# OMNeT++ based Opportunistic Routing Protocols Simulation: A Framework

Zhongliang Zhao, Torsten Braun
Institute of Computer Science Applied Mathematics, University of Bern
Neubrückstrasse 10, 3012 Bern, Switzerland
Email: {zhao, braun} @iam.unibe.ch

***Abstract*—This paper describes a framework for simulating opportunistic routing protocols in the INETMANET framework of OMNeT++. The proposed modules adopt an abstraction of the generic functions of the most representative opportunistic routing algorithms. The main contribution is an OMNeT++ modeling architecture that could be extended to implement different opportunistic routing schemes. Our work provides an analysis of the most representative opportunistic routing algorithms. We decouple the opportunistic routing schemes into four procedures - *Forwarder Candidate Selection*, *Forwarder Selection*, *Forwarder Role Change Notification* and *Collision Avoidance*. Different protocols should have specific implementations of each procedure. In the framework, these four procedures are defined as virtual functions and act as implementation stubs such that different protocols could be implemented by overriding them in the derived function according to their distributed strategies.**

## I. Introduction

Opportunistic networks are one of the most interesting evolutions of MANET, in which mobile nodes are enabled to communicate with each other even if there is no source-destination route available. Routes are built dynamically, while messages are en route between the sender and the destination. Any node may opportunistically act as next relay, given that it is likely to bring the message closer to the final destination. One important feature is that the multiple receivers of a message negotiate and decide which packet to forward but not the sender (receiver-based forwarding).

OMNeT++[1] is an open-source modular simulation platform that has primarily been used for simulating wired and wireless communication networks. It includes, and is continuously complemented by, multiple modeling frameworks like INET, INETMANET, MiXiM, etc. The INETMANET framework includes multiple radio wave propagation models, simple battery models and supports multi-radio communications, which are the key characteristics for simulation of opportunistic routing protocols. MiXiM is another popular framework for wireless network simulation. However, it focuses more on the modeling of low layer protocols and because our focus is at routing, we prefer INETMANET as our implementation supporting platform.

The idea of this work is based on the fact that OMNeT++ (including INETMANET) lacks support for some of the key features of opportunistic routing protocols. Opportunistic routing tries to take advantage of the time-varying nature of wireless environment to provide hop-by-hop packet forwarding in scenarios where traditional MANET routing may not perform well. The goal is to implement a framework for simulating opportunistic routing protocols with the INETMANET framework in the OMNeT++ simulator.

## II. Related Work

Opportunistic routing protocols make use of the broadcast nature of wireless communications during data forwarding by taking advantage of the transient nature of channel and node availability. This design principle seems to be a countermeasure for the situation of mobile ad hoc networks, where nodes are highly mobile and wireless propagation is inherently instable, making network topology changes frequently. Typical MANET routing protocols may not perform ideally in those scenarios.

Various opportunistic routing protocols have been proposed. BLR [2] is a geographic routing protocol that uses location data to minimize the routing overhead by eliminating the periodic *Hello* message. Data packets are broadcasted and the protocol takes care that just one of the receiving nodes forwards the packet. ExOR [3] pioneers the concept of being opportunistic when wireless links are weak by using the broadcast nature of wireless communication. In ExOR, the sender specifies a list of candidate nodes in the packet header, which are potential forwarders of the packet. The receivers relay the packet according to its priority in the list by negotiating with surrounding nodes. MORE [4] is a MAC independent protocol that combines the idea of opportunistic routing and network coding to utilize spatial reuse. MIXIT [5] improves the throughput by applying network coding at the physical layer. SOAR [6] uses priority-based timers to make sure that the most preferred node forwards the packet with little coordination overhead. MCExOR [7] extends opportunistic routing to multi-channel environments. The use of multiple non-overlapping RF channels contributes to the reduction of overall interference and the throughput increases superproportionally. In Chapter 3, we recapitulate the general strategies of the most representative opportunistic routing protocols and abstract them as the core of the framework.

