for k_1, using our hypothesis that $p^c_{k_1 l} > 0$ Eq. (7) becomes:

$$v_{k_1 l} = \frac{v_{ll} \cdot p^c_{k_1 l}}{p^c_{k_1 l}} = 1. \quad (8)$$

Further, let us take $k_{\alpha-1}$ such that $U_l > U_{k_1} > \cdots > U_{k_{\alpha-1}} > U_j$ for all $j \neq l, k_1, \ldots, k_{\alpha-1}$; this means $k_{\alpha-1}$ is the node with the α-th highest utility in the network. Moreover, assume $v_{k_1 l} = \ldots = v_{k_{\alpha-1} l} = 1$. Then, we will prove that for k_α, the node with the $(\alpha+1)$-th highest utility, $v_{k_\alpha l} = 1$. From the hypothesis, for this node we have either $p^c_{k_\alpha l} > 0$ or there exists $1 \leqslant \beta < \alpha$ such that $p^c_{k_\alpha k_\beta} > 0$ or both. Then:

$$v_{k_\alpha l} = \frac{v_{ll} \cdot p^c_{k_\alpha l} + \sum\limits_{1 \leqslant \beta < \alpha} v_{k_\beta l} \cdot p^c_{k_\alpha k_\beta}}{p^c_{k_\alpha l} + \sum\limits_{1 \leqslant \beta < \alpha} p^c_{k_\alpha k_\beta}} = \frac{p^c_{k_\alpha l} + \sum\limits_{1 \leqslant \beta < \alpha} p^c_{k_\alpha k_\beta}}{p^c_{k_\alpha l} + \sum\limits_{1 \leqslant \beta < \alpha} p^c_{k_\alpha k_\beta}} = 1. \quad (9)$$

Therefore, we have proven that $v_{il} = 1$ for all $i \neq l$ and thus *Greedy Forwarding* is correct.

"Necessary": If *Greedy Forwarding* is correct, then $\forall\, i \neq l$, $\exists\, j \in \mathcal{N}$ with $U_i < U_j$, such that $p^c_{ij} > 0$. The assumption that *Greedy Forwarding* is correct means that $v_{il} = 1$. Using Eq. (6) and our hypothesis, $v_{il} = 1$, we obtain that $p^c_{il} + p_{ii} + \sum\limits_{U_i < U_j} p^c_{ij} = 1$. Assuming $p^c_{il} = 0$ and $p^c_{ij} = 0$, for all $j \in \mathcal{N}$ with $U_i < U_j$ would mean $p_{ii} = 1$, i.e., i is an absorbing state which implies that $v_{il} = 0$, a contradiction. Therefore, there must exist at least one $j \in \mathcal{N}$ with $U_i < U_j$ such that $p^c_{ij} > 0$. This ends the proof for $L = 1$.

($L > 1$ case) The proof for the case of $L > 1$ is more involved but similar in substance. For the sake of presentation we will argue for the case of $L = 2$. The proof is similar in substance for $L > 2$. Denote by l_1 and l_2 the two nodes in $\mathcal{L}^\star$. The state space of the Markov chain is L-dimensional, therefore it explodes with the increase of L. For the current case, $L = 2$, the state/solution space is $\Omega = \mathcal{N} \times \mathcal{N}$. Transitions still happen only at contacts, hence transitions are only along one dimension at a time. In the 2-dimensional case, these transitions are governed, as previously, by factors of the form $A_{ij(k)} = \mathbb{1}_{\{j \neq k, U_i < U_j\}}$, where i, j represent the transition dimension and k is the value of the other dimension.

"Sufficient": If $\forall\, i \neq l_1, l_2$, $\exists\, j_1, j_2 \in \mathcal{N}$ with $U_i < U_{j_1}$ and $U_i < U_{j_2}$, such that $p^c_{ij_1} > 0$ and $p^c_{ij_2} > 0$, then *Greedy Forwarding* is correct. We must prove here that, for all initial $i_1, i_2 \neq l_1, l_2$:

$$\begin{aligned} v_{(i_1 i_2)(l_1 l_2)} &= p_{i_1 l_1} \cdot p_{i_2 l_2} + p_{i_1 l_2} \cdot p_{i_2 l_1} + v_{(i_1 i_2)(l_1 l_2)} \cdot p_{i_1 i_1} \cdot p_{i_2 i_2} \\ &\quad + \sum_{\substack{1 \leqslant k_1, k_2 \leqslant N \\ k_1 \neq l_1, l_2, i_1 \\ k_2 \neq l_1, l_2, i_2}} v_{(k_1 k_2)(l_1 l_2)} \cdot p^c_{i_1 k_1} \cdot p^c_{i_2 k_2} \cdot A_{i_1 k_1(i_2)} \cdot A_{i_2 k_2(i_1)} \\ &= 1. \end{aligned} \quad (10)$$

An expansion of Eq. (10) gives the result presented in Eq. (11). Using Eq. (11) and a complete induction argument as above, it is easy to prove that *Greedy Forwarding* is correct.

"Necessary": If *Greedy Forwarding* is correct, then $\forall\, i \neq l_1, l_2$, $\exists\, j_1, j_2 \in \mathcal{N}$ with $U_i < U_{j_1}$ and $U_i < U_{j_2}$, such that $p^c_{ij_1} > 0$ and $p^c_{ij_2} > 0$. The assumption that *Greedy Forwarding* is correct means that $v_{(i_1 i_2)(l_1 l_2)} = 1$. Using Eq. (10) and our hypothesis, $v_{(i_1 i_2)(l_1 l_2)} = 1$, we obtain that $p^c_{i_1 l_1} \cdot p^c_{i_2 l_2} + p^c_{i_1 l_2} \cdot p^c_{i_2 l_1} + p_{i_1 i_1} \cdot p_{i_2 i_2} + \sum\limits_{U_{i_1} < U_{j_1}, U_{i_2} < U_{j_2}} p^c_{i_1 j_1} \cdot p^c_{i_2 j_2} = 1$.

Clearly, assuming $p^c_{i l_1} = 0$, $p^c_{i l_2} = 0$ and $p^c_{ij} = 0$, for all $j \in \mathcal{N}$ with $U_i < U_j$ would mean $p_{i_{(.)} i_{(.)}} = 1$, i.e., $i_{(.)}$ are absorbing states which implies that $v_{(i_1 i_2)(l_1 l_2)} = 0$, a contradiction. Therefore, there must exist at least one $j \in \mathcal{N}$ with $U_i < U_j$ such that $p^c_{ij} > 0$.

Further, let us assume that for at least two nodes $i \neq l_1, l_2$ there exists exactly one $j \in \mathcal{N}$ with $U_i < U_j$, such that $p^c_{ij} > 0$. Consider the case when for both nodes $j = l_1$, the highest utility node in the network. In Eq. (10), this would once again mean that $p_{i_{(.)} i_{(.)}} = 1$, a contradiction. In conclusion, to ensure that *Greedy Forwarding* is correct, i.e., $v_{(i_1 i_2)(l_1 l_2)} = 1$, we must *always* have at least one of the products $p^c_{i_1 j_1} \cdot p^c_{i_2 j_2} > 0$. This can be ensured only if $\forall\, i \neq l_1, l_2$, $\exists\, j_1, j_2 \in \mathcal{N}$ with $U_i < U_{j_1}$ and $U_i < U_{j_2}$, such that $p^c_{ij_1} > 0$ and $p^c_{ij_2} > 0$. ∎

Remark: Finally, it is important to note that this theorem can be generalized to other distributed optimization problems,

$$
\begin{aligned}
v_{(i_1 i_2)(l_1 l_2)} &= p^c_{i_1 l_1} \cdot p^c_{i_2 l_2} + p^c_{i_1 l_2} \cdot p^c_{i_2 l_1} + v_{(i_1 i_2)(l_1 l_2)} \cdot \left(1 - \sum_{\substack{1 \leqslant k_1, k_2 \leqslant N \\ k_1 \neq i_1, k_2 \neq i_2}} p^c_{i_1 k_1} \cdot A_{i_1 k_1 (i_2)} \cdot p^c_{i_2 k_2} \cdot A_{i_2 k_2 (i_1)} \right) + \sum_{\substack{1 \leqslant k_1, k_2 \leqslant N \\ k_1 \neq l_1, l_2, i_1 \\ k_2 \neq l_1, l_2, i_2 \\ U_{i_1} < U_{k_1}, U_{i_2} < U_{k_2}}} v_{(k_1 k_2)(l_1 l_2)} \cdot p^c_{i_1 k_1} \cdot p^c_{i_2 k_2} \\
&= p^c_{i_1 l_1} \cdot p^c_{i_2 l_2} + p^c_{i_1 l_2} \cdot p^c_{i_2 l_1} + v_{(i_1 i_2)(l_1 l_2)} - v_{(i_1 i_2)(l_1 l_2)} \cdot \sum_{\substack{1 \leqslant k_1, k_2 \leqslant N \\ k_1 \neq i_1, k_2 \neq i_2 \\ U_{i_1} < U_{k_1}, U_{i_2} < U_{k_2}}} p^c_{i_1 k_1} \cdot p^c_{i_2 k_2} + \sum_{\substack{1 \leqslant k_1, k_2 \leqslant N \\ k_1 \neq l_1, l_2, i_1 \\ k_2 \neq l_1, l_2, i_2 \\ U_{i_1} < U_{k_1}, U_{i_2} < U_{k_2}}} v_{(k_1 k_2)(l_1 l_2)} \cdot p^c_{i_1 k_1} \cdot p^c_{i_2 k_2} \\
0 &= p^c_{i_1 l_1} \cdot p^c_{i_2 l_2} + p^c_{i_1 l_2} \cdot p^c_{i_2 l_1} - v_{(i_1 i_2)(l_1 l_2)} \cdot \sum_{\substack{1 \leqslant k_1, k_2 \leqslant N \\ k_1 \neq i_1, k_2 \neq i_2 \\ U_{i_1} < U_{k_1}, U_{i_2} < U_{k_2}}} p^c_{i_1 k_1} \cdot p^c_{i_2 k_2} + \sum_{\substack{1 \leqslant k_1, k_2 \leqslant N \\ k_1 \neq l_1, l_2, i_1 \\ k_2 \neq l_1, l_2, i_2 \\ U_{i_1} < U_{k_1}, U_{i_2} < U_{k_2}}} v_{(k_1 k_2)(l_1 l_2)} \cdot p^c_{i_1 k_1} \cdot p^c_{i_2 k_2} \\
v_{(i_1 i_2)(l_1 l_2)} &= \frac{p^c_{i_1 l_1} \cdot p^c_{i_2 l_2} + p^c_{i_1 l_2} \cdot p^c_{i_2 l_1} + \sum_{\substack{1 \leqslant k_1, k_2 \leqslant N \\ k_1 \neq l_1, l_2, i_1 \\ k_2 \neq l_1, l_2, i_2 \\ U_{i_1} < U_{k_1}, U_{i_2} < U_{k_2}}} v_{(k_1 k_2)(l_1 l_2)} \cdot p^c_{i_1 k_1} \cdot p^c_{i_2 k_2}}{\sum_{\substack{1 \leqslant k_1, k_2 \leqslant N \\ k_1 \neq i_1, k_2 \neq i_2 \\ U_{i_1} < U_{k_1}, U_{i_2} < U_{k_2}}} p^c_{i_1 k_1} \cdot p^c_{i_2 k_2}} = \frac{\sum_{\substack{1 \leqslant k_1, k_2 \leqslant N \\ k_1 \neq i_1 \\ k_2 \neq i_2 \\ U_{i_1} < U_{k_1}, U_{i_2} < U_{k_2}}} v_{(k_1 k_2)(l_1 l_2)} \cdot p^c_{i_1 k_1} \cdot p^c_{i_2 k_2}}{\sum_{\substack{1 \leqslant k_1, k_2 \leqslant N \\ k_1 \neq i_1, k_2 \neq i_2 \\ U_{i_1} < U_{k_1}, U_{i_2} < U_{k_2}}} p^c_{i_1 k_1} \cdot p^c_{i_2 k_2}}. \qquad (11)
\end{aligned}
$$

