Advanced Topology Analysis in Three Wireless Community Networks

Michele Pittoni

2014 - VI

Contents

Introduction

Wireless Community Networks (WCNs), a particular kind of wireless mesh networks, have become more and more popular in recent years. These networks are different from traditional ones in various aspects, such as the link nature and the arising topologies. For this reason, generic network protocols may not be well suited for WCNs and much research effort is being dedicated to the development of new routing protocols for this networks and for measuring their efficiency.

This works provides an introduction to the topic of Wireless Community Networks and a description of the three networks which are subsequently analysed. Also provided is an introduction to OLSR, the routing protocol which is used in the analysed networks. The biggest part focuses on the analysis of some efficiency metrics for the chosen networks and a comparison with some well known random network models.

1 Wireless Community Networks

The phrase "Wireless Community Network" has been used in literature to indicate various kinds of projects. The most common usage refers to wireless mesh networks operated by a community of citizens, as opposed to those controlled by a single entity (corporation or government). Other authors also use the term for the public hotspot networks run by municipalities, or more generally for networks that provide wireless connection to the public. In this document the term is used with the first meaning, only for networks which are run – not merely accessible – by the community.

History

The appearence of the first Wireless Community Networks can be dated to the late 1990s-early 2000s, when the IEEE 802.11 protocol (WiFi) was first introduced. They started as experiments by radio enthusiasts who wanted to explore the potential of this new technology, pushing it to the limit: WiFi was designed as a protocol for local communication, but it was shown that it can perform well also on long distance links, with the right equipment. The experimentation was made easier by the unlicensed spectrum of frequencies in which WiFi operates.

With WiFi gaining popularity and wireless enabled devices becomnig mainstream, the cost of WiFi equipment decreased and so did the barrier to participate in WCNs. At the same time, wireless protocols evolved and improved – for example 802.11a worked in the less crowded 5GHz band, allowing more separation between the used frequencies and thus less interference. In addition, a key factor to the diffusion on WCNs was the scarce availability of broadband connections in certain areas. In such cases, networks were not an experiment anymore, but were used to provide services were commercial initiatives were lacking.

Today, the WCN scenario is varied: in some places they are used to mitigate the digital divide, elsewhere they remaind experimental testbeds or are more focused on the social/political aspect of being an autonomous network owned and run by citizens. In the latter, the focus is on the services provided inside the network rather than on Internet access, even if it may be available. Some

WCNs have grown to the point of having an Autonomous System (AS) number assigned and being able to peer with other networks at Internet eXchange Points (IXPs).

WCNs usually have a cultural background of Open Source enthusiasts and hackers[1] in general. Some of them are run by a formal associations, other by informal groups of citizens, but in every case node owners know each other and there is a sense of community. More experienced participants share their technical knowledge (and their time), making it possible for new members to enter the network with minimum effort.

Because of their nature of not asking permissions, it is difficult to determine precisely the number of active WCNs as there is no central registry to look at. However, there is a Wikipedia page[2] which, albeit incomplete, lists 262 WCNs at the time of writing. The dimensions of such networks are varied, from just a handful of nodes to nearly 25,000 as in Guifi[3], which is widely regarded as the world's largest WCN.

Technology

Figure 1.1: Antennas of a WCN node

WCNs were born together with the IEEE 802.11 protocol family and continue to use it for various reasons, such as hardware availability and low cost, unlicensed frequency spectrum operation and the constant improvement of subsequent versions. Other solutions for the pyhsical layer are sometimes used –

[1] as in "curious", not as in "criminal"

[2] ("List of Wireless Community Networks by Region. Wikipedia, the Free Encyclopedia" 2014)

[3] http://guifi.net

for example proprietary protocols, maybe operating in licensed frequencies, or even methods not based on radio[4] – but they are an uncommon last resort when WiFi can not work (due to interference or other reasons).

A node of a WCN is a router connected with one or more radio interfaces. There is not a standard for the construction of a node, but over time every community has gathered some best practices and guidelines based on experience and trial-and-error. The equipment used varies from consumer WiFi routers (such as the very popular Linksys WRT54GL) with homemade antennas, to professional and more expensive equipment dedicated to long-range WiFi links. Smaller nodes, especially if they are near enough to other ones, may use a single omnidirectional antenna to connect to different nodes. Usually, however, directional antennas with a limited beam width are used, to reduce interference leveraging different channels, avoid receiving noise from all directions and achieve an higher gain. Some nodes use both kinds of antennas, directional for backbone links and omnidirectional to provides an hotspot access.

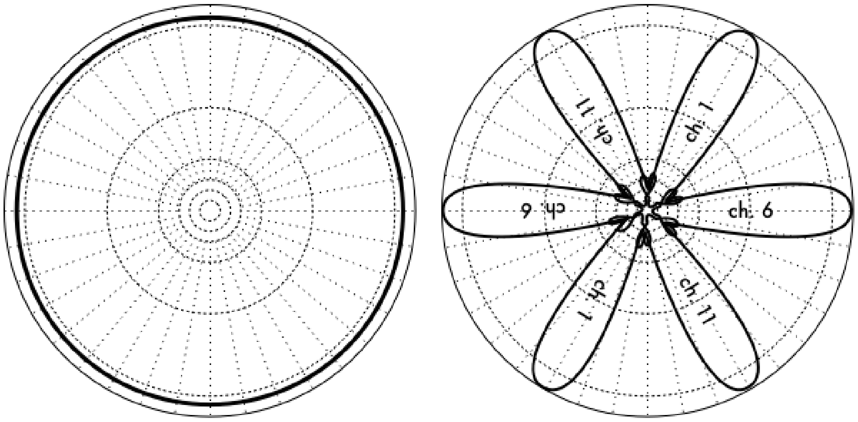


Figure 1.2: The difference between an omnidirectional and multiple monodirectional antennas.

Routing is one of the biggest challenges in wireless mesh networks. Due to the very nature of wireless links, traditional routing protocols thought for wired networks perform poorly when applied to them. In recent years, many new routing protocols have been proposed to address this issue and today there is a competition with no clear winner. The two most widely known routing protocols for wireless mesh and ad-hoc networks are OLSR and BATMAN. The former, which is used in the three WCNs analysed in this work, will be the subject of chapter 3. *some words on BATMAN?*

[4] Ronja, http://ronja.twibright.com/

2 Network topology and graphs

Some mathematical instruments are required to do any kind of description or analysis of the topology of a network, or to explain the functioning of a routing protocol.

