Use 'em or lose 'em: On unidirectional links in reactive routing protocols

Thomas Heide Clausen, Juan-Antonio Cordero, Jiazi Yi, Yuichi Igarashi

Use 'em or Loose 'em: On Unidirectional Links in Reactive Routing Protocols

Thomas Clausen[a], Juan-Antonio Cordero[a], Jiazi Yi[a,*], Yuichi Igarashi[b]

[a] *École Polytechnique, Route de Saclay, Palaiseau, France*
[b] *Hitachi Ltd., Research & Development Group, Japan*

Abstract

In reactive unicast routing protocols, Route Discovery aims to include only bidirectional links in discovered routing paths. This is typically accomplished by having routers maintain a "blacklist" of links recently confirmed (through Route Reply processing) to be unidirectional – which is then used for excluding subsequent Route Discovery control messages received over these links from being processed and forwarded.

This paper first presents an analytical model, which allows to study the impact of unidirectional links being present in a network, on the performance of reactive routing protocols. Next, this paper identifies that despite the use of a "blacklist", the Route Discovery process may result in discovery of *false forward routes*, *i.e.,* routes containing unidirectional links – and proposes a counter-measure denoted *Forward Bidirectionality Check*. This paper further proposes a *Loop Exploration* mechanism, allowing to properly include unidirectional links in a discovered routing topology – with the goal of providing bidirectional connectivity even in absence of bidirectional paths in the network.

Finally, each of these proposed mechanisms are subjected to extensive network simulations in static scenarios. When the fraction of unidirectional links is moderate ($15-50\%$), simulations find *Forward Bidirectionality Check* to significantly increase the probability that bidirectional routing paths can be discovered by a reactive routing protocol, while incurring only an insignificant additional overhead. Further, in networks with a significant fraction of unidirectional links ($\geq 50\%$), simulations reveal that *Loop Exploration* preserves the ability of a reactive routing protocol to establish bidirectional communication (possibly through non-bidirectional paths), but at the expense of a substantial additional overhead.

Keywords: unidirectional links, routing, ad hoc network, reactive protocol

1. Introduction

The routing protocols for ad hoc networks have two main categories: proactive and reactive. Proactive protocols rely on periodic routing control message exchange to build routes before they are actually required. Reactive protocols, on the other hand, trigger routing control messages only when a path is desired. The basic operation of reactive routing protocols, *e.g.,* DSR [1], AODV [2], LOADng [3], is route discovery, illustrated in figure 1: a router with a packet to deliver to a destination, and which does not have a valid entry in its routing table for that destination, will issue a *Route Request* (`RREQ`) (figure 1b), diffused through the network (figure 1b-c-d) so as to reach all other routers. When a router forwards this `RREQ`, it records an entry in its routing table towards the originator of that RREQ – a *reverse route* indicating the eventual path from the destination to the originator. If the destination is present in the network, it will eventually receive the `RREQ`, and respond by a unicast *Route Reply* (`RREP`) (figure 1e), sent along the previously installed reverse route towards the originator of the `RREQ`. The routers forwarding the `RREP` will install a *forward route* towards the destination. Once the `RREP` arrives at the originator of the corresponding `RREQ`, a bidirectional path has thus been installed, and is available for use.

If multiple `RREQs` (with the same sequence number) are received by a destination D from the same source S, D will send a `RREP` in reply to "the `RREQ` corresponding to the shortest path". If an intermediate router x receives two `RREQs` from the same S, for the same D, and with the same sequence number, x will forward "the `RREQ` corresponding to the shortest path" and ignore subsequent `RREQs`, as illustrated in figure 1d-e, where the `RREQ` received by router B from router E is ignored.

When a link is detected to be broken (typically through a link-layer notification of a data-packet failing to be delivered to a next hop), the detecting router may engage in a *route-repair* operation – essentially a new `RREQ`/`RREP` cycle to discover a path to the destination – and if that fails, issue a *route-error* (`RERR`) message to inform the source of the failed data-packet of the error.

This *basic* Route Discovery process, will neither discover nor avoid unidirectional links. If, with reference to figure 1, a unidirectional link $A \rightarrow B$ exists on that short-

*Corresponding author
Email addresses: Thomas@ThomasClausen.org (Thomas Clausen), Juan-Antonio.Cordero-Fuertes@polytechnique.edu (Juan-Antonio Cordero), Jiazi@JiaziYi.com (Jiazi Yi), yuichi.igarashi.hb@hitachi.com (Yuichi Igarashi)

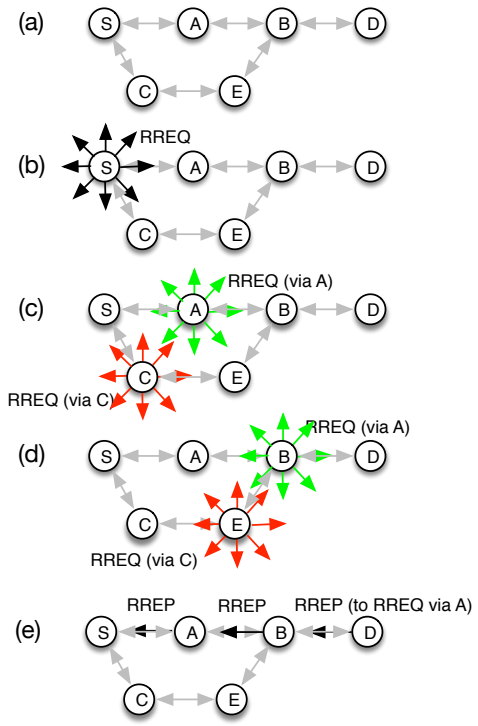

Figure 1: An example illustrating the Route Discovery process for a reactive routing protocol. The gray arrows indicate the underlaying L2 connectivity (a). S seeks a route to D, issues an `RREQ` (b) which is flooded through the network (b-c-d) until it reaches the destination D – which responds by generating and unicasting an `RREP` (e).

est path, the `RREP` will never reach S — which will experience a timeout and restart Route Discovery. Subsequent Route Discoveries will face the exact same situation: even if a bidirectional path $S \leftrightarrow C \leftrightarrow E \leftrightarrow B \leftrightarrow D$ exists, it will not be discovered because of the unidirectional link $A \rightarrow B$.