Most of the existing opportunistic routing algorithms are evaluated with specific simulators. ONE [8] is probably the most successful simulator specifically designed for evalu-

ating DTN and opportunistic routing protocols. It allows users to create scenarios based upon different synthetic movement models or real-world traces to offer a framework for implementing routing and application protocols. However, ONE focuses on the modeling of the behavior of store-carry-forward networking, and hence refrains from detailed modeling of the low layer mechanisms such as signal attenuation and congestion of the physical medium. Instead, the radio link is modeled as a communication range and a link with a certain bit-rate, which are assumed to be constant over the simulation. All these limitations make ONE imperfect for simulating opportunistic routing protocols, which heavily and inherently make use of links that packet delivery is possible with low variational probability and should easily accommodate with channel fluctuation. INETMANET contains numerous implementations for each of the ISO/OSI layers, from physical/data link layer to application layer, which are essential to build the opportunistic routing framework. Later in Chapter 3, we will show how to adopt the existing components of INETMANET, such as mobility model, battery module, multi-radio module, radio propagation module and multiple link layer protocol implementations(802.11.a/e/g and 802.15.4) to set up our framework.

There have been earlier works in the MANET community to develop frameworks, with integrated functionalities for implementing ad hoc routing protocols. ASL [11] and FRAd-Hoc [12] present such routing frameworks in mobile ad-hoc networks. [13] provides a MANET routing protocol framework for the OMNeT++ community. [14] designs a framework for opportunistic routing protocols in ad-hoc networks, but it targets to emphasize that the throughput gain achieved by opportunistic routing is not clearly attributed to the opportunistic selection of forwarder but also partly due to its acknowledgement and scheduling features which may also be implemented by traditional MANET routing protocols. It does not focus on the compositional architecture of generic opportunistic routing protocols. Our framework, consisting of abstract components and common functionalities, builds an architecture for designing and implementing opportunistic routing protocols.

## III. Design & Implementation

In this section, we describe the general architecture of our framework.

### A. Framework Architecture

Our implementation is based on the work of the OPPONET project [9] [10], which provides basic mechanisms for simulating opportunistic and delay-tolerant networks in OMNeT++. OPPONET allows simulating *open* systems of wireless mobile nodes where synthetic or real mobility traces are used to drive the simulations. One possible application scenarios of opportunistic routing are Unmanned Aerial Vehicle(UAV) ad hoc networks, where UAV might leave the networks for charging and rejoin the system afterwards. However, OPPONET is too much limited to the mobility modeling (scripted mobility) and object creation. Moreover, it does not focus on routing aspects.

An example of an opportunistic routing node in the OMNeT++ simulator can be found in Figure 1. The ***Navigator*** module is responsible for the movement of nodes. It is designed as a module interface, which should be implemented as specific mobility model like Random Walk or Random Waypoint. Module implementation is derived from OPPONET with slight modifications. During the simulation, the configuration file *omnet.ini* could be set to use the generated xml-formatted trace file, which was beforehand produced by the mobility generation tool of OPPONET to control the movement and subsistence of the nodes.

The ***Controller*** module is simply in charge of the initialization of the node. It mainly includes the channel utilization, packet storage management and other functionalities.

The ***WNIC*** module is an implementation of a wireless network interface controller, composed of physical and MAC layer. We choose IEEE 802.11 from INETMANET as our WNIC implementation, which includes the IEEE 802.11a/e/g implementation. A node could have multiple WNIC modules working at different radio frequencies to support multi-radio communication. The *ChannelControlExtend* module from INETMANET is adopted to implement the multichannel related functions.

The ***EnergyManager*** module is derived from the INETMANET *InetSimpleBattery* module and is a simple energy related implementation. The *InetSimpleBattery* module provides a linear model of batery usage with fairly coarse estimate of battery consumption, together with little computational overhead.

