Simulating Opportunistic Networks: Survey and Future Directions

Jens Dede[1], Anna Förster[1], Enrique Hernández-Orallo[3], Jorge Herrera-Tapia[3,4], Koojana Kuladinithi[2], Vishnupriya Kuppusamy[1], Pietro Manzoni[3], Anas bin Muslim[1], Asanga Udugama[1], Zeynep Vatandas[2]
[1]*University of Bremen, Germany*
[2]*Hamburg University of Technology, Germany*
[3]*Universitat Politècnica de València, Spain*
[4]*Universidad Laica Eloy Alfaro de Manabí, Ecuador*

***Abstract*—Simulation is one of the most powerful tools we have to evaluate the performance of Opportunistic Networks. In this survey we focus on available tools and models, compare their performance and precision and experimentally show the scalability of different simulators. We also perform a gap analysis of state-of-the-art Opportunistic Network simulations and sketch possible further developments and research directions.**

This survey is targeted at students starting to work and research in this area, but it is also a valuable source of information for experienced researchers.

***Index Terms*—Simulation, Opportunistic Networks, OMNeT++, the ONE, Adyton, ns-3, SUMO, BonnMotion, Mobility Models, Radio Propagation Models, Traffic Models, Data Propagation, Energy Consumption Models, Simulation Scalability.**

I. Introduction

Opportunistic networks (OppNets for short) have faced a constantly growing interest from the research community since their first appearance. Their main idea is to exploit direct, localised communications instead of infrastructure-based communications. This is also their main advantage: They can operate anywhere where end-user devices are present. For this reason they are considered extremely useful for emergency scenarios, in rural or remote areas and for overloaded or sabotaged networks. They do not compete with mainstream 4G / 5G networks: instead, they complement them.

The motivation for this survey comes mainly from the observation that the deployment of opportunistic services and applications still proceeds too slowly [1]. In our opinion, one of the main reasons for this is the low readiness level of the technologies being considered. However, in this case we think this is a “chicken and egg” problem:

- Researchers need large-scale deployments with real users, real devices and real scenarios to increase the robustness and reliability of the services being designed.
- Users do not want to use research-level, non-reliable services.

Researchers need therefore to depend on small-scale deployments and simulated environments to perform their investigations. Consequently, we consider that the best way out of this cycle is to have well-performing, realistic, reliable, and highly scalable simulation tools.

This survey focusses exactly on this objective. It provides a thorough review of existing simulation models and tools available for opportunistic networks. We compare them in terms of their precision, scalability and performance. We focus on four large simulation tools, free to use and widely accepted by the OppNets community, namely (in alphabetical order): Adyton, OMNeT++, ONE and ns-3.

There exist various relevant and thorough surveys on OppNets related topics in the literature, one of the most widely used being the one by Mota et al. [2] where various protocols, mobility models and tools are described. In [3], the authors provide a detailed survey of mobility models available and study how mobility parameters affect the performance. Finally, in [4], [5], [6], the authors focus on the routing and data dissemination techniques, whereas in [7] researchers concentrate more on the vehicular context.

However, when starting research in this topic, many questions arise, which we were not able to answer with the help of existing surveys and tutorials. For example, *"Should we use Poisson or constant traffic?", "Is it OK to use a trivial radio transmission model?", "Which tool should I use and when?", "Which models are available and where?", "What is the impact of individual models on the performance of the simulation tool?" or "Which tool scales better?"*. Furthermore, we go beyond a traditional survey which simply lists and compares available options. We identify the gaps in the current state-of-the-art and propose possible future directions of the research in this area necessary to eventually reach our desired goal.

Therefore, we strongly believe that this survey gives a number of important guidelines for students as well as experienced researchers who want to explore the research area of OppNets. In the process of presenting these explanations, we have attempted to explain each aspect (e.g., concepts, models, simulators, etc.) relevant to OppNets in a simple manner so that a newcomer can follow and understand the content of this survey with relative ease.

This survey continues as follows: The next Section II defines an opportunistic network (OppNet) and narrows down the scope of the survey. Section III explores in detail available tools and methods for evaluating the performance of OppNets, in terms of simulation, theoretical analysis, testbeds and real deployments. Section IV presents the available tools (not models), able to simulate OppNets behavior. Section V explores the models relevant to simulating OppNets, such as mobility, traffic, radio propagation, and many more. Section VI addresses shortly the question of data propagation algorithms, which is typically the main research area in OppNets. However, a full overview of these algorithms and protocols is out of scope. Then, Section VII presents the evaluation metrics used for OppNets, before Section VIII presents a comparative study of the four simulation tools against each other in terms of their performance. Section IX summarises the findings of this survey and offers a list of best practices. Finally, Section X identifies gaps in current best practices and proposes some further concrete steps towards more scalable and realistic OppNets simulations.

II. Opportunistic Networks: Definition and Concepts

The term opportunistic networks has been used for sometimes different technologies. Here, we define Opportunistic Networks as the set of **applications and services running on end-user devices (e.g., smartphones, tablets and similar digital gadgets) that use *direct communication opportunities* among them to exchange information.**

A typical sample application is disaster alert, where an alert message (e.g. about a tsunami or earthquake) is sent immediately to all neighbouring nodes. In this way, all devices in the area can be easily reached, without the need of using a special application or paying for mobile data subscriptions. Furthermore, if the communication infrastructure, like 3G or 4G, is defective or not existing at all, the messages are still exchanged. The term "opportunistic" refers to the fact that communication opportunities with other devices are used as they come.

There are many similar use cases which exploit the OppNets concept, such as delay-tolerant networks (DTN), space communications, vehicle-to-vehicle communications and underwater communications. A good overview of such applications is provided by Mota et al. in [2]. Such networks and their relevant applications are close to OppNets, but their properties in terms of mobility and application requirements are quite different and thus we do not consider them here further.

	Destination-oriented (unicast or multicast)	**Destination-free**
User-driven	Announcements, messaging, chatting	Crowd sourcing, people as sensors, recommendations, public announcements
Automatic / event-based	Security and safety, newsletters, health alerts (allergies, smoke, etc.)	Public safety alert (floods, strong sun radiation etc.)
Automatic / periodic	Patient or baby monitoring, critical device status	Public reports (traffic, weather, health)

TABLE I: Various application examples used for motivating and evaluating OppNets.

An additional aspect related to the use cases we consider is the classification of the applications used by users. Table I gives the main properties and examples of the various application types. We basically want to differentiate between: (1) destination-oriented vs. destination-free models, and (2) user-driven vs. automatic applications. Destination-oriented applications assume that each message has a dedicated destination, being it a single user or a group of users. Destination-less applications do not know this from the beginning - they assume all users might or might not be interested in their data. It remains thus the goal of the propagation protocols to decide where to deliver them.

This classification, though simple, is very important since it dictates the characterisation of the data propagation to be used and the metrics to consider depending on the class of applications. Some protocols are designed to serve destination-less applications and will not perform well in destination-oriented applications and vice versa.

III. Performance Evaluation of Opportunistic Networks

Although this survey is focused on simulating methods, it is important to put it in context and to briefly describe other methods for evaluating OppNets. Performance evaluation can be defined as quantifying the service delivered by a computer or a communication system [8]. In order to perform an evaluation, we must define our evaluation goals and then the system, the load and the metrics.

First, we must define our goals. It may be obvious, but establishing clearly these evaluation goals is critical to this evaluation process, and will help in setting the type of evaluation, defining the system and the load, and selecting the metrics to analyse. For example, the goal can be the comparison of different data propagation protocols, so we must evaluate them in different scenarios and under different loads in order to determine which protocol performs better under what context. On the other hand, if our goal is designing a new real-world application the evaluation must be performed using a load similar to the expected one in the real deployment scenario.

The performance evaluation process consists of three main elements as shown in Figure 1. In detail, we have:

- **System**. The system, in our case, is the opportunistic network to be evaluated and comprises all its elements from hardware to software that can affect the performance of these networks, such as communication range, data transmission speed, buffer capacity, overhead, user behaviour, etc.
- **Load**. The load (or workload) represents the type and quantity of requests in a system, that in OppNets are primarily mobile nodes with devices that transmit messages. More precisely, we can distinguish between:

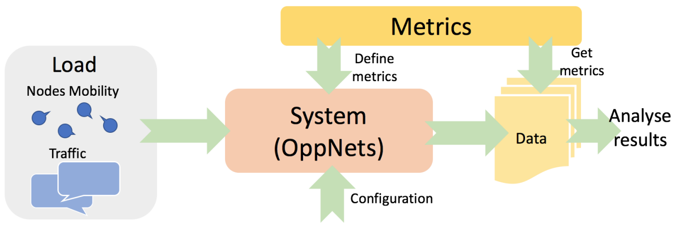


Fig. 1: Flow diagram of the process for the performance evaluation of OppNets

	Implementation (Real System)	Simulation	Analytical Model
System	Real	Depends on the simulator, from simple to complex	Simple. Strong assumptions and simplifications
Load	Real or synthetic (system benchmarking)	Configurable: From simple loads (synthetic mobility models and simple scenarios) to realistic loads (trace-based node mobility and traffic)	Very simple (contact rate, simple area, no spatial consideration, single message)
Metrics	Experiment-bound	Custom metrics. Depends on the simulator, can be similar to real systems.	Deterministic values for population processes, and distributions for Markov Chain models
Pros	The most realistic method.	Very flexible: Full control of workload, scenario model, metrics, etc. with different resolution levels. Cost and time efficient	Fastest
Cons	Very limited scenarios (it is not easy to evaluate the effect of varying parameters or real users). Very time and cost consuming	Can deviate significantly from real systems. Can be very time-consuming for very large scenarios	Oversimplified results, considering ideal conditions only. Models very complex to obtain

TABLE II: Summary of the available performance techniques for OppNets.

 - **Node mobility**: Defines the movement of the nodes, and their interactions are especially important for the evaluation of OppNets. Node movements are usually restricted to the **scenario** to evaluate. The complexity of these scenarios can range from simple restricted square areas to realistic map-based scenarios. This topic will be further discussed in Subsection V-A.
 - **Traffic load**: Refers to all the messages and network requests generated by the nodes. This traffic load can be as simple as the diffusion of one message to all the nodes in the network or to more complex messaging patterns resembling the use of social or messaging mobile applications. Subsection V-F details traffic models further.
- **Metrics**: Finally, we also need to define the metrics to evaluate the performance of OppNets. These metrics are mainly focused on the evaluation of the information diffusion in terms of the diffusion / dissemination time, delivery rate or number of hops. Regarding the nodes and network infrastructure utilisation, we can also obtain metrics about buffer occupation, network overhead, energy consumption, etc. When the performance evaluation is done, we usually have a large amount of data that we must process in order to get the desired metrics to proceed on the analysis of the results. Regarding the statistical nature of the metrics we can obtain only deterministic values (for example, a mean), or we can determine its stochastic distribution. This will be discussed in more detail in Section VII.

In terms of how to perform such evaluations, we have the following options:

- **Implementation** (real or testbed systems): Experiments performed using real scenarios and equipment can be very expensive and sometimes impossible to perform. Nevertheless, some complete evaluations in controlled places have been performed in [9], [10], [11]. Other experiments are focused on obtaining traces about node mobility that can be used to simulate these scenarios, as shown in Subsection V-A.
- **Simulation**: It is usually a simplified model of the system and the load implemented by software. A common approach is to combine a network simulation tool with realistic mobility traces, in order to reproduce the real dynamics and interactions of mobile nodes. Nevertheless, as we will see in this survey, simulations can be computationally intensive (especially for large or complex loads) and its parametrisation is not trivial.
- **Analytical Model**. It is a mathematical model of the system and the load. Analytical models can avoid some of the drawbacks of simulations and real testbeds providing a faster and broader performance evaluation, where we can identify the key mechanisms underlying the information diffusion. Analytical models usually require strong assumptions or simplifications about the system to evaluate and the load model considered is very simple. Usually the diffusion of a single message in a network of nodes with a given contact pattern is poisson distributed (considering that the inter-contact times distribution between pairs of nodes is exponentially distributed with a given contact rate) [12]. Despite that, it is not trivial to obtain such models. Two main classes of analytical models have been proposed for modeling such network dynamics: Deterministic models based on population or epidemic processes [13], [14], [15], [16], [17] and Markovian models [12], [13], [18], [19], [20], [21]. Markovian models, being stochastic, have the benefit of obtaining the probability distribution of the metrics at the cost of an increased computational cost when no closed-form expressions can be obtained.

We summarise the main performance evaluation methods in Table II. One of the most significant aspects of selecting the type of evaluating method is its precision and cost. In Figure 2 we can see the relation between these two factors. On one side, we have the cost, used as a broader term, comprising both the time cost and the economical cost (that can be very high in real experiments). On the other side, we have the accuracy of the obtained results compared to the real-world deployments. Analytical methods are very fast, but the results can be unrealistic. Simulation can obtain a precision very close to real testbeds using sophisticated simulation models, despite its computational cost. Thus, in this survey we focus on simulation models and how to obtain the most realistic and best performing scenarios.

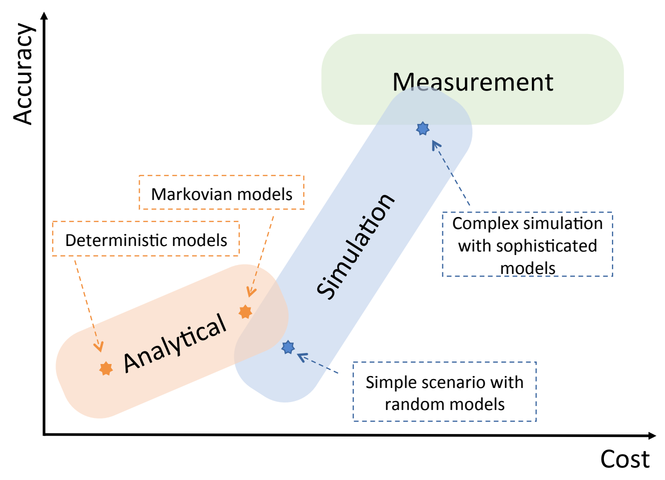


Fig. 2: Accuracy versus cost of the different methods of performance evaluation.

IV. Simulation Tools

In this section we first give an overview of existing simulation tools and platforms suitable for evaluating OppNets. While there exist many more simulation tools, we focused on free to use and widely accepted ones. We explore them in terms of platforms supported, available OppNets models and general advantages / disadvantages. We further explore them experimentally in Section VIII, where we provide a direct comparison.

A. *OMNeT++*

OMNeT++ is an extensible, object oriented, general purpose discrete event simulator (DES) written in C++. It was first published in late 1990 and its current version is 5.0. OMNeT++ has a free academic license and provides the mechanisms to build and simulate any type of network, from sickness dissemination to wireless networks. The OMNeT++ simulator itself only provides the building blocks to build nodes in networks and to model their behaviour and interactions. So called frameworks have emerged over the years, where pre-defined nodes with some special behaviours are available: Sensor nodes, IP-based nodes, satellites, cars, etc. INET is such a framework that implements the protocols of the Internet Protocol (IP) suite. By using INET, the user can easily construct a network of IP-based nodes and focus on parameters, applications, scenarios, etc. OMNeT++ is especially well suited for such broad studies, as it offers a user-friendly environment for automatic parameter studies.

1) Available Models for OppNets: INET itself is not very well suited directly for OppNets simulations, as it is focused on IP-based networks and their protocol stack. This stack is overcomplex for efficient OppNets simulations, as we will identify also later throughout this survey. Also, INET does not readily provide OppNets data dissemination protocols. We (some of the authors of this survey) have used INET as a basis for our OPS framework[1]. It offers the possibility to switch on and off the IP stack of INET and offers currently some simple data dissemination protocols. This is also the framework, which we use in this survey with the latest OMNeT++ 5.0 version (2016) and INET Framework version 3.4.0 (2016).

Additionally, there have been other frameworks and models developed for OppNets in OMNeT++. However, all these models have been developed for earlier versions of OMNeT++ (mainly 4.1-4.2, 2010-2013) and need large-scale refactoring. Such frameworks are OPPONET [22], [23], [24], [25] and OppSim [26].

2) Advantages: OMNeT++ has a very sophisticated user interface, with many possibilities to visualise and inspect various scenarios. The latest version also includes a sophisticated 2D and 3D graphics support to visualise the scenarios. It is also highly modular and separates clearly between node behaviour and node parameters, which makes it very easy to run large parameter studies. Its performance is also very good, it runs on all major operating systems and has very well maintained documentation, including user guides, tutorials, wiki page, etc. It can also be parallelised.

3) Limitations: OMNeT++ does not offer the possibility to transfer simulation models to real implementations, e.g. directly to Linux distributions.

B. *ns-3*

Network Simulator 3 (ns-3) is a discrete event simulator primarily focussed on simulating IP based networks with an emphasis on the network layer (layer 3) and the above layers of the protocol stack. The simulation scenarios can be created using C++ or Python, and are run as command line applications without a GUI. However, *NetAnim* is a tool shipped with ns-3 for visualizing node mobility during or after a simulation. ns-3 is licensed under the GPLv2 open source license. ns-3 was completely rewritten, although it can be considered the successor of ns-2. For this reason, ns-3 is incompatible with ns-2 and the simulation models have to be adapted to be used in ns-3. At present, ns-2 is only lightly maintained as the focus of the developers is on ns-3 which therefore should be used for new projects. The version used in the experiments is ns-3.27.

1) Available Models for OppNets: ns-3, similarly to OMNeT++, is a general purpose network simulator. In order to be used for OppNets simulations, it needs to be configured with the proper protocols at all levels—e.g. a OppNets data propagation protocol, mobility, traffic, link technology, etc. It is not possible currently to switch off the link technology model.