such as the resource allocation problem defined in Section 2. If one considers the (multi-dimensional) Markov Chain **P** defined in Section 2 for the resource allocation problem, then Theorem 2 simply states that greedy algorithm works, if and only if there are no absorbing states in **P**. Nevertheless, the state space of **P** quickly explodes even for modestly complex resource allocation problems, making the task of calculating **P** challenging. It is also less obvious to infer how mobility patterns and utility functions impact the algorithm.

Summarizing, given a distributed optimization problem (mapped into a utility function), and a mobility scenario (captured in a contact probability matrix), Theorem 2 converts the (often difficult) task of deciding whether a simple greedy algorithm would suffice, into the (often simpler) task of checking the rows of a matrix (the contact matrix) for enough non-zero entries.

3.4 Utility Ascent Paths in Mobility Traces

The conditions of Theorem 2 appear less strict than those of Theorem 1. Yet, in addition to the actual mobility scenario and its contact pattern, it is not difficult to see that these conditions (and the correctness or lack thereof) could be affected by the following factors: (a) the value of L, (b) the choice of utility function. We return to our datasets, introduced in Section 2.2, to examine whether and when these conditions hold in practice.

Figure 3 shows the percentage of node pairs that are reachable by a utility ascent path as a function of TTL. While for the whole trace duration, over 90% of the node pairs are greedily reachable, for smaller TTLs, many node pairs do not have paths for the Greedy algorithm to use. This is due either the nodes being unreachable or to existence of absorbing local maxima.

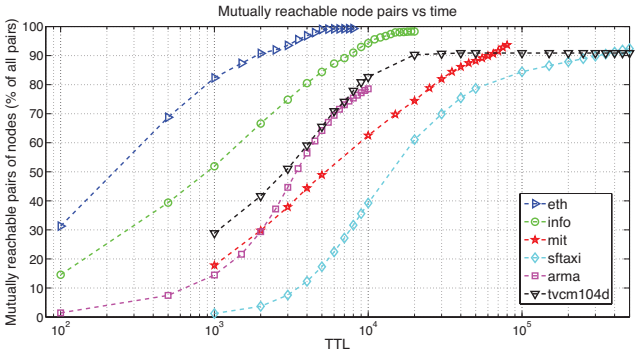
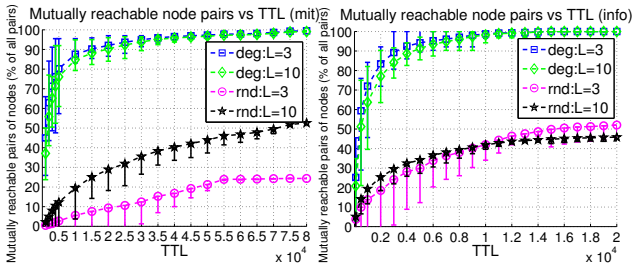


Figure 3: *Multi hop greedy paths in various traces*

In Figure 4, we investigate more closely the effect of L and the relation between the utility rank and the mobility rank. It provides a comparison among two relevant utilities: (i) the *degree centrality*, correlated to mobility [24, 12], and (ii) a random utility, which has little or no correlation to mobility.



(a) Utility ascent paths in MIT (b) Utility ascent paths in Infocom

Figure 4: *Comparison of several utilities*

Two observations ensue from Figure 4. First and foremost, there do not always exist utility ascent paths leading to op-

timal solutions. This means local maxima are present even for simple utility functions. This becomes more pronounced when L increases. Second, the correlation or lack thereof between the utility rank and the mobility rank considerably affects the navigability properties of the contact graph. This stresses the need to choose utility functions carefully.

Summarizing, while the *Greedy Forwarding* algorithm has less strict requirements than the *Direct Placement* algorithm, it is still relatively fragile in the face of parameters like TTL and the choice of utility function. To cope with these issues, stochastic optimization algorithms come in handy.

3.5 Algorithm 2: Markov Chain Monte Carlo

When the utility function is not known beforehand or there is inherent uncertainty about the network and application parameters involved, *stochastic* algorithms offer a more robust solution than deterministic utility ascent algorithms. Introducing randomization allows one to escape local extrema and explore additional configurations. Yet, randomization alone will not suffice, as this would essentially correspond to a *random walk* over the configuration space, and would not guarantee convergence.

In the context of Opportunistic Networking, the type of solution-space-traversal permissible by occurring contacts, as described in Section 2, can be naturally mapped to Markov Chain Monte Carlo (MCMC) methods [25]. While MCMC methods are often used to simulate and sample complex (and non-invertible) functions, they also provide a powerful tool for stochastic optimization. Unlike the greedy algorithm, they allow moves to lower utility states, but calibrate the probability of such moves so as to provably converge to an optimal solution [14, 26].

We will show how to modify the Greedy Forwarding using *Metropolis-Hastings sampling* [25]. This consists of building an *ergodic* Markov chain, whereof the states are feasible solutions of the optimization problem in hand. Once again, we use the formalism introduced in § 2: the transition probabilities of the Markov chain, $p_{\mathbf{xy}}$ still have two components, as in Eq. (1): a mobility component, p^c_{ij} and an algorithm component, $A_{\mathbf{xy}}$, with $p_{\mathbf{xy}} = p^c_{ij} \cdot A_{\mathbf{xy}}$.

The ***mobility component*** is the probability of "proposing" a new configuration in Metropolis-Hastings terms. This component cannot be changed regardless of the algorithm: it is the contact probability between two nodes. Therefore, we act on the ***algorithm component***, $A_{\mathbf{xy}}$, which is the acceptance probability. In the Greedy Forwarding scheme, this component was either 0 (for a lower utility proposed configuration) or 1 (for a higher utility proposed configuration). In contrast, in the MCMC Forwarding scheme, we keep the 1 for higher utility states, but replace the 0 with a well chosen probability. Then, the algorithm component is:

$$A_{\mathbf{xy}} = \min\left(1, \frac{\pi(\mathbf{y})}{\pi(\mathbf{x})}\right). \tag{12}$$

$\pi(\mathbf{x})$ is the *desired* stationary distribution of the Markov chain formed by the feasible solutions $\mathbf{x} \in \Omega$ of our system (for the *Content Placement* problem, all possible L-node subsets in the network). The Markov chain has a unique stationary distribution only if it is ergodic.

Moreover, for the chain to converge to the optimal solution, *the stationary distribution must be concentrated around that solution.* Hence, we must choose $\pi(\mathbf{x})$ in such a way that, with high probability, the Markov chain "walks" towards a solution with the highest utility. The Gibbs distribution has this property [25]:

$$\pi(\mathbf{x}) = \frac{\exp\left(\frac{U_{\mathbf{x}}}{T}\right)}{\sum\limits_{\mathbf{y}\in\Omega} \exp\left(\frac{U_{\mathbf{y}}}{T}\right)}, \tag{13}$$

where Ω is the space of all possible configurations and T is a system parameter, the "temperature". When T is small, the distribution is concentrated around the large values of $U_{\mathbf{x}}$ and thus, the algorithm will converge to a "good" solution with high probability. Luckily, the normalizing constant in the denominator of Eq. (13) cancels out resulting in the following acceptance probability (algorithm component):

$$A_{\mathbf{xy}} = \min\left(1, \frac{\exp\left(\frac{U_{\mathbf{y}}}{T}\right)}{\exp\left(\frac{U_{\mathbf{x}}}{T}\right)}\right) = \min\left(1, \exp\left(\frac{U_{\mathbf{y}} - U_{\mathbf{x}}}{T}\right)\right) \tag{14}$$

The MCMC-based optimization algorithm for the content placement can then be defined as follows:

Definition 3 (MCMC Forwarding)**.** *Let* $\mathbf{x}$ *denote the current configuration of node carriers, with node* i *holding a content copy:* $\mathbf{x}(i) = 1$*. Let* i *encounter a node* j *without a copy:* $\mathbf{x}(j) = 0$*. Then,* i *forwards a copy to* j *(or the chain goes from state* $\mathbf{x}$ *to state* $\mathbf{y}$*) with probability* $A_{\mathbf{xy}} = A_{ij} =$

$$\min\left(1, \exp\left(\frac{U_{\mathbf{x}} - U_{\mathbf{y}}}{T}\right)\right) \text{ or } \min\left(1, \exp\left(\frac{U_j - U_i}{T}\right)\right), \tag{15}$$

if U *is decomposable.*

The following theorem is a direct application of the Markov chain convergence theorem.