Excluding unidirectional links is reasonable, even necessary, when the underlying data-link layer itself requires bidirectionality for data traffic forwarding – such as if data-link layer acknowledgement of successful packet delivery are used, *e.g.,* ContikiMAC [4], X-MAC [5], IEEE 802.11b DCF or IEEE 802.15.4-2015 TSCH [6]. Reactive protocols therefore include mechanisms for detecting unidirectional links, and for excluding them from routing paths, and from being used as part of routing protocol operation. These mechanisms are *Reverse Bidirectionality Check* (*RBC*) (expecting *Route Reply Acknowledgements*, denoted `RREP_ACKs`, sent in response to `RREPs`) and *Blacklists* (ignoring subsequent route discovery messages from a neighbour, who did not send an `RREP_ACK` in response to an `RREP`) – both detailed in section 4.

1.1. Statement of Purpose

Even when the underlying layer 2 protocol does not require link bidirectionality, unidirectional links, when present in the network, pose a number of challenges for reactive routing protocols. These links can be detected (by way of *RBC*) and excluded (by way of *Blacklists*), at best, at the cost of repeated (costly) Route Discovery operations, but this entails the risk of rendering a network disconnected. Indeed, excluding unidirectional links from route discovery and computation may cause artificial network partitioning and prevent communication between nodes for which bidirectional connectivity is topologically available. Figure 2 illustrates a case in which routers A and D can communicate with each other (via unidirectional paths $A \rightarrow B \rightarrow D$ and $D \rightarrow C \rightarrow A$), although no bidirectional path between A and D is available.

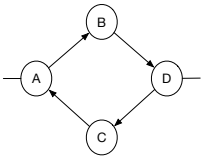

Figure 2: Example of unidirectional loop: links (A, B), (B, C), (C, D) and (D, A) are unidirectional. There is bidirectional connectivity between A and B, but there is no bidirectional path between A and B.

The exclusion of unidirectional links in this case would partition the network, in which any pair of routers *could* otherwise communicate, in four isolated regions – one per router. Moreover, detecting and excluding unidirectional links between A and D, by using existing mechanisms such as *RBC* and *Blacklisting*, would actually not be sufficient to prevent the installation of *"false forward routes"*, *i.e.,* paths which contain unidirectional links in the direction $A \rightarrow D$.

The objective of this paper is (i) to study the performance characteristics of reactive routing protocols, when facing unidirectional links, (ii) to analyse the cases in which existing mechanisms (notably *RBC* + *Blacklisting*) fail to properly exclude unidirectional links from discovered routing topologies and cause *false forward routes* to be installed, and (iii) to develop and propose mechanisms rendering reactive routing protocols resilient to, and able to properly exploit unidirectional links without relying on any propagation model nor specific layer 2 types when feasible.

This paper proposes two new mechanisms, the *Forward Bidirectionality Check* (*FBC*) and the *Loop Exploration* (*LE*). FBC is useful for properly identifying unidirectional links in the network (without incurring in "false forward routes") and *LE* allows to exploit them, if possible, for route discovery. FBC can thus be used over any data-link technology, whether data-link acknowledgements are used or not. LE, in contrast, is only useful over a data-link technology that does not filter out unidirectional links.

1.2. Paper Outline

The remainder of this paper is organized as follows: section 2 overviews related work on unidirectional links in reactive routing protocols, for the purpose of positioning and providing a context for the work of this paper.

Section 3 provides – by way of an analytical model – a study of the performance and behaviour of reactive routing protocols, when exposed to network topologies containing unidirectional links.

Unidirectional links may, in different ways, cause a network to appear disconnected (to the routing protocol, and thus to higher layers). Section 4 first explores the unidirectional link exclusion mechanism of *Reverse Bidirectionality Check* (*RBC*) + *Blacklisting* in detail. This mechanism, part of the `RREP` forwarding, is found to be insufficient to identify all unidirectional links, and may cause *false forward routes* (*i.e.,* paths containing unidirectional links in the opposite direction of the intended traffic flow) to be installed as part of the Route Discovery process. To avoid this, this paper proposes the alternative mechanism *Forward Bidirectionality Check* (*FBC*), integrated in the `RREQ` flooding.

Section 5 presents an algorithm, inspired by the approach taken by [7, 8], for detecting and utilising unidirectional links in reactive unicast routing protocols, using the *Loop Exploration* (*LE*) technique.

Both proposed algorithms (*FBC* and *LE*) are implemented and tested in a network simulator, and a comprehensive simulation study is presented in section 6. Section 7 concludes this paper.

2. Related Work

Routing protocols either generally assume that links in the network are bidirectional (symmetric), or include specific mechanisms for ensuring that unidirectional (asymmetric) links are excluded from being considered for inclusion in the routing topology [9]. Yet, unidirectional links are common in wireless multi-hop networks [10, 11, 12], in particular in wireless sensor networks (WSNs), and in low-power and lossy networks (LLNs) [13, 14].

The impact of unidirectional links on the performance of flat routing protocols, in particular on a distance-vector protocol (DSDV), is studied in [10, 15]; these studies conclude that adapting distance-vector routing to use unidirectional links requires increasing the control message exchange rate (*i.e.,* the control overhead) from $O(n)$ to $O(n^2)$. This is confirmed in [11], which concludes that managing the topology in a network with unidirectional links leads to a substantial increase of the routing overhead, and provides a questionable performance benefit. Furthermore, [12] observes that the effect of unidirectional links in a generic network cannot be exclusively deduced from the fraction of links that are unidirectional, and proposes a set of additional metrics (reachability, neighbours, paths, change rate) to model and characterise the routing impact in mobile ad hoc networks.