2) Advantages: All communication modules have been designed in a way that the interfaces can be matched easily to the standard Linux APIs. Thus, it is possible to easily move a simulation setup to the real world and vice versa. It is also possible to interact with existing real-world networks using virtual TAP (layer 2) devices, connecting simulated and real nodes.

3) Limitations: A good user interface with debugging and visualisation possibilities is clearly missing. Furthermore, the simulator structure is rather complex and not easy to parametrise or change even for experienced researchers.

[1]https://github.com/ComNets-Bremen/OPS

C. The One

The ONE (Opportunistic Network Environment) [27], is a simulation tool especially designed for OppNets. It was first developed in 2009, at Aalto University, and it is now maintained cooperatively by Aalto University and the Technische Universität München. The version used in this survey is 1.6.0, released in July 2016, and is available on GitHub[2]. It is written in Java.

1) Available Models for OppNets: The ONE is especially designed for OppNets and thus offers a wide variety of models, which also grows continuously. The ONE allows to generate node movements using different models, to reproduce message traffic and routing, cache handling and to visualise both mobility and message passing through its graphical user interface. It can also produce a variety of reports such as node movements to message passing and general statistics.

2) Advantages: In terms of available models for OppNets, the ONE is very sophisticated. It is easy to use and has a solid user community.

3) Limitations: The ONE does not perform well for large simulations, mainly because of the used programming language. In terms of models, it does not offer any link technology models nor radio propagation models.

D. Adyton

Adyton [28] is the newest tool in our survey, it was released in 2015. It is written in C++ and is also dedicated to simulating OppNets scenarios. It is only available for Linux and does not have a graphical user interface. It is freely available on Github[3].

1) Available Models for OppNets: Similar to the ONE, Adyton was especially designed for OppNets. It has a wide variety of data propagation models, it offers also buffer and cache size management. In terms of mobility, Adyton took a different path than the other simulators described here. Instead of actually moving nodes around, it pre-calculates contacts between them. This is a very processing-efficient way, but only few mobility traces are made available by the developers. To use other traces, they require pre-processing by an external tool.

2) Advantages: The performance of Adyton is very promising and seems to solve the main problem of the ONE. In terms of available models, it also offers a continuously growing variety of data propagation models.

3) Limitations: A good graphical user interface is missing for debugging and validation. Furthermore, the availability of mobility models needs to be improved to allow for more independent studies. It does not support link technologies, radio propagation models nor energy consumption models.

E. Further tools

Additionally to the simulation tools described above, there are also some other mobility-oriented tools, which are also very useful for simulating OppNets.

- **BonnMotion** is a tool than can create and analyze mobility scenarios and is most commonly used for the investigation of mobile ad hoc network characteristics [29]. BonnMotion's main objective is to create mobility traces using mobility models, as well as trace-based mobility scenarios (both described in Subsection V-A). Furthermore, the generated traces can be exported to a notable number of formats used by network simulators like for example *ns-3*, *Cooja*, *ONE*, *OMNeT++*, *GloMoSim / QualNet* etc.
- **Legion** is a commercial tool for generating mobility scenarios used by architects and civil engineers for urban and traffic planning. Using this mobility simulator, we can create our own scenarios (mainly buildings or city areas) and define the number of pedestrians / vehicles, its type of movement and destination. The underlying model is patented and validated with large traces of real pedestrian movement. The output generated by these tools can be used as input to an OppNets simulator after re-formatting. A good example is described in [30].
- **PedSim** (Pedestrian Simulator) [31] is similar to Legion, but free for use.
- **SUMO** (Simulation of Urban MObility) is also a mobility traffic generator written by the German Aerospace Center (DLR) [32]. The main focus is on the simulation of public transport, pedestrians and vehicles including speed limits, traffic lights etc. From these simulations, mobility traces can be exported for trace-driven mobility in OppNets simulations.

Summing up, the presented mobility-generating tools can be very helpful in relieving the OppNets simulator from some of its tasks and to produce very sophisticated large-scale mobility traces.

V. Simulation Models

The evaluation of opportunistic networks involves various different elements, that combined, contribute to make the simulation results more realistic. In this section, we focus on the most relevant of these elements, presenting the available models and their properties. Without claiming to provide an exhaustive survey, the next subsections give an executive overview of the state-of-the-art models for mobility, user behaviour, radio transmission and interference, battery and energy consumption, traffic generation and link technologies.

Figure 3 illustrates the main purpose of the individual models explored here. The models are presented from the perspective of a single simulated node.

A. Mobility Models

This subsection focuses on how the movement of the users is modelled in simulators. The mobility models are designed to describe the movement patterns using the user location and the velocity change over time. Since mobility provides opportunities for contacts and therefore for communication, the understanding of the human movement patterns is considered an essential element when evaluating a protocol performance. It is desirable for mobility models to emulate the movement

[2]https://akeranen.github.io/the-one/

[3]https://npapanik.github.io/Adyton/

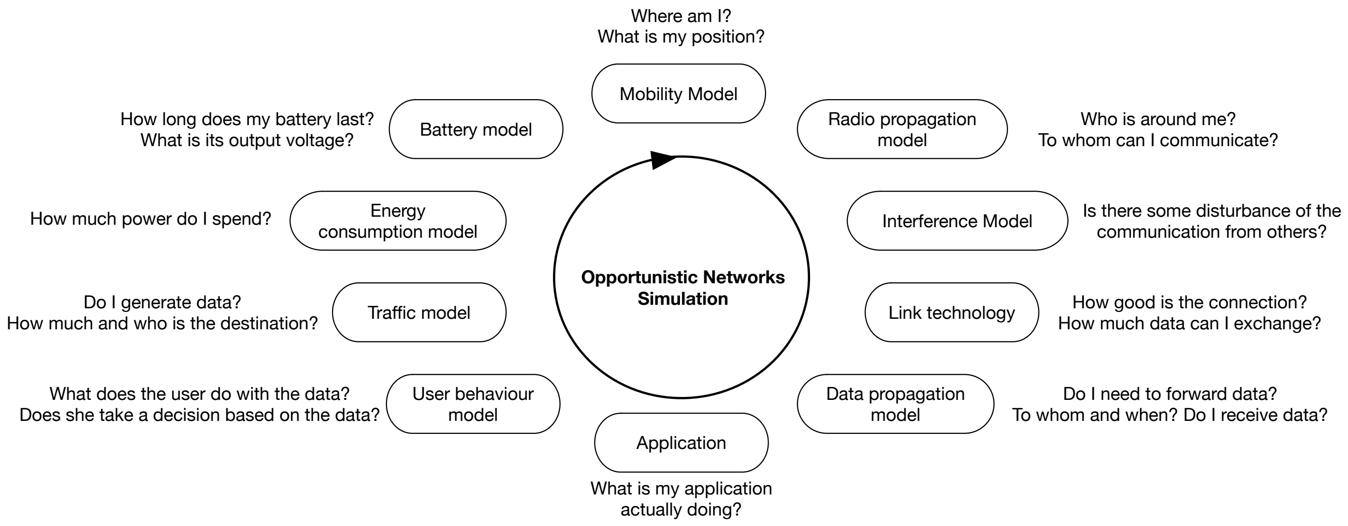

Where am I?
What is my position?
Mobility Model
How long does my battery last?
What is its output voltage?
Battery model
Radio propagation model
Who is around me?
To whom can I communicate?
How much power do I spend?
Energy consumption model
Opportunistic Networks Simulation
Interference Model
Is there some disturbance of the communication from others?
Do I generate data?
How much and who is the destination?
Traffic model
Link technology
How good is the connection?
How much data can I exchange?
What does the user do with the data?
Does she take a decision based on the data?
User behaviour model
Application
Data propagation model
Do I need to forward data?
To whom and when? Do I receive data?
What is my application actually doing?

Fig. 3: The OppNets simulation models explored in this survey from the perspective of a single simulated node.

pattern of targeted real life applications in the most realistic way possible. The general structure of a mobility algorithm typically follows the steps indicated in Figure 4.

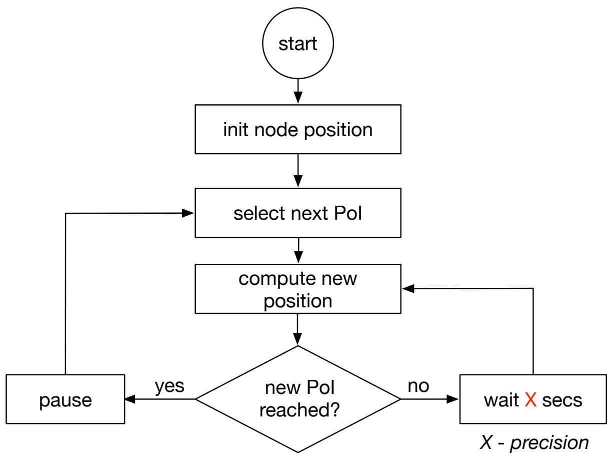


Fig. 4: The general structure of a mobility algorithm.

Mobility models can be categorised mainly into three types based on how the next *Point of Interest (PoI)* is selected.

a) ***Random Mobility Models:*** Models with random movements employ stochastic movement patterns to move a node within a given area. The most frequently used mobility model in many simulations is the Random Way Point (RWP) model due to its simplicity of implementation. The mobile node selects a destination randomly and moves towards it with a randomly selected speed. The other extended mobility models based on RWP can further be characterised as mobility models with *temporal dependency* (i.e., the next movement will depend on the previous history), with *spatial dependency* (i.e., tend to change the movement in a correlated manner) and with *geographic restriction* (i.e., movement is restricted by streets or obstacles). Almost all the available random mobility models are discussed in detail in [33] and [34].

These models are not very useful for evaluating OppNets as the mobility patterns used in OppNets should mimic human behaviour where the movement of a user is also influenced by their daily activities (at home, school or work), by their means of transportation (walking, bike, bus, etc.) or by the behaviour of other users in their neighbourhood / environment. Many of these activities are related to user behaviour and social relationships in addition to the above mentioned three characteristics. We will show these properties and their impact on OppNets performance also experimentally at the end of this subsection.

b) ***Real Mobility traces:*** There is a large collection of datasets obtained from the observation of nodes mobility in real scenarios. These traces provide accurate information, especially when the trace is being collected among a large number of participants and for a longer period of time. We can find two classes of traces: Contact based and location based. A contact based trace is obtained by measuring the times when contacts between pairs of nodes occur for a given time interval. Well known datasets such as *Infocom* [35], *Cambridge* [36], *Milano* [37], *MIT (or Reality)* [38] among others, have been extensively used, and their statistical properties have been studied in depth [39], [40]. The simplicity of these traces allows the analytical evaluation and simple simulation of OppNets.

Their main drawback is that they do not allow to simulate the impact of communication protocols. Therefore, to evaluate these aspects, we must use location based traces. These traces are the result of obtaining the location of the nodes (mainly GPS coordinates) periodically (or when they move). There are several types of traces, such as the taxis mobility in Shanghai [41], or the student mobility in the National Chengchi University campus [42], among others. The Crawdad repository [43] contains most of the publicly available traces, and it can be considered the first place to look for the required traces for simulations.

Though the trace based mobility models represent very realistic movement patterns, they may not be very useful when it comes to new scenarios where the collection of traces has not been performed yet. Furthermore, they are fixed, i.e., they cannot be scaled up or extended in time. Also, they cannot react to any changes from the user's perspective, e.g. modelling the user running away from an unexpected dangerous situation. They are also expensive to use in simulations, because the information about the next points has to be read from external files. We will demonstrate this at the end of this subsection.

*c) **Hybrid Mobility Models**:* This category shows a combination of the first two described above. In these models, some parameters (e.g., frequency of user movements w.r.t. locations) for a random model are derived based on a collection of traces or based on user experience. For example, the Small Worlds In Motion (SWIM) model [44] is based on the assumption that a user either selects as next PoI a location close to her home or a very popular location (e.g. a popular restaurant in town). Thus, hybrid models attempt to model real properties of human movements by taking into account "common sense" assumptions, but also statistics from traces. Hybrid models achieve better performance and scalability compared to real mobility traces.

Many studies are available that aim at providing realistic mobility models based on social relationship (e.g., users periodic travels over short distances, movements coordinated by social relationships, etc.) between users and location preferences. Some of these models are: TLW (Truncated Levy Walk) [45], SLAW (Self-similar Least Action Walk) [46], SMOOTH [47], SWIM [44], HCMM [48], WDM (Working Day Model) [49], TVC (Time Variant Community model) [50], HHW [51] and SOLAR (Sociological Orbit models) [52].

Differently from other surveys (like [53], [54]) in Table III we offer a comparison among these models according to the human mobility properties they model, a more convenient approach when dealing with OppNets. We have organised the table according to the steps shown in Figure 4.

We first explored how the PoIs are identified answering to the following questions: Are those coordinates random? Are the coordinates taken from maps / traces, but we do not have any context information about them (e.g., a cafe or a concert hall)? Or, is all this information available?

Next, we explored how the next PoI are selected. Is this done purely randomly, or do we consider some human behaviour properties? For example, it could depend on the time-of-the-day, on previously visited locations (we tend to re-visit locations), but also on how much somebody likes going out.

Then we considered the pause time in a particular PoI. Again, random time can be assumed or the time spent in a location could depend on its specific type (e.g., only 10-30 minutes in a supermarket, but 2-3 hours in a cinema).

Finally, we considered how the user moves between PoIs. Is it on a straight line with constant speed or is it more realistic, e.g. sometimes taking the bus, sometimes walking and or even a combination of those? It is interesting to note the difference between the "Map-driven" line with respect to the "Trace-driven" and "Depends on real traffic". In defining the exact path, the last two are more precise by considering the fact that daytime traffic conditions could be different (using information provided for example by Google Traffic) or, even more realistic, it could be taken from real traces with real people (who in general do not behave predictably).

*d) **Performance comparison**:* To compare the performance of the different mobility models in terms of the simulation time needed (i.e. the wall clock time), but also in terms of other OppNets metrics, we designed a set of experiments whose results are presented in Table IV.

For this, we took a set of real traces from San Francisco taxis [55] (also referred to as "*SFO trace*"; more details about these traces are given in Section VIII) and evaluated them statistically in terms of covered area, number of nodes, duration of trace, mean speed of the nodes, etc. Then we parametrised a random model (RWP) and a hybrid model (SWIM) as closely as possible to the real trace. In order to keep the simulation duration viable, we used only 50 nodes.

From these results we can conclude that real mobility traces are very time consuming to use as the simulation took more than 20 times longer to complete compared to the corresponding simulation using a random mobility model. The hybrid model, in this case SWIM, positions itself between the two extremes. However, when comparing using the other specific OppNets performance metrics, we see that RWP delivers completely different results than the real trace, which we assume to be the most correct one. Thus, its behaviour is obviously too far away from real movements and should not be used for OppNets simulations. SWIM delivers results similar to the real trace results in terms of delivery rate and delay, but looking into the contacts information it becomes obvious that it is quite different in its behaviour. This is however due to the fact that SWIM is modelling walking users behaviour, while the real trace delivers information about users on vehicles (taxis).

Take-Away Message 1:
Random mobility models are not well suited for simulating OppNets. One should preferably use sophisticated hybrid models, but if these models are inadequate for the scenarios being evaluated, consider using real traces.

What we miss in all available models is the possibility to **react** to some message, such as run away from fire, go to a party, etc. We will discuss this in Section X, where we summarise all future ideas and lacking functionality.

B. Radio Propagation Models

Radio propagation models simulate how the wireless signal propagates through different environments, over different distances and obstacles. Since all opportunistic networks are typically wireless, these models are very relevant. At the same time, there is a tradeoff between the precision of the radio propagation models against simulation time (i.e. wall clock time) and scalability of simulations. A more sophisticated radio propagation model will more correctly identify impossible or interrupted transmissions. However, when we are interested in the general behaviour of very large OppNets, a sophisticated radio propagation model would probably not significantly contribute to the understanding or evaluation of the system, while using too many resources.

Property / Model	**RWP**	**SLAW**	**SMOOTH**	**TLW**	**HCMM**	**TVC**	**SOLAR**	**SWIM**	**HHW**	**HWDM**
Distributions of PoIs (Point of Interests) are										
Random	✓	-	-	✓	✓	-	✓	-	✓	✓
Based on Truncated Power Law (TPL) distribution	-	-	✓	-	-	-	-	-	-	-
Based on traces / maps – without context	-	✓	-	-	-	✓	-	✓	-	✓
Based on traces / maps – with context (e.g. home, work, supermarket)	-	✓	✓	-	-	✓	✓	✓	✓	-
Next PoI selection based on...										
Random	✓	-	-	✓	-	-	✓	-	✓	✓
Based on previously visited PoI	-	-	✓	-	-	-	-	-	-	-
Distance from last location	-	✓	-	-	-	-	-	✓	-	-
Time of day / week	-	-	-	-	-	-	-	-	✓	✓
Based on social relationship	-	-	-	-	✓	-	-	-	-	-
Traces	-	-	✓	-	-	✓	-	✓	-	-
Vicinity to home location	-	-	-	-	-	-	-	✓	-	-
Popularity of places in general	-	-	-	-	-	-	✓	✓	✓	-
Popularity among friends	-	-	-	-	-	✓	-	-	✓	-
Move with a community (e.g. friends)	-	-	-	-	✓	-	-	-	-	✓
Differentiation between communities (e.g. friends, family, collogues)	-	-	-	-	-	-	-	-	✓	-
Personal preferences / contextual information (visit often cinemas but no restaurants)	-	-	-	-	-	✓	✓	-	-	-
Level of personal mobility (some people move a lot, some doesn't)	-	-	-	-	-	-	✓	-	-	-
Stay in a PoI for...										
Random time	✓	-	-	✓	✓	✓	✓	-	-	✓[a]
Based on TPL	-	-	✓	-	-	-	-	-	-	-
Trace-driven	-	✓	✓	-	-	-	-	✓	-	-
Course grained context (e.g. home: 16 hrs, work: 8 hrs)	-	-	-	-	-	-	-	-	✓	✓
Fine grained context (e.g. supermarket: 30 min, bar: 1 hr, concert: 3 hrs)	-	-	-	-	-	-	-	-	-	-
Type of movements between PoIs...										
In a straight line with the same speed	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
With different modes of transportation (different speeds)	-	-	-	-	-	-	-	-	-	✓
Map-driven (e.g. Navigator like)	-	-	-	-	-	-	-	-	-	✓
Trace-driven (e.g. GPS traces)	-	-	-	-	-	-	-	-	-	-
Depends on real draffic – day / night (Google Maps like)	-	-	-	-	-	-	-	-	-	-
Group-based (e.g. move together with friends)	-	-	-	-	✓	-	-	-	-	✓
Allow scheduling / planning										
Handle influences of user behaviour	-	-	-	-	-	-	-	-	-	-

[a] defined in settings

TABLE III: A comparison between the different hybrid mobility models for OppNets (RWP is used here only as a reference).