Theorem 3. *For all source nodes* i *and all initial copy allocations* $\mathcal{L}$*,* MCMC Forwarding *is correct if: (i) the Markov chain corresponding to the contact matrix,* $\mathbf{P}^c$*, is irreducible and aperiodic, (ii) the stationary distribution or this Markov chain is highly concentrated around the nodes in* $\mathcal{L}^\star$*.*

Proof. (i) According to Eq. (15) the acceptance probability for any new state is always larger than 0. Hence,

$$\exists\, n : \{\mathbf{P}^{(n)}\}_{ij} > 0 \Leftrightarrow \{(\mathbf{P}^c)^{(n)}\}_{ij} > 0.$$

In other words, the transition matrix $\mathbf{P}$ for the algorithm is irreducible and aperiodic, if and only if the contact matrix $\mathbf{P}^c$ is irreducible and aperiodic, which in this case, the Markov chain convergence theorem guarantees converge (see e.g., [25]). (ii) For $T \to 0$, the stationary distribution of the chain has almost all its mass on the configuration(s) maximizing the utility function. ■

The Role of the Temperature T: We make here a final note about temperature T. For very small T, the Markov chain will almost surely converge to the maximum $U_{\mathbf{x}}$. However, the convergence time increases. For larger values of T, the chain will escape local maxima easier but the stationary distribution is more flat; over time less optimal configurations will have a higher probability to appear in steady state. To overcome these difficulties, *simulated annealing* is often used, where the temperature starts with a relatively high T but gradually cools down. While theoretical results exist, showing that appropriate cooling schedules (e.g. logarithmic) are *guaranteed* to find a global optimum [27], the effect of cooling schedules is beyond the scope of this paper.

3.6 Space-Time Paths in Mobility Traces

Let us now verify whether the properties of Theorem 3 hold in practice. As for the previous algorithms, we will use the real mobility traces, introduced in Section 2.2. This time, the main requirement is that the contact graph of the network is irreducible. With an appropriate choice of acceptance probabilities, this implies that MCMC is always correct given sufficient time.

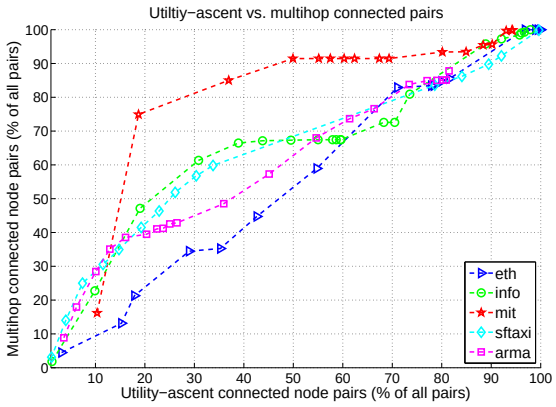


Figure 5: *Multihop vs. utility-ascent paths in various traces*

Figure 5 shows the percentage of node pairs that are reachable in multiple hops over time versus the percentage of node pairs reachable by greedy paths at the same point in time. In most traces, many more node pairs are reachable much earlier in the trace by simple multihop paths than by greedy paths. This means that real scenarios lend themselves well to the usage of MCMC algorithms for object placement or resource allocation.

4 Performance of Distributed Optimization Algorithms in DTNs

In the previous section, we proved necessary and sufficient conditions for each of the algorithms to reach the globally optimal configuration (Markov chain state) from *any* initial configuration. In addition to *worst case performance*, in practice we are also interested in the following performance metrics for the various algorithms:

(a) *The convergence probability* of the algorithm to the global optimum. Even if a state is *reachable*, this does not guarantee that it is *reached*, if local maxima exist.
(b) *The convergence time* of the algorithm. In the case of MCMC algorithms, while a path may exist, the probability of taking that path may be very small (due to high utility differences at hops along the path), resulting in high convergence delays.

Existing analytical models for DTNs, often used successfully to treat simple epidemic-based routing protocols, do not lend themselves easily to (more complex) algorithms where forwarding decisions are utility-based [28, 11]. What is worse, even a small departure from the assumption of independent identically distributed (IID) contact times complicates the problem significantly [29]. Yet, studies of real mobility settings (based on many of the traces we use, as well), reveal significant heterogeneity in pair-wise contact patterns. To address the problem of performance prediction for distributed optimization algorithms in DTNs, we take a different approach. Specifically, we will go back to the Markov chain defined in Section 2 and show that convergence probability and mean convergence time of each algorithm can be mapped to statistics of *absorption* or *first passage* quantities on the Markov chain. These statistics can often be derived in closed form after some matrix algebra.

4.1 Node Tie Strength and Contact Probabilities

To define the Markov chain $\mathbf{P}$ for each algorithm, we need $A_{\mathbf{xy}}$ and p_{ij}^c. We have shown how to obtain $A_{\mathbf{xy}}$ for each algorithm in Section 3. Here, we will show how to obtain the required contact probabilities p_{ij}^c, for a given mobility scenario[3].

In order to estimate p_{ij}^c values, we are interested in the strengh of the relationship ("tie") between nodes i and j. Different metrics such as the age of last contact [30, 7], contact frequency [31, 24] or aggregate contact duration [24] have been used as tie strength indicators in DTN routing protocols.

Here we use a metric based on the residual intercontact time of each node pair. Our Markov chains function in discrete time measured in number of contacts, since it is a contact that engenders each transition. Hence, the time between two transitions is a residual intercontact time. For a pair of nodes i and j, each contact occuring in the network is a *Bernoulli trial*: either the contact is between i and j (with probability p_{ij}^c), or not[4]. Therefore, the number of contacts until a node

[3]Note that the analysis and simulation results of Section 3 were only concerned with path *existence*, that is whether the probability of a single or multi-hop path was non-zero.

[4]We are assuming that contacts are mutually independent. While the veracity of this assumption may be disputable, our results show that it is sufficiently good for the study.

pair meets again is geometrically distributed with parameter p_{ij}^c.

We measure these parameters for each node pair, by sampling at exponentially distributed time intervals, using a Poisson process [32]. This guarantees that we obtain the actual average residual intercontact time for each pair. We apply the method of Maximum Likelihood estimation [33] for the geometric distribution to the above samples, to obtain the probabilities p_{ij}^c.

Summarizing, for a given mobility scenario, we will derive the matrix $\mathbf{P}^c$, as described above, use it to produce the algorithm's Markov chain $\mathbf{P}$, and then resort to transient analysis of this chain to derive useful statistics about the algorithm at hand.

4.2 Performance of Greedy Forwarding

We will first look at the performance of the *Utility Ascent (Greedy) Forwarding* algorithm described in Section 3.3. The Markov Chain $(\mathbf{X}_n)_{n\in\mathbb{N}_0}$ described by $\mathbf{P}$ corresponding to the solution space traversal for the *Greedy Forwarding* algorithm is an *absorbing Markov chain*. We therefore use the theory of absorbing Markov chains to characterize the algorithm's performance [34].

Denote by $\mathbf{x}^\star$ the globally optimal configuration corresponding to the maximum utility solution. By definition, this is an absorbing state in the Markov chain. Without loss of generality, let us assume that only one such configuration exists. In the general case, the solution space Ω for the greedy algorithm might also contain local maxima. The local maxima correspond to other absorbing states in $\mathbf{P}$. We denote the set of such states

$$\mathcal{LM} \subset \Omega \quad \text{(local maxima)}.$$

$\mathcal{LM}$ contains all solutions (states) $\mathbf{x} \in \Omega \setminus \{\mathbf{x}^\star\}$, such that for every other state $\mathbf{y} \in \Omega$ with $d(\mathbf{x},\mathbf{y}) = 2$, either $p_{ij}^c = 0$ or $U_\mathbf{y} < U_\mathbf{x}$. This condition is equivalent to that of Theorem 2. Every other state in Ω is a transient state. Denote by $\mathrm{TR} \subset \Omega$, the set of transient states. Then, $\Omega = \{\mathbf{x}^\star\} \cup \mathcal{LM} \cup \mathrm{TR}$.

In order to derive absorption related quantities, we write the matrix $\mathbf{P}$ in *canonical form*, where states are re-arranged such that transient states (TR) come first, followed by absorbing states corresponding to local maxima ($\mathcal{LM}$), followed by the maximum utility state $\mathbf{x}^\star$:

$$\mathbf{P} = \begin{array}{c} \begin{array}{ccc} \mathrm{TR} & \mathcal{LM} & \mathbf{x}^\star \end{array} \\ \left(\begin{array}{c|c|c} \mathbf{Q} & \mathbf{R_1} & \mathbf{R_2} \\ \hline \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} & 1 \end{array}\right) \end{array} \begin{array}{c} \mathrm{TR} \\ \mathcal{LM} \\ \mathbf{x}^\star \end{array}$$

Denote $|\mathcal{LM}| = r_1$ and $|\mathrm{TR}| = t$, i.e., there are r_1 local maxima and t transient states. Then, $\mathbf{I}$ is an $r_1 \times r_1$ identity matrix, $\mathbf{Q}$ is a $t \times t$ matrix, $\mathbf{R_1}$ is a non-zero $t \times r_1$ matrix, and $\mathbf{R_2}$ is a non-zero t-element column vector.