Different approaches have been proposed to address unidirectional links. Some explore cross-layer or data-link layer solutions [16, 17, 18, 19, 20] to make unidirectional links transparent to the network layer. Others propose to handle unidirectional links at the data-link layer and thus hide their impact on upper layers – in particular, on routing protocols – entirely [18]. This "bidirectional abstraction" is also explored in [19], which proposes to detect unidirectional links via a HELLO message exchange, then to build tunnels at the data-link layer. Assuming positioning information, [20] proposes that routers adjust their transmission power to overcome detected asymmetries.

While these approaches can be useful for specific data-link layer types, they are not adapted to provide a transparent support for asymmetry in heterogeneous networks, *i.e.,* networks comprised of multiple different data-link layers (wireless, wireline, PLC, etc.). Asymmetric links in such heterogeneous networks must necessarily be handled above the data-link layer.

Network-layer approaches to handle unidirectional links, mostly, explore modifications of existing routing protocols. Such is the case for [8, 21], which address multicast routing over unidirectional links by extending ODMRP [22] and ADMR [23], respectively – and which have inspired the *Loop Exploration* mechanism, proposed in this paper.

The Unidirectional Link-state Protocol (ULP) is proposed in [24]. For every unidirectional link detected, ULP looks for – and maintains – an "inclusive cycle" allowing communication in both directions, via pure and limited flooding. Alternatively, [25, 26] propose a periodic HELLO message exchange to detect unidirectional links, and maintain a connected, dominating and absorbent core of routers, in a way such that every non-core router can maintain bidirectional communication with the core (and thereby with any other router in the network). Decentralised construction and maintenance of this core requires limited flooding; as in ULP, the number of hops that an update is forwarded determines the maximum length of the "inclusive cycle" that can be discovered around a unidirectional link.

For reactive unicast routing protocols, operating in networks with unidirectional links, RODA [20] proposes a Route Discovery process with two rounds of request and reply: the first round of flooding (`RREQ` and `RREP`) to identify routes in both directions, and the second round with unicast transmission of Forward and Reverse Packets to install identified routes. Route maintenance is, then, performed by way of periodic beaconing.

AODV [2] describes two mechanisms for explicitly excluding unidirectional links from being included in a routing topology: *Reverse Bidirectionality Check + Blacklisting*, as discussed in section 1, as well as a periodic HELLO message exchange verifying a priori and proactively with which neighbours bidirectional links exist, and so from which neighbours a `RREQ` can be accepted. These mechanisms are evaluated in [11], which also proposes, and compares to, a third mechanism, *RemotePathSearch*. This mechanism allows intermediate routers, and the destination, receiving an `RREQ` to perform a depth-first-like search of reverse bidirectional paths via `RREPs` – thus excluding paths with unidirectional links. EUDA [27] is an exten-

sion to AODV that determines link unidirectionality upon RREQ reception by estimating the link distance with a propagation model and comparing it with the receiver's range. Common for all these is, that they target detecting and preventing unidirectional links from being included in the routing topology.

A similar detection mechanism to [27] is used in [28], which also introduces a *monitoring node* that can provide a 2-hop reverse route to include unidirectional links in AODV route computation. Thus [28] is able to utilize unidirectional links, but only under the condition that a *monitoring node*, *i.e.,* a common neighbour of both endpoints of the link, exists. A mechanism based on unidirectional link counter is proposed in [29] that depends on a neighbourhood discovery protocol and discovering 1-hop detours in between neighbours.

In [17], an extensive work is performed in improving the sensor network performance in the presence of unidirectional links. Five protocols are proposed for different network scenarios and assumptions. For example, *Buckshot* is based on source routing with minimum complexity possible. *Buckshot with Distance Vectors (BuckshotDV)* applies the distance vector mechanism to reduce the network load by learning the "next-but-one" hop in the path. *Overhearing Supported Buckshot DV (OSBRDV)* further improves the number of message transmissions by overhearing the next-but-one hop. *Unidirectional Link Triangle Routing* and *Unidirectional Link Counter* are based on the assumption that unidirectional links are known beforehand with certain neighborhood discovery protocol.

Dynamic Source Routing [1] can deal with unidirectional links by using two network-wide floodings of RREQs and RREP messages. However, the source-routing based approach is very different from other table-driven protocols: it requires per-data-packet overhead to keep the source routing information, and the RREQ messages have to carry the accumulated path, which tends to be significant. Both aspects (per-data-packet overhead and network-wide flooding) should be avoided or minimized as much as possible, especially for low power and lossy networks (LLNs).

The IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL) [30] [31] builds a collection tree to one or several "root" routers. It assumes links to be bi-directional and depends on external mechanisms such as the Neighbour Unreachability Detection (NUD) or Bidirectional Forwarding Detection (BFD) to discard (*i.e.,* ignore) unidirectional links.

3. The Impact of Unidirectional Links on Reactive Routing Protocol Performance

In this section, the effect of asymmetric links in route discovery is studied, and the impact of asymmetry in routing performance is quantified by way of a stochastic model of the RREQ and RREP transmission mechanisms. Section 3.1 presents the *Idealized Optimized Flooding* (IOF) assumed in the model. Section 3.2 introduces the network model, key parameters, notation, and definitions for quantifying the impact of asymmetry. Sections 3.3 and 3.4 describe RREQ flooding and Route Discovery (RREQ+RREP) as Markov chains, and compute the corresponding probability of Route Discovery success. Section 3.5 studies the probability that bidirectional connectivity in the network can be established – this provides an upper bound for the success probability of Route Discovery. Finally, section 3.6 particularises the previously obtained results for the two-level (Bernouilli) model of the probability of successful transmission over a link.

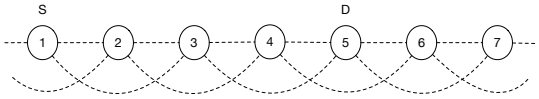

Figure 3: Family ($N = 5, m = 2$).