Model	RWP	SWIM	SFO trace
Simulation Time	4 min	59 min	109 min
Memory used	74 MB	86 MB	127 MB
Average delivery rate	3 %	96%	92 %
Average delivery delay	20.6 h	16.25 h	13.16 h
Total number of contacts	190	46,752	155,757
Average contact duration	117.14 sec	150.12 sec	584.39 sec

TABLE IV: A comparison between a simple random, real and hybrid mobility models in OMNeT++. The network consists of 50 nodes.

Similarly to mobility models, radio propagation models (radio models for short) can be divided into trace-based and synthetic ones, plus some hybrid variants.

1) Synthetic Radio Propagation Models: Synthetic radio models attempt to mathematically describe the propagation of the radio signal through the environment and to predict its strength at the receiver. In general, they calculate the so called **Path Loss**, which is the loss of strength between the sender and the receiver as:

$$P_R = P_T + G_T + G_R - PL - L_T - L_R \tag{1}$$

where PL is the path loss, P_T and P_R are the transmitted and the received power, G_T and G_R are the gains of the transmitting and the receiving antennas, and L_T and L_R are the losses of the transmitting and receiving systems (e.g. coax, connectors, etc.). All parameters are in dB. This equation is often called also the **link budget** equation [56]. Very often, the system losses get neglected in simulation models and only the antenna gains are used.

The path loss is due to various factors, such as:

- Spreading: When the signal is emitted from the sender, it does not travel in a line, but spreads uniformly around it. This happens of course also in vacuum. The signal gets weaker with distance according to the inverse square law.
- Attenuation: When the signal looses some of its power due to interaction with the medium. In perfect vacuum, the attenuation should be zero, but perfect vacuum does not exist even in space.
- Fading: The attenuation itself changes over time and / or distance.
- Doppler effect: When the sender is mobile, the signal gets stretched or contracted. This impacts the frequency of the signal as it travels and is a special case of fading.
- Shadowing / Superposition: When the signal interacts with other electromagnetic signals or reflections of itself. This is often also referred to as interference. For OppNets, relevant interference sources are all devices working in the same frequency bands as OppNets link technologies (mostly 2.4 GHz) and some other more unexpected devices, such as microwaves.

Most often, models use a more or less sophisticated version of the following equation:

$$PL(d)[dB] = \overline{PL}(d)[dB] + X_\sigma[dB] \tag{2}$$

where $\overline{PL}(d)$ is the mean path loss for some distance d from the transmitter and X_σ is a probability distribution, typically Gaussian with zero mean and standard deviation σ. This distribution function models the above mentioned factors as *noise* in the system.

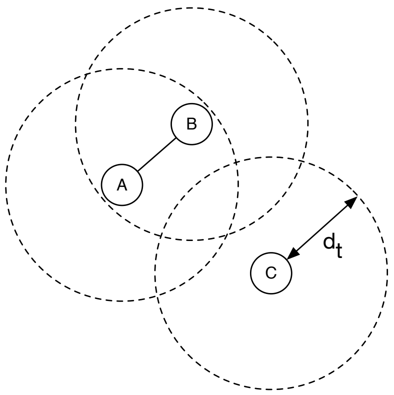


Fig. 5: The Unit Disk Graph model.

Models target usually either the calculation of the path loss (deterministic models) or the calculation of the noise (non-deterministic models). Reality is mirrored best with combined methods. However, most of the path loss models can be easily combined with the noise models, as it is a simple sum (see Equation (2)).

The simplest model is the so called **Unit Disk Graph (UDG)**. The idea is shown in Figure 5: Two nodes in a network are connected to each other if the distance between them is below some threshold d_t. In this case, the path loss is assumed to be zero. Otherwise, they cannot communicate to each other at all and their path loss is considered to be some maximum value, which makes the transmission impossible. Often this model is also called the **Binary model**, since it either allows for perfect communication or none at all. Obviously, this model is not very realistic, but very efficient.

Looking back into our Equation (2), UDG can be expressed as:

$$PL(d)[dB] = \begin{cases} 0, & \text{if } d \leq d_t \\ \infty, & \text{otherwise} \end{cases} \tag{3}$$

where the noise is always 0. This can be easily extended to include also noise and to make the borders of the unit disk more fuzzy. The noise distribution can be Gaussian with zero mean and some standard deviation σ, but it can also be some other distribution.

Typically, the noise is considered to be independent spatially and timely. With other words, every time you need to calculate the path loss at some destination point, you do so independently of what you calculate at other positions (spatial independence) or for the same position at a different time (timely independence). However, many researchers have shown that this is not the behaviour of real world wireless links, so both dimensions (time and position) are highly relevant.

Some researchers have considered the spatial dependence in [57]. They have shown that the noise at a particular angle is dependent on the noise at neighbouring angles, which is defined by Equation (4). This model is referred to as the **Radio Irregularity Model (RIM)**:

$$\begin{gathered} PL_d^{DOI} = PL_d \cdot K_i, \\ \text{where} \\ K_i = \begin{cases} 1, & \text{if } i = 0 \\ K_{i-1} \pm rand \cdot DOI, & \text{if } 0 < i < 360 \end{cases} \end{gathered} \tag{4}$$

In other words, instead of adding noise to the path loss independently of the position of the destination, it is calculated dependent on the noise for destinations close by. The DOI (degree of irregularity) is a parameter of how strong the noise is. The larger the DOI, the more irregular is the transmission area around the sender. A DOI of 0 results in a UDG model. The random distribution applied is the Weibull distribution, which is often used to model natural phenomena [57]. Any PL model can be easily plugged into Equation 4.

The second problem, the timely dependence, has been described in [58]. There, the authors have explored the property of link burstiness. It means that if the communication between two nodes fails at some point, the probability that it will fail again grows and the opposite. The authors have defined for this a parameter, which they call the **beta (β) factor**. The

β factor represents this dependency: The larger β is, the more dependent is the transmission result on previous ones. The paper did not describe an updated radio transmission simulation model but it can be easily done analogous to Equation (4) and is one of our identified future directions (see Section X).

On the other side, many efforts have been invested in the deterministic part of the path loss $\overline{PL}$. Here, many models exist to reflect the behaviour of the signal in various environments, such as outdoor free space, outdoor city, indoor, etc. Again, the more realistic the model is, the larger its overhead in terms of processing and memory usage. For simplicity, we omit the mean sign over the path loss in the next equations.

The simplest model is the so called **Free Space Path Loss (FSPL)** or **Friis Equation**.

$$PL(d)[dB] = -10 \cdot log\left[\frac{G_T G_R \lambda^2}{(4\pi)^2 d^2}\right] \tag{5}$$

where λ is the signal wavelength in meters and d is the distance between transmitter and receiver.

The assumption here is that the receiver and the transmitter are in line of sight of each other and that there are no objects around to cause reflection or diffraction. It is important to understand that this equation is not defined for $d = 0$ and it only holds for larger distances, the so-called far field ($d \gg \lambda$).

Obviously, the free space path loss increases with a factor of square of the distance. If we have already calculated or measured the path loss at some distance d_0, we can easily calculate the path loss at some other distance $d > d_0$ without having all parameters of the system, such as antenna gains:

$$P_R(d) = P_R(d_0)\left(\frac{d}{d_0}\right)^2 \tag{6}$$

$$PL(d)[dB] = PL(d_0)[dB] + 10 \cdot 2 \cdot log\left(\frac{d}{d_0}\right) \tag{7}$$

Since the expression in dB is logarithmic, this model is also called **Log Distance Path Loss**. Note that we have written $10 \cdot 2$ in the equation, instead of simply 20. This is due to the next extension to the model, which reflects how well a signal penetrates a particular medium. Until now, we have considered vacuum, which has a **Path Loss Exponent** of 2. Other environments, such as a grocery store, has a lower path loss exponent, typically around 1.8, while a crowded office environment can have a path loss exponent of 3.0. The generalised equation is then:

$$PL(d)[dB] = PL(d_0)[dB] + 10 \cdot n \cdot log\left(\frac{d}{d_0}\right) \tag{8}$$

where n is the path loss exponent.

Once the received signal strength has been calculated at the receiver's side, the propagation model can proceed with comparing it with the sensitivity threshold of the simulated radio transceiver. If the received power was below the threshold, the transmission has failed. Thus, in some way, these models can be considered as calculating a more complex shape for the transmission area than UDG.

The models can also be applied at different precision levels. If you apply the model to packet level, the result is either a successfully received packet or a failed packet. If you apply them on a bit level, then the higher communication layers can attempt to repair individually failed bits. Again, the tradeoff is between precision and processing overhead.

The models presented here are only a few and are rather very simple. There are many more models, attempting to catch also the Doppler effect [59], indoor environments with walls [60], and many more. However, more sophisticated ones are too expensive in terms of performance to be used for large-scale OppNets simulations and thus, out of scope of this survey. The interested reader can turn to [56] or [61].

Finally, **Ray Tracing** is a technique from the computer graphics domain, where a scene is rendered with rays, which propagate through the space and get reflected / diffracted / scattered / etc. In fact, it is a very precise technique, which delivers extremely realistic results (as we can see from 3D rendered movies). However, it requires a very precise description of the scene itself—individual objects and their form, structure, materials, etc. Of course, this can be done also for network simulation scenarios, but requires a level of detail, which we typically cannot provide. Additionally, the rendering process is very slow. Nevertheless, it has been applied to network simulation and even some optimisations have been achieved [62], [63].

2) Trace-Based Radio Propagation Models: Another possibility to simulate the radio propagation for OppNets is to gather real data from real experiments and re-run it in a simulation tool. In the context of OppNets, this can be done in two different ways: Gather wireless link quality properties or abstract away and gather only the so called contact data.

a) Wireless Link Traces: The general idea is the following: You prepare an experiment with some nodes, which use the communication and link technologies relevant to your scenarios. For example, you could take 10 smartphones communicating over Bluetooth. You implement an application, which simply broadcasts a packet very often (e.g. every 100 ms) and records the received pings from other neighbours together with their wireless quality properties, such as Received Signal Strength Indicator (RSSI) or Link Quality Indicator (LQI).

Later in the simulation, the algorithm works in reverse way: When a node needs to send a packet to another node, the simulator looks up in the trace file for that transmission and checks whether it was successful or not.

This way of implementing trace based radio propagation models is mostly used for wireless sensor networks (WSNs) [64], [65]. It allows for pretty good precision, but it is obviously not scalable, as it can only simulate as many nodes as the original experiment included and only in the original experimental setting. Recording such traces is also not trivial. For OppNets, the abstraction to contact times is better suited.

b) Contact times: The idea is to abstract away from individual links and their properties and to record simply when was who connected to whom. We have already presented them in the scope of mobility traces in Subsection V-A. For example, in the above scenario with 10 smartphones connected by Blue-

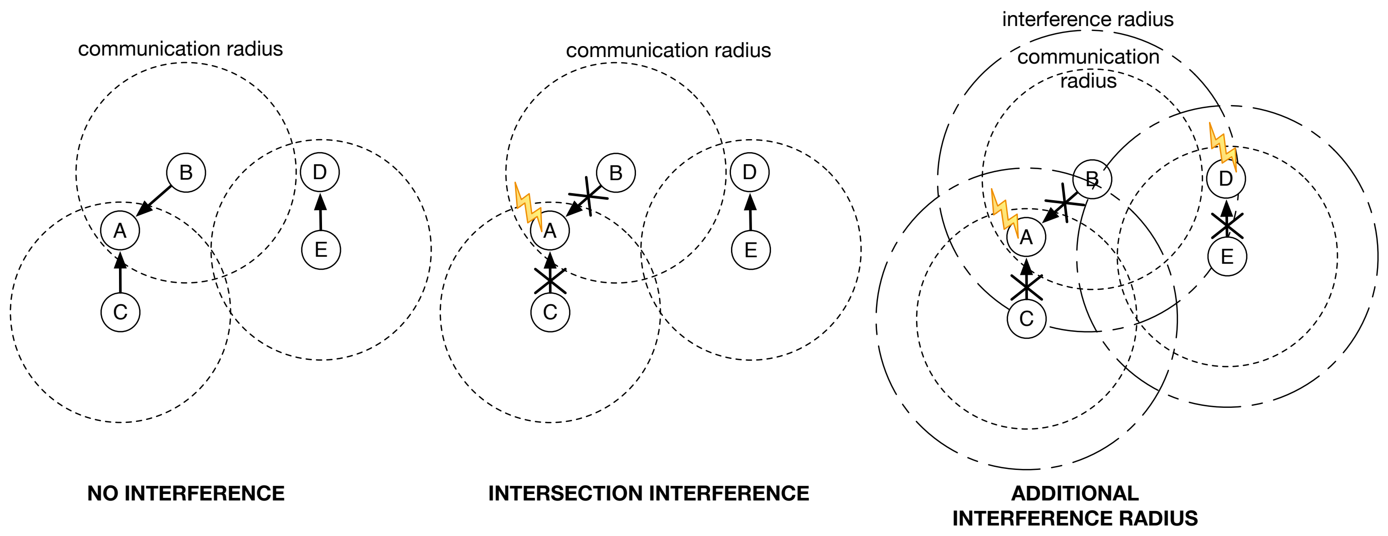


Fig. 6: Various interference models for the UDG radio propagation model.

tooth, you will implement an application, which simply checks which other devices are around every predefined interval (for example, every second) and records this information. Later you can process the information to identify how long was a particular contact between two devices. Since OppNets require exactly this type of information, such models are very well suited and very processing-efficient. They can also be used to analyse the properties of the individual scenarios [66] to abstract further away and implement a hybrid model, described next.

3) Hybrid Radio Propagation Models: Analysing trace files leads us to the next abstraction for radio propagation modelling, where a simple mathematical model is used, but with parameters coming from real experiments. For example, the WSN TOSSIM simulator [67] uses a graph model, where each link is represented by an edge in the graph and assigned a bit error rate in both directions. The graph itself can be easily calculated with the help of the UDG model, as described above. The bit error rate comes from analysing real experiments, such as in [64].

This model has been used widely for simulating WSNs, but is not very popular in OppNets probably because it has a higher overhead. We discuss it again in Section X.

4) Performance Evaluation: In order to evaluate the performance of a more realistic radio propagation model against UDG, we have designed an experiment with ns-3. The results are presented in Table V. We use a Nakagami fading model, as readily available in ns-3, to compare against UDG. Both are configured to have an average transmission radius of 50 meters. It can be seen that using a more sophisticated radio propagation model results in better overall network performance, especially in terms of delivery delay. The Nakagami model was also configured to a 50 meters transmission radius, but its behaviour allows for communication also outside this area. Thus, this model increases the communication opportunities between nodes, improving the general network performance. However, this comes at the cost of longer simulation times and increased RAM usage. It remains open which of the two models is actually closer to reality.

Take-Away Message 2:
Sophisticated radio propagation models have a significant impact on the general network performance, but result in longer simulation durations.

C. Interference Models

Interference is generally defined as noise in wireless communications. It can come from natural sources, like interstellar radiation or lightning, or from other devices operating in the same or close frequencies. Here we explore the inter-device interference, which has by far the larger impact on wireless propagation.

The above described radio propagation models all model individual connections, as if those are always taking place alone (no other connections running). This is especially true for all synthetic models and for wireless link traces. Contact time traces can be considered as already taking care of interference, because the effect is inherently included. For example, when 10 smartphones are continuously sending and receiving ping packets, they do so at the same time and in a real environment with many other smartphones around.

Of course, the first interference model is no model at all, which is a very often used option. This model is also fine to use with large-scale environments and few connections going on at the same time, because the interference impact will not be very large anyway. However, it is not a good idea for many connections and dense experiments with many nodes.