We can now define the fundamental matrix $\mathbf{N}$ for the absorbing Markov chain as follows:

$$\mathbf{N} = (\mathbf{I} - \mathbf{Q})^{-1} = \mathbf{I} + \mathbf{Q} + \mathbf{Q}^2 + \cdots \tag{16}$$

The last equality is easy to derive; a proof can be found in [34], page 45. $\mathbf{N}$ is a $t \times t$ matrix whose entry $n_{\mathbf{xy}}$ is the expected number of times the chain is in state $\mathbf{x}$, starting from state $\mathbf{y}$, before getting absorbed. Consequently, the sum of each line of the fundamental matrix of an absorbing Markov chain is the expected number of steps until absorption, when starting from the respective state.

Theorem 4 (Success Probability: Greedy)**.** *The probability that the* Greedy Forwarding *algorithm will succeed in finding the optimal solution, starting from any initial state with equal probability, is given by*

$$p_g = \frac{1}{t} \cdot \sum_{\mathbf{x}\in TR} b_{\mathbf{x}\mathbf{x}^\star}, \tag{17}$$

where $\mathbf{B}^\star = \{b_{\mathbf{x}\mathbf{x}^\star}\}$ *is a t-element column vector given by* $\mathbf{B}^\star = \mathbf{N}\mathbf{R_2}$.

Proof. Starting from transient state $\mathbf{x}$, the process may be captured in the optimal state, $\mathbf{x}^\star$, in one or more steps. The probability of capture on a single step is $p_{\mathbf{x}\mathbf{x}^\star}$. If this does not happen, the process may move either to an absorbing state in $\mathcal{LM}$ (in which case it is impossible to reach $\mathbf{x}^\star$), or to a transient state $\mathbf{y}$. In the latter case, there is probability $b_{\mathbf{y}\mathbf{x}^\star}$ of being captured in the optimal state. Hence we have:

$$b_{\mathbf{x}\mathbf{x}^\star} = p_{\mathbf{x}\mathbf{x}^\star} + \sum_{\mathbf{y}\in TR} p_{\mathbf{x}\mathbf{y}} \cdot b_{\mathbf{y}\mathbf{x}^\star}, \tag{18}$$

which can be written in matrix form as $\mathbf{B}^\star = \mathbf{R_2} + \mathbf{Q}\mathbf{B}^\star$. Thus $\mathbf{B}^\star = (\mathbf{I} - \mathbf{Q})^{-1}\mathbf{R_2} = \mathbf{N}\mathbf{R_2}$. $\mathbf{B}^\star$ is the vector of success probabilities starting from each of the t transient states. We obtain the probability of success starting from any state uniformly, as follows:

$$\frac{1}{t} \cdot b_{\mathbf{x}_1\mathbf{x}^\star} + \cdots + \frac{1}{t} \cdot b_{\mathbf{x}_t\mathbf{x}^\star} = \frac{1}{t} \cdot \sum_{\mathbf{x}\in TR} b_{\mathbf{x}\mathbf{x}^\star}. \tag{19}$$

The idea for the proof of this theorem came from [34], page 52. ∎

Theorem 5 (Convergence Delay: Greedy)**.** *The expected time for the* Greedy Forwarding *algorithm to find the optimal solution, starting from any initial state with equal probability, given that it does not get absorbed in any local maximum is given by*

$$\mathbb{E}[T_g] = \frac{1}{t} \cdot \sum_{\mathbf{x}\in TR} \tau_\mathbf{x}, \tag{20}$$

where $\boldsymbol{\tau} = \{\tau_\mathbf{x}\}$ *is a t-element column vector given by* $\boldsymbol{\tau} = \mathbf{D}^{-1}\mathbf{N}\mathbf{D}\mathbf{c}$. $\mathbf{c}$ *is a t-element column vector with ones, and* $\mathbf{D}$ *is a diagonal matrix with entries* $b_{\mathbf{x}\mathbf{x}^\star}$ *for* $\mathbf{x} \in TR$. *Furthermore, the variance of the convergence time is given by*

$$\mathbb{V}[T_g] = \frac{1}{t} \cdot \sum_{\mathbf{x}\in TR} \tau_{2,\mathbf{x}} + \left(\frac{1}{t} \cdot \sum_{\mathbf{x}\in TR} \tau_\mathbf{x}^2 - \mathbb{E}[T_g]^2\right), \tag{21}$$

where $\boldsymbol{\tau_2} = \{\tau_{2,\mathbf{x}}\}$ *and is given by* $\boldsymbol{\tau_2} = (2\mathbf{D}^{-1}\mathbf{N}\mathbf{D} - \mathbf{I})\boldsymbol{\tau} - (\boldsymbol{\tau})^2$. $(\boldsymbol{\tau})^2$ *is obtained from* $\boldsymbol{\tau}$ *by squaring each entry.*

Proof. Assume we start in a non-absorbing state $\mathbf{x}$ of our Markov chain $(\mathbf{X}_n)_{n\in\mathbb{N}_0}$ and compute all probabilities relative to the hypothesis that the process ends up in the optimal state, $\mathbf{x}^\star$. Then we obtain a new absorbing chain $(\mathbf{Y}_n)_{n\in\mathbb{N}_0}$ with a single absorbing state $\mathbf{x}^\star$. The non-absorbing states will be as before, except that we have new transition probabilities. We compute these as follows. Let $\mathbf{a}$ be the statement "$(\mathbf{X}_n)_{n\in\mathbb{N}_0}$ is absorbed in state $\mathbf{x}^\star$". Then if $\mathbf{x}$ is a non-absorbing state, the transition probabilities for $(\mathbf{Y}_n)_{n\in\mathbb{N}_0}$ are:

$$\begin{aligned}\mathbb{P}\left[\mathbf{Y}_{n+1}=\mathbf{y}\,|\,\mathbf{Y}_n=\mathbf{x}\right] &= \mathbb{P}\left[\mathbf{X}_{n+1}=\mathbf{y}\,|\,\mathbf{a}\wedge\mathbf{X}_n=\mathbf{x}\right]\\ &= \frac{\mathbb{P}\left[\mathbf{X}_{n+1}=\mathbf{y}\wedge\mathbf{a}\,|\,\mathbf{X}_n=\mathbf{x}\right]}{\mathbb{P}\left[\mathbf{a}\,|\,\mathbf{X}_n=\mathbf{x}\right]}\\ &= \frac{\mathbb{P}\left[\mathbf{a}\,|\,\mathbf{X}_{n+1}=\mathbf{y}\right]\cdot\mathbb{P}\left[\mathbf{X}_{n+1}=\mathbf{y}\,|\,\mathbf{X}_n=\mathbf{x}\right]}{\mathbb{P}\left[\mathbf{a}\,|\,\mathbf{X}_n=\mathbf{x}\right]}\\ \hat{p}_{\mathbf{xy}} &= \frac{b_{\mathbf{yx}^\star}p_{\mathbf{xy}}}{b_{\mathbf{xx}^\star}}.\end{aligned}$$

The standard form for $\hat{\mathbf{P}}$, the transition matrix of $(\mathbf{Y}_n)_{n\in\mathbb{N}_0}$, may be obtained as follows. The matrix $\hat{\mathbf{R}}$ is a column vector with $\hat{\mathbf{R}} = \{\frac{p_{\mathbf{xx}^\star}}{b_{\mathbf{xx}^\star}}\}$. Let D be a diagonal matrix with diagonal entries $b_{\mathbf{xx}^\star}$, for $\mathbf{x}$ non-absorbing. Then $\hat{\mathbf{Q}} = \mathbf{D}^{-1}\mathbf{Q}\mathbf{D}$ and consequently, $\hat{\mathbf{N}} = \mathbf{D}^{-1}\mathbf{N}\mathbf{D}$. $\hat{\mathbf{B}}^\star$ is now a t-element column vector of ones.

The derivations of $\boldsymbol{\tau}$ and $\boldsymbol{\tau_2}$ as a function of $\hat{\mathbf{N}}$ can be found in [34], page 51. The initial state is chosen uniformly in TR, therefore, using $\boldsymbol{\tau}$, $\boldsymbol{\tau_2}$ and the laws of total expectation and respectively total variance, we obtain the results in equations 20 and 21.

This proof is inspired from [34]. ∎

Corollary 6. *The expected time for the* Greedy Forwarding *algorithm to converge to any solution, locally or globally optimal, is given by* $\frac{1}{t}\cdot\sum_{\mathbf{x}\in TR} T_{\mathbf{x}}$, *where* $\mathbf{T} = \{T_{\mathbf{x}}\}$ *is given by* $\mathbf{T} = \mathbf{N}\mathbf{c}$. $\mathbf{c}$ *is a* t*-element column vector with ones.*

4.3 Performance of MCMC Forwarding

We now turn our attention to the *Markov Chain Monte Carlo* algorithm. In the context of Opportunistic Networks, we are interested in mobility scenarios whose contact graph is "connected", that is, "space-time" paths [35] exist between every two nodes in the network. If this is not the case, the network is considered permanently partitioned (within the time horizon we consider) and should be treated as two separate networks, since no DTN algorithm allows nodes in the two partitions to communicate.

In the case of the Greedy algorithm, as we saw earlier, the acceptance matrix may remove links and make the resulting Markov chain described by $\mathbf{P}$ *reducible*, even if the contact matrix $\mathbf{P}^c$ is irreducible.

Lemma 7 (Ergodicity)**.** *The Markov Chain defined by the transition matrix* $\mathbf{P}$ *for the MCMC algorithm is ergodic, if the respective contact matrix* $\mathbf{P}^c$ *defines an ergodic chain. Furthermore, the probability that the MCMC algorithm discovers the optimal solution, as time goes to infinity, goes to* 1.