3.1. Idealized Optimized Flooding

RREQs are sent through the network, typically [2, 3] by way of pure flooding, *i.e.,* all routers receiving a RREQ retransmit (& process) it on first receipt, then drop subsequently received copies. This is known to lead to redundant retransmissions, possible collisions and, in general, poor performance in dense dynamic networks [32]. Several optimizations have been proposed and discussed for use in reactive routing protocols to avoid interference between simultaneous transmission and search for the shortest path [33]. Since flooding optimization is not the focus of this paper, such an optimization is taken for granted. Thus, this section considers an *Idealized Optimized Flooding* (IOF), in which a transmitted RREQ is *only* retransmitted (once) by the *best possible* relay(s), *i.e.,* the router(s) covering more new routers on a transmission.

3.2. System Model and Definitions

The scenario depicted in figure 3 represents the model for wireless transmission in unidimensional networks, and with unidirectional links, used in this section. In this network, the number of RREQs from a source S to a destination D depend on the maximum and minimum length (in hops) of loop-free paths between S and D (distance $N-1$, with $N = 5$ in fig. 3), which depend on the distance between S and D, N, and the neighbour density (*right-side density*) of the network, m ($m = 2$ in fig. 3).

Formally, a unidimensional, multi-hop, wireless network with infinite (countable) routers is modeled with a graph $G = (V, E)$, where $V = \{..., 1, 2, ..., n, ...\}_{n\in\mathbb{N}}$, is the set of vertices (routers), with $|V| = \infty$, and $E = \{(i, j) : i, j \in V, |i - j| \leq m\}$ is the set of edges. The *right-side density* of such a network, m, is the maximum number of neighbours that a router $i \in V$ can have with id $j > i$.

Transmission of a RREQ (and a RREP) is modeled as follows. Without loss of generality, it is considered a Route Discovery in which router $S \equiv 1$ (source) performs a RREQ (broadcast RREQ) towards router $D \equiv N$ (destination)

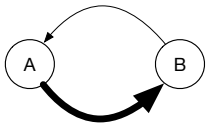

Figure 4: Asymmetric link (A, B), with strong direction $A \longrightarrow B$ and weak direction $B \longrightarrow A$.

– *i.e.,* a right-directional IOF with source S. The *best possible* relay for a router $x \in V$, $1 \leq x < N$, is $\min(x + m, N)$. The study is thus restricted to the subgraph $G' = (V', E')$, where:

$$\begin{cases} V' = \{n \in V : 1 \leq n \leq N + m\} \\ E' = \{(i, j) : i, j \in V'\} \end{cases}$$

Given a link $(i, j) \in E$, $p_{i,j} \in [0, 1]$ denotes the probability that a transmission from i is received successfully at j, and $\dot{p}_{i,j}$ denotes its complementary (*i.e.,* the probability that a transmission from i is *not* received successfully at j), with $\dot{p}_{i,j} = 1 - p_{i,j}$. Events resulting from transmissions over different links (links involving different routers) are probabilistically independent. A link (i, j) is *asymmetric* if the probability of successful transmission in one direction (the *strong* direction) is higher than in the other (the *weak* direction). If the strong direction of link (i, j) is $i \longrightarrow j$, then $p_{i,j} >> p_{j,i}$. Figure 4 illustrates an asymmetric link.

Unidirectional links are particular cases of asymmetric links, in which the weak direction has probability of successful transmission $= 0$.

With $\alpha \in [0, 1]$ being the fraction of asymmetric links in the network, a link is asymmetric with probability α (P-model [34]). A network (N, m) with $100 \times \alpha\%$ of asymmetric links can be described with a tuple (N, m, α).

An infinite network (N, m) is fully described with an infinite matrix $\mathbf{P} = (p_{i,j})_{i,j \in \mathbb{N}}$. A route request/reply procedure from $S \equiv 1$ and $D \equiv N$ can be studied in its finite restriction $\mathbf{P}' = (p_{i,j})_{0 < i,j \leq N+m}$.

Table 1 summarizes the notation used throughout this section.

3.3. Probability of `RREQ` *Transmission Success*

Given a (N, m) network, $T_n = i$ is the random variable describing the last router that newly received successfully the n-th transmission of the `RREQ` ($n \geq 1$, $i \in \{0, 1, 2, ..., N, ..., N + m\}$). If no new router received the n-th transmission, then $T_n = 0$. If transmission n reaches the destination router N ($T_n = N$), the `RREQ` is assumed successful (with path length n if $T_{n-1} < N$), and $T_{n'} = N$, $\forall n' > n$. If transmission n reaches (at most) router $j : N < j \leq N + m$, but is not received by router N, then $T_n = j$.

Transmission n of a `RREQ` by a router j, with $N < j \leq N + m$, makes the system evolve towards state N ($T_n = N$) if it is (newly) received by the destination; otherwise the system evolves to (a) state $k \neq 0, k > j$, if the next transmitter can still reach the destination N (*i.e.,* , if $k \leq N+m$), or (b) to state 0 is the transmission fails or further transmitter cannot reach N – in this case, the request has been unsuccessful.

The process $T = \{T_n\}_{n \geq 1}$ is a discrete-time Markov chain indexed in n and defined in Lemma 1:

Lemma 1. *The set of states of T is:*

$$S_T = \{0, 1, ..., N, ..., N + m\} \tag{1}$$

The transition matrix of T is $\mathbf{T} = (t_{i,j})_{0 \leq i,j \leq N+m}$, where:

$$\begin{cases} t_{i,j} = p_{i,j} \prod_{k=j+1}^{i+m} (1 - p_{i,k}) & , 0 < i < j < N \\ t_{i,N} = p_{i,N} & , i \neq 0, N \end{cases} \tag{2}$$

and:

$$\begin{cases} t_{0,0} = 1 & \\ t_{0,i} = 0 & , \forall i > 0 \\ t_{i,i} = 0 & , \forall i \neq 0, N \\ t_{N,N} = 1 & \\ t_{i,j} = \dot{p}_{i,N} p_{i,j} \prod_{k=j+1}^{i+m} \dot{p}_{i,k} & , i \neq 0, N; \\ & N < j \leq N + m \\ t_{i,0} = \prod_{k=i+1}^{i+m} \dot{p}_{i,k} & , 0 < i < N \\ t_{i,0} = \dot{p}_{i,N} \prod_{k=i+1}^{N+m} \dot{p}_{i,k} & , N < i < N + m \\ t_{N+m,0} = \dot{p}_{N+m,N} & \end{cases} \tag{3}$$

States 0 and N are *absorbing states* in T, *i.e.,* once the process has reached these states, they cannot evolve to any other state.