The mathematical structure which is used to describe networks is the graph. A *simple, undirected graph* is an ordered pair $G = (V, E)$, where elements of V are the *nodes* (also called *vertices*) of the graph and are usually denoted with letters $u, v, w, \ldots$, while $E \subseteq \binom{V}{2}$ is the set of the *edges* of the graph. For convenience, $e_{ij} \coloneqq \{v_i, v_j\} \in E$.

An edge e_{ij} is said to be *incident* to the vertices v_i and v_j; equivalently, v_i and v_j are incident to e_{ij}. Two vertices incident to the same edge are said *adjacent*.

The graph is said simple since there are no loops (i.e. $\{u, u\} \notin E$) and each pair of vertices is connected by at most one edge. The above mentioned graph is also said undirected. On the other hand, a *directed graph* is a pair $D = (V, A)$ where $A = \{(u, v) | u, v \in V,\ u \neq v\}$ – the elements of A are usually called *arcs*.

For the purposes of describing networks, graphs are considered to be *finite*, so N and E are finite sets. Many well known finite graph properties do not hold in the infinite case.

A *weighted graph* is a graph in which every edge has an associated label *weight*, usually a real number. It is useful to define a function $w : E \to \mathbb{R}$ which associates weights to edges; in the case of unweighted graphs, it can be assumed $w : e \mapsto 1\, \forall e \in E$.

To model networks as graphs, each node of the network is represented by a node in the graph and a link between two nodes is represented by an edge between those nodes. If there are unidirectional link, a directed graph is used. For the purposes of this work, every link is considered bidirectional and symmetric, so from now on “graph” will be use for simple, undirected graphs.

Terminology

Order and size

The *order* of a graph is the number of its nodes, $|V|$. The *size* of a graph is the number of its edges, $|E|$.

Degree

The *degree* k_v of a vertex v is the number of edges incident to that vertex. A vertex of degree 0 is an *isolated vertex.* A vertex of degree 1 is a *leaf.*

The *total degree* of a graph is $\sum_{v \in V} k_v$.

The *degree sequence* of a graph is the list of degrees in non-increasing order. Not every non-increasing sequence of integers is the degree sequence of some graph.

The *degree distribution* of a graph is a function $p_k : \mathbb{N} \to [0,1]$ such that

$$p_k(n) = \frac{|\{v \in V \text{ st. } k_v = n\}|}{|V|}$$

The degree distribution is a discrete probability distribution since $\sum_k p_k = 1$.

Subgraphs

A *subgraph* $G' = (V', E')$ of G is a graph such that $V' \subseteq V$ and $E' \subseteq E \restriction_{V'}$, where $E \restriction_{V'} = \{\{v_i, v_j\} \in E \text{ st. } v_i, v_j \in V'\}$.

Walks, paths

A *walk* is a sequence of vertices $P = (v_0, v_1, \ldots, v_n) \in V \times V \times \ldots \times V$ such that $e_ij \in E$, $0 \leq i < n$. A walk is *closed* if $v_0 = v_n$, *open* otherwise. The *length* of the walk is n. The *weight* of the walk is $w_P = \sum_0^{n-1} w(e_ij)$. In unweighted graphs, $w_P = N$.

A *path* is a walk with no repeated vertices. A *cycle* is a closed walk with no repeated vertices, except obviously the starting one which is repeated once at the end.

Given a graph with no negative-weight cycles, a *geodesic path*, also called *shortest path*, between v_0 and v_n is a walk $P = (v_0, v_1, \ldots, v_n)$ such that $\nexists P' = (u_0, u_1, \ldots, u_m)$ with $u_0 = v_0$, $u_m = v_n$ st. $w_{P'} < w_P$. Note that P is a path: if $\exists v_i, v_j \in P$ st. $v_i = v_j$, then $\exists P' = (v_0, \ldots, v_i, v_{j+1}, \ldots, v_n)$ st. $w_{P'} < w_P \Rightarrow P$ is not the geodesic path.
In a graph with negative-weight cycles, the geodesic path is not defined, since it is possible to have walks with $w_P = -\infty$.
The length of a geodesic path (which is the length of all of them) from u to v is called *geodesic distance* of u and v, indicated with d_{uv}

Neighbours

Each vertex adjacent to v is also called a *neighbour* of v. The set of the neighbours of v is called *neighbourhood* of v.

The concept of neighbourhood may be extended to vertices at any distance. For example, the *2-hop neighbourhood* of v is the set of vertices u such that there is a walk from v to u whit lenght 2. Similarly, the *strict 2-hop neighbourhood* of v is the set of vertices which are in the 2-hop neighbourhood, excluding v itself and its direct neighbours.

Connectivity

A graph is called *connected* if, for each pair $\{u, v\}$ of nodes, there is a path between u and v.

A *connected component* of G is a maximally connected subgraph of G.

Centrality

In network science there is substantial interest in the concept of the centrality of a vertex (or an edge) in a graph. The idea behind this metric is to determine the most “important” components of a network. The meaning of “important” varies depending on the context: in social networks importance is usually defined by the influence of a node, measured by the size of it neighbourhood. In communication networks, the most important nodes are those who participates most communications, either by forming circuits or by relaying packets. These are just two examples of different notions of importance that require different ways to be measures. In the following paragraphs, the centrality metrics relevant to this work are presented.

Degree centrality

The degree of a vertex is the simplest possible measure of centrality and it is the only one that is only based on local properties. This is an advantage from the computational point of view, but it also implies that degree centrality is the least significant centrality metric. Nonetheless, depending on the graph, it can approximate quite well the behaviour of other metrics.

$$C_D(v) = k_v \tag{2.1}$$

Betweenness centrality

Betweenness centrality of vertex v is defined as the fraction of shortest paths between any two vertices that pass through v. Formally, define $\sigma(s, t)$ the number of shortest paths from s to t and $\sigma(s, t|v)$ the number of those paths that pass through v. If $s = t$, $\sigma(s,t) = 1$. There is not a consensus in literature if a path "passes through" its endpoint; in this case, it is assumed not: $\sigma(s,t|s) = \sigma(s,t|t) = 0$. The betweennes centrality is

$$C_B(v) = \sum_{s,t \in V} \frac{\sigma(s,t|v)}{\sigma(s,t)} \tag{2.2}$$

Betweennes centrality is especially useful in the study of communication networks because information usually travels through the shortest path, so C_B helps estimating how much traffic a node will see, in a way other centrality measures do not consider. For example, in the classic Barbell graph − two complete graphs connected by a path − vertices on the path have a very small degree but since every path between the two complete graphs passes through them they have high betweenness. This reflects the great control they have over the communications between other nodes.