Proof. The first part of the Lemma: "$\mathbf{P}^c$ ergodic $\Rightarrow$ $\mathbf{P}$ ergodic", is proven as in item i of Th. 3. The second part is also closely related to item ii of Th. 3. Indeed, since the stationary distribution of the Markov chain described by $\mathbf{P}$ is chosen such that most of its mass is on the optimal configuration, the chain will inevitably end up in that configuration as time goes to infinity. ∎

The above theorem states that, given enough time, the MCMC algorithm is guaranteed to discover (visit) the optimal state. We next look at how long this actually takes, on average.

Since $\mathbf{P}$ for the MCMC algorithm is an ergodic chain, we cannot anymore use the theory of absorbing matrices of Section 4.2 to calculate performance quantities of interest. The fundamental matrix $\mathbf{N}$ cannot be defined, as $(\mathbf{I}-\mathbf{Q})$ is not reversible anymore [34]. In this context, we must look instead at the statistics of the *first hitting time* to the Markov chain state corresponding to the optimal configuration, starting from any other state. This is also often referred to as *first passage time*. Since the Markov chain is ergodic, hitting the optimal state does not guarantee that the algorithm stays there forever. Nevertheless, the *mean first passage time* can be seen as a lower bound on the convergence delay of the MCMC. If the simulated annealing algorithm is designed correctly (see Section 3), this bound can become tight.

Hitting times on finite graphs or Markov chains can be derived in different ways, including first step analysis, the electric network analogy, or through spectral properties. Here, we will use an approach that is similar to the approach of Section 4.2, which derives from first step analysis. While $\mathbf{N}$ cannot be defined as in the absorbing Markov chain case, a different *fundamental matrix for ergodic chains* can still be defined that plays a similar role:

$$\mathbf{Z} = (\mathbf{I}-\mathbf{P}+\boldsymbol{\Pi})^{-1}, \tag{22}$$

where $\boldsymbol{\Pi}$ is a matrix with each of its rows being the *stationary probability vector* for transition matrix $\mathbf{P}$.

Theorem 8 (Convergence Delay: MCMC)**.** *The expected time for the* MCMC Forwarding *algorithm to converge to the optimal configuration/state* $\mathbf{x}^\star$*, starting from any state with equal probability, is lower bounded by*

$$\frac{1}{|\Omega|-1}\sum_{\mathbf{y}\in\Omega\setminus\{\mathbf{x}^\star\}}\frac{z_{\mathbf{x}^\star\mathbf{x}^\star}-z_{\mathbf{yx}^\star}}{\pi_{\mathbf{x}^\star}}, \tag{23}$$

where $\pi_{\mathbf{x}^\star}$ *is the stationary probability of the optimal state and* $|\Omega|$ *is the size of the solution space.*

Proof. To lower bound the expected convergence time, we will use the mean first passage time from a given state $\mathbf{y}$ to the optimal configuration $\mathbf{x}^\star$. In [34], pages 78-79, the authors

prove using matrix algebra that the mean first passage times from any state to any state contained in the matrix **M** are given by:

$$\mathbf{M} = (\mathbf{I} - \mathbf{Z} + \mathbf{E}\mathbf{Z}_{\mathbf{dg}})\mathbf{D}, \tag{24}$$

where **E** is a square matrix with all entries 1, $\mathbf{Z}_{\mathbf{dg}}$ agrees with **Z** on the main diagonal, but is 0 elsewhere, and **D** is a diagonal matrix with diagonal elements $d_{\mathbf{yy}} = \frac{1}{\pi_{\mathbf{y}}}$.

From a given state **y** to the optimal configuration $\mathbf{x}^\star$, the mean first passage is element $m_{\mathbf{yx}^\star}$ of **M**, which from equation 24 can be written as:

$$m_{\mathbf{yx}^\star} = \frac{z_{\mathbf{x}^\star\mathbf{x}^\star} - z_{\mathbf{yx}^\star}}{\pi_{\mathbf{x}^\star}}. \tag{25}$$

Using equation 25 above and averaging over all non-optimal states ($|\Omega| - 1$ in total), we obtain the expected first passage time starting from any state with equal probability to the optimal configuration/state $\mathbf{x}^\star$ as shown in equation 23. This quantity is a lower bound for the convergence time of the MCMC algorithm, since the probability for the chain to leave $\mathbf{x}^\star$ is, albeit very small, larger than 0. ■

The temperature parameter discussed in Section 3.5 is important for this result. It determines the $A_{\mathbf{xy}}$ components of the Markov chain **P**. The formula is exact for fixed temperature values, when the Markov chain is time-homogeneous. However, MCMC algorithms of this type are generally used with some cooling method to speed up convergence, which makes the Markov chain inhomogeneous. In this case, the formula can still be used with e.g., the mean temperature value of the cooling schedule to obtain a lower bound.

Remark: The probability and delay estimations for both the greedy and the MCMC algorithms require the inversion of matrices. This is relatively easy for a Markov chain the size of the network, as is the case, e.g., for the content placement problem with $L = 1$ copy (as shown in Section 2). As the state space of the Markov chain increases (e.g., $L > 3$), it becomes difficult to calculate inverses and to handle the chain in general. The state space explosion problem has been mitigated in other context through e.g., state lumping, Petrinet-based models etc. We recognize this as a scalability problem for our framework and plan to address it in the future.

4.4 From Contact Clock Ticks to Real Time

It is important to note here that, in the above results concerning convergence time, we have measured time in terms of "contact clock ticks". We look at the sequence of consecutive contacts occurring (between any two nodes) and derive the expected number of contacts until the optimal solution is found. This might be desirable in many cases, in order to fairly compare performance in the different mobility scenarios considered, due to the (sometimes artificially) varying sparsity of contact measurements (e.g., average inter-contact time is much higher in MIT than in ETH).

We show here how one could go back from the results of Th. 5 and 8 to real time. In a somewhat different context focusing on distributed estimation, the authors of [36] assume exponential clock ticks and use the law of large numbers to show that the exact time of the n-th tick is highly concentrated around its average. We argue here that a slightly more generic statement can be made based on Renewal Theory and *Wald's equation* [23].

Lemma 9. *Let the expected time between consecutive contact events be $E[C]$. Let further T_{st} denote the delay or convergence time of a given process over an opportunistic contact graph* **W***, in terms of number of contacts. Then, the expected delay of this process is equal to*

$$E[C] \cdot E[T_{st}]. \tag{26}$$

Proof. Let the times of consecutive contact events be renewals. If time is counted at renewal times (contact times), it is easy to see that delay quantities derived in Section 4, are *stopping times*. We can then apply Wald's equation [23] to get Eq. (26). ■

5 Validation of Analytical Results

We assess here the validity of the above theorems using the same traces as previously. We use the p_{ij}^c as calculated in Section 4.1 and some decomposable utilities for the calculation of the A_{ij} components. The various utility functions we test, originate in a weighted graph, $\mathbf{W} = \{w_{ij}\}$ for the network. The weights may express the relationship between the two nodes or they may be uncorrelated to node relationships. For each node, we calculate utility as its weighted degree:

$$d_i = \sum_{j \in \mathcal{N}} w_{ij}. \tag{27}$$

In this work, we use the following four utility functions:

i) **Frequency utility**. As the name suggests, this utility is based on the contact frequency between each node pair. This means that w_{ij} is equal to the number of times the two nodes have met throughout the trace. The utility of a node is the its degree as in Eq. (27).

ii) **PCA utility**. This is a combined utility based on contact frequency and duration. We first calculate the frequency and duration matrices for each node pair. Then, we transform them to scalar weights to obtain **W** using principal component analysis (PCA) [33]. Once again, the utility of a node is the its degree as in Eq. (27).

iii) **Tie strength utility**. This utility is based on the strength of the tie between each node pair, as calculated in Section 4.1. Here, $w_{ij} = p_{ij}^c$, the contact probability of the two nodes. The utility of a node is the its degree as in Eq. (27).

iv) **Random utility**. Finally, for this utility, we assign random weights w_{ij} to each node pair and use these to calculate the utility of each node as d_i.

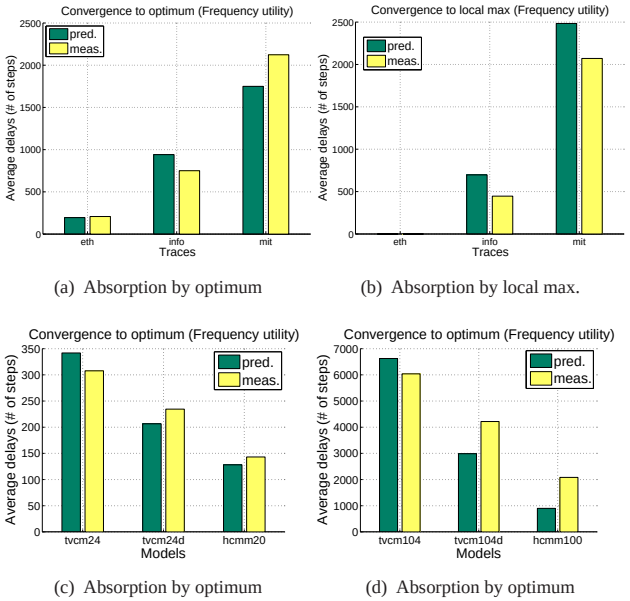
		ETH	INFO	MIT	TVCM24	TVCM24d	HCMM20	TVCM104	TVCM104d	HCMM100
Optimum	pred.	1.0000	0.5637	0.9733	1.0000	1.0000	1.0000	1.0000	1.0000	0.2427
	meas.	1.0000	0.5659	0.7066	1.0000	1.0000	1.0000	1.0000	1.0000	0.2386
Local max.	pred.	N/A	0.3762	0.0266	N/A	N/A	N/A	N/A	N/A	0.2368
	meas.	N/A	0.3726	0.0266	N/A	N/A	N/A	N/A	N/A	0.2118

Table 2: *Absorption probabilities with frequency utility*

In the following, we present for each of the four utilities: (i) absorption probability results for the Greedy algorithm, corresponding to Theorem 4, (ii) absorption delay results for the Greedy algorithm, corresponding to Theorem 5[5]., and (iii) lower bound results on the convergence delay for the MCMC algorithm, corresponding to Theorem 8.For the two algorithms, we show both delays averaged over all nodes in the network, as well as individual delay results for each node.