From that, it is immediate to observe that:

Lemma 2. *The probability that a* `RREQ` *is successful in a system (N, m) is:*

$$\lim_{n \longrightarrow \infty} [\mathbf{T}^n]_{1,N} = [(\mathbf{T})^{N+m-1}]_{1,N} \tag{4}$$

This corresponds to the probability that a request has reached state N in at most $N + m - 1$ transmissions.

3.4. Probability of Route Discovery (`RREQ`*+*`RREP`*) Success*

The Markov chain $\hat{T}$ is instrumental for computing the probability that a Route Discovery is successful, *i.e.,* that both `RREQ` and `RREP` are sent and received successfully. $\hat{T}$ is based on the previously defined chain T, and has a transition matrix $(\hat{t}_{i,j})$, $\hat{t}_{i,j} = Pr\{$RREQ on (i, j) successful; RREQ on $(i, k), k > j, k \leq i + m$, not successful; RREP on (j, i) successful$\}$.

By definition of the model, successful transmissions at (i, j) and (i, k) are independent if $j \neq k$. In a system with a fixed fraction α of asymmetric links, however, successful transmission in (i, j) is *not* necessarily independent from successful transmission in (j, i), and the relationship between the former and the latter is specific to the link

Name	*Description*
$G = (V, E)$	Network graph.
$N \in \mathbb{N}$	Number of routers.
$m \in \mathbb{N}$	Right-side density.
$\alpha \in [0, 1]$	Fraction of asymmetric links.
$G' = (V', E')$	Subgraph for Route Discovery.
$\mathbf{P} = (p_{i,j})_{1 \leq i,j \leq N+m}$ $p_{i,j} \in [0, 1]$	Matrix of probabilities of successful transmission in G'.
$\mathbb{E}\{X\}$	Expected value of random variable X.
T_n $\hat{T}_n$	State of route request and discovery, respectively, at time n.
$\mathbf{T} = (t_{i,j})_{0 \leq i,j \leq N+m}$ $\hat{\mathbf{T}} = (\hat{t}_{i,j})_{0 \leq i,j \leq N+m}$	Transition matrices for T and $\hat{T}$, respectively.
$S \equiv 1 \in V$	Node 1 (also denoted S), source of the Route Discovery.
$D \equiv N \in V$	Node N (also denoted D), destination of the Route Discovery.

Table 1: Notation.

characteristics[1]. Consequently, there cannot be a general closed form of equations (2) and (3) for T.

$\hat{t}_{i,j}$ describes the probability of transition in $\hat{T}$ between state i and state j in a single step. In this case, transition to $j \neq 0$ implies that a `RREQ` transmission from i happened, this transmission was successfully received by router j, and not by any router $k : k > j$, and `RREP` *will* be successfully transmitted from j to i. This leads to Lemma 3.

Lemma 3. *The average probability of discovery success (`RREQ`+`RREP`) can be computed as follows:*

$$\lim_{n \longrightarrow \infty} \mathbb{E}\{(\hat{\mathbf{T}}^n)_{1,N}\} = \mathbb{E}\{(\hat{\mathbf{T}}^{N+m-1})_{1,N}\} \tag{5}$$

3.5. Probability of Bidirectional Connectivity

Topology and link characteristics of the system, described by $\mathbf{P}$, allow to compute the upper bound for the probability of success in route discovery (`RREQ`/`RREP` packets exchange between S and D). This upper bound does *not* depend on specific mechanisms, it is a characteristic of the system itself.

In the described model, this upper bound corresponds to the probability that a `RREQ` reaches D in (at most) $N + m - 1$ transmissions, *and* the corresponding `RREP` reaches S in (at most) $N + m - 1$ transmissions, *i.e.:*

$$\lim_{n \longrightarrow \infty} [(\mathbf{T}^n)_{1,N}]^2 = [(\mathbf{T}^{N+m-1})_{1,N}]^2 = [1 - (\mathbf{T}^{N+m-1})_{1,0}]^2 \tag{6}$$

[1]In particular: if a link direction $(j, i) \in E$ is weak, direction (i, j) is necessarily strong (*i.e.,* with high probability of success in transmission); if (j, i) is strong, (i, j) could be strong or weak (with a probability related to α).

3.6. Particular Case: Bernoulli Asymmetry

The Bernoulli model for asymmetry describes links as follows:

- Each link is asymmetric with probability α, symmetric with probability $(1 - \alpha)$.
- In symmetric links, probability of transmission success is p_b in both directions, probability of failure $(1 - p_b)$ (Bernoulli distribution).
- In asymmetric links, probability of transmission success is p_b in the strong communication direction and $p_u << p_b$ in the weak.
- Probability of transmission success over links is *static* (does not change with time) and transmissions over different links or times are *independent.*
- There is no distinction between broadcast and unicast transmissions.