The UDG model can be extended to include interference in two different ways: Intersection based or with an additional interference radius. The difference is presented in Figure 6. The left-most picture illustrates the case where three transmissions are going on and all three are successful, as no interference between them is assumed. The center picture considers interference, if two or more transmission radii are

	50 nodes		100 nodes	
	UDG	Nakagami	UDG	Nakagami
Simulation Time	10.24 h	16.71 h	83.85 h	95 h
Memory used	350 MB	400 MB	840 MB	1300 MB
Average delivery rate	98%	99%	97%	97%
Average delivery delay	2.1 h	1.49 h	1.52 h	1.22 h

TABLE V: A comparison between a simple UDG and a Nakagami fading model in ns-3.

overlapping at one receiver (intersection model). In this case, both transmissions from nodes B and C to node A fail, because they overlap at node A and cancel each other. The third option is shown at the right, where an additional interference radius is shown. It is larger than the transmission radius and assumes that a transmission might not be possible at that distance, but the signal still interferes with other connections. In this case, a transmission is considered failed if one or more transmission or interference radii overlap at a receiver.

The above presented models can be also easily used with other, non-UDG radio propagation models, such a Friis equation or RIM. There, the overlapping areas have simply different shapes than circular. When using a non-UDG model, we can also use the calculated received power at the receiver directly to decide whether a transmission is successful or not. If the difference between two transmissions is large enough, the stronger signal is received successfully. If the difference is too small, both transmissions fail [68]. Note that this model is not that different from the radius-based one, described above.

For wireless trace models, we can also easily implement interference models, where simultaneous transmissions to the same node get cancelled.

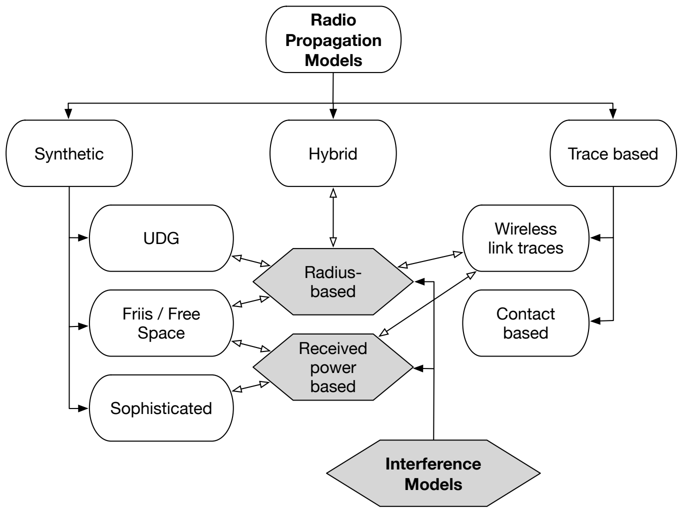


Fig. 7: Taxonomy of Radio Propagation and Interference Models.

Figure 7 presents a taxonomy of the here presented radio propagation and interference models. It is not exhaustive, but it provides a visualisation of the available options and their main properties.

D. Link Technologies and Models

The link layer as referred to in simulations and wireless networks describes a combination of the data link and the physical layer of the OSI model. The objective of the link layer is to adapt the data from the higher layers (i.e., application data) for the used media (i.e., wireless channel). These aspects are important when simulating OppNets since they impact the delivery rate and delays through buffer management, retransmissions, connections, etc.

The most well known link technology which can be used freely, i.e., without being bound to any operator, is the WiFi technology, which is based on the IEEE 802.11 standard [69], [70] defined by the Institute of Electrical and Electronics Engineers (IEEE). Various specifications belong to this technology, the most widely known being 802.11n and 802.11ac, which are currently available in most of the modern handheld devices. But several other specifications exist which are relevant for OppNets. For example, IEEE 802.11s is a standard specifically designed for mesh networks where the participating nodes create a layer 2 mesh network which can also be described as a kind of *MAC-relaying* infrastructure. IEEE 802.11ah is a new upcoming standard optimised for energy restricted devices such as sensor nodes and machine-to-machine communication. The focus of IEEE 802.11ah is on supporting a larger number of nodes (up to several thousands) and a lower energy consumption compared to the common WiFi standard. Relevant to OppNets is *WiFi direct*, a technology that enables WiFi devices to connect directly, allowing an easy pairing of devices for short-term data transmission without the need of an infrastructure. It basically uses the ability of modern handheld devices to become an access point. This technology was developed to overcome the practical limitations of the *Ad-Hoc mode*, originally defined in the IEEE 802.11 standard with the same objectives. From the link layer point of view, WiFi direct based networks behave as classical WiFi networks.

Another widely used technology is Bluetooth in all its variants [71]. Especially with version 4, known as *Bluetooth Low Energy* (BLE) and with the new features introduced with Bluetooth 5.0, it offers energy efficient functionalities for wireless communication in constrained environments and OppNets.

The main advantage of Bluetooth and WiFi is their wide availability on a variety of end-user devices ranging from smartphone to IoT devices. However, some other specialised technologies exist which should also be considered for OppNets, since they may become relevant for some specific applications scenarios and environments, such as national park monitoring or data gathering in less densely population areas.

First of all, referring to the very active area of Wireless

Sensor Networks and *Smart Things*, we would like to highlight IEEE 802.15.4 [72]. Several higher level protocols like *ZigBee*, *WirelessHART* and *Thread* are based on this standard. Its main advantage is the low energy consumption and the optimisation for low power devices.

Currently, LoRa [73] and Sigfox [74], two standards that fall into the category of *LPWAN* (Low-power Wide-area network), are becoming more and more widely used in the area of IoT as they claim to have a very long transmission range, up to several kilometres. The low-bandwidth offered, the possible scalability issues and the actual benefits of such long transmission ranges are still being observed and tested, but in the future they may become a good option for implementing specific OppNets-based services.

Some simulators offer realistic implementations of the above described link technologies. WiFi is well supported by OMNeT++ (using INET) and ns-3. Both can simulate Ad-Hoc and mesh networks as well as the classical infrastructure based ones. The new IEEE 802.11ah standard is not yet officially being supported by both simulators. IEEE 802.15.4 is also supported by OMNeT++ and ns-3 where the focus of ns-3 is more on the network layer and IP-based networks (6LoWPAN) whereas OMNeT++ simulates down to the physical layer. Bluetooth is neither directly supported by OMNeT++ nor by ns-3. For both simulators, several community driven projects with different qualities exist where the variety of OMNeT++ based ones is higher. Thus, implementing those link technologies is possible and typically requires only good understanding of the standard itself.

The impossibility to completely and exactly model any specific real scenario where various details should be considered, like the presence of physical obstacles, the mobility of the nodes or the density of the nodes, motivates the use of a so called "ideal link layer" to simulate high-level data propagation OppNets protocols and algorithms. Basically, most of the simulators, including those discussed in this survey, assume that if two nodes are in *contact*, i.e., if their distance is less than a certain threshold, they can exchange messages. Typically the presence of a contact is re-validated every x seconds. The ONE and Adyton only have ideal link models. ns-3 has only real link technologies and the implementation of an ideal one is not trivial. OMNeT++ has the potential of offering both environments, but it requires a complete new implementation of the OppNets data propagation protocols; in this survey, we used the ideal link layer for OMNeT++.

As we will see in Section VIII, the results with ideal and real link models are very much comparable even among different simulators. In conclusion, we can say that the use of such a conceptual link makes simulations more computationally efficient and yields results that allow comparing alternative solutions among them.

E. User Behavior Models

A user behaviour model is answering the question: What happens after the user receives the data? Typical application models cover possibilities such as delete the data, store the data and sometimes like / dislike the data. However, a real user behaviour model goes beyond this and offers options like decide to go somewhere (e.g., after reception of a concert notification for tomorrow), cancel an already scheduled movement (e.g., after receiving the weather forecast) or, very importantly, run away from danger. Other behaviour models include answering a message, creating a message in response to some external event (e.g., sending a message to all friends to cancel the biking tour because of bad weather forecast) and many more.

Why is the user behaviour model important? Because it changes both the mobility and the traffic generation in the simulation. To understand this better, let us explore the example of receiving fire alarms as depicted in Figure 8. The initial situation is the center of a city, with people moving around. Suddenly, there is a fire alarm in one of the buildings. This alarm propagates quickly via OppNets and all users around. However, a state-of-the-art simulation does not change the behaviour of people moving around—they will continue moving as if nothing has happened. The reality looks differently: The emergency service workers would run to the place of fire to organise the evacuation, while visitors run away quickly out of the danger area and gather around it.

There is one common property of all user behaviour models: They change at least the mobility of the simulated user. As we already identified in Subsection V-A, there are no mobility models currently, which support this kind of scheduling and planning. Thus, also user behaviour models are largely missing from OppNets simulations.

Take-Away Message 3:
The user behaviour has a great impact on application traffic and user mobility and should be modelled properly in a simulation.

F. Traffic Models

One of the main parameters when simulating a real-world scenario is how much data is created and circulated in a particular network. Also in this case, the used models should follow the real-world case characteristics as closely as possible. The traffic model is considered very important in OppNets, since the creation time of a particular message is crucial for the resulting delivery delay. For example, if the user creates the message at home with no contacts to other devices, it can be only delivered on the next days, when she starts moving again, thus introducing a high delivery delay.

When talking about traffic models, we need to differentiate between traffic size and traffic frequency. The former models how much data is created at once (e.g. always 1 kB, 1 GB or other random sizes), while the latter dictates how often data is created (once per second or week, randomly, etc.). Additionally, it is important to note that there are destination-oriented and destination-less traffic models.

A very good overview of existing models is provided in the book of Wehrle et al. [75]. We consider here only some of these models, which are more relevant to OppNets, namely:

- **Constant (periodic) traffic:** This is the simplest model, where a data packet is created every x time interval. It is considered by many researchers as being not realistic enough but, as we will show later in this section, for very

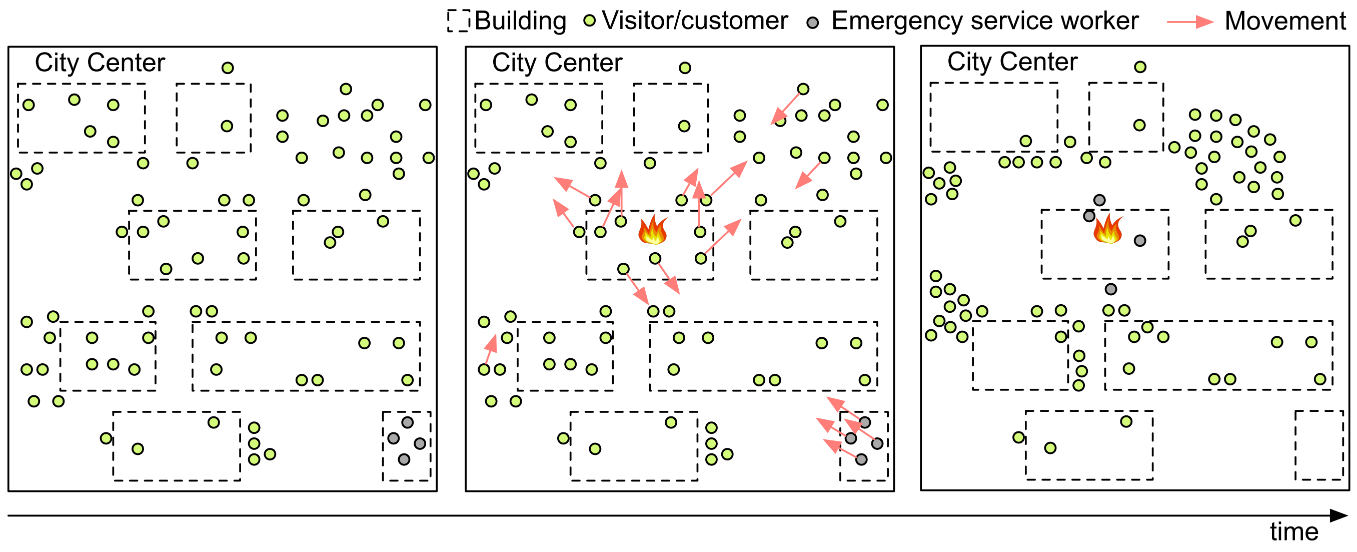


Fig. 8: Implications of User Behaviour: A fire at a building in the city center results in people running away from the danger zone, but gathering around it, while emergency workers rush to the fire.

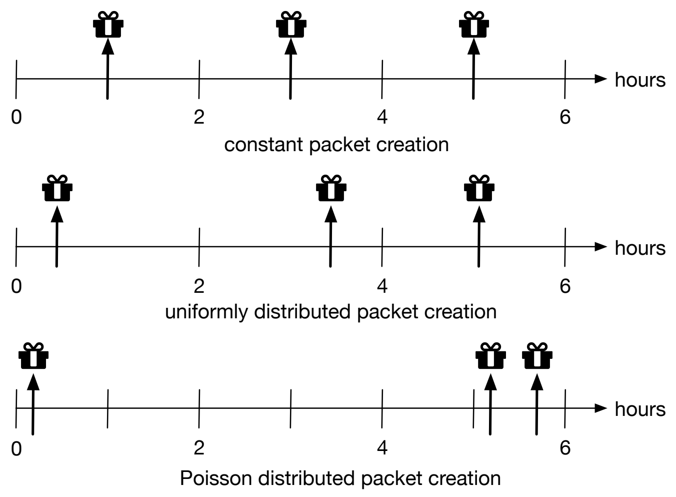

hours
0
2
4
6
constant packet creation
hours
0
2
4
6
uniformly distributed packet creation
hours
0
2
4
6
Poisson distributed packet creation

Fig. 9: Traffic patterns for the three most widely used traffic generators, the mean of all generators is the same.

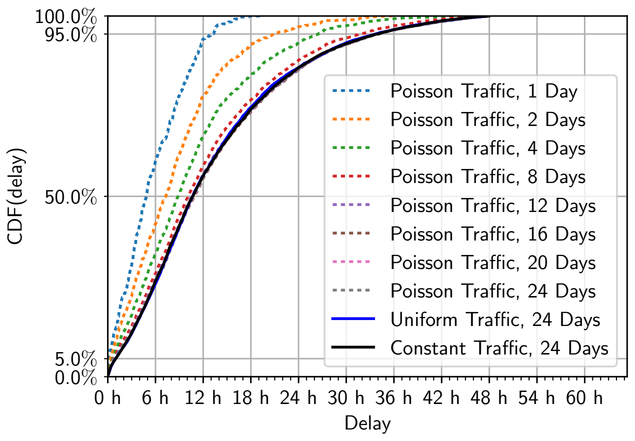


Fig. 10: Performance of different traffic generators with different simulation durations. Cumulative distribution function of the delivery delay. The mean of all generators was set to 2 hours, experiments performed with OMNeT++ and 50 nodes. The packet delivery rate varies between 91% and 97%.

large simulations it performs equally well as other models considered more realistic.

- **Uniform traffic:** This model is also based on a constant x time slot but the data packet is created anytime within the time slot instead of at the beginning (or at the end) of it. The creation instant inside the slot is randomly computed using a uniform distribution.
- **Poisson traffic:** This is probably the best known model, where the creation instant is randomly computed using a Poisson distribution. It has been shown that this distribution models very closely the traffic in user-driven network traffic, such as web browsing, phone calls, sending text messages, etc.

Figure 9 graphically compares the three traffic patterns assuming a 2 hours time slot. The “Constant” pattern generates packets periodically—at a well predefined time. The “Uniform” pattern generates one per period of 2 hours, but the exact timing of the packet is uniformly distributed inside the interval. Finally, the “Poisson” pattern creates a packet at random times, thus producing sometimes very long inter-packet times and sometimes creating “bursts” of packets, i.e., many packets with a very short inter-packet delay.

As commented before, Poisson traffic is usually considered superior to Constant or Uniform, since it is more “random” and has been shown to model user-driven traffic very well. We decided to test this assumption and we compared the three traffic generators using the OMNeT++ simulator and a simple opportunistic routing protocol (RRS, see Section VI). The scenario we used had 50 mobile nodes moving according

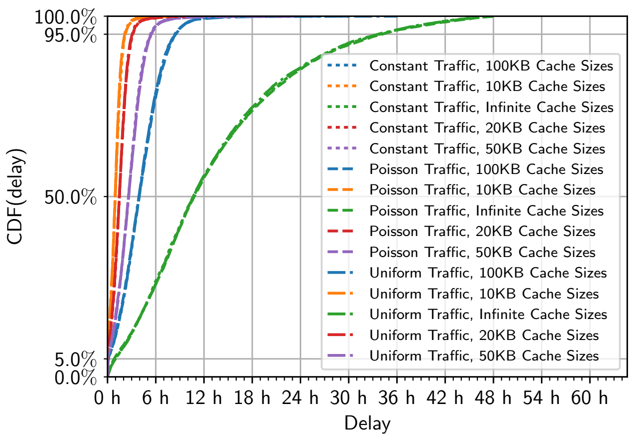


Fig. 11: Performance of different traffic generators with different cache sizes. Cumulative distribution function of the transmit delay. The mean of all generators was set to 2 hours, experiments performed with OMNeT++ and 50 nodes. The packet delivery rate varies between 39% and 92%. It can be seen that same-coloured curves overlap, which means that there is no significant difference between the three traffic generators.

to the GPS trace from San Francisco taxis; as we already did in Subsection V-A and will do in Section VIII. The mean of all traffic generators was set to 2 hours and the cache size of every node (to buffer data) was considered being infinite.

The resulting CDF (Cumulative Distribution Function) of the delivery delays is shown in Figure 10. As can be seen from the graph, when the simulation time is short (e.g., 1 day), the delivery delay of the packets vary slightly compared to the simulations of longer durations. However, when the simulation time increases, CDF curves of the delay quickly converge. **Take-Away Message 4:**

Constant, Uniform and Poisson traffic generators have the same impact on average performance metrics for long simulation runs.