5.1 Results for the Frequency Utility

The frequency utility is the first and simplest utility one may consider in the context of the Content Placement problem and possibly in other contexts as well. Since the goal is to make a piece of content as available as possible to the entire network, choosing the node that meets other nodes the most frequently seems a reasonable strategy. This utility is relatively strongly correlated with the tie strength as defined in Section 4.1.



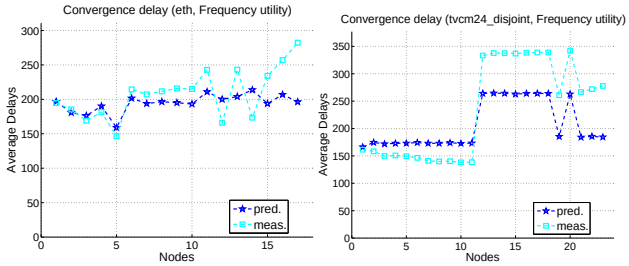
(a) Absorption by optimum (b) Absorption by local max.

(c) Absorption by optimum (d) Absorption by optimum

Figure 6: *Average absorption delays with frequency utility (Greedy)*

Table 2 shows absorption probabilities predicted using Thm. 4 and measured in the traces. The first two rows give the probability of absorption by the global optimum, the last two – the probability of absorption by a local maximum. In most cases, the prediction is very accurate, both with a single absorbing state, the global maximum (in ETH and TVCM24) and when local maxima are present, resulting in multiple absorbing states (in Infocom, HCMM100).

Fig. 6 compares the predicted and measured values of the average delay over all nodes for the greedy algorithm using the frequency utility above. The prediction is very accurate in most cases. For larger scenarios (MIT, TVCM104, TVCM104d, HCMM100), a much longer trace or simulation time is needed for accurate results. This is because delays are longer, therefore more time is required to get sufficient samples for statistical significance.



(a) ETH trace (b) TVCM24d scenario

Figure 7: *Individual absorption delays (Greedy)*

In Fig. 7, we show delay for individual nodes rather than aggregated values. Nodes are ordered by increasing absolute difference between predicted and measured values for clarity. Some nodes are not present, as they are disconnected from the network (we prune links with less than 50 contacts for statistical significance). Other nodes are local maxima, hence they are absorbing states themselves. In general, most nodes are accurately predicted.

However, a few outliers are also present: these are nodes with irregular activity: bursts of very many contacts in very little time, followed by much longer periods of inactivity. For these nodes, the assumption of geometrically distributed meeting probabilities does not hold, since their contacts are no longer independent of each other.

A good illustration of this fact is the TVCM24d scenario, in Figure 7(b). Here, bridges and nodes membership in communities and are clearly identifiable by the accuracy of the prediction. For the larger community (first 11 nodes), where the optimal relay is almost surely located (the 12th node in the community), the absorption delays are very accurately predicted, as no irregular links (e.g., with bridge nodes) are involved. For the other community and for the bridges, absorption by the optimal relay located in the first community

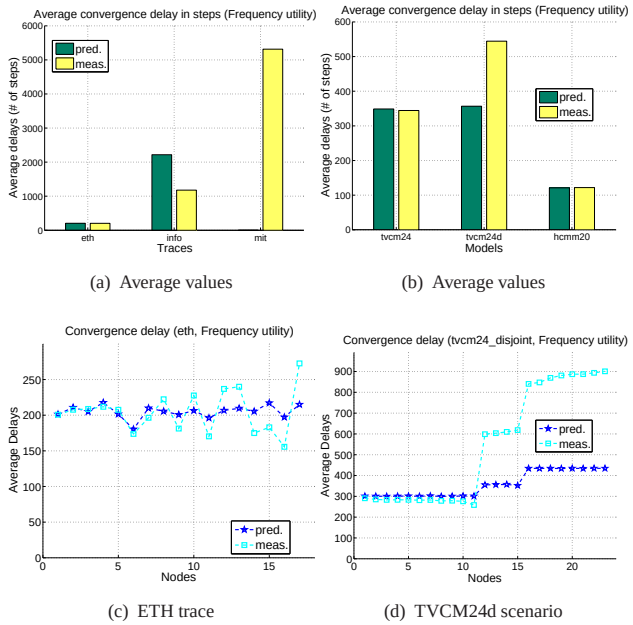
[5]Note that both Thm. 4 and Thm. 5 also allow for the calculation of local maxima quantities.

		ETH	INFO	MIT	TVCM24	TVCM24d	HCMM20	TVCM104	TVCM104d	HCMM100
Optimum	pred.	1.0000	0.5267	0.6772	1.0000	0.6086	1.0000	1.0000	0.7862	0.2141
	meas.	1.0000	0.5435	0.5561	1.0000	0.5979	1.0000	1.0000	0.6893	0.2185
Local max.	pred.	N/A	0.4132	0.2957	N/A	0.3913	N/A	N/A	0.2138	0.2404
	meas.	N/A	0.3936	0.1465	N/A	0.4020	N/A	N/A	0.3106	0.2377

Table 3: *Absorption probabilities with PCA utility*

must go through the links with irregular activity of bridge nodes. Therefore, the predictions are rougher.

To improve the predictions, we are planning to use a more sophisticated model for these nodes in the future.



(a) Average values (b) Average values

(c) ETH trace (d) TVCM24d scenario

Figure 8: *Convergence delays bounds with frequency utility (MCMC)*

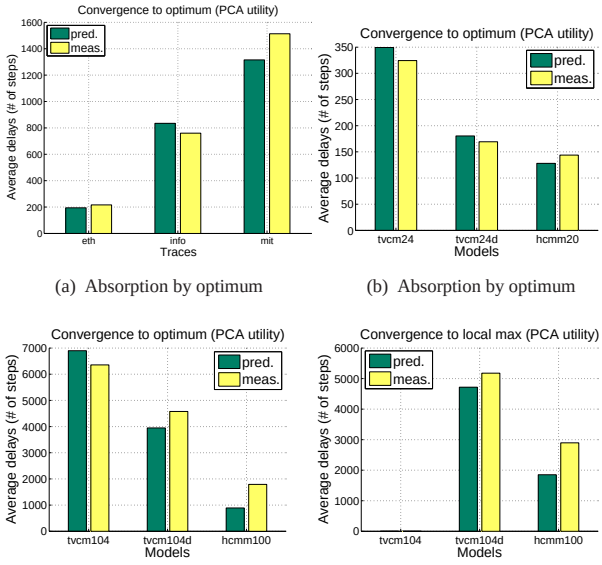
Finally, Fig. 8 shows aggregated (8(a), 8(b)) and individual results (8(c), 8(d)) for the convergence delay of the MCMC algorithm. These results correspond to Thm. 8. In the majority of traces, the predicted and measured values for both algorithms are consistent. For some traces, certain combinations of utility and temperature may render the Markov chain non-ergodic, by creating either closed communicating classes or periodicity or both (e.g., the MIT trace in Figure 8(a)). The delay prediction method is able to detect this.

In addition, the effect of nodes with irregular activity is also here obvious. Once again, the bridge nodes and various communities are clearly identifiable in the TVCM24d scenario in Figure 8(d).

5.2 Results for the PCA Utility

In the frequency utility, we only account for how often a pair of nodes meets. However, in some application contexts, we may also be interested in the duration of each meeting. For exemple, in the content placement problem with large sizes of the shared content, making the content the most available does not only mean finding the relay with the most frequent meetings, but the relay with the most frequent and longest meeting, so that there is enough time for the transmission to take place. To account for that, we analyze here a PCA-based utility, combining frequency and duration of encounters. Similarly to the frequency one, this utility is also relatively strongly correlated with the tie strength as defined in Section 4.1.

Table 3 shows absorption probabilities predicted using Thm. 4 and measured in the traces. The first two rows give the probability of absorption by the global optimum, the last two – the probability of absorption by a local maximum. In most cases, the prediction is very accurate, both with a single absorbing state, the global maximum (in ETH and TVCM24) and when local maxima are present, resulting in multiple absorbing states (in Infocom, MIT, TVCM24d).



(a) Absorption by optimum (b) Absorption by optimum

(c) Absorption by optimum (d) Absorption by local max.

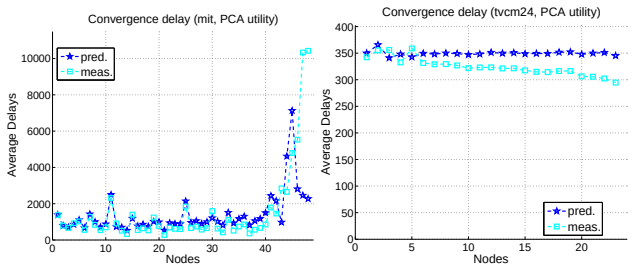
Figure 9: *Average absorption delays with PCA utility (Greedy)*

Fig. 9 compares the predicted and measured values of the average delay over all nodes for the greedy algorithm using the PCA utility above. The prediction is very accurate in most cases. For larger scenarios (MIT, TVCM104, TVCM104d, HCMM100), a much longer trace or simulation time is needed for accurate results. This is because delays

		ETH	INFO	MIT	TVCM24	TVCM24d	HCMM20	TVCM104	TVCM104d	HCMM100
Optimum	pred.	1.0000	0.9419	0.9733	1.0000	1.0000	1.0000	1.0000	1.0000	0.6086
	meas.	1.0000	0.9393	0.7066	1.0000	1.0000	1.0000	1.0000	1.0000	0.5979
Local max.	pred.	N/A	0.0580	0.0266	N/A	N/A	N/A	N/A	N/A	0.3913
	meas.	N/A	0.0606	0.0266	N/A	N/A	N/A	N/A	N/A	0.4020

Table 4: *Absorption probabilities with tie strength utility*

are longer, therefore more time is required to get sufficient samples for statistical significance.