Closeness centrality

Closeness centrality is also based on shortest paths, but has a different approach and a different meaning. It is based on the mean distance between v and the other vertices. If d_{vu} is the geodesic distance between v and u, the *mean geodesic distance* from v to u, averaged over all vertices u is

$$\mathcal{L}_v = \frac{1}{n} \sum_u d_{vu} \tag{2.3}$$

Since usually centrality measures have high values for more central nodes, closeness centrality is usually defined as the inverse of the mean distance $\mathcal{L}_v$.

$$C_C(v) = \frac{1}{\mathcal{L}_v} = \frac{n}{\sum_u d_{vu}} \tag{2.4}$$

Closeness centrality, despite being often used in network studies, has come shortcomings. For example, the above definition is only valid if the graph is connected, since d_{vu} is defined to be infinite if there is no path from v to u. In graphs with more than one connected components, C_C would then be zero for every vertex. The usual solution is to compute the closeness centrality for each connected component separately: this works, but since distances are usually smaller in small components, vertices in those components tend to have higher closeness centrality, which may be undesirable.

Another issue with closeness centrality is that its values are often cramped in a small range from lowest to highest. In most networks distances tend to be small, typically increasing with the logarithm of the size n of the graph. So, the lower and upper bound for $\mathcal{L}_v$ are, respectively, 1 and $\log n$. Similarly, the range for C_C is limited.

Random graph models

A random graph is a graph generated by a random process. The reason for using random graph models in network analysis is that they can produce graphs with known degree distributions, which can be used to prove mathematically or otherwise analyse empirically their structural and dynamical properties.

Erdős-Rényi random graph

The random graph model originally proposed by Erdős and Rényi is also called $G(n, M)$ model, since it consists in the uniform random selection of a graph from the set of all graphs with n nodes and M edges.

The model used here is a variaton first introduced by (Gilbert 1959), called the $G(n, p)$ model. The algorithm starts form a graph with n nodes and no edges. Then, for each unordered pair of nodes $\{i, j\}.i \neq j$, the edge ij is added with probability p.

The $G(n, p)$ models has some interesting properties which are not obvious at a first look. For example, the number of edges is not known as in the $G(n, M)$ models, but the expected number of edges can be determined to be $\binom{n}{2}p$. Another important aspect is connectedness: for $p > \frac{(1+\epsilon)\ln n}{n}$ the graph will almost surely be connected, while for $p < \frac{(1-\epsilon)\ln n}{n}$ it will almost surely have isolated vertices.

Finally, the degree distribution has the form

$$p_k = \binom{n-1}{k} p^k (1-p)^{n-1-k} \tag{2.5}$$

Barabási-Albert graph

A scale free network is a network whose degree distribution follows a power law of the form

$$p_k = Ck^{-\alpha} \tag{2.6}$$

A method for generating graphs with a power law degree distribution, using a preferential attachment mechanism, was devised by A. L. Barabási and R. Albert in (Barabási and Albert 1999). This is the method implemented by NetworkX. Given a target number n of nodes and a parameter m which controls the density of the network, the algorithm starts from a graph with m nodes and no edges. Then other nodes are added and from each new node m edges are created. The new edges are attached preferentially to the nodes with higer degree. This continues until there are n nodes in the graph, meaning the final graph will contain $(n - m)m$ edges.

3 OLSR summary

Optimized Link State Routing (OLSR) is a proactive routing protocol standardized by the IETF in RFC 3626[1] and designed to have a better performance on wireless mesh and ad-hoc networks than traditional protocols for wired networks.

In link state routing protocols, each node is supposed to know the entire topology of the network in order to calculate the routes. This means that each time the topology changes, the new information must be propagated to every node. This is traditionally done by flooding link-state advertisement packet through the network.

In the case of traditional networks with wired links, this method is acceptable since the topology seldom changes. In WCN (and wireless networks in general), however, links change their cost quite often and may also disappear temporarily. Flooding in this situation imposes a great effort on the network and may degrade the performance consistently.

OLSR addresses this concern with an optimized flooding mechanism which significantly reduces the overhead by using only selected nodes, called multipoint relays (MPRs), to broadcast link-state advertisements. The next sections outline the details of this mechanism and of OLSR in general.

Generic packet format

OLSR uses different types of messages in its specification. In order to take advantage of the maximal frame size provided by the network, one or more messages are encapsulated in a packet which has the same format for all types of messages. This facilitates the extensibility of the protocol and allows the transmission of different kinds of information in a single packet.

Each message is flooded through the network with a TTL. The transmission to neighbours is just a special case of flooding. Duplication is eliminated locally, since each node maintains a set of messaged it has already processed, and minimized globally by the MPR mechanism.

[1] Clausen and Jacquet (2003)