However, there is another important parameter to consider, namely cache sizes. We designed another experiment with the three traffic generators and limiting caches to different sizes (10 kB, 20 kB, and infinite). The results are depicted in Figure 11 where it can be seen that the differences in the delays become significant when compared to the previous experiment in Figure 10. Moreover, when investigating the delivery rate of these simulations, we found that the PRR changed drastically depending on the cache sizes (e.g., 10 kB caches: ~39 %, 20 kB caches: ~50 %, infinite caches (1 GB): ~92 %). However, different traffic generators with the same cache size showed exactly the same behaviour (overlapping lines in Figure 11).

This behaviour has a logical explanation, viz., OppNets have a store and forward feature which means that when the user creates a message, this message might remain in the cache for a very long time because there are no contacts to forward it to. Thus, once the user starts moving and contacting other users, her cache is full of messages, created earlier. Thus, the exact time of creation becomes less important (although it affects the absolute delivery delay) than their number or the cache size itself.

Take-Away Message 5:
Large cache sizes are crucial for achieving high delivery rates.

The use of the previous synthetic traffic generation is the most common approach to evaluate the performance, as we can simply set and regulate the average load (messages) in our simulation experiments. Nevertheless, for some evaluations (for example, considering the application usage or social aspects) we must use more complex traffic patterns, such as **trace-driven** approaches. For example, the authors in [11] presented a trace of 50 users using *Twitter* on their smartphones with a Bluetooth based opportunistic network. These traces can be used to evaluate in depth several aspects such as network usage or device's power consumption. Nevertheless, this reproduction imposes a serious limitation on the evaluation, since the reproduced traffic is fixed.

Due to these limitations, a good approach is to use **hybrid traffic**, where the traffic is synthetically generated resembling typical user application patterns. A simple approach is to generate messages with message sizes and frequencies based on known application usage statistics. For example, in [76] three message sizes and frequencies were considered: (1) a short text message (1 kB) every hour; (2) a photo (1 MB) every 18 hours; and (3) a video or high-resolution picture (10 MB) every 96 hours. These frequencies were based on the statistics of *Whatsapp (Facebook, Inc.)* message usage from [77], while sizes are approximations of the content produced by current mobile phone hardware. More statistics about the use of this kind of mobile applications can be found in [78], [79].

G. Energy Consumption and Battery Models

All operations on end-user devices need energy. However, the exact behaviour of both the energy consuming parts (processing, visualisation, communication, etc.) and the batteries on the other side, is very complex and exhibits non-linear stochastic properties. This complexity has resulted in the development of a number of models that characterise and emphasise different aspects of handling energy usage. Considering the basic functionality, the available models can be classified as follows:

- The **Battery Model** should ideally mimic the behavioural characteristics of real world batteries like capacity, maximum voltage and current, charging and discharging, temperature dependency, lifetime, etc. Simple models assume the battery functions like a bucket: A limited number of energy coins are available for usage. Once these are exhausted, the battery dies. This is usually referred to as the bucket model. More complex models consider also the self-discharge and self-recovery effects of the battery or take into consideration also different loads and their impact on the battery lifetime. These models are covered well in the survey of Jongerden et al. [80].

- The recharging of the battery is handled by **Power Generator Models**. They simulate components such as charging stations, solar panels and similar components. They are very often combined with the battery model.
- The **Energy Consumption** or **Energy Expenditure** models reflects the energy usage by device components and / or activities. In these models, the energy consumption of the simulated devices is implemented as energy requirements of a certain state like idle, transmitting, receiving, processing etc., especially due to the operations of the network devices (802.15.4, Bluetooth, etc.). Depending on the granularity, the modelling of the device states can become very complex as the energy consumption of the complete device has to be taken into account. Another task of the energy consumption model is to shut down the node when the battery model signals that its capacity was depleted, and to restart, when the battery is recharged.

The complexity of these models has resulted in differing implementations in simulators.

The **ns-3** simulator supports all three models, i.e., generation, storage and consumption of energy. The generator model in ns-3 is a simple on-off generator while there are a couple of battery models that include a model for Lithium-ion batteries according to [81]. Additionally, ns-3 includes an analytical battery model which implements the Rakhmatov-Vrudhula model [82]. The energy consumption is modelled in ns-3 only for the WiFi physical layer, addressing the consumption levels of the different states of WiFi.

The **OMNeT++** simulator, through the INET Framework, provides a set of models to characterise energy related hardware properties. The framework provides generator, consumer and battery models that support a basic set of functions. Each of these models can be attached to any other module, providing it with the respective properties. Readily available models include an on-off generator (as in ns-3), an ideal battery model with infinite capacity and a residual battery model, with finite capacity. Both battery models follow the bucket algorithm. The energy consumption is modelled using a two state mechanism that supports the states of operating and sleeping, with an energy consumption parameter for both states.

The **ONE** simulator supports a model where the energy level can be set as a parameter before a simulation, together with the amount of energy required for different states such as scanning, transmitting, etc. of wireless interfaces. When the energy is depleted due to node activities, the node is shut down.

The **Adyton** simulator does not offer any models for energy or power consumption in its implementation.

As indicated before, modelling energy consumption and battery characteristics are complex tasks. Below, we present a simple scenario that highlights how the different user behaviour patterns would influence the operations of a battery based on a specific usage and consumption pattern.

Our scenario considers the OppNets service as a secondary service on user-held devices. There are two users, Alice and Bob, who have the same OppNets chatting application on their devices. Generally, the power consumption of their applications depend mainly on the device used and the activities of the applications. However, it depends also on the user herself, assuming that Alice has always all her communication interfaces switched on compared to Bob who switches the interfaces on only when needed. When Alice's application tries to send messages, it works much faster (no power-up delay) and the overhead of using them is small, compared to Bob. With Bob, sending a message is much more expensive due to the switch on, send, switch off cycle. As a result, the exact impact on the device battery lifetime is very hard to measure directly for individual users. Therefore, it makes more sense to consider an average overhead of OppNets services instead of lifetime or energy consumption.

Take-Away Message 6:
Consider average communication and processing overhead for OppNets instead of device lifetime and energy consumed.

VI. Propagation Protocols

During the last years various proposals emerged describing novel data propagation protocols for OppNets. It is not an aim of this survey to detail and classify all of these proposals, but in the following we will describe the most relevant and referenced proposals, indicating the availability on OppNets simulators analysed in this work. For a wider overview of the existing state-of-the-art in the area of opportunistic routing and forwarding we suggest to refer to the following works [4] or [83]. Some of the widely known protocols are presented below.

- *Epidemic* [84] is one of the most basic approaches, which is very often used as a reference for new proposals. Its basic idea is that when two nodes meet, they first exchange a list of their data items and then synchronise them. In this way, when they divide and the given time was sufficient, they will have exactly the same data caches.
- *Randomised Rumor Spreading (RRS)* [85] is similar to Epidemic and also floods the networks. However, when a node meets other nodes, it randomly selects one item in its cache and sends it out to all neighbours. Thus, it cannot be guaranteed that a useful item is sent out.
- *PRoPHET* [86] is a context-aware protocol, which uses the history of contacts to determine the probability that a node can deliver a message to a particular destination. It can reduce the network overhead by about one order of magnitude compared to simpler approaches like Epidemic.
- *Spray and Wait* [87] is a simple but very effective method based on controlled replication. In this approach the source nodes assign a maximum replication number to a message. A copy of this message is then distributed to a number of relay nodes and the process continues until one of the relay nodes meets the destination or the maximum number of replicas is reached. The idea is very efficient in controlling the overhead of individual messages. Based on this initial strategy many other protocols were proposed that basically focus on improving the efficiency of the data delivery by using different strategies for the replication phase.

- *BubbleRap* [88] is probably one of the first solutions that focused on using some social aspects of the users' context. They proposed the concept of *community* and of *centrality*. When messages are spread, nodes with higher centrality and from the same community are preferred with the assumption that those nodes will meet the destination sooner.

Adyton supports a wide range of protocols, e.g. Epidemic, Direct Delivery, PRoPHET, Spray and Wait, SimBet [89], and BubbleRap. Furthermore, it supports many variants of Spray and Wait, e.g. Most-Mobile-First (MMF) spraying and Most-Social-First (MSF) spraying by [90], LSF Spray and Focus [91], Encounter-Based Routing [92], Delegation Forwarding [93], Coordinated Delegation Forwarding [94], and some others.

The ONE offers also a wide variety of protocols, e.g. Epidemic, Spray and Wait, PRoPHET, PRoPHET v2 [95], First Contact [96], Direct Delivery, Maxprop [97] and some others.

OMNeT++ provides a number of data propagation protocols for OppNets. However, most of them are not compatible with the newer versions of OMNeT++, as explained in Subsection IV-A. There we find implementations of Epidemic, Publish-Subscribe [24], ExOR [25], and MORE [25]. The OPS framework[4] is up-to-date and offers a RRS implementation.

Several implementations are available also for ns-3, e.g. Bundle [98], Licklider [99] and Epidemic [100].

VII. Performance Metrics

Performance metrics provide the means to evaluate the performance of a given system. The type of metrics to use in evaluating a system differs from system to system. Due to the nature of OppNets, commonly used metrics for evaluating networks (e.g., throughput) have a lesser importance than certain others. In the following, a list of widely used metrics are presented together with how they could be computed.

A. Delivery Delay

The delivery delay provides a metric of how fast a message can be delivered to an intended recipient (or a group of recipients) considering a per-node scope or a per-network scope. For example, in case of emergency messages, it is always critical to know the speed of data propagation, while in other scenarios, it is important to know how far the data is propagated depending on different node densities, mobility patterns, user preferences and so on. The timeliness of data can be evaluated as *Average delay time to receive certain data per node* and *Average time to propagate data through the network*. The network wide *Mean Delivery Delay* δ is computed in Equation (9) where N is the number of nodes in the network.

$$\delta = \frac{1}{N}\sum_{i=1}^{N}\delta_i \tag{9}$$

where δ_i refers to the *Node Delivery Delay* and is computed according to Equation (10).

[4]https://github.com/ComNets-Bremen/OPS

$$\delta_i = \frac{1}{M_{rx,i}}\sum_{j=1}^{M_{rx,i}}(\tau_{rx,i,j} - \tau_{tx,i,j}) \tag{10}$$

where M_i is the total number of packets (i.e., messages or data) received by the i^{th} node in the network. The transmission time $\tau_{tx,i,j}$ is the time when the origin generates the j^{th} packet. The reception time $\tau_{rx,i,j}$ refers to the time when the packet is received by the node i.

In case a destination-less scenario is used, the delivery rate per node is computed according to Equation (11).

$$\delta_i = \frac{1}{M}\sum_{j=1}^{M}(\tau_{rx,i,j} - \tau_{tx,j}) \tag{11}$$

where M is the total number of messages created in this network, $\tau_{tx,j}$ is the time the j^{th} message was generated and $\tau_{rx,i,j}$ is the time node i received it.

Beside computing only mean values, it is also helpful to plot the CDF of the delays of individual nodes and all nodes in the network.

B. Delivery Rate

The delivery rate or reception ratio provides a metric of how many packets were delivered to an intended recipient before the packet was removed from the network. One main reason for the removal of a packet from the network is due to the data in the packet reaching its expiration time (time-to-live, TTL). There are other reasons as well, such as the removal from caches of the nodes due to their limited sizes and the caching policies adopted. The Delivery Rate η is calculated according to Equation (12).

$$\eta = \frac{M_{rx,i}}{M_{tx,i}} \tag{12}$$

where $M_{tx,i}$ is the total number of packets created by all the nodes in the network with destination node i and $M_{rx,i}$ is the number of packets received by the intended recipient node i.

In case of destination-less scenarios, the delivery rate η is computed according to Equation (13).

$$\eta = \frac{M_{rx,i}}{M} \tag{13}$$

where M is again the total number of messages created in this network and $M_{rx,i}$ is the number of messages received at node i.

C. Overheads

The term overhead refers to the additional activities needed to achieve an intended objective. In OppNets, the *Overheads* are a measure in terms of how much of these additional activities are required to deliver packets to the intended recipients. Following are some of the sub-metrics relevant in OppNets w.r.t. overhead computations.

- *Overhead of irrelevant data and duplicated data per node*: In OppNets, nodes receive also uninteresting data and duplicated copies. Therefore, the percentage of the

number of *irrelevant data received / forwarded* and the *number of duplicated copies received* with regard to the total number of data received / sent gives an indication of overhead per node.
- *Overhead of cache usage*: This shows the utilization of the cache for OppNets with regard to total memory available in the node. This can further be analyzed with regard to types of data which are propagated. For example, after receiving the data, even though the node is not interested in the data, it might have to store and carry the data until it encounters another node to forward because others might be interested in it. This will cause also an overhead in cache usage for irrelevant data w.r.t. the preferences of a node.
- *Overhead of energy consumption*: All of the above overheads cause also battery usage, e.g. for sending out messages. However, also other activities of the OppNets use energy – for comparing newly arrived messages with already existing ones, for sorting caches, for learning activities, etc.

Additionally to simply computing the number of irrelevant data stored or forwarded, we could also compute the fairness of overhead among all nodes in the network. We may use the Coefficient of Variation (CoV) as a means of determining the balance of the load in the network. The *Load Balancing Metric* (LBM) is computed in Equation (14).

$$\begin{aligned} \text{LBM} &= CoV(f(i)) \\ &= \frac{\sqrt{Var[f(i)]}}{E[f(i)]} \end{aligned} \tag{14}$$

where $f(i)$ is the function that returns the number of packets forwarded at a certain node i. Smaller values of LBM are an indication of the loads being well spread over the network while larger values are an indication of unevenly spread loads. The same can be done also for cache overhead or in general for energy overhead.

VIII. Comparative Study of Simulation Tools

In this section we first describe a use case for simulating an opportunistic network, that we later use for comparing the performance and output of the four different simulators we explore in this survey.

A. Use Case Scenario

We used a popular OppNets scenario, where people are moving through a city and exchanging short messages about events in the city. A typical message could be: "Coffee sampling event at Coffee Corner, 23.03.2017, 2-6 pm". We varied the number of nodes to test the scalability of the tools. In terms of simulation models, we used the following configuration (summarised also in Table VI):

1) Mobility Model: For our comparison we used a trace-based model (see Subsection V-A). Here also comes the first challenge of implementing our scenario. In order to evaluate the different simulators we required a large trace with many nodes throughout many days. We evaluated different traces, and the best option was the mobility trace of taxi cabs in San Francisco, USA [55] (also referred to as SFO trace), due to its high resolution, number of nodes and duration. This dataset contains GPS coordinates of approximately 500 taxis collected over 24 days in the San Francisco Bay Area, as shown also in Figure 12. This trace has been processed with the BonnMotion tool [29] in order to generate the appropriate trace format required for each evaluated simulator. However, it has to be noted that each simulator *uses* the trace in different ways. For example, Adyton pre-computes the contacts between individual nodes and uses this contact trace for its simulations. OMNeT++, ns-3 and the ONE have different parameters on how often they re-compute the current position (move on a straight line between two trace points). OMNeT++ and the ONE update the position according to a parameter, while ns-3 updates the position every time a higher layer requests it. In our case, this is approximately every second.

2) Radio Propagation and Interference Model: We used Unit Disk Graph (UDG) with a communication radius of 50 meters. We do not consider interference.

3) Link Technology: For scalability reasons and with the takeaway from Subsection V-D in mind, we do NOT use any communication protocol. However, message transmission depends on the bandwidth (fixed to 2.1 Mbps) and the contact duration calculated from the traces, in order to obtain more realistic evaluation. This simple model checks who is around every second and communicates directly with these neighbours, while the contact is still there. This was not possible for ns-3, where we use the complete IP stack and a WiFi implementation. Furthermore, Adyton does not have any link technology, as it pre-computes contact traces from GPS traces.

4) Data Propagation: Epidemic [84] is used for Adyton and the ONE. In OMNeT++, there is only the RRS implementation available, which we use in this evaluation. For ns-3, we use an implementation of Epidemic routing according to [101] which is based on WiFi Ad-Hoc and IP traffic. It is implemented as an IPv4 routing protocol.

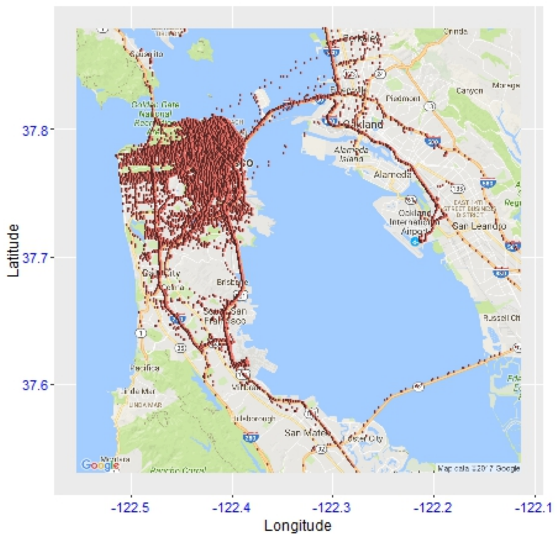
5) User Behaviour Model: The assumption is that the user simply reads the message.

6) Traffic Model: The users produce the messages with a Poisson traffic generator with a mean of one message per two hours. The size of the message is 1 KB. There is a single, random destination for each message.