(a) MIT trace (b) TVCM24d scenario

Figure 10: *Individual absorption delays with PCA utility (Greedy)*

In Fig. 10, we show delay for individual nodes rather than aggregated values. Nodes are ordered by increasing absolute difference between predicted and measured values for clarity. Some nodes are not present, as they are disconnected from the network (we prune links with less than 50 contacts for statistical significance). Other nodes are local maxima, hence they are absorbing states themselves. In general, most nodes are accurately predicted.

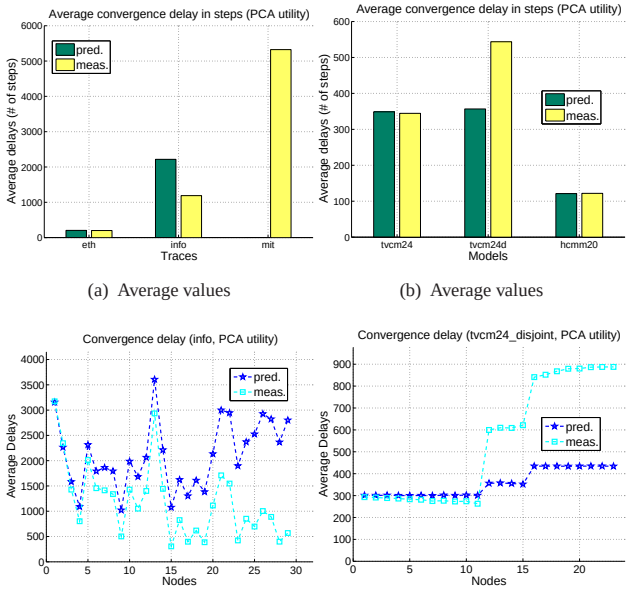
However, a few outliers are also present. As mentioned in Section 5.1, these are usually nodes with irregular activity. When these nodes are not present (e.g., Figure 10(b)), the results are much more accurate and we can no longer distinguish between different node classes (bridges, communities).

Finally, Fig. 11 shows aggregated (11(a), 11(b)) and individual results (11(c), 11(d)) for the convergence delay of the MCMC algorithm. These results correspond to Thm. 8. In the majority of traces, the predicted and measured values for both algorithms are consistent. For some traces, certain combinations of utility and temperature may render the Markov chain non-ergodic, by creating either closed communicating classes or periodicity or both (e.g., the MIT trace in Figure 11(a)). The delay prediction method is able to detect this.

In addition, the effect of nodes with irregular activity is also here obvious. Once again, the bridge nodes and various communities are clearly identifiable in the TVCM24d scenario in Figure 11(d).

5.3 Results for the Tie Strength Utility

With the frequency and PCA utilities, we are essentially trying to estimate the strength of a relationship between two



(a) Average values (b) Average values

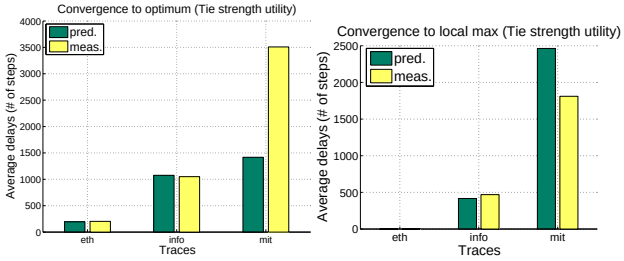
(c) Infocom trace (d) TVCM24d scenario

Figure 11: *Convergence delays bounds with PCA utility (MCMC)*

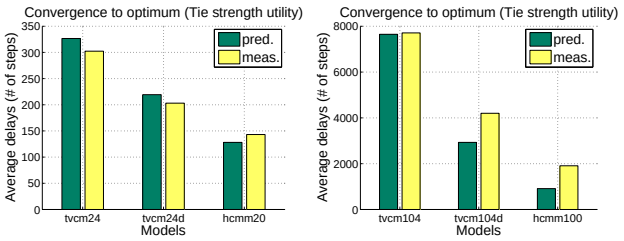
nodes. In the Content Placement problem, this is very useful in order to find the node with the strongest ties to the rest of the network. Intuitively, this node would provide maximum availability for any content it stores. Another way to estimate the strength of the tie between two nodes is the one we introduce in Section 4.1. In this section, we use this estimate to calculate the utility of each node and evaluate our prediction theory. In this case, the meeting probabilities and the utilities are perfectly correlated.

Table 4 shows absorption probabilities predicted using Thm. 4 and measured in the traces. The first two rows give the probability of absorption by the global optimum, the last two – the probability of absorption by a local maximum. In most cases, the prediction is very accurate, both with a single absorbing state, the global maximum (in ETH and TVCM24) and when local maxima are present, resulting in multiple absorbing states (in Infocom, HCMM100).

Fig. 12 compares the predicted and measured values of the average delay over all nodes for the greedy algorithm using the tie strength utility above. The prediction is very accurate in most cases. For larger scenarios (MIT, TVCM104, TVCM104d, HCMM100), a much longer trace or simulation time is needed for accurate results. This is because delays are longer, therefore more time is required to get sufficient



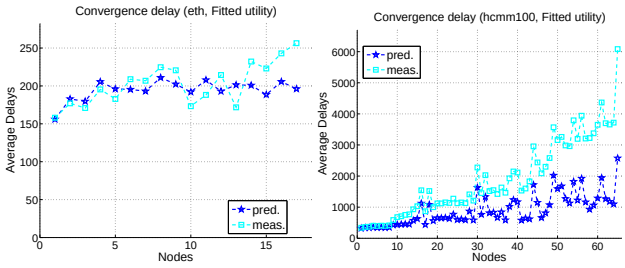
(a) Absorption by optimum (b) Absorption by local max.



(c) Absorption by optimum (d) Absorption by optimum

Figure 12: *Average absorption delays with tie strength utility (Greedy)*

samples for statistical significance.



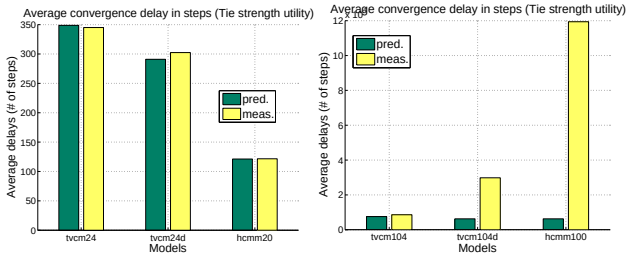
(a) ETH trace (b) HCMM100 scenario

Figure 13: *Individual absorption delays with tie strength utility (Greedy)*

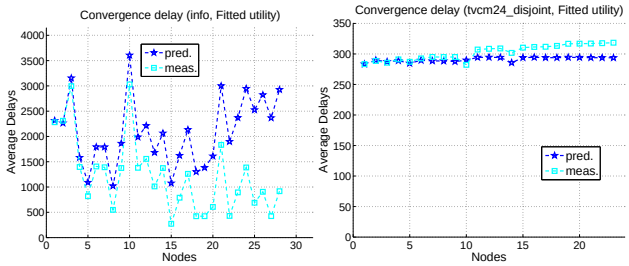
In Fig. 13, we show delay for individual nodes rather than aggregated values. Nodes are ordered by increasing absolute difference between predicted and measured values for clarity. Some nodes are not present, as they are disconnected from the network (we prune links with less than 50 contacts for statistical significance). Other nodes are local maxima, hence they are absorbing states themselves. In general, most nodes are accurately predicted.

However, a few outliers are also present. As mentioned before, these are usually nodes with irregular activity. Particularly, the HCMM100 scenario shows the relative robustness of our prediction method. The HCMM model is specifically designed to follow recent findings that, in practice, pairwise intercontact times follow a power law. Consequently, our geometric assumption from Section 4.1 is clearly violated. In spite of this violation, the prediction is still decent. This sustains the thesis expressed in [37], that pairwise intercontact times have exponential tails.

Finally, Fig. 14 shows aggregated (14(a), 14(b)) and individual results (14(c), 14(d)) for the convergence delay of the



(a) Average values (b) Average values



(c) Infocom trace (d) TVCM24d scenario

Figure 14: *Convergence delays bounds with tie strength utility (MCMC)*

MCMC algorithm. These results correspond to Thm. 8. In the majority of traces, the predicted and measured values for both algorithms are consistent.

While previously the effect of nodes with irregular activity was obvious, in Figure 14(d), this effect is much more toned down. This may be related to the perfect correlation between the node utilities and tie strengths.

5.4 Results for the Random Utility

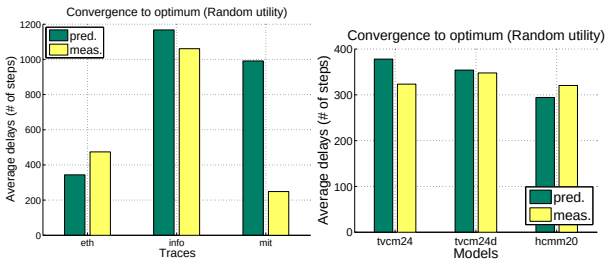
To finish our evaluation, we conduct a series of experiments using a random utility, which should have little or no correlation with the tie strength defined in Section 4.1. While this is of little practical interest, it will help better understand the relationship between utility and tie strength. Moreover, it will also verify whether the proposed prediction method is applicable to a wider range of utility functions.

Table 4 shows absorption probabilities predicted using Thm. 4 and measured in the traces. The first two rows give the probability of absorption by the global optimum, the last two – the probability of absorption by a local maximum. In most cases, the prediction is very accurate, both with a single absorbing state, the global maximum (in TVCM24 and TVCM104) and when local maxima are present, resulting in multiple absorbing states (in Infocom, TVCM104d, HCMM100).