The ***NotificationBoard*** is employed for modules to notify each other about the "event" of state changes, such as interface status changes (up/down), mobile node position updates, etc. The NotificationBoard acts as an intermediary between modules, where state changes can occur, and modules that are interested in learning about those changes. Modules should "subscribe" to the notification categories they are interested in. The *NotificationBoard* module from INETMANET is adopted in our framework.

The ***OppRoutingProtocol*** module is the core component of the framework. It is implemented as a simple module such that it could be easily extended. It abstracts the general functions of the most representative opportunistic routing algorithms and modularizes them such that a specific protocol could be implemented by extending the module. For example OppRoutingProtocolExOR is the ExOR implementation module by extending the OppRoutingProtocol.

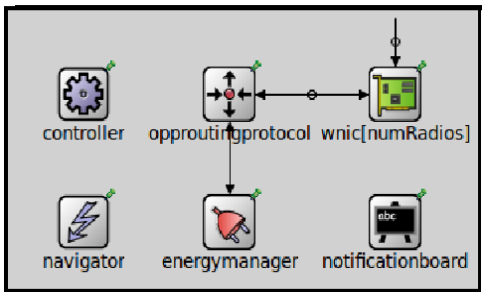


Figure 1. Opportunistic Routing Node Structure in OMNeT++ Simulator.

### *B. OppRoutingProtocol Module*

Opportunistic routing differs from the traditional ad hoc routing in that it exploits the broadcast nature of the wireless medium by deferring the route selection decision to the receiver side. Clearly, this feature copes well with the unreliable and unpredictable characters of wireless transmission. However, opportunistic routing requires coordination among the potential forwarding nodes, which means additional overhead. The major challenge in designing opportunistic routing is to maximize the routing progress of each individual packet transmission and keep the additional coordination overhead as low as possible.

As our focus is mainly on the opportunistic routing protocols, it makes sense to analyze the kernel of the most representative protocols and perceive some fundamental structures. By interpreting the five distinguished opportunistic routing protocols presented in Chapter 2 and references, we describe a general picture of opportunistic routing and decompose it into four phases. We think these should be the key features of an efficient opportunistic routing algorithm. The four phases are:

- *Forwarder Candidates Selection*
- *Forwarder Selection*
- *Forwarder Role Change Notification*
- *Collision Avoidance*

**Forwarder Candidates Selection** is the first procedure of the routing protocol. The sending node utilizes the peer-discovery service provided by the WNIC module. It periodically polls the node factory to check the nodes inside its range. Certain attributes (e.g., geographic region or nodes movement tendency) are adopted additionally to build the set of potential next-hop nodes. The design of these attributes should take into account that only the nodes that are closer to the destination or that have the movement towards it, should be the candidates. The frequency of the polling operation should be correlated with the nodes' speed and the rapid change of the network topology.

**Forwarder Selection** defines rules how the actual forwarding node is picked from the candidates set. Each node inside the candidates set will be added into a peer collection and marked as unreachable once the WNIC reports its unreachability. Unreachable peers will remain in the collection for a period, which enables the re-acquisition of nodes that are temporarily unreachable in an intermittent environment. One design proposal is that the sending node periodically broadcasts a message containing its current available channels, transmission bit-rate, movement statistical information. Candidates that successfully receive these packets will consider the status of these information, its remaining battery lifespan, and the pre-calculated Expected Transmission Count(ETX)/Expected Any-path Transmission(EAX) metrics to the destination. A comprehensive utility function will be executed, based on the combination of the ETX/EAX value and the relationships between the received and local data. Each candidate will return an utility value and all the successfully received candidates will share its value with others. The candidate with the highest value will be the one winning the election process.

**Forwarder Role Change Notification** enables the winning forwarder to announce its new role and responsibility to surrounding nodes, to make them aware of the selection winner and stop the competition. This procedure is important because if it is well-designed, the duplicated transmission could be avoided. Otherwise duplicated transmission leads to retransmissions, which means additional overhead. A possible implementation could be that the selected forwarder broadcasts a "StartToSend(STS)" packet to indicate the start of data transmission, including the adopted channel usage and bit-rate. The data transmission will start if no more messages are received within an interval after the STS. In the framework, we implement this module as a broadcast function. The data transmission to a peer will be aborted whenever the node is detected as lost by the WNIC module.