The mean of the probability of successful transmission over a link $(i, j) \in E$ (*i.e.,* , $|j - i| \leq m$) is:

$$\mathbb{E}\{p_{i,j}\} = \frac{\alpha}{2} p_u + \left(1 - \frac{\alpha}{2}\right) p_b \tag{7}$$

Since equation (7) does not depend on the indices i, j (as long as $|j - i| \leq m$), and for the sake of simplicity on the notation, hereafter $\bar{p} \equiv \mathbb{E}\{p_{i,j}\}$. Based on equations (2) and (3), $\mathbb{E}\{t_{i,j}\}$ is as follows (non-exhaustive list):

- If $|j - i| \leq m$, with $0 < i < j < N$:

$$\mathbb{E}\{t_{i,j}\} = \bar{p} \, (1 - \bar{p})^{i+m-j}$$

- If $|j-i| \leq m$ and $i \neq 0, N$, $j = N$:

$$\mathbb{E}\{t_{i,j}\} = \bar{p}$$

- If $|j-i| < m$ and $i \neq 0$, $N < j \leq N+m$:

$$\mathbb{E}\{t_{i,j}\} = \bar{p}\,(1-\bar{p})^{i+m-j+1}$$

- If $|j-i| > m$, $\mathbb{E}\{t_{i,j}\} = 0$.

The probability that transmissions over links (i,j) and (j,i) are *both* successful has the average: $\bar{q} = (\alpha p_b p_u + (1-\alpha)p_b^2)$. Therefore, $\mathbb{E}\{\hat{t}_{i,j}\}$ is as follows:

- If $|j-i| \leq m$, with $0 < i < j < N$:

$$\mathbb{E}\{\hat{t}_{i,j}\} = \bar{q}\,(1-\bar{p})^{i+m-j}$$

- If $|j-i| \leq m$ and $i \neq 0, N$, $j = N$:

$$\mathbb{E}\{\hat{t}_{i,j}\} = \bar{q}$$

- If $|j-i| < m$ and $i \neq 0$, $N < j \leq N+m$:

$$\mathbb{E}\{\hat{t}_{i,j}\} = \bar{q}\,(1-\bar{p})^{i+m-j+1}$$

- If $|j-i| > m$, $\mathbb{E}\{\hat{t}_{i,j}\} = 0$.

These allow solving equations (4), (5) and (6).

Figure 5 depicts the average probability that the route discovery is successful (`RREQ`+`RREP`) and the upper bound for the probability of success in Route Discovery, and figure 6 the average probability that a `RREQ` from 1 to N is successful, as a function of the fraction α of asymmetric links, for $(N = 5, m = 2)$ networks with P-model link characteristics.

As expected, performance degrades substantially with an increasing fraction of asymmetric links. In particular, when all links are asymmetric ($\alpha = 1$), the probability that Route Discoveries succeed is close to zero (0.002, *i.e.,* only 0.2% of discoveries would succeed in this case), even when bidirectional connectivity is available 19% of the times. Adapting Route Discovery mechanisms to leverage bidirectional connectivity instead of relying on bidirectional paths may substantially increase Route Discovery quality – in this example, the gap between the upper bound and the legacy route discovery mechanism probability of success is between 25% and 35%, for scenarios with a significant portion (15% to 80%) of unidirectional links in the network.

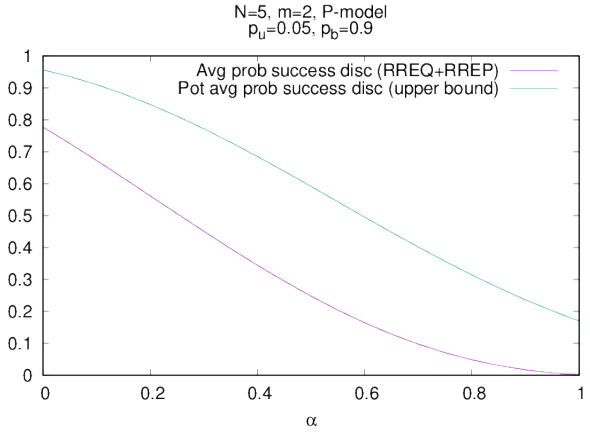

Figure 5: Average probability of discovery success (`RREQ`+`RREP`) and upper bound for probability of bidirectional connectivity, for $(N = 5, m = 2)$, as a function of the fraction of asymmetric links in the network ($\alpha \in [0,1]$).

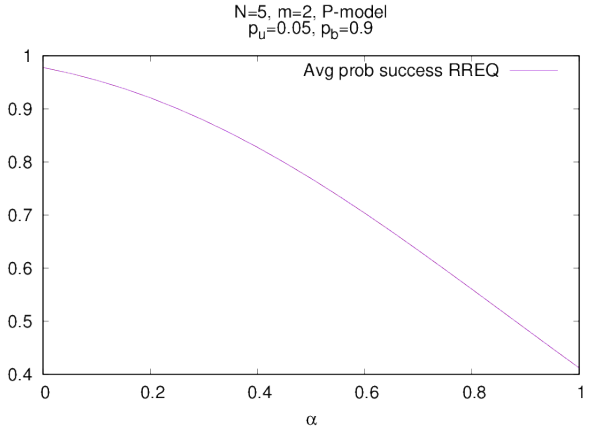

Figure 6: Average probability of route request success (`RREQ`), for $(N = 5, m = 2)$, as a function of the fraction of asymmetric links in the network ($\alpha \in [0,1]$).

3.6.1. Sensitivity to Parameters

Figure 7 depicts the impact of right-side density (m) in the probability of discovery success, for different fractions of unidirectional links in the network (α). For dense networks ($m \geq 12$), the probability of success increases abruptly when the minimum shortest path from S to D reduces its length, as a result of increasing the right-side density ($m = \{12, 16, 24\}$). Step increases in the average probability of success discovery occur when a new path with less hops becomes possible due to the increase of right-side density m.

Figure 8 shows the decrease in the probability of discovery success with the number of hops between S and D $(= N-1)$, for a relatively sparse network ($m = 2$). As expected, the probability of success decreases exponentially, more dramatically in presence of unidirectional links.