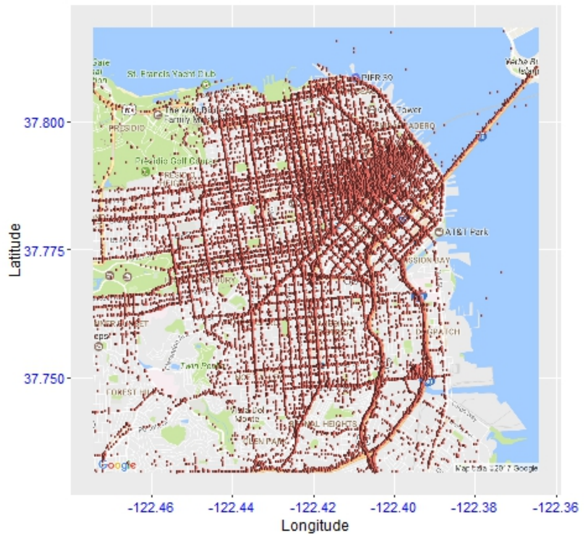
7) Power Consumption and Battery Model: We do not consider them in our scenario.

In terms of metrics, we compared the output of the simulators in terms of delivery latency (delay) and delivery rate at all nodes after simulating the complete 24 days of traces. We started with a baseline scenario with 50 nodes and we varied the number of users between 50 and the maximum of 536 nodes which corresponds to the total number of available traces. Furthermore, we compared the memory used and the real (i.e., wall clock) time needed to simulate the complete 24 days of traces.

All experiments have been performed in virtualised Linux servers with 8 processor cores (Intel Xeon, Sandy Bridge) with 48 GB RAM. The operating system was Ubuntu Yakkety



(a) San Francisco (Overview)



(b) San Francisco (Zoom-In)

Fig. 12: Vehicular GPS trace sample (San Francisco taxis-cabs).

Model	Adyton	ONE	OMNeT++	ns-3
Mobility	SFO trace, 24 days, contact trace	SFO trace, 24 days, 1 sec position update	SFO trace, 24 days, 1 sec position update	SFO trace, 24 days, on position update request (1 sec)
Radio propagation	UDG, 50 m	UDG, 50 m	UDG, 50 m	UDG, 50 m
Interference	none	none	none	none
Link technology	direct contact, 1 sec scan	direct contact, 1 sec scan	direct contact, 1 sec scan	WiFi
Data propagation	Epidemic	Epidemic	RRS	Epidemic
Application	Single destination	Single destination	Single destination	Single destination
User behavior	none	none	none	none
Traffic	Exponential, 1 pkt / 2 hours	Poisson, 1 pkt / 2 hours	Poisson, 1 pkt / 2 hours	Poisson, 1 pkt / 2 hours
Power consumption	none	none	none	none
Battery	none	none	none	none

TABLE VI: Summary of the used simulation models for all simulators and their main parameters.

16.10 using a 4.8 standard kernel. For the virtualisation, KVM (versions 2.1) has been used.

In general, the parametrization of ONE and Adyton was straight forward and relatively easy to accomplish. These two simulators are clearly targeted towards OppNets and provide exactly the parameters we needed, without options for link technologies, radio propagation models, power consumption, etc. OMNeT++ was also not difficult to setup and parametrize, but we are biased in this case because of our year-long experience with it. The setup of ns-3 proved to be the most difficult of all, because of the higher complexity (IP based simulation) and resulting higher number of parameters which influence the overall simulation results.

B. Scalability

We start the comparison between the simulators in terms of their own scalability and performance. Figure 13 shows the simulation time depending on the number of simulated nodes. Note that the values for the simulations with 536 nodes are extrapolated as none of the four simulations have completed even after 4 months. The extrapolation has been done after perusing the logs generated up to now. There is a clear message in this graph: The currently available tools are not scalable: 536 nodes are not a lot and 24 days are realistic.

There is also a clear trend to be seen: Simpler models and using C++ as the programming language provide a significant advantage. Thus, the ONE is the slowest (the only one written in Java), followed by ns-3, which has a complex link technology model, then comes Adyton and OMNeT++ is the fastest. Note that OMNeT++ has been optimised for performance for many years, while Adyton is a newcomer. Thus, we can still expect that Adyton's performance can improve dramatically soon. Regarding the memory used, we can see in Figure 14 that the ONE uses more than 5 times more RAM than the others. It is followed by ns-3 because of the IP stack simulation. However, all of the simulators stay in reasonable limits in their RAM usage, even for larger simulations. At the same time, it is probably not advisable to use a simple desktop machine for such large-scale simulations.

Additionally, we explored the size of the trace files for each simulator and summarised the results in Table VII. As can be seen, the ONE uses huge trace files, while the files for ns-3 and OMNeT++ are comparable to the original file. It is also visible that using contact traces instead of GPS traces plus radio propagation models saves a lot of memory (Adyton). We have to note here that Adyton in fact *loads the complete contact trace* into RAM before starting the simulation and still, its memory usage is one of the lowest (Figure 14).

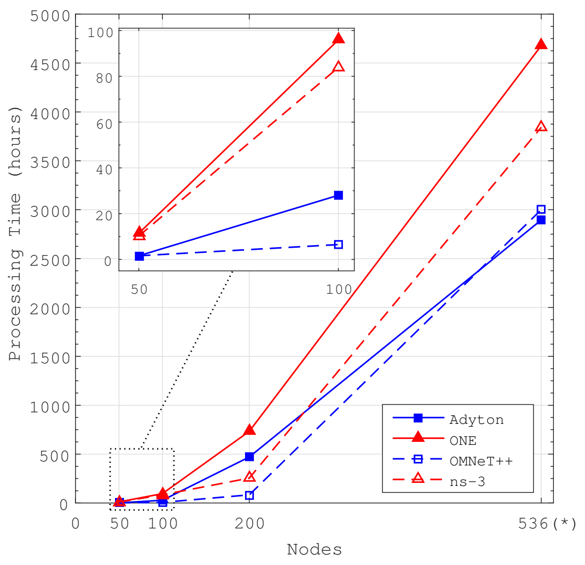


Fig. 13: Simulation time (i.e., wall clock time) required for the four simulators for 50-536 nodes.
∗ Values are extrapolated

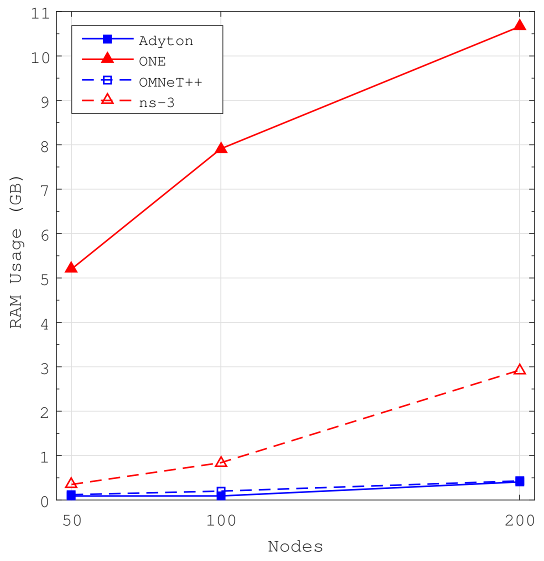


Fig. 14: RAM used for the four simulators for 50-200 nodes.

Take-Away Message 7:
Current tools are not scalable in handling the dimensions that OppNets require (thousands of nodes).

C. Performance Metrics

Let us now turn to the OppNets evaluation results, i.e., the obtained metrics from the different simulators. Figure 15 presents the delivery rate and the mean delay results for all scenarios and all simulators. The outlier is clearly OMNeT++ and the reason is the slightly different data propagation model (RRS instead of Epidemic for all others). This is also the reason why the delivery delay for OMNeT++ is much higher.

Trace file	Size, 50 nodes	Size, 200 nodes	Size, 536 nodes
Original trace	87 MB	323 MB	876 MB
Adyton	0.88 MB	16.03 MB	117.6 MB
ONE	4,800 MB	20,000 MB	52,000 MB
OMNeT++	49 MB	183 MB	494 MB
ns-3	99 MB	371 MB	1009 MB

TABLE VII: Trace file sizes for all simulators compared with the size of the original trace.

Take-Away Message 8:
Slight differences in the data propagation model result in large performance differences. Be aware of the exact implementation of a particular protocol!

D. Impact of Trace Resolution

We could potentially reduce the resolution of our mobility traces to overcome the performance problems of the simulators. We decided to test this hypothesis. The original traces have a resolution of one second. Thus, we generated a new trace with a resolution of five seconds, comparing the performance of a simulation with both traces. The results are summarised in Table VIII. As can be seen, reducing the resolution of the trace file from one second to five seconds in fact reduces the simulation time significantly, especially for the ONE. However, the results have also changed: Especially the delivery delay increases very significantly. For OMNeT++, also the delivery rate decreased. The reason of this behaviour is clear: For greater resolutions some of the possible contacts are missed, so the opportunity to transmit a message is lost, reducing the overall performance.

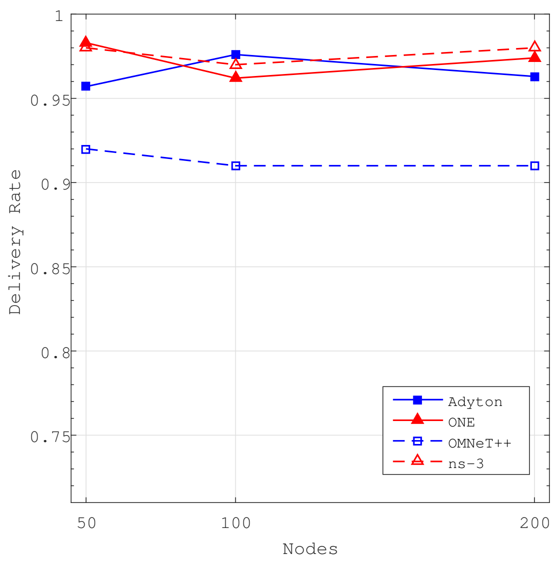
Take-Away Message 9:
Results from the same trace file but with different resolutions are not comparable with each other.

Simulator	ONE	ONE	OMNeT++	OMNeT++
Resolution	**1 sec**	**5 sec**	**1 sec**	**5 sec**
Delivery rate	98%	97%	92%	79%
Delivery delay	2.9 h	6.5 h	13.16 h	19.01 h
RAM	5.2 GB	5.14 GB	0.127 GB	0.124 GB
Simulation time	11.43 h	2.03 h	1.81 h	0.7 h

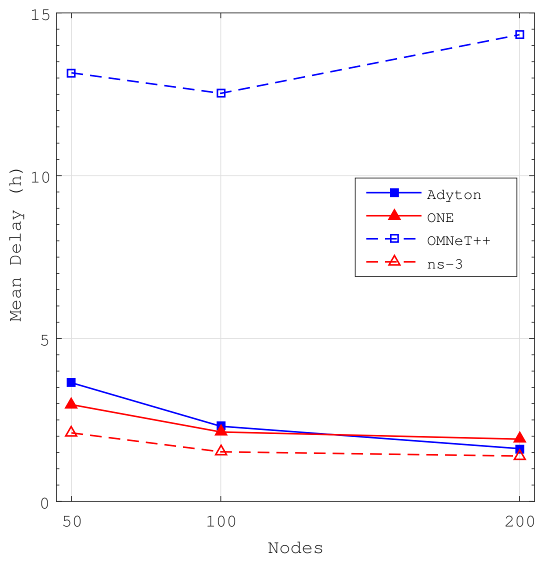
TABLE VIII: Comparison between 1 and 5 seconds resolution of the trace files.

E. Who is the fairest of them all?

It is not easy to answer the question which of the simulators is the best. Table IX offers a high-level comparison between the simulators, their readily available models and their user friendliness. This table can be used for a first overview to compare them in terms of the desired requirements, programming skills and application scenarios. However, what is probably more important, especially for newcomers in the area of OppNets, is whether you can get proper help. Thus, if there



(a)



(b)

Fig. 15: Delivery Rate (a) and Mean Delay (b) for the four simulators for 50-200 nodes.

is already a simulator in use in your community / research group, even if used for other types of network simulations, you should stick to that simulator. None of the here presented ones is perfect for any of the relevant applications anyhow.

If there are no simulators already in use, then you should consider joining the development of some simulator. The most promising ones are currently Adyton and OMNeT++. While Adyton is richer in models and has very promising performance, OMNeT++ is more sophisticated and mature, with very good support, documentation and user interface. It probably depends on your programming skills and preferences which one to choose. What we explicitly do not recommend is to start yet a new home-brewed simulator. There are enough choices and enough opportunities to shape the development of existing simulators.

Take-Away Message 10:
Prefer a simulator with which you or your group has already experience. Publish all new protocols and algorithms you develop open-source.

The next section offers also a list of suggestions to follow when conducting OppNets performance analysis with any of the presented simulators.

IX. Best Practices

Along the text of this survey we have included take-away messages for the reader that summarised the gained knowledge obtainable from the material presented, and made suggestions on which of the existing models to use and when. In the next section, we will identify what is still missing in existing models and suggest some possible concrete ideas on how these deficiencies could be addressed.

In this section we would like to summarise some further findings of this survey, which we did not state previously.

- Mobility models are the most important driver of performance in OppNets. **Carefully select an appropriate mobility model considering the specific use case.** Preferably select scalable, sophisticated hybrid models.
- Radio propagation, interference, battery and energy consumption models are not required for simulations in OppNets. **It is better to evaluate these factors in testbeds or to outsource the complex computations to a pre-step, as described in Subsection X-D.**
- Using models for link technologies introduces too much overhead in the simulator and furthermore affects the performance metrics of the OppNets. **Unless they are important per-se for the evaluation, it is worthwhile using a simplified link technology.**
- Simulating use cases in OppNets (similar to other environments) is a game of mixing-and-matching the appropriate models and setting their parameters to suit the given use case. **Meticulously document the models and the parameters you use for later comparison and reasoning.**

Our last finding is the most important one. To this end, we provide a sample experiment journal of one of our experiments (Table IV) in Appendix A. We have used such journals to document all of the presented experiments, which enables someone to easily reproduce our results and identify strong or weak points in the setup. The logs and a blank template are available online[5]. If always provided, these logs could significantly increase the readability of new results and will enable much more productive discussions among researchers and faster progress of the research area.

X. Future Directions

Section VIII provided a description of which models are already available in which simulator. It is hard to conclude on

[5]https://github.com/ComNets-Bremen/OppNets-Survey-Material

Tool	**Adyton**	**ONE**	**OMNeT++**	**ns-3**
Platforms	Linux	Java (JDK 6+)	Win, Linux, Mac	Linux, Mac, FreeBSD
Programming language	C++	Java	C++, NED	C++, Python
Parallelizable	-	-	+	o
BSD / Linux API compatibility	-	-	-	+ (DCE framework)
Documentation	+	+	++	+
Mailing lists and tutorials	o	++	++	+
User interface	-	+	++	-
Mobility models	o	+	++	o
Radio propagation models	-	o	+	+
Interference models	-	o	+	+
Link technologies	-	-	+	+
OppNets data propagation models	++	++	o	o
User behavior models	-	-	-	-
Traffic models	++	++	++	++
Energy consumption models	-	o	+	+
Battery models	-	o	+	++
Scalability	+	-	+	o

- no support, o partial support, + adequate support, ++ well supported

TABLE IX: High-level comparison between OppNets simulators.

which is the best simulator to use, since none of them has the complete spectrum of models, the desired performance and feature support level. It is left to the readers to decide which of these tools to select for their work and to try to compensate for the missing bits.

In the next paragraphs, we will sketch some concrete ideas about further developments of simulation models, irrespective of the tool being used. These ideas arose during the writing of this survey and can be used by researchers to improve the current simulation-based performance evaluations of OppNets.

A. *User Behavior Models*

Typically, OppNets deliver messages to users and users react to these messages. Such user behaviour models, as already discussed in Subsection V-E, practically do not exist right now. These models should include two basic actions that could be taken by the user:

- Move as a reaction to a message – either move immediately or schedule a move in the future. Of course, the mobility model needs to be able to react to such an action. Such a user behaviour model is needed for applications delivering also this type of information: Event announcements, traffic information, danger alerts, etc.
- Create one or several messages as a response to a message—this is a typical behaviour of a user, who just received a message from a friend. Such a model is important for all applications where users create messages (and not machines).

The second action is very simple to implement and only requires trivial changes in the traffic models. The first one is however more important as it directly influences the mobility of the user, which we have identified to be the driving force of OppNets. The model itself needs to accommodate some random but realistic behaviour of the user. For example, if Alice receives 10 event announcements per day, we cannot expect her to go to all these events. Somehow the selection of the event should be driven by Alice's preferences (which might be hidden from the OppNets system), but may also be influenced by the behaviour of her friends (are the friends attending too?) or some purely random selection.

B. *Reactive Mobility Models*

In order to support the above described user behaviour model, we need a reactive mobility model, able to schedule a move in the future or to move somewhere immediately. Any hybrid mobility model, as described in Subsection V-A, can be changed to accommodate such a behaviour. What is needed is an interface between the user behaviour model and its mobility model. However, mobility models based on real movement patterns (i.e., traces) are not able to do this due to their dependance on the position coordinates and the corresponding times listed in the trace.

C. *Realistic and Scalable Mobility Models*

In general, real mobility models (i.e., trace based) have many disadvantages, mainly related to flexibility, scalability and trace generation. Even if these mobility models are very realistic and very easy to use, OppNets performance analysis should focus on developing more sophisticated hybrid models. Here we sketch an idea of how to combine the advantages of real and hybrid mobility models:

1) Extract Points of Interests (PoIs) from real traces. Geographical information systems can be used very efficiently to identify good PoIs and their contextual information [102].
2) Break the trace into segments, where each segment is a real movement between two PoIs; each segment will become a "mini-trace".
3) Build a graph where nodes are PoIs and links are segments. There might be more than one link between two nodes, as the movement between a pair of PoIs might have occurred several times and with various means of transportation or speeds.