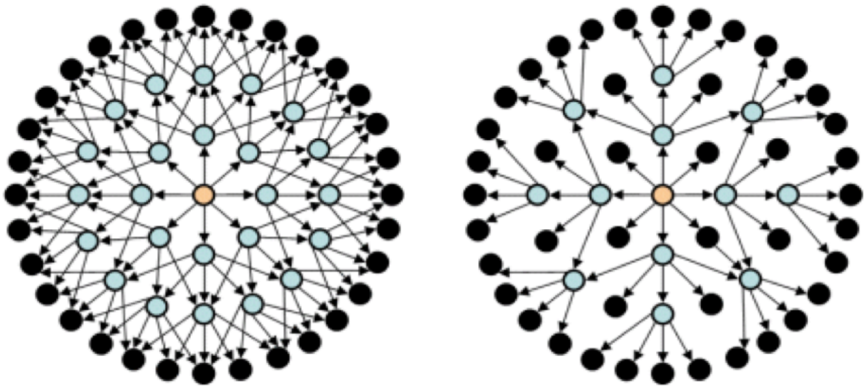


Figure 3.1: Flooding vs. MPR forwarding

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Packet Length         |    Packet Sequence Number     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Message Type |     Vtime     |         Message Size          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Originator Address                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Time To Live |   Hop Count   |    Message Sequence Number    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
:                            MESSAGE                            :
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Message Type |     Vtime     |         Message Size          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Originator Address                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Time To Live |   Hop Count   |    Message Sequence Number    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
:                            MESSAGE                            :
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                                                               :
               (etc.)
```

Link sensing and neighbour discovery

MPRs are selected locally by each node between its neighbours. The requirement is that the MPRs of a node v must cover, with the union of their neighbourhoods, the 2-hop neighbourhood of v. Thus, in order to select its MPRs, a node must know its 2-hop neighbours and how to reach them.

The message type used in OLSR for this purpose is the `HELLO` message, which is transmitted by each node to its neighbours and contains the addresses of the neighbour it knows. Of course, the first `HELLO` messages are empty and only serve the purpose of link sensing. After each node populates its neighbour set, it includes this information in its `HELLO` messages, along with some information on the links and the neighbours (e.g. if the links are verified to be symmetric). This allows all nodes to gather the necessary knowledge of their 2-hop neighbourhood.

`HELLO` messages are generated and transmitted at a regular interval (`HELLO_INTERVAL`). Lost links are also advertised for some time (with a link type of `LOST_LINK`).

MPR selection and signalling

Using the information from the `HELLO` messages, each node can select a set of its neighbours such that every node in its 2-hop neighbourhood is at 1 hop from the set. Formally, call $N_1(u)$ the neighbourhood, $N_2(u)$ the strict 2-hop neighbourhood, select $\mathrm{MPR}(u) \subseteq N_1(u)$ such that

$$\forall v \in N_2(u) \exists s \in \mathrm{MPR}(u) \text{ st. } v \in N_1(s)$$

This requirement essentially means that each node in the strict 2-hop neighbourhood can be reached through a MPR. The protocol specification suggests that the MPR set of each node should be as small as possible, but does not require it. Once a node has selected its MPRs, it needs to signal its choice to the neighbours, so that the selected nodes know that they must retransmit its broadcasts (and the other nodes know not to do that). `HELLO` messages are used also for this purpose: the selected MPRs are advertised with a neighbour type of `MPR_NEIGH`.

Message forwarding

Observing the `HELLO` messages it receives, each node populates and maintains an `MPR Selector Set`. This is the set of nodes that selected it as an MPR or, in other words, the set of nodes whose transmissions are to be forwarded by the node in question.

When a node receives a message (except for `HELLO` messages, which are never forwarded), it checks if the time-to-live has reached zero or if the message was already processed (by examining the duplicate set). If the message passes this preliminary checks, it is forwarded only if it was received from a selector.

This strategy obviously reduces the number of times a single message is forwarded. Thanks to the construction of the MPR set, the strategy also ensures that each message can reach every node in the network.

Topology control

The link-state information is propagated throughout the network with `Topology Control` messages (`TC`). This messages are generated only by the nodes which have been selected as MPR for some other node and propagated following the above described rules. Each `TC` message contains the identification of the node who generated the message and a list of the addresses of some of its neighbours.