Figure 9 shows the impact of the network scale (factor F) in the request performance. The figure depicts the average probability of success (`RREQ` IOF+unicast `RREP` and

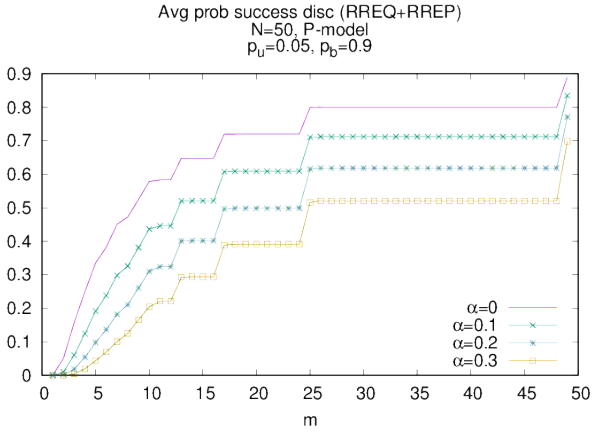

Figure 7: Average probability of discovery success (RREQ+RREP) for different values of α (fraction of unidirectional links) and m (right-side density).

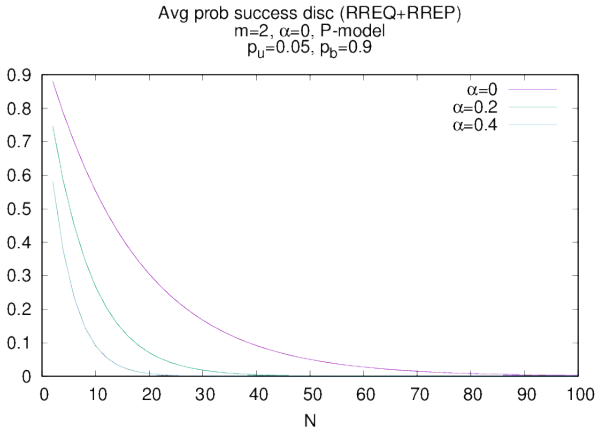

Figure 8: Average probability of discovery success (RREQ+RREP) depending on N (length of the discovery), for $m = 2$, for different values of α.

upper bound) for different networks (N, m), in which N/m is constant, $N = N_0F$, $m = m_0F$, for different values of F. Networks with higher value of F are denser (routers have more neighbours), but requests need to traverse a similar number of hops.

In these networks, the presence of asymmetric links has limited impact on the (potential) ability to establish bidirectional communication between S and D (above 90% for $F > 1$, for all considered values of α). However the performance of route discovery (RREQ flooding and unicast RREP) degrades substantially as α increases: less than 20% of the requests are successful for $\alpha = 0.4$, no matter how dense (F) the network is. This suggests that this Route Discovery procedure is not adapted for these scenarios (dense and strongly asymmetric networks), but Route Discovery performance can be substantially improved with more flexible mechanisms.

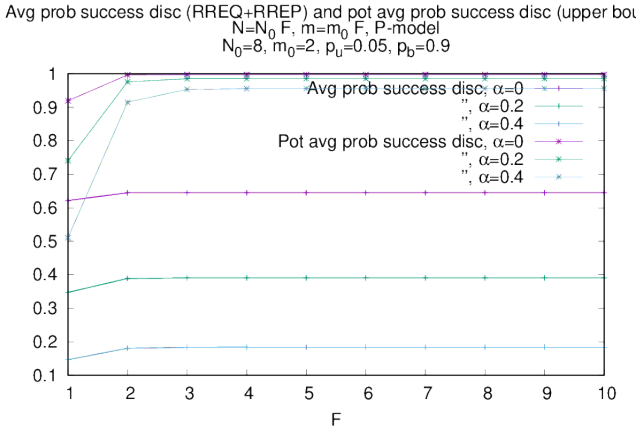

Figure 9: Average probability of discovery success (RREQ+RREP) and potential average probability of success (upper bound) depending on F (scale factor), for $N_0 = 8$, $m_0 = 2$, for different values of α.

4. Unidirectional Link Detection

Reactive routing protocols assume that links are bidirectional, and include mechanisms to check for (and, exclude) unidirectional links during the Route Discovery procedures. A mechanism for unidirectional link detection, denoted *Reverse Bidirectionality Check* (*RBC*), is specified for (and used by) different reactive routing protocols, *e.g.,* LOADng [35, 36, 3], DSR [1], or AODV [2].

In this section, the existing approach *RBC* is detailed in section 4.1. *RBC* is performed during the RREP phase. If a link does not pass *RBC*, then it is *blacklisted* so as to exclude it from further Route Discoveries. However, as detailed in section 4.1.2, this entails a risk of installing *false forward routes*.

In order to avoid the risk of *false forward routes*, section 4.2 proposes a *Forward Bidirectionality Check* (*FBC*), consisting of testing link bidirectionality in the RREQ phase, *i.e.,* when the RREQ is being flooded towards the destination.

4.1. Reverse Bidirectionality Check (RBC)

RBC consists of verifying bidirectional connectivity of each link in a route during the RREP forwarding phase, *i.e.,* when a RREQ has reached the destination D and D has sent a unicast RREP towards the source D, so that every intermediate router installs a forward route towards D.

More precisely, given a source, S, a destination, D, and a shortest path from S to D, $p = \{S \equiv x_0, x_1, ..., x_n \equiv D\}$, when an intermediate router $x_{i+1} \in p$, transmits a RREP to a router $x_i \in p$, router x_{i+1} indicates, by a flag in the RREP, that it expects to receive an acknowledgement (in this paper denoted RREP_ACK) in a timely fashion. Failure to receive an expected RREP_ACK is interpreted as the link (x_i, x_{i+1}) being unidirectional and only working in the direction opposite of which the RREP was sent (*i.e.,* $x_i \rightarrow$