4) For each node (PoI), extract statistics for the pause time from the original trace and attach this information to the respective nodes.
5) Define a mobility model, which starts at one PoI, pauses there accordingly, then moves to the next PoI by following one of the available segments (mini-traces).

This approach has two very important and novel advantages. First, different traces can be merged together incrementally over time. This will give all researchers a very large database for performing very large, realistic mobility studies. Second, the exact algorithm of traversing the graph can be changed easily and can accommodate ideas and research from current hybrid mobility models, e.g., moving with friends or following a day-night schedule, etc.

The performance of such a model is expected to be better than real mobility models, but worse than state-of-the-art hybrid models. This is due to the fact that the simulated user does not move on a straight line between two PoIs, but on a mini-trace. Therefore, smart multithread-based model implementations will be required to address issues such as efficient operation of the models, e.g. to load the new segment data in a separate thread while the node is pausing in the current PoI.

D. Contact-based Mobility and Communication

Contact-based traces are very efficient in terms of performance as our results show with Adyton. In fact, they combine the impact of mobility, radio propagation and link technology models into one input trace. Thus, the overhead of computation during simulation runtime is minimised significantly. However, the main question is: Is the resulting behaviour realistic?

Currently, there are two possibilities to obtain contact traces: Experimentally and model-based. Obtaining them experimentally sounds like a very good idea. However, reality is different, as we know from our own studies [66]. In such experiments, nodes broadcast beacons regularly and log beacons from other nodes. When looking into the logged beacons from a pair of nodes, we most often observe a highly asymmetric behaviour: Node A has logged many more beacons from node B or vice versa. A post-processing step is needed to decide when the contact actually occurred. In fact, we do not know whether and for how long the nodes would have been able to actually exchange data. For example, link setup time and re-connect time are not taken into account at all. Thus, we need more sophisticated experiments, where the length of contact is obtained by taking into account real data communications over appropriate link technologies meaningful for the use case (i.e., Bluetooth LE instead of WiFi).

The second option for obtaining contact traces by models is maybe more practical to achieve. Here, we would need a sophisticated simulation environment with good mobility (for example, the idea above), radio propagation, interference and link technology models. This simulation can produce contact traces similar in quality compared to the experimentally obtained ones. These contact traces can be then used for simulating large scale OppNets scenarios.

An open challenge is how to combine user behaviour models with contact-based traces. One possible but complex option is to pre-compute contact data "just in time" in a separate thread of the simulator. Thus, if the mobility of a particular user changes, the contact data can be re-computed on the fly.

E. Radio and Interference Models

Purely synthetic radio propagation models are highly complex, not only in terms of performance, but mainly in terms of correct usage and parametrisation; this hides a highly probable source of error. On the other hand, hybrid models which use statistical trace data, are much more reliable and simple to use. Many research efforts have been put into such models for wireless sensor networks [67], but not for OppNets. This is a clear gap to be closed soon, especially considering the above idea of simulating realistic contact traces.

The main real-world properties to be considered in such a model are:

- Asymmetric links, mainly due to non-circular radio propagation patterns. REMI [103] is modelling this behaviour very well already.
- Bursty links, where the success of the next transmission depends on the history of transmissions. This has been observed and quantified with the beta-factor in [58].

Unfortunately, there is currently no available synthetic model taking into account all these properties together.

F. Credible and Realistic Traffic Models

We have shown in this survey that the used traffic models do not significantly affect the results of large simulations (Subsection V-F). We have also identified that cache sizes are crucial to increase the delivery rate in the network. At the same time, there is a mutual impact between the size of individual data messages and real contact length (considering also the link technology used). This connection needs to be explored.

Furthermore, the correct data size selection process seems rather random in state-of-the-art research. For example, many applications consider exchange of large video files—but the question is: Which realistic OppNets service would exchange videos? Short messages, such as text messages or small images are much more realistic and can be better served with OppNets. The exact parameters need to be extracted from real services and applications.

Moreover, even if we have shown that constant, uniform and exponential traffic patterns deliver comparable results, a realistic traffic model is still missing. For example, an exponential traffic pattern still generates traffic during the night—but, this is a very rare behaviour of real users. Such models are still missing and their impact on the performance of OppNets needs to be explored.

G. Scalability

All of the above described ideas target also an increase of the scalability of OppNets simulations. However, while they will most probably increase the general performance, they will not conceptually allow for very much larger scale simulations, e.g., millions of nodes. Thus, more far-reaching new concepts are required.

One novel idea has been presented in [104], where a packet-based simulation has been exchanged with a flow-based simulation. This saves a significant amount of discrete events, as only the start, the end and the status changes of a communication session between two nodes is modelled. It could be interesting to transfer this idea to OppNets simulations.

Furthermore, mathematical models are also a good alternative, especially for very large networks. We have already explored shortly the current state-of-the-art in Section III.

H. Metrics

Current metrics used in OppNets performance analysis are rather simple and assume that the user is happy to receive either everything (irrespective of how old or relevant the data is to her) or to receive the messages destined to her only. However, especially in OppNets applications, we often observe that people would like to receive "relevant" data. The question anyway is how to define relevance: Is it the receipt of timely data or of popular data?

Generally speaking, it depends on the data itself. Expired data, such as a finished event or outdated traffic information, is not useful at all, unless it was so popular that you would like to read about it later. For example, the fact that there was a traffic jam somewhere in the city yesterday is probably irrelevant. However, the news about a large demonstration in the city yesterday might be of great interest to many people.

One possibility to evaluate the performance of a particular data propagation algorithm is to allow users to evaluate data messages offline. For example, we can assume that all users have some preferences, defined as a set of keywords. Those preferences might or might not be visible to the OppNets application or algorithms. Furthermore, we allow each user to take a random decision about whether she likes the message, based on her preferences. After the simulation has finished, we let the simulated user evaluate each created message, irrespective of whether she actually received it or not and if received, when. Later we can compare this list with the actually received messages and compute the user-based delivery rate. Furthermore, if some of the messages were received too late, we exclude them from the delivered ones. This idea can be extended to also consider popular, but late messages.

Another challenge to address is the perceived quality of service (QoS) for a particular user. While the average delivery rate and delay might be very good for a particular scenario or network, there might be some users who never received any messages or with much higher delays than the average. These outliers need to be identified and examined carefully, too.

XI. Conclusion

In this survey we have explored all relevant simulation models and tools for analysing the performance of Opportunistic Networks. We have offered a taxonomy for most of the available models and we have explored their impact on the analysis results and their performance. We have compared the four most important OppNets simulators in terms of their performance, user friendliness and available models.

Furthermore, we have identified lacks of current models and have proposed several concrete improvements. We encourage the community to take these ideas, develop them further and thus contribute to a more sophisticated, realistic and comparable research in Opportunistic Networks.

Acknowledgment

This work was partially supported by the *Ministerio de Economia y Competitividad, Programa Estatal de Investigación, Desarrollo e Innovación Orientada a los Retos de la Sociedad, Proyectos I+D+I 2014*, Spain, under Grant TEC2014-52690-R and by the Universidad Laica Eloy Alfaro de Manabí (ULEAM), and the Secretaría Nacional de Educación Superior, Ciencia, Tecnología e Innovación (SENESCYT), Ecuador.

References

[1] S. Trifunovic, S. T. Kouyoumdjieva, B. Distl, L. Pajevic, G. Karlsson, and B. Plattner, "A decade of research in opportunistic networks: Challenges, relevance, and future directions," *IEEE Communications Magazine*, vol. 55, no. 1, pp. 168–173, Jan. 2017.

[2] V. F. Mota, F. D. Cunha, D. F. Macedo, J. M. Nogueira, and A. A. Loureiro, "Protocols, mobility models and tools in opportunistic networks: A survey," *Computer Communications*, vol. 48, pp. 5 – 19, July 2014.

[3] S. Batabyal and P. Bhaumik, "Mobility models, traces and impact of mobility on opportunistic routing algorithms: A survey," *IEEE Communications Surveys Tutorials*, vol. 17, no. 3, pp. 1679–1707, Thirdquarter 2015.

[4] N. Chakchouk, "A survey on opportunistic routing in wireless communication networks," *IEEE Communications Surveys Tutorials*, vol. 17, no. 4, pp. 2214–2241, Fourthquarter 2015.

[5] F. Xia, L. Liu, J. Li, J. Ma, and A. V. Vasilakos, "Socially aware networking: A survey," *IEEE Systems Journal*, vol. 9, no. 3, pp. 904–921, Sept. 2015.

[6] M. R. Schurgot, C. Comaniciu, and K. Jaffres-Runser, "Beyond traditional DTN routing: social networks for opportunistic communication," *IEEE Communications Magazine*, vol. 50, no. 7, pp. 155–162, July 2012.

[7] S. M. Tornell, C. T. Calafate, J. C. Cano, and P. Manzoni, "DTN protocols for vehicular networks: An application oriented overview," *IEEE Communications Surveys Tutorials*, vol. 17, no. 2, pp. 868–887, Secondquarter 2015.

[8] J.-Y. Le Boudec, *Performance Evaluation of Computer and Communication Systems*. EPFL Press, CRC Press, 2010.

[9] N. Banerjee, M. D. Corner, D. Towsley, and B. N. Levine, "Relays, base stations, and meshes: Enhancing mobile networks with infrastructure," in *Proceedings of the 14th ACM International Conference on Mobile Computing and Networking*, ser. MobiCom '08. New York, NY, USA: ACM, Sept. 2008, pp. 81–91.

[10] V. Vukadinovic and S. Mangold, "Opportunistic wireless communication in theme parks: A study of visitors mobility," in *Proceedings of the 6th ACM Workshop on Challenged Networks*, ser. CHANTS '11. New York, NY, USA: ACM, Sept. 2011, pp. 3–8.

[11] N. Ristanovic, G. Theodorakopoulos, and J. Y. L. Boudec, "Traps and pitfalls of using contact traces in performance studies of opportunistic networks," in *2012 Proceedings IEEE INFOCOM*, Orlando, FL, USA, Mar. 2012, pp. 1377–1385.

[12] R. Groenevelt, P. Nain, and G. Koole, "The message delay in mobile ad hoc networks," *Performance Evaluation*, vol. 62, pp. 210–228, Oct. 2005.

[13] Z. J. Haas and T. Small, "A new networking model for biological applications of ad hoc sensor networks," *Networking, IEEE/ACM Transactions on*, vol. 14, no. 1, pp. 27–40, Feb. 2006.

[14] X. Zhang, G. Neglia, J. Kurose, and D. Towsley, "Performance modeling of epidemic routing," *Computer Networks*, vol. 51, no. 10, pp. 2867 – 2891, July 2007.

[15] C. S. De Abreu and R. M. Salles, "Modeling message diffusion in epidemical DTN," *Ad Hoc Networks*, vol. 16, pp. 197–209, May 2014.

[16] Q. Xu, Z. Su, K. Zhang, P. Ren, and X. S. Shen, "Epidemic information dissemination in mobile social networks with opportunistic links," *Emerging Topics in Computing, IEEE Transactions on*, vol. 3, no. 3, pp. 399–409, Sept. 2015.

[17] E. Hernández-Orallo, M. Murillo-Arcila, C. T. Calafate, J. C. Cano, J. A. Conejero, and P. Manzoni, "Analytical evaluation of the performance of contact-based messaging applications," *Computer Networks*, vol. 111, pp. 45 – 54, Dec. 2016.

[18] T. Spyropoulos, K. Psounis, and C. Raghavendra, "Efficient routing in intermittently connected mobile networks: The multiple-copy case," *Networking, IEEE/ACM Transactions on*, vol. 16, no. 1, pp. 77 –90, Feb. 2008.

[19] E. Hernandez-Orallo, M. Serrat Olmos, J.-C. Cano, C. Calafate, and P. Manzoni, "CoCoWa: A collaborative contact-based watchdog for detecting selfish nodes," *Mobile Computing, IEEE Transactions on*, vol. 14, no. 6, pp. 1162–1175, June 2015.

[20] M. Karaliopoulos, "Assessing the vulnerability of DTN data relaying schemes to node selfishness," *Communications Letters, IEEE*, vol. 13, no. 12, pp. 923 –925, Dec. 2009.

[21] J. Whitbeck, V. Conan, and M. Dias de Amorim, "Performance of opportunistic epidemic routing on edge-markovian dynamic graphs," *Communications, IEEE Transactions on*, vol. 59, no. 5, pp. 1259–1263, May 2011.

[22] O. R. Helgason and K. V. Jónsson, "Opportunistic networking in OMNeT++," in *Proceedings of the 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems and Workshops, Simutools 2008, Marseille, France, March 03 - 07*, 2008.

[23] O. R. Helgason and S. T. Kouyoumdjieva, "Enabling multiple controllable radios in OMNeT++ nodes," in *Proceedings of the 4th International ICST Conference on Simulation Tools and Techniques, SIMUTools 2011, Barcelona, Spain, March 21 - 25*, 2011.

[24] S. T. Kouyoumdjieva, S. Chupisanyarote, O. R. Helgason, and G. Karlsson, "Caching strategies in opportunistic networks," in *Proceedings of the IEEE International Symposium on World of Wireless, Mobile and Multimedia Networks, WoWMoM 2012*, San Francisco, CA, USA, June 2012.

[25] Z. Zhao, B. Mosler, and T. Braun, "Performance evaluation of opportunistic routing protocols: a framework-based approach using OMNeT++," in *Proceedings of the 7th Latin American Networking Conference*. Medellin, Columbia: ACM, Oct. 2012, pp. 28–35.

[26] R. Zhang, A. R. Chandran, N. Timmons, and J. Morrison, "OppSim: A simulation framework for opportunistic networks based on MiXiM," in *Proceedings of the IEEE 19th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks, CAMAD 2014*, Athens, Greece, Dec. 2014.

[27] A. Keränen, J. Ott, and T. Kärkkäinen, "The ONE simulator for DTN protocol evaluation," in *Proceedings of the 2nd International Conference on Simulation Tools and Techniques*, ser. Simutools '09, Rome, Italy, Mar. 2009, pp. 55:1–55:10.

[28] N. Papanikos, D.-G. Akestoridis, and E. Papapetrou, "Adyton: A network simulator for opportunistic networks," [Online]. Available: https://github.com/npapanik/Adyton, 2015.

[29] N. Aschenbruck, R. Ernst, E. Gerhards-Padilla, and M. Schwamborn, "BonnMotion: A Mobility Scenario Generation and Analysis Tool," in *Proceedings of the 3rd International ICST Conference on Simulation Tools and Techniques*, ser. SIMUTools '10. ICST, Brussels, Belgium, Belgium: ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), Mar. 2010, pp. 51:1–51:10.

[30] Ó. Helgason, S. T. Kouyoumdjieva, and G. Karlsson, "Opportunistic communication and human mobility," *Mobile Computing, IEEE Transactions on*, vol. 13, no. 7, pp. 1597–1610, July 2014.

[31] C. Gloor, "Pedestrian simulator," Available: http://pedsim.silmaril.org (Accessed 2017-03-04).

[32] M. Behrisch, L. Bieker, J. Erdmann, and D. Krajzewicz, "Sumo–simulation of urban mobility: an overview," in *Proceedings of SIMUL 2011, The Third International Conference on Advances in System Simulation*. Barcelona, Spain: ThinkMind, Oct. 2011.

[33] F. Bai and A. Helmy, "A survey of mobility models," 2004.

[34] T. Camp, J. Boleng, and V. Davies, "A survey of mobility models for ad hoc network research," vol. 2, no. 5. Wiley Online Library, 2002, pp. 483–502.

[35] P. Hui, A. Chaintreau, J. Scott, R. Gass, J. Crowcroft, and C. Diot, "Pocket switched networks and human mobility in conference environments," in *Proceedings of the 2005 ACM SIGCOMM Workshop on Delay-Tolerant Networking*, ser. WDTN '05. New York, NY, USA: ACM, Aug. 2005, pp. 244–251.

[36] J. Leguay, A. Lindgren, J. Scott, T. Friedman, and J. Crowcroft, "Opportunistic content distribution in an urban setting," in *Proceedings of the 2006 SIGCOMM Workshop on Challenged Networks*, ser. CHANTS '06. New York, NY, USA: ACM, Sept. 2006, pp. 205–212.

[37] S. Gaito, E. Pagani, and G. Rossi, "Fine-grained tracking of human mobility in dense scenarios," in *Proceedings of the 6th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks Workshops, 2009. SECON Workshops '09.*, Rome, Italy, June 2009, pp. 1–3.

[38] N. Eagle and A. Pentland, "Social serendipity: mobilizing social software," *Pervasive Computing, IEEE*, vol. 4, no. 2, pp. 28–34, Mar. 2005.

[39] A. Passarella and M. Conti, "Characterising aggregate inter-contact times in heterogeneous opportunistic networks," in *Proceedings of the 10th International IFIP TC 6 Conference on Networking - Volume Part II*, ser. NETWORKING'11. Berlin, Heidelberg: Springer-Verlag, May 2011, pp. 301–313.

[40] E. Hernández-Orallo, J. C. Cano, C. T. Calafate, and P. Manzoni, "New approaches for characterizing inter-contact times in opportunistic networks," *Ad Hoc Networks, Special Issue on Modeling and Performance Evaluation of Wireless Ad Hoc Networks*, vol. 52, pp. 160 – 172, Dec. 2016.