The OLSR specification requires that the nodes in the `MPR Selector Set` of a node be in the `TC` messages it generates. To add redundancy, each node can advertise, in addition, the neighbours selected by it as MPRs, or even all of its neighbours. The added redundancy comes at the cost of longer `TC` messages, which may be more susceptible to congestion.

Link quality

OLSR implements a mechanism to avoid using “bad” links (i.e. links which are usually too weak but may let `HELLO` messages pass from time to time). Since `HELLO` messages are transmitted at a regular interval, each node knows how many of them to expect from each neighbour over a period of time. Comparing this with the number of received messages it computes a measure of the Link Quality (LQ). This metric was originally only used to decide if a link was reliable enough to use. New versions of OLSR have put more importance on link quality.

It is common in WCNs to use the ETX metric to express link quality. ETX stands for Expected Transmission Count and was proposed in De Couto (2004). It indicates the expected number of transmissions (including retransmissions) required to successfully deliver a packet.

In OLSR, ETX is derived directly from LQ. `HELLO` messages contain the calculated values, so each node has for every link two measures: its own (LQ) and its neighbour’s (NLQ). Since each packet transmission requires an acknowledgement, the estimated probability of success is LQ · NLQ. ETX is calculated as

$$\mathrm{ETX} = \frac{1}{\mathrm{LQ} \cdot \mathrm{NLQ}} \tag{3.1}$$

4 The analysed networks

The three WCNs which are analysed later are Ninux, Funkeuer Wien and Funkfeuer Graz. The study considers 50 snapshots of the networks taken from ... to ... .

Ninux

ninux.org

WIRELESS COMMUNITY NETWORK

Ninux[1] is the largest italian WCN. It was started in 2001 in Rome and now consists of about 250 active nodes, located in different "Ninux islands" all over Italy. The name "Ninux" originally was a tribute to the project founder, Nino, but now the project members usually take it with the meaning "Neighbourhood Internet, Network Under eXperiment".

Ninux is managed in an informal way: every member is the owner and responsible of its node (or nodes), but there is no formal association. This is a deliberate choice of the Ninux community, motivated by the excessive bureaucratic effort it would require. Moreover, all associations must have a president responsible for the activities of the association itself, and Ninux members prefer the responsibility to be decentralised (as the network is).

The different islands use a variety of protocols and have different topologies:

- in Pisa, Sicily and Friuli there are three mesh networks, with routing based on B.A.T.M.A.N.
- in Rome, the biggest Ninux island uses a backbone with point-to-point links combined with some mesh areas, employing OLSR for all the routing

[1] http://wiki.ninux.org/

- in recent years other networks have been created in Florence, Viterbo, Catanzaro, Cosenza and Reggio Calabria, all based on OLSR

The islands are connected together by tunnels using a variety of protocols.

Ninux is an Autonomous System (AS# 197835)[2] and it is a member of the NaMeX[3] Internet Exchange Point.

In this work, the biggest Ninux island has been analyzed (since it is the biggest OLSR routing domain), which is Rome's newtork. It consists of 132 nodes connected by 154 links (average degree of 2.333).

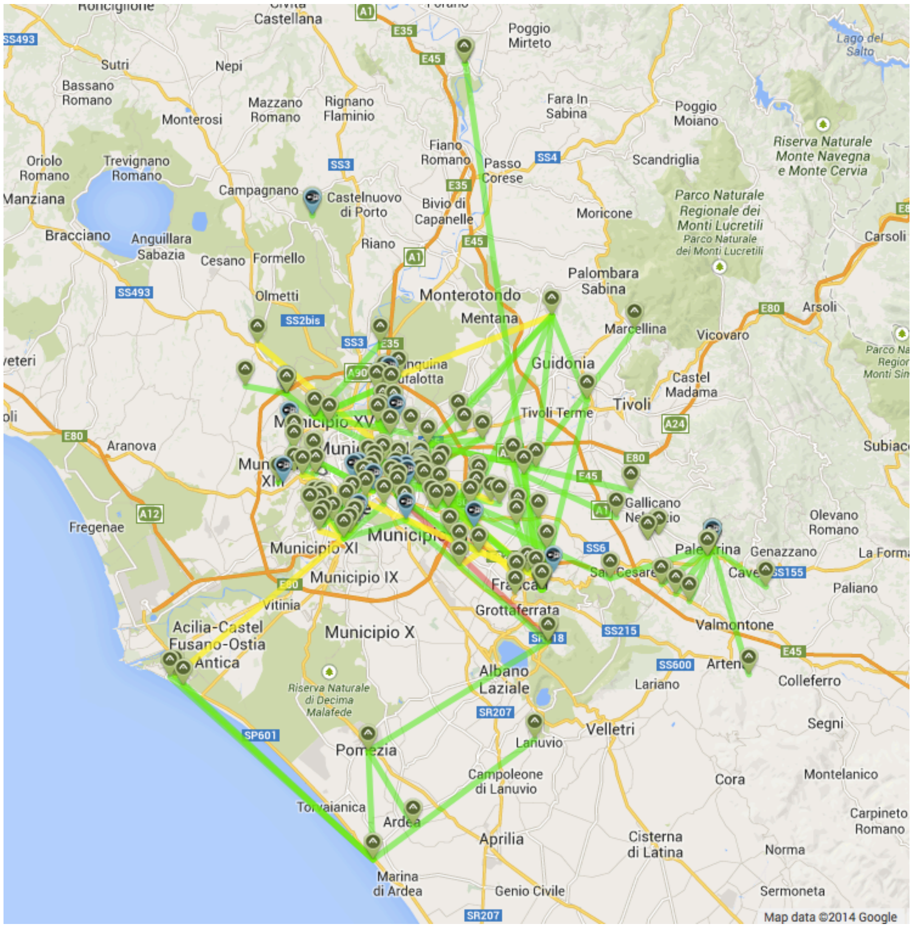


Figure 4.1: Map of the Rome Ninux island

[2] https://apps.db.ripe.net/search/query.html?searchtext=AS197835&object_type=aut-num#resultsAnchor

[3] http://www.namex.it/en/who/members

FunkFeuer: Wien

FunkFeuer[4] is a project that comprises networks in different parts of Austria (Vienna, Graz, parts of Weinviertel and Bad Ischl). The literal meaning of "funkfeuer" in german is "radio beacon". FunkFeuer is also a registered association in Austria, differently from Ninux.