**Collision Avoidance** concerns how the nodes, which wish to access the wireless medium at the same time and contend for the channel. A subsequent contention resolution mechanism must be defined. Contention could happen in two cases: the first case is imperfect design of the Forwarder Role Change Notification process, which leads to duplicated transmission; the second case is when two or more nodes want to send packets at the same time, which could result in packet collisions. To avoid this, multiple channel access mechanisms could be applied, such as CSMA-CA.

These four procedures are defined as ***virtual functions***, which just have the general interfaces with necessary data structures. Concrete routing modules need to be created for respective protocol by extending the *OppRoutingProtocol* module. The four virtual functions are defined as following:

- ***candidateSelection(Src, Dst)***: This function returns a vector of nodes by selecting the candidate nodes as the potential relays to a given destination(Dst), from the neighbors of a given node(Src) based on specific rules.
- ***forwarderSelection(HostVector)***: This function returns a forwarder from the candidates set(HostVector).
- ***roleChangeNotification(Host)***: This function broadcasts a message notifying the Host's surrounding nodes about its new role. The receiving nodes will stop competing for the channel access.

- ***collisionAvoidancne()***: This function avoids that two nodes attend to access the medium at the same time.

Besides the core virtual functions, there are some other common functionalities, which are fundamentals for most opportunistic routing protocols. The framework also includes the implementation of these shared functions. Although there might be differences for each protocol(some protocols may not explicitly include all the four procedures), we believe that most of the protocols could be easily adapted to use the common mechanisms provided by the framework. These common utility tasks include:

- *Neighbor Discovery & Management*
- *Packet Broadcasting*
- *Packet Buffer Management*
- *Transmission Reliability Control*
- *Time Scheduling*
- *Node Interface Management*
- *ETX/EAX Calculation*

**Neighbor Discovery & Management:** Nodes need to detect neighbors that are physically reachable in one hop. Neighbor detection is essential for opportunistic routing because a well-designed neighbor detection mechanism acts as a basis for forwarder selection. Neighbor management service of INETMANET is adopted to control neighbors.

**Packet Broadcasting:** In almost all routing protocols, nodes have to distribute information throughout the network. An adapted implementation from OMNeT++/INETMANET is provided.

**Packet Buffer Management** is another compulsory operation for nodes. Nodes need to store received packets and do other manipulations. Potential data structures and corresponding operations are defined inside the framework to fulfill this task.

**Transmission Reliability Control:** In the simulation, the delivery of a packet from one node to another has a predetermined probability. OMNeT++ assigns three parameters to each link: propagation delay, bit error rate and data rate. INETMANET includes numbers of channel propagation models, which provides a detailed simulation basis for transmission control. The INETMANET link layer implementation is adapted to control the packet transmission.

**Time Scheduling** plays a vital role in opportunistic routing, because nodes need to schedule their transmission based on the information they observe from the Transmission Reliability Control. An accurate time scheduling mechanism could avoid collision.

**Node Interface Management:** Nodes inside the network may be equipped with more than one physical antennas to increase the network throughput. The management for multiple interfaces is necessary for benefiting from more antennas, i.e., to support multichannel communication. This function is to be implemented in the future.

**ETX/EAX Calculation:** Most of the "Candidate Selection" processes of the opportunistic routing protocols are based on the same principle that the source node predetermines a forwarder priority list based on the estimates of the path loss rates according to ETX/EAX value. This function is implemented to calculate the ETX/EAX of each node pair.