[41] H. Zhu, L. Fu, G. Xue, Y. Zhu, M. Li, and L. M. Ni, "Recognizing exponential inter-contact time in VANETs," in *Proceedings of the 29th Conference on Information Communications*, ser. INFOCOM'10. Piscataway, NJ, USA: IEEE Press, Mar. 2010, pp. 101–105.

[42] T.-C. Tsai and H.-H. Chan, "NCCU Trace: social-network-aware mobility trace," *IEEE Communications Magazine*, vol. 53, pp. 144–149, Oct. 2015.

[43] University of Dartmouth, "CRAWDAD: A community resource for archiving wireless data." [Online]. Available: http://www.crawdad.org

[44] A. Mei and J. Stefa, "SWIM: A simple model to generate small mobile worlds," *CoRR*, vol. abs/0809.2730, Apr. 2008.

[45] I. Rhee, M. Shin, S. Hong, K. Lee, S. J. Kim, and S. Chong, "On the levy-walk nature of human mobility," *Networking, IEEE/ACM Transactions on*, vol. 19, no. 3, pp. 630–643, June 2011.

[46] A. Munjal, W. C. Navidi, and T. Camp, "Steady-state of the SLAW mobility model," *Journal of Communications*, vol. 9, no. 4, pp. 322–331, Apr. 2014.

[47] A. Munjal, T. Camp, and W. Navidi, "SMOOTH: a simple way to model human mobility," in *Proceedings of the 14th International Symposium on Modeling Analysis and Simulation of Wireless and Mobile Systems, MSWiM 2011, Miami, Florida, USA, October 31 - November 4*, 2011, pp. 351–360.

[48] C. Boldrini and A. Passarella, "Hcmm: Modelling spatial and temporal properties of human mobility driven by users' social relationships," *Computer Communications*, vol. 33, no. 9, pp. 1056–1074, June 2010.

[49] F. Ekman, A. Keränen, J. Karvo, and J. Ott, "Working day movement model," in *Proceedings of the 1st ACM SIGMOBILE Workshop on Mobility Models*, ser. MobilityModels '08. New York, NY, USA: ACM, May 2008, pp. 33–40.

[50] W. j. Hsu, T. Spyropoulos, K. Psounis, and A. Helmy, "Modeling time-variant user mobility in wireless mobile networks," in *IEEE INFOCOM 2007 - 26th IEEE International Conference on Computer Communications*, Barcelona, Spain, May 2007, pp. 758–766.

[51] S. Yang, X. Yang, C. Zhang, and E. Spyrou, "Using social network theory for modeling human mobility," *IEEE Network*, vol. 24, no. 5, Sept. 2010.

[52] J. Ghosh, S. J. Philip, and C. Qiao, "Sociological orbit aware location approximation and routing (solar) in MANET," *Ad Hoc Networks*, vol. 5, no. 2, pp. 189 – 209, Mar. 2007.

[53] D. Karamshuk, C. Boldrini, M. Conti, and A. Passarella, "Human mobility models for opportunistic networks," *IEEE Communications Magazine*, vol. 49, no. 12, pp. 157–165, Dec. 2011.

[54] P. Pirozmand, G. Wu, B. Jedari, and F. Xia, "Human mobility in opportunistic networks: Characteristics, models and prediction methods," *Journal of Network and Computer Applications*, vol. 42, pp. 45 – 58, June 2014.

[55] M. Piorkowski, N. Sarafijanovic-Djukic, and M. Grossglauser, "CRAWDAD dataset epfl/mobility (v. 2009-02-24)," Downloaded from http://crawdad.org/epfl/mobility/20090224, Feb. 2009.

[56] T. S. Rappaport, *Wireless Communications*. Upper Saddle River, NJ: Prentice Hall, 2001.

[57] G. Zhou, T. He, S. Krishnamurthy, and J. A. Stankovic, "Models and solutions for radio irregularity in wireless sensor networks," *Sensor Networks, ACM Transactions on*, vol. 2, no. 2, pp. 221–262, May 2006.

[58] K. Srinivasan, M. Kazandjieva, S. Agarwal, and P. Levis, "The Beta-Factor: Improving Bimodal Wireless Networks," in *6th ACM Conference on Embedded Networked Sensor Systems (SenSys'07)*, Raleigh, NC, USA, Nov. 2008.

[59] J. Bok and H. G. Ryu, "Path loss model considering doppler shift for high speed railroad communication," in *16th International Conference on Advanced Communication Technology*, Pyeongchang, South Korea, Feb. 2014, pp. 1–4.

[60] M. Lott and I. Forkel, "A multi-wall-and-floor model for indoor radio propagation," in *Proceedings of the IEEE VTS 53rd Vehicular Technology Conference, Spring*, vol. 1, Rhodes, Greece, May 2001, pp. 464–468 vol.1.

[61] H. Karl and A. Willig, *Protocols and Architectures for Wireless Sensor Networks*. John Wiley & Sons, 2005.

[62] Z. Ji, B. Li, H. Wang, H. Chen, and T. Sarkar, "Efficient ray-tracing methods for propagation prediction for indoor wireless communications," *IEEE Antenna and Propagation Magazine*, vol. 43, pp. 41–49, Apr. 2001.

[63] Z. Yun and M. F. Iskander, "Ray tracing for radio propagation modeling: Principles and applications," *IEEE Access*, vol. 3, pp. 1089–1100, July 2015.

[64] K. Garg, A. Förster, D. Puccinelli, and S. Giordano, "A tinyos based tool for gathering real-world wireless traces," in *Proceedings of the Sixth ACM International Workshop on Wireless Network Testbeds, Experimental Evaluation and Characterization (WinTECH)*, Las Vegas, Nevada, USA, Sept. 2011.

[65] A. Marchiori, L. Guo, J. Thomas, and Q. Han, "Realistic performance analysis of wsn protocols through trace based simulation," in *Proceedings of the 7th ACM Workshop on Performance Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks (PeWASUN)*, ser. PE-WASUN '10. Bodrum, Turkey: ACM, Oct. 2010, pp. 87–94.

[66] A. Förster, K. Garg, H. A. Nguyen, and S. Giordano, "On context awareness and social distance in human mobility traces," in *Proceedings of the 3rd International Workshop on Mobile Opportunistic Networks (MobiOpp)*, Zurich, Switzerland, Mar. 2012.

[67] P. Levis, N. Lee, M. Welsh, and D. Culler, "TOSSIM: accurate and scalable simulation of entire tinyos applications," in *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems (SenSys)*, Los Angeles, CA, USA, Nov. 2003, pp. 126–137.

[68] Q. Chen, F. Schmidt-Eisenlohr, D. Jiang, M. Torrent-Moreno, L. Delgrossi, and H. Hartenstein, "Overhaul of IEEE 802.11 modeling and simulation in ns-2," in *Proceedings of the 10th ACM Symposium on Modeling, Analysis, and Simulation of Wireless and Mobile Systems*, ser. MSWiM '07. New York, NY, USA: ACM, Oct. 2007, pp. 159–168.

[69] "IEEE 802.11 Wireless Local Area Networks – The Working Group for WLAN Standards," http://www.ieee802.org/11/, Accessed: 2017-02-20.

[70] IEEE, "IEEE Standard for Information technology," *IEEE Std 802.11-2012*, Mar. 2012.

[71] Bluetooth, SIG, "Bluetooth specification version 5.0," *Bluetooth SIG*, 2014.

[72] "IEEE Standard for Local and metropolitan area networks–Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs)," *IEEE Std 802.15.4-2011 (Revision of IEEE Std 802.15.4-2006)*, pp. 1–314, Sept. 2011.

[73] "LoRa Alliance – Wide Area Networks for IoT," https://www.lora-alliance.org/, Accessed: 2017-02-20.

[74] "Sigfox," http://www.sigfox.com/, Accessed: 2017-02-20.

[75] K. Wehrle, M. Günes, and J. Gross, *Modeling and Tools for Network Simulation*. Springer, 2010.

[76] J. Herrera-Tapia, E. Hernández-Orallo, A. Tomás, P. Manzoni, C. Tavares Calafate, and J.-C. Cano, "Friendly-sharing: Improving the performance of city sensoring through contact-based messaging applications," *Sensors*, vol. 16, no. 9, p. 1523, Sept. 2016.

[77] F. Richter, "An average whatsapp user sends messages per month," 2015. [Online]. Available: http://www.statista.com/chart/1938/monthly-whatsapp-usage-per-user

[78] P. Fiadino, P. Casas, M. Schiavone, and A. D'Alconzo, "Online social networks anatomy: On the analysis of facebook and whatsapp in cellular networks," in *2015 IFIP Networking Conference (IFIP Networking)*, Toulouse, France, May 2015, pp. 1–9.

[79] L. Zhang, C. Xu, P. H. Pathak, and P. Mohapatra, "Characterizing instant messaging apps on smartphones." in *Passive and Active Measurement. PAM 2015. Lecture Notes in Computer Science*, vol. 8995, Mar. 2015.

[80] M. Jongerden and B. Haverkort, "Which battery model to use?" in *Published in IET Software*, vol. 3, no. 6. IET, Dec. 2009, pp. 445–457.

[81] O. Tremblay, L.-A. Dessaint, and A.-I. Dekkiche, "A generic battery model for the dynamic simulation of hybrid electric vehicles," in *Vehicle Power and Propulsion Conference, 2007. VPPC 2007.* Arlington, TX, USA: IEEE, Sept. 2007, pp. 284–289.

[82] D. Rakhmatov, S. Vrudhula, and D. A. Wallach, "Battery lifetime prediction for energy-aware computing," in *Proceedings of the 2002 International Symposium on Low Power Electronics and Design*. Monterey, California, USA: ACM, Aug. 2002, pp. 154–159.

[83] *Opportunistic Routing*. John Wiley & Sons, Inc., 2013, pp. 419–452.

[84] A. Vahdat and D. Becker, "Epidemic routing for partially connected ad hoc networks," *Technical report number CS-200006, Duke University*, pp. 1–14, 2000.

[85] R. Karp, C. Schindelhauer, S. Shenker, and B. Vocking, "Randomized rumor spreading," in *Foundations of Computer Science, 2000. Proceedings. 41st Annual Symposium on*. Redondo Beach, CA, USA, USA: IEEE, Nov. 2000, pp. 565–574.

[86] A. Lindgren, A. Doria, and O. Schelén, "Probabilistic routing in intermittently connected networks," pp. 239–254, 2004.

[87] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, "Spray and Wait: An Efficient Routing Scheme for intermittently connected mobile networks," in *in Proceedings of the 2005 ACM SIGCOMM Workshop on Delay-Tolerant Networking*, Philadelphia, Pennsylvania, USA, Aug. 2005, pp. 252–259.

[88] P. Hui, J. Crowcroft, and E. Yoneki, "BUBBLE Rap: Social-based forwarding in delay-tolerant networks," *Mobile Computing, IEEE Transactions on*, vol. 10, no. 11, pp. 1576–1589, Dec. 2011.

[89] E. M. Daly and M. Haahr, "Social network analysis for routing in disconnected delay-tolerant MANETs," *Proceedings of the 8th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pp. 32–40, Sept. 2007.

[90] T. Spyropoulos, T. Turletti, and K. Obraczka, "Routing in delay-tolerant networks comprising heterogeneous node populations," *Mobile Computing, IEEE Transactions on*, vol. 8, no. 8, pp. 1032–1047, Aug. 2009.

[91] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, "Spray and focus: Efficient mobility-assisted routing for heterogeneous and correlated mobility," in *Proceedings - Fifth Annual IEEE International Conference on Pervasive Computing and Communications Workshops, PerCom Workshops 2007*, White Plains, NY, USA, Mar. 2007, pp. 79–85.

[92] S. C. Nelson, M. Bakht, and R. Kravets, "Encounter – Based Routing in DTNs," in *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM)*, Rio de Janeiro, Brazil, Apr. 2009, pp. 846–854.

[93] V. Erramilli, A. Chaintreau, M. Crovella, and C. Diot, "Delegation forwarding," in *Proceedings of the 9th ACM International Symposium on Mobile Ad Hoc Networking and Computing - MobiHoc '08*, Hong Kong, Hong Kong, China, May 2008, pp. 251–259.

[94] N. Papanikos and E. Papapetrou, "Coordinating replication decisions in multi-copy routing for opportunistic networks," in *International Conference on Wireless and Mobile Computing, Networking and Communications*, Larnaca, Cyprus, Oct. 2014, pp. 8–13.

[95] RFC-6693, "Probabilistic Routing Protocol for Intermittently Connected Networks." [Online]. Available: https://tools.ietf.org/html/rfc6693

[96] S. Jain, K. Fall, and R. Patra, "Routing in a Delay Tolerant Network," in *SIGCOMM '04: Proceedings of the 2004 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, Portland, Oregon, USA, Sept. 2004, pp. 145–158.

[97] J. Burgess, B. Gallagher, D. Jensen, and B. N. Levine, "MaxProp: Routing for vehicle-based disruption-tolerant networks," in *IEEE INFOCOM 2006 - 25th IEEE International Conference on Computer Communications*, Barcelona, Spain, Apr. 2006.

[98] RFC-5050, "Bundle protocol specification." [Online]. Available: https://tools.ietf.org/html/rfc5050

[99] RFC-5326, "Licklider Transmission protocol - specification." [Online]. Available: https://tools.ietf.org/html/rfc5326

[100] "Issue 13831049: Epidemic routing protocol addition for review," https://codereview.appspot.com/13831049/, Accessed: 2017-02-20.

[101] M. J. Alenazi, Y. Cheng, D. Zhang, and J. P. Sterbenz, "Epidemic routing protocol implementation in ns-3," in *Proceedings of the 2015 Workshop on ns-3*. Barcelona, Spain: ACM, May 2015, pp. 83–90.

[102] K. A. Tran, S. J. Barbeau, and M. A. Labrador, "Automatic identification of points of interest in global navigation satellite system data: A spatial temporal approach," in *Proceedings of the 4th ACM*

SIGSPATIAL International Workshop on GeoStreaming, ser. IWGS '13. Orlando, Florida, USA: ACM, Nov. 2013, pp. 33–42.

[103] J. Manojlovic, A. Förster, and M. Malek, "In search of reality: Remi - refined empirical model with irregularity consideraions for indoor channel propagation," 2017, under review.

[104] G. Anggono and T. Moors, "FLEO: A flow-level network simulator for traffic engineering analysis," in *2015 International Telecommunication Networks and Applications Conference (ITNAC)*, Sydney, NSW, Australia, Nov. 2015, pp. 131–136.

Appendix A
Sample Experiment Journal

General											
Experimental goal	Compare SWIM with SFO traces and RWP										
Dates (start - end)	Feb, 25th, 2017 - Feb 26th, 2017										
Machine	hebe virtual machine, Ubuntu 16.10, 4.8 standard kernel, 8 processor cores, 48 GB RAM										
Simulation tool version	OMNeT++ 5.0										
Simulation tool add-ons, framework etc. version	INET 3.4.0, OPS 0.1										
Final logs archive location	earth:/dev/null										
Responsible	Asanga										
Simulation Models											
Application	Personal message passing, people sending messages to each other										
Mobility model	**No. of nodes**	**Trace resolution**	**Position update resolution**	**Trace duration**	**Mean speed**	**Pause time**	**Environment size**	**Number of locations**	**alpha (SWIM)**	**Comments**	**Implementation**
SFO traces	50	1 sec	1 sec	24 days	inherent	inherent	inherent	inherent	n/a		BonnMotion
SWIM	50	n/a	1 sec	24 days	12.2 m/s	300 s	45000 m × 45000 m	n/a	n/a		self-implemented, published
Random Waypoint	50	n/a	1 sec	24 days	12.2 m/s	300 s	45000 m × 45000 m	n/a	n/a		INET
Radio propagation model	**Transmission Radius**	Path loss exponent									**Implementation**
UDG	50 m	n/a									OPS 0.1
Interference model	**Interference Radius**										**Implementation**
None	n/a										n/a
Link technology	**Scan Interval**	**Protocol**	**Bandwidth**	**Link delay**	**Cache size**	**Cache del. strategy**	**Cache add. strategy**				**Implementation**
Simple direct-contact	1 s	n/a	2.1 Mbps	0	infinite	n/a	LIFO				OPS 0.1
Data Propagation model	**Beacon Interval**	**TTL**	**Cache size**	**cache del. strategy**	**cache add. strategy**						**Implementation**
RRS	1 s	48 h	infinite	oldest first	n/a						OPS 0.1
User behavior model	**Change mobility**	**Reply to messages**									**Implementation**
None	n/a	n/a									n/a
Traffic model	**Shape (const, uniform, posson etc.)**	**Mean number of messages per hour**	**Message size**	**Destination**							**Implementation**
Constant	constant	2	1 KB	random single destination							OPS 0.1
Energy consumption model	**Energy per message**	**Energy per bit**									**Implementation**
None	n/a	n/a									n/a
Battery model	**Capacity**										**Implementation**
None	n/a										n/a
Metrics											
	Network-wide	**Mean per node**	**Full logs per node / message**								
Delivery rate for all messages	-	-									
Delivery rate for wished messages only	computed	computed	✓								
Delivery delay for all messages	-	-									
Delivery delay for wished messages only	computed	computed	✓								
Individual contacts	computed	computed	✓								
Messages received (as destination)	computed	computed	✓								
Messages created	computed	computed	✓								
Messages forwarded	computed	computed	✓								
Messages dropped	n/a	n/a	n/a								
Energy spent	-	-	-								
Simulation duration (wall clock time)	✓	-	-								
Simulation memory	✓	-	-								
Comments and Notes											
Trace files were created in real time on file server (earth) Wall clock time measured using *time* command RAM usage measured using *pidstat -r -u -C*											