The origins of FunkFeuer are in the experiments of an austrian ISP based in Vienna, Silver Server, which, during the 1990s, explored the commercial viability of wireless radio data links. After a test phase, Silver Server ultimately decided that the technology was not ready to be used commercially; however, the infrastructure was already in place and it was handed off to two associations, Team Teichenberg and Public Voice Lab. With direction from Franz Xaver and Roland Jankowski, they further expanded the network, bringing the node count to 15, but failed to create easy end-user access.

The network was ultimately decentralised, giving the opportunity to the citizens to buy the hardware of the nodes. The advent of cheap GNU/Linux based embedded WiFi products promoted the growth of the network and an association was founded to have a more structured organisation and address the issues of decentralisation. The existence of a formal association also enables to request sponsoring from local administrations.

FunkFeuer Wien (FFWien)[5] is the biggest of FunkFeuer networks, covering 1/3 of the city. The active node in the analysed snapshots were 237, with 433 links (an average degree of 3.654).

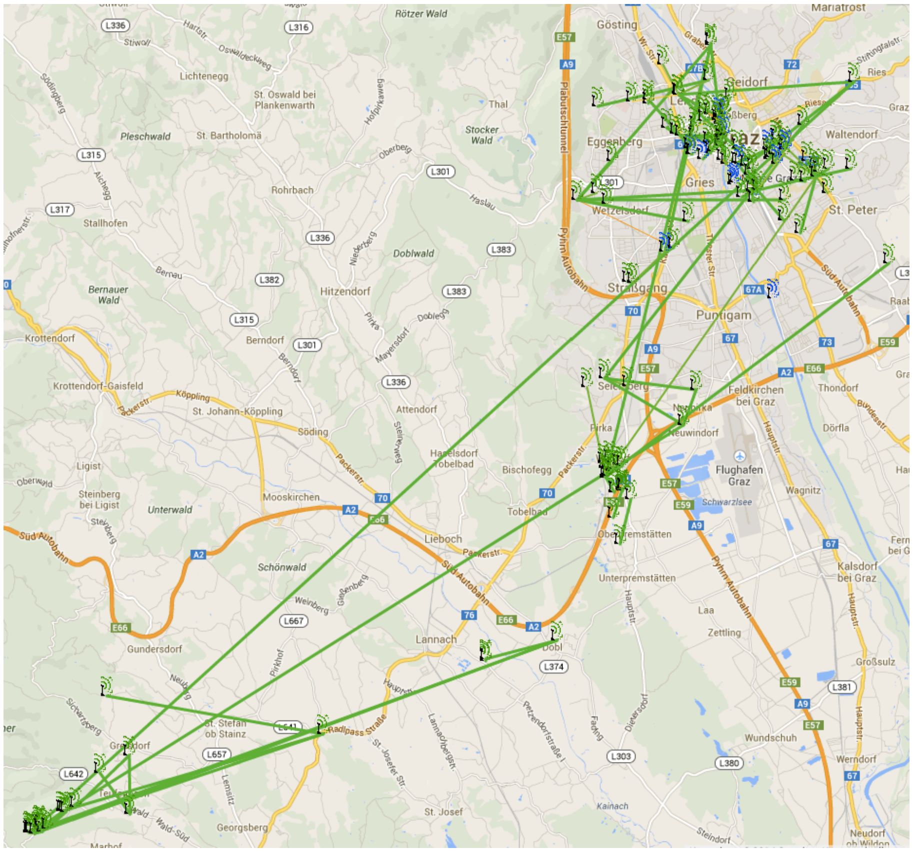
FunkFeuer: Graz

FunkFeuer Graz (FFGraz)[6] is the "smaller sister" of the FFWien network, situated in the homonymous city. It was founded after FFWien by Othmar Gsenger, Erwin Nindl and Roland Jankowski and has its own association to apply for local sponsoring. It consists of 144 nodes and 199 edges, with an average degree of 2.764.

[4] http://www.funkfeuer.at/

[5] http://www.funkfeuer.at/Vienna.206.0.html?&L=1

[6] http://graz.funkfeuer.at/



Degree	Ninux	FFWien	FFGraz
1	69.02	77.96	64.72
2	22.64	27.46	27.28
3	16.34	36.38	18.80
4	9.66	36.24	11.72
5	4.36	18.34	2.04
6	1.98	11.40	7.32
7	1.00	8.50	2.80
8	4.00	3.62	2.24
9	2.00	4.62	1.84
10	1.00	4.44	2.04
11	0.00	1.98	1.24
12	0.00	2.94	1.40

Degree	Ninux	FFWien	FFGraz
13	0.00	0.46	0.28
14	0.00	0.48	0.08
15	0.00	1.18	0.28
16	0.00	0.32	1.60
17	0.00	0.50	0.00
18	0.00	0.02	0.00
19	0.00	1.30	0.00
20	0.00	0.62	0.00
21	0.00	0.06	0.00

Table 4.1: Average degree frequencies in the three WCNs, over 50 samples

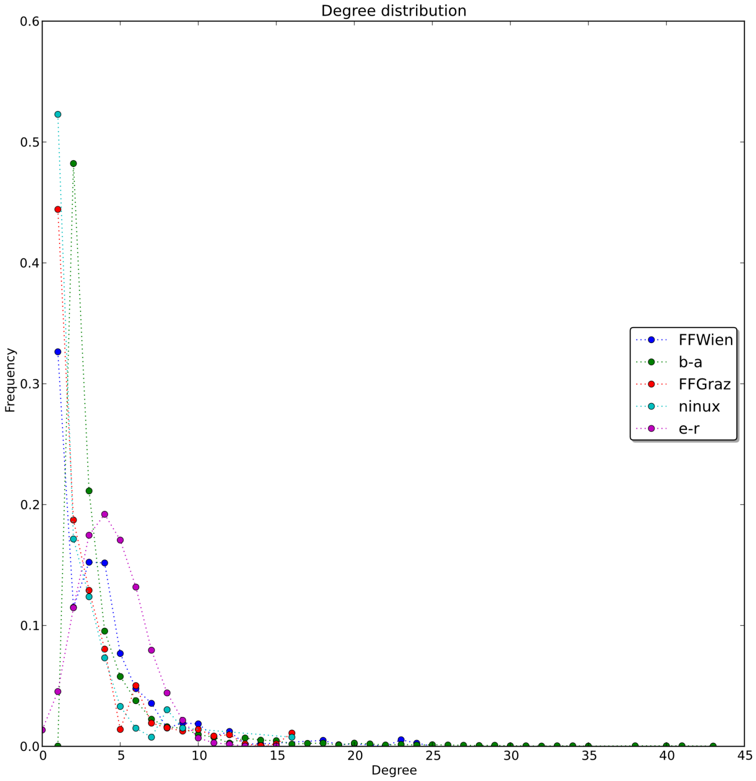


Figure 4.2: Degree distributions of the three WCNs, compared with the Erdős-Rényi and Barabási-Albert models

5 Robustness analysis

The first metric analysed is the robustness of the network. The chosen methodology is a variation of the percolation problem described in Chapter 16 of (Newman 2010).

Defining robustness is not trivial. While it is intuitive that removing nodes or links affects the network ability to successfully transport information, it is not obvious how this ability can be quantified. Moreover, nodes and links in a network are note all equal, neither in the impact of their removal nor in their probability of failure in the real world.

The first concern is traditionally addressed in literature by considering the connected components $C_0, \ldots C_n$ of the remaining graph. Specifically, define $|C_0|$ the order of the largest connected component. The robustness metric is then defined as the ratio

$$S = \frac{|C_0|}{|V|} \tag{5.1}$$

Then the inequalities of nodes and link must be taken into account. This is a more complicated matter because there is not a single correct solution. The approach largely depends on the real world situation to be analysed. For example, to evaluate the robustness of a network to random equipment failures the nodes can be removed in a random order. On the other hand, to simulate a targeted attack scenario the nodes with highest degree can be removed first, assuming attackers will direct their action to cause the highest possible damage. Other node or link metrics can be used in the same way, to simulate other scenarios.

Here different criteria have been used to gather a broad set of data. Specifically, nodes were first removed randomly with uniform probability, as in the classic percolation problem. Then they were removed following the order based on different centrality metrics (see chapter 3).

Practical considerations

To obtain meaningful results, the synthetic graphs generated with th random models need to be comparable with the real topologies at least in some aspects, the most obvious being average degree. In order to achive this, the parameters of the generators must be adjusted remembering that:

- for the Erdős-Rényi model, the expected average degree is