Table I
BASIC DATA STRUCTURES

| *Data Structure* | *Implementation* | *Function* |
|---|---|---|
| Packet Message | *typedef struct packet* Packet | Packet Format |
| Packet Buffer | std:list $\langle Packet \rangle$ PacketQueue | Stores received packets |
| Host Node Entry | *typedef* struct HostEntryExtendedExOR HostEntryExtended | Node structure type definition from ChannelControlExtended |
| Host Node Entry Vector | std::vector $\langle HostRefExtended \rangle$ HostRefExtendedVector | Vector of pointers to the HostEntryExtended, stores the pointers to nodes |

### C. Message Format & Data Structure

The message format is also an important issue of the OMNeT++ simulation, because it triggers the basic event handlers. In our framework, implementation of the message format follows the C++ language structures *struct*, such that all the fields could be easily manipulated. A general message structure of the opportunistic routing protocols is defined in the *OppRoutingProtocol* simple module, which includes message id, source/destination node id, etc. One important composition would be a set consisting of ranked nodes, which are selected as the candidate forwarders based on certain metrics. We implement this as a *vector* container from C++ Standard Template Library(STL) libraries using C++ generic programming. Because this will facilitate the operation of a user-defined data type according to interested metrics, e.g., prioritize nodes according to the ETX value. Some additional data structures are listed in the Table I.

## IV. EVALUATION STRATEGY

To illustrate the usefulness of the framework, we now demonstrate its use in building the well-known ExOR routing protocol, show the process how to define modules to accommodate the main procedures of ExOR. Figures 2&3 are the flow charts of source/forwarder node of ExOR protocol. The defined virtual procedures and implemented common functionalities could be found embedded to show their roles (presented as dashed frame).

Demonstration of the ExOR implementation flow charts evaluates our framework in terms of its configurability and expressibility. Next we plan to provide a preliminary evaluation of the overhead of the framework. This will be based on a performance comparison between an existing standalone ExOR implementation and the framework-based ExOR implementation to see the performance loss by moving from a dedicated to a framework-based implementation. The evaluation metrics we will use include: forwarding overhead, end-to-end delay, memory footprint, initialization

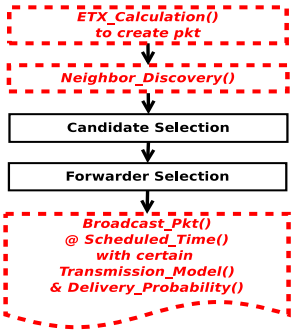


Figure 2. Flow Chart of Source Node in ExOR.

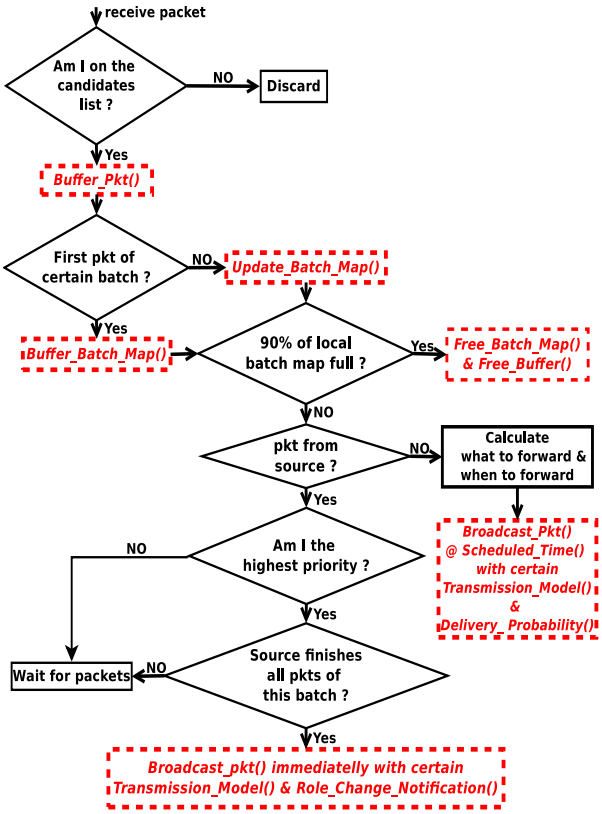


Figure 3. Flow Chart of Forwarder Node in ExOR.

time, etc. The gain of the framework to protocol developers will also be analyzed, e.g., in terms of the number of lines of code required to implement a protocol.