Fig. 15 compares the predicted and measured values of the average delay over all nodes for the greedy algorithm using the random utility. The prediction is very accurate in most cases. For larger scenarios (MIT, TVCM104, TVCM104d, HCMM100), a much longer trace or simulation time is needed for accurate results. This is because delays are longer,

		ETH	INFO	MIT	TVCM24	TVCM24d	HCMM20	TVCM104	TVCM104d	HCMM100
Optimum	pred.	0.4881	0.8675	0.0145	1.0000	1.0000	1.0000	1.0000	0.5195	0.6086
	meas.	0.9000	0.8794	0.0370	1.0000	1.0000	1.0000	1.0000	0.5028	0.5979
Local max.	pred.	0.5118	0.0297	0.2737	N/A	N/A	N/A	N/A	0.2086	0.3913
	meas.	0.0999	0.0476	0.1225	N/A	N/A	N/A	N/A	0.2019	0.4020

Table 5: *Absorption probabilities with random utility*

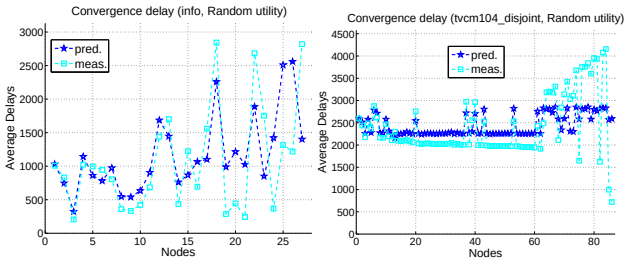


(a) Absorption by optimum (b) Absorption by optimum

(c) Absorption by optimum (d) Absorption by local max.

Figure 15: *Average absorption delays with random utility (Greedy)*

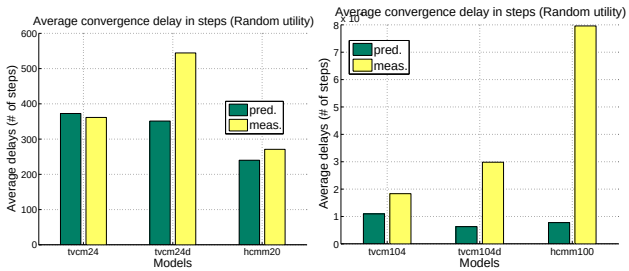
therefore more time is required to get sufficient samples for statistical significance.



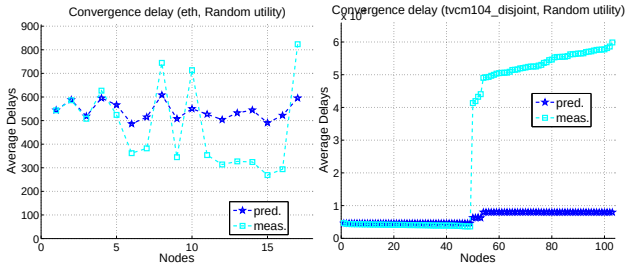
(a) Infocom trace (b) TVCM104d scenario

Figure 16: *Individual absorption delays with random utility (Greedy)*

In Fig. 16, we show delay for individual nodes rather than aggregated values. Nodes are ordered by increasing absolute difference between predicted and measured values for clarity. Some nodes are not present, as they are disconnected from the network (we prune links with less than 50 contacts for statistical significance). Other nodes are local maxima, hence they are absorbing states themselves. In general, most nodes are accurately predicted. However, a few outliers are also present. As mentioned before, these are usually nodes with irregular activity.



(a) Average values (b) Average values



(c) ETH trace (d) TVCM104d scenario

Figure 17: *Convergence delays bounds with random utility (MCMC)*

Finally, Fig. 17 shows aggregated (17(a), 17(b)) and individual results (17(c), 17(d)) for the convergence delay of the MCMC algorithm. These results correspond to Thm. 8. While the results are not of the same quality as previously, predicted and measured values for both algorithms are still relatively consistent.

Once again, the effect of nodes with irregular activity is obvious. In Figure 17(d), the convergence delay of a content starting from any source node in the largest community (size 50) is perfectly predicted. In contrast, predictions for delays from nodes in the other communities not at all accurate. This indicates that the degree of correlation of the utility to the tie strength is important for the accuracy of the prediction. Most importantly however, it prompts us to find a better model for ties which do not follow our current model.

6 Conclusion

This work has dealt with Distributed Optimization in the context of Opportunistic Networks. We proposed an analytical framework and used it to study deterministic (Greedy) and stochastic utility ascent (Markov Chain Monte Carlo) algorithms. Using this framework, we proved necessary and

sufficient conditions for their correctness, and derived closed form solutions for their performance (convergence probability and delay), in generic mobility scenarios. We used real and synthetic mobility traces to validate our findings, and found a close match between predicted and measured quantities.

In the short term future, we plan to refine our framework to account for inaccuracies (e.g., nodes with irregular activity etc). On the longer term, we will demonstrate and evaluate the applicability of this framework to the more general distributed optimization problems as described in Section 2.

References

[1] Widmer J and Le Boudec JY. *Network coding for efficient communication in extreme networks*. WDTN 2005.

[2] Vahdat A and Becker D. *Epidemic routing for partially connected ad hoc networks*. Tr, Duke University, NC, USA, 2000.

[3] Spyropoulos T, Psounis K et al. *Spray and Wait: an efficient routing scheme for intermittently connected mobile networks*. SIGCOMM WDTN 2005.

[4] Zhao W, Ammar M et al. *Multicasting in delay tolerant networks: semantic models and routing algorithms*. WDTN 2005.

[5] Balasubramanian A, Levine B et al. *DTN routing as a resource allocation problem*. SIGCOMM CCR, 2007.

[6] Pitkänen MJ and Ott J. *Redundancy and distributed caching in mobile dtns*. MobiArch 2007.

[7] Lindgren A, Doria A et al. *Probabilistic routing in intermittently connected networks*. SIGMOBILE Mob Comput Commun Rev, 7(3), 2003.

[8] Sarafijanovic-Djukic N, Piórkowski M et al. *Island hopping: Efficient mobility-assisted forwarding in partitioned networks*. IEEE SECON 2006.

[9] Jones EP and Ward PA. *Routing strategies for delay-tolerant networks*, 2006. URL `http://ccng.uwaterloo.ca/~pasward/Publications/dtn-routing-survey.pdf`.

[10] Boyd S and Vandenberghe L. Convex Optimization. 2004.

[11] Gao W, Li Q et al. *Multicasting in delay tolerant networks: a social network perspective*. MobiHoc 2009.

[12] Picu A and Spyropoulos T. *Minimum Expected *-cast Time in DTNs*. BIONETICS 2009.

[13] Lenders V, May M et al. *Wireless ad hoc podcasting*. SIGMOBILE Mob Comput Commun Rev, 2008.

[14] Hu L, Le Boudec JY et al. *Optimal channel choice for collaborative ad-hoc dissemination*. Infocom 2010.

[15] Krifa A, Barakat C et al. *Optimal buffer management policies for delay tolerant networks*. IEEE SECON. 2008.

[16] Eagle N and Pentland A. *Reality mining: sensing complex social systems*. Pers Ubiq Comput, 10(4), 2006.

[17] Hui P, Chaintreau A et al. *Pocket switched networks and human mobility in conference environments*. WDTN 2005.

[18] Lenders V, Wagner J et al. *Measurements from an 802.11b mobile ad hoc network*. WOWMOM 2006.

[19] Hsu WJ, Spyropoulos T et al. *Modeling spatial and temporal dependencies of user mobility in wireless mobile networks*. IEEE/ACM Transactions on Networking, 2009.

[20] Boldrini C and Passarella A. *Hcmm: Modelling spatial and temporal properties of human mobility driven by users' social relationships*. Comput Commun, 33:1056–1074, 2010.

[21] Musolesi M and Mascolo C. *A community based mobility model for ad hoc network research*. REALMAN 2006.

[22] Newman MEJ. *The structure and function of complex networks*. SIAM Review, 45(2):167–256, 2003.

[23] Ross S. Stochastic Processes. 1996.

[24] Hui P, Crowcroft J et al. *Bubble rap: social-based forwarding in delay tolerant networks*. MobiHoc 2008.

[25] Brémaud P. Markov chains : Gibbs fields, Monte Carlo simulation, and queues. 2001.

[26] Kauffmann B, Baccelli F et al. *Measurement-based self organization of interfering 802.11 wireless access networks*. Infocom 2007.

[27] Geman S and Geman D. *Stochastic relaxation, Gibbs distributions and the bayesian restoration of images*. Journal of Applied Statistics, 20(5), 1993.

[28] Spyropoulos T, Psounis K et al. *Efficient routing in intermittently connected mobile networks: the single-copy case*. IEEE/ACM Trans Netw, 16(1), 2008.

[29] Spyropoulos T, Turletti T et al. *Routing in delay-tolerant networks comprising heterogeneous node populations*. IEEE Transactions on Mobile Computing, 8, 2009.

[30] Dubois-Ferriere H, Grossglauser M et al. *Age matters: efficient route discovery in mobile ad hoc networks using encounter ages*. MobiHoc 2003.

[31] Burgess J, Gallagher B et al. *MaxProp: Routing for Vehicle-Based Disruption-Tolerant Networks*. Infocom 2006.

[32] Wolff RW. *Poisson Arrivals See Time Averages*. Operations Research, 30(2), 1982.

[33] Duda R, Hart P et al. Pattern Classification. 2000.

[34] Kemeny J and Snell L. Finite Markov Chains. 1960.

[35] Borrel V, Ammar MH et al. *Understanding the wireless and mobile network space: a routing-centered classification*. CHANTS 2007.

[36] Boyd S, Ghosh A et al. *Randomized gossip algorithms*. IEEE/ACM Transactions on Networking, 2006.

[37] Karagiannis T, Le Boudec JY et al. *Power law and exponential decay of inter contact times between mobile devices*. MobiCom 2007.