$$\langle k \rangle = \frac{2\binom{n}{2}p}{n} = \frac{2}{n}\frac{n!}{2!(n-2)!}p = \frac{1}{n}\frac{(n)(n-1)(n-2)!}{(n-2)!}p = (n-1)p \tag{5.2}$$

- for the Barabási-Albert model the exact average degree is known

$$\langle k \rangle = \frac{2(n-m)m}{n} \tag{5.3}$$

In the second case, since $m \in \mathbb{N}$, the value of $\langle k \rangle$ can not be controlled precisely once fixed n.

Results

As can be seen in figure 1-3, the networks have a similar behaviour when removing random nodes, but in the "attack" scenarios they exhibit very different properties. Of course, removing the more central nodes makes the networks fail much sooner, but there are marked differences.

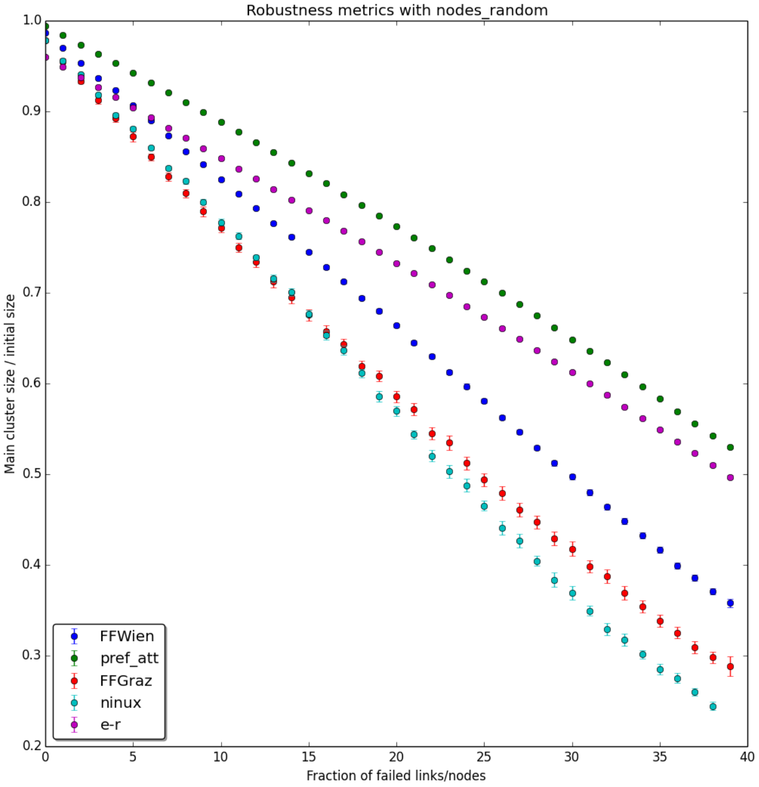


Figure 5.1: Random removal

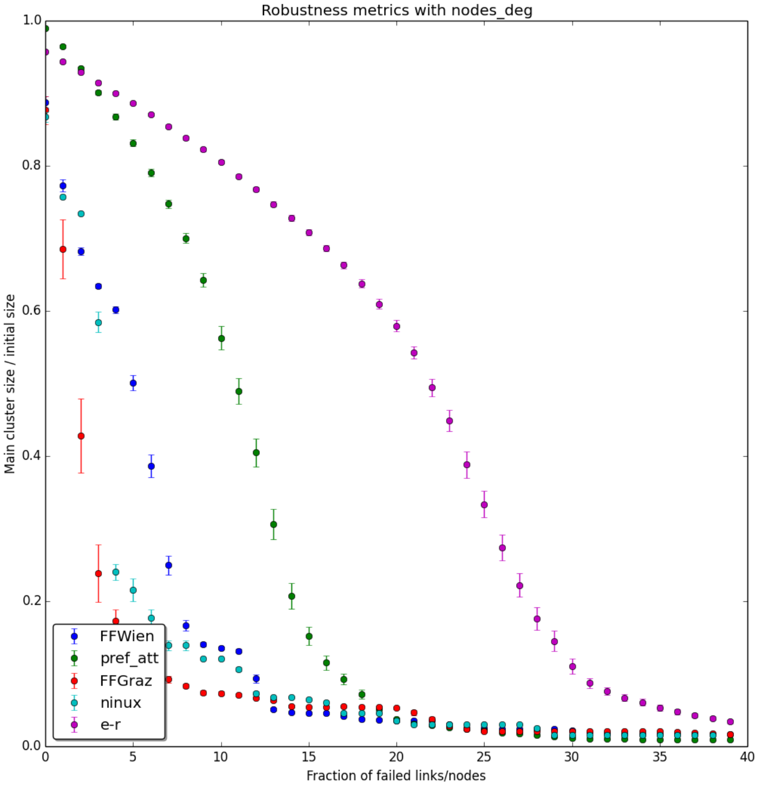


Figure 5.2: Removal by degree

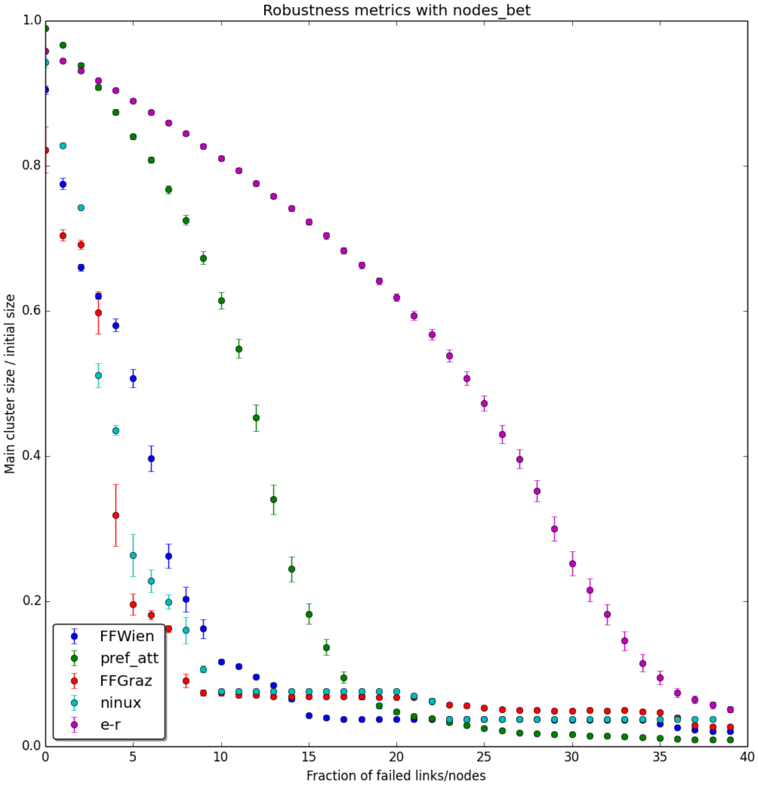


Figure 5.3: Removal by betweenness centrality

6 Message propagation analysis

The importance of routing

The robustness of a network is based on a static analysis of the connectivity of the network graph when removing nodes or links. A communication network, however, is a dynamic system where information needs to move between nodes. Moreover, the decentralised nature of computer networks means that the complete topology of the network is not necessarily the topology used to transmit information, depending on the routing protocol used for the network.

Given this, in order to understand the behaviour of a communication network we need to study the behaviour of its routing protocol with different underlying topologies. We are interested in the phase of topology discovery, where each node receives information on the existence of the other nodes in the network and (part of) the route to reach them.

Topology discovery in link state routing protocols is usually performed with each node flooding the network with some kind of link-state advertisement message. The possible variations are the flooding policy and the contents of the message. The most popular routing protocols used in mesh networks behave as follows:

- B.A.T.M.A.N. uses the simplest possible flooding (each node just performs a duplicate detection to avoid loops) and the advertisement message contains the sender address and a sequence number (for the duplicate detection)
- OLSR employs a more sophisticated flooding mechanism based on MPRs and the advertisement message contains the whole neighbourhood of the sender
- versions of OLSR used in practice usually force each node to be an MPR,[1] thus having an hybrid behaviour with flooding as in B.A.T.M.A.N.

Problem definition

The network is represented by a weighted graph $G = (V, E)$, where weights represent the probability of losing a packet on each link (we use the ETX

[1] Maccari (2013)

metric of OLSR for this purpose).

Each node creates a message with information on its neighbourhood and propagates it to each neighbour. Each node also propagates the message it receives, with a simple duplicate detection based on the sender to avoid loops.

Further iterations of the analysis will consider a subset of nodes generating and propagating the messages and a different protocol for loop avoidance.

Given the above situation, we define

$$T_u = \forall v \in N.\ \text{“node } v \text{ has a route to node } u\text{”}$$

$$R_u = \forall v \in N.\ \text{“node } u \text{ has a route to node } v\text{”}$$

Determine the probability of T_u and R_u for each node u in the graph.

Methodology

The propagation of a message with duplicate detection can be simulated with a Breadth First Search (BFS) over the graph. The most important variation is that before traversing an edge a random number is generated and compared to the packet loss probability of that link, to check if the transmission succeeds. During the BFS, the simulation keeps track of which nodes received the message and based on the content of the message determines the couples of nodes which have a known route between themselves. The search is repeated for every node as the starting point. The union of the results is then used to verify T_u and R_u.

The random simulation is run 1000 times to gather a significant figure of the probability of T_u and R_u.

The propagation for a node is as follows in pseudocode:

function propagate(Graph g, Node u)