## V. Conclusion & Future Work

The main contribution of the work is an OMNeT++ modeling framework that could be extended to implement different opportunistic routing schemes. Our framework provides a straightforward analysis of the most representative opportunistic routing algorithms. We decouple the opportunistic routing schemes into four tasks - *Forwarder Candidate Selection*, *Forwarder Selection*, *Forwarder Role Change Notification* and *Collision Avoidance*. Different protocols should have specific implementations of each phase. These four functions are defined as virtual functions in the framework and act as implementation stubs such that different protocols just override them in the derived function according to their distributed strategies.

In the future, our short-term goal is to fulfill the framework and finish the existing opportunistic routing protocol implementations, such as ExOR and MORE. As a result, we aim to build an opportunistic routing protocol library. Long-term goals include developing new opportunistic routing mechanisms and mobility models by learning from real-world measurement of specific scenarios (i.e. UAV ad hoc networks) using the components provided by the framework.

## Acknowledgment

The work presented in this paper is partly supported by the Swiss National Science Foundation under grant number 200021-130211/1.

## References

[1] A.Varga. The omnet++ discrete event simulation system. *Proceeding of European Simulation Multiconference*, June 2001.

[2] T. M.Heissenbutel, T.Braun and M.Walchli. Blr: beacon-less routing algorithm for mobile ad hoc networks. *Computer Communication Journal*, 2004.

[3] S.Biswas and R.Morris. ExOR: Opportunistic routing in multi-hop wireles networks. *Proceedings of ACM SIGCOMM*, Philadelphia, Pennsylvania, August 2005.

[4] S.Chachulski. Trading structure for randomness in wireless opportunistic routing. *Proceeding of ACM SIGCOMM*, 2007.

[5] S.Katti and D.Katabi. Symbol-level network coding for wireless mesh networks *Proceeding of ACM SIGCOMM*, Seattle, WA, USA, August 2008.

[6] E.Rozner and J.Seshadri. Simple opportunistic routing for wireless mesh networks. *Wireless Mesh Networks*, 48-54, Reston, VA, USA, 2006.

[7] A.Zubow and M.Kurth. Multi-channel opportunistic routing. *European Wireless*, 2007.

[8] Ari Keränen and Jörg Ott and Teemu Kärkkäinen. The ONE Simulator for DTN Protocol Evaluation. *SIMUTools '09: Proceedings of the 2nd International Conference on Simulation Tools and Techniques*, Rome, Italy, 2009.

[9] O.R.Helgason and K.V.Jonsson. Opportunistic Networking in OMNeT++ *SIMUTools '08: Proceedings of the 1st International Conference on Simulation Tools and Techniques*, 2008.

[10] K.V.Jonsson. A Gateway for Wireless Dissemination of Delay-Tolerant Content. *Master Dissertation*, Janurary 2008.

[11] V. Kawadia, Y. Zhang and B. Gupta. System Services for Ad-Hoc Routing: Architecture, Implementation and Experiences. *MobiSys '03: Proceedings of the 1st international conference on Mobile systems, applications and services*, 2003.

[12] U. Correa, C. Montez, V. Mazzola, M. A. R Dantas. Frad-Hoc: A Framework to Routing AD-Hoc Networks. *IFIP International Federation for Information Processing*, Vol. 212, page 71-82, 2006.

[13] A. Ariza-Quintana, E. Casilari and A. Trivio Cabrera. Implementation of MANET routing protocols on OMNeT++. *SIMUTools '08: Proceedings of the 1st International Conference on Simulation Tools and Techniques*, 2008.

[14] N. Gazoni, V. Angelakis, V. A. Siris and B. Raffaele. A framework for opportunistic routing in multi-hop wireless networks. *PE-WASUN '10: Proceedings of the 7th ACM workshop on Performance evaluation of wireless ad hoc, sensor, and ubiquitous networks*, 2010.