```
message_sender <- u
message_content <- neighbourhood_of(u)
q <- Queue()
route_knowledge <- set()
u.visited <- True
for v in u.neighbours append (u,v) to q
while q is not empty do
    pop (u,v) from q
    n <- random()
    if (not v.visited) and (n   weight(u,v))
        v.visited <- True
        for i in message_content
            add (i,v) to route_knowledge
        for w in v.neighbours append (v,w) to q if w   u
return route_knowledge
```

The function is run for each node in the graph an the results are collected.

function propagate_all(Graph g)

```
rk <- set()
t, r <- array()
for u in g
    rk <- rk   propagate(g, u)
for u in g
    if (u,v)   rk   node v   u
        t[u] <- 1
    else
        t[u] <- 0
    if (v,u)   rk   node v   u
        r[u] <- 1
    else
        r[u] <- 0
return t, r
```

function run_simulation(Graph g, Integer n)

```
pt, pr <- array()
for n times
    t, r <- propagate_all(g)
    pt += t                                % sum each element
    pr += r                                % "
divide each element of pt by n
divide each element of pr by n
return pt, pr
```

Rationale

The feasibility of calculating T_u and R_u exactly has been evaluated. However, the computational complexity of this approach seems very high and likely does not justify its use in place of the Monte Carlo simulation.

Going into the details, the probability of a message propagating from node u to node v is easily calculated by… ~~summing the probabilities of success for every simple path between the two nodes.~~ With this result for every possible destination $v1 \ldots vn$ of a message transmitted by u, it's theoretically possible to calculate T_u but it's not easy: the events "reaching v1"…"reaching vn" are not independent, so the probability of “reaching every node” is not the product of their probabilities.

For example, if w is in any simple path from u to v, the probability of success between u and v changes if it is known that node w has been reached.

$P(v) = P(w) \cdot P(v|w) + P(\neg w) \cdot P(v|\neg w)$

The value of $P(v|w)$ and $P(v|\neg w)$ is not so obvious:

- $P(v|w)$ is the sum of the probabilities of the subpaths $w \to v$ for every simple path uv
- $P(v|\neg w)$ is the sum of the probabilities of success for simple paths from u to v excluding the paths that contain w

This must be computed for every w that appears in at least one simple path $u \to v$. Again, this computation must be repeated for every possible source-destination pair u, v.

7 Conclusions

Bibliography

Barabási, Albert-László, and Réka Albert. 1999. "Emergence of Scaling in Random Networks." *Science* 286 (5439): 509–12. doi:10.1126/science.286.5439.509. http://www.sciencemag.org/content/286/5439/509.

Clausen, T., and Philippe Jacquet. 2003. "Optimized Link State Routing Protocol (OLSR)." IETF. http://tools.ietf.org/html/rfc3626.

De Couto, Douglas SJ. 2004. "High-Throughput Routing for Multi-Hop Wireless Networks." PhD thesis, MIT. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.117.3059&rep=rep1&type=pdf.

Gilbert, E. N. 1959. "Random Graphs." *The Annals of Mathematical Statistics* 30 (4): 1141–44. doi:10.1214/aoms/1177706098. http://projecteuclid.org/euclid.aoms/1177706098.

"List of Wireless Community Networks by Region. Wikipedia, the Free Encyclopedia." 2014. June 10. https://en.wikipedia.org/w/index.php?title=List_of_wireless_community_networks_by_region&oldid=612342893.

Maccari, Leonardo. 2013. "An Analysis of the Ninux Wireless Community Network." In *Wireless and Mobile Computing, Networking and Communications (WiMob), 2013 IEEE 9th International Conference on*, 1–7. IEEE. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6673332.

Newman, Mark. 2010. *Networks: An Introduction.* Oxford University Press, USA.