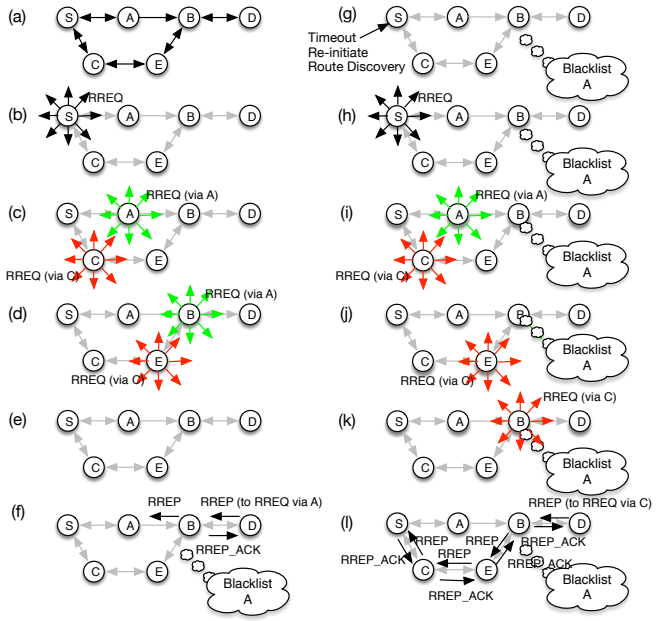

Figure 10: An example Route Discovery process when faced with unidirectional links. The grey arrows indicate the underlaying L2 connectivity. S seeks a route to D, issues a `RREQ` (b) which is flooded through the network (b-c-d) until it reaches the destination D – which responds by generating and unicasting an `RREP` (f).

x_{i+1}), and triggers an action of "blacklisting" that link, as described below.

4.1.1. Blacklisting

Figure 10 displays a blacklisting example: the grey arrows indicate underlaying L2 connectivity, and the L2 connectivity between A and B is unidirectional (figure 10a) in the direction A→B. Route Discovery proceeds, as in figure 1, until figure 10f, where router B does not receive an expected `RREP_ACK` – and therefore "blacklists" router A. When eventually S times out and re-initiates the next Route Discovery, the `RREQ` received by B from A is ignored, *i.e.,* is neither forwarded nor processed (figure 10i), thus the `RREQ` received by way of the path $S \rightarrow C \rightarrow E \rightarrow B$ is forwarded to D – and, the path $S \leftrightarrow C \leftrightarrow E \leftrightarrow B \leftrightarrow D$ is established.

Therefore, the `RREP_ACK` mechanism aims at eliminating unidirectional links in the route, at the cost of increased overhead and delay.

4.1.2. False Forward Routes

While failure to receive an `RREP_ACK` may be indicative that the `RREP` was not delivered (with reference to figure 10f, that would be indicating that layer 2 communication $B \rightarrow A$ is not possible), it may also occur when the `RREP` was actually delivered (thus communication $B \rightarrow A$ is possible) but the `RREP_ACK` itself was lost (thus, no communication $A \rightarrow B$ confirmed).

This case is particularly harmful as it will cause a "false forward route" to be installed: if in figure 10f the `RREP` was received by router A, but it was the `RREP_ACK` (and not the `RREP`) that was lost, router A would forward the `RREP` to S – which would assume that a bidirectional route existed, and would start using it for data traffic.

Recovery from this situation would only occur when failure to deliver data traffic across the "false forward route" is detected, triggering `RERR` message generation, and a renewed Route Discovery procedure to be initiated.

This is not a hypothetical situation, nor is it one requiring highly time-varying link properties — it can occur as a consequence of different transmission rates of broadcasts (such as is used for `RREQs`) and unicasts (such as is used for `RREP` and `RREP_ACK`), causing a `RREQ` to "reach further" than a `RREP_ACK`. WiFi is one example of a data-link layer exhibiting possibly different transmission rates for broadcast/multicast and unicast transmissions [37] and where this phenomenon can easily be observed.

4.2. Forward Bidirectionality Check (FBC)

By performing a bidirectionality test during forwarding of `RREQ` from the source S to the destination D, *FBC* avoids installing a "false forward route", as discussed in section 4.1.2: a router does not forward a `RREQ` until such time that it has been confirmed that bidirectional unicast communication is possible between itself and the neighbour from which it received the `RREQ` – in other words, a reverse route is never installed if the corresponding forward route is not verified to be available.

Specifically, on receiving a multicast `RREQ`, a router:

1. Buffers the received multicast `RREQ`;
2. Temporarily installs a route to the neighbour, from which the `RREQ` was received (but, not to the source of the `RREQ`);
3. Unicasts a `RREQ` to that neighbour, with its own IP address as originator, with the IP address of that neighbour as Destination, and with a hop-limit of 1;
4. Awaits a unicast `RREP` from that neighbour to be received.
   - If a `RREP` is received, the buffered multicast `RREQ` is processed as specified by the unicast routing protocol, *e.g.,* as per [35, 36, 3].
   - Otherwise, if no `RREP` is received:
     - the buffered multicast `RREQ` is silently discarded;
     - the temporary route to the neighbour is removed;
     - the neighbour is "blacklisted" as specified by the unicast routing protocol for when an expected `RREP_ACK` is absent.

This ensures that a unicast `RREQ` / `RREP` exchange between two neighbouring routers occurs before the `RREQ` is processed and flooded and, thus, guards against a "false forward route" having been installed.

Note that *FBC* also helps to avoid repeated Route Discovery operations (corresponding to the flooding operations of `RREQs` carrying paths including the unidirectional link). This is, however, at the expense of a possibly slower progression of `RREQ` flooding throughout the network (since each new link has to be tested before `RREQ` forwarding) and longer delays for successful Route Discovery processes. Section 6 will quantify both the gain and the incrementally slower `RREQ` progression resulting from this mechanism.

5. Loop Exploration (LE)

The RBC and FBC mechanisms, both discussed in section 4, aim at eliminating the unidirectional links from the route discovery; all discovered paths are thus bidirectional. On the other hand, in some scenarios, especially when fully bidirectional paths are not available, it is preferable to make use of paths possibly including unidirectional links. In this section, *Loop Exploration* is proposed to discover paths with unidirectional links. Naturally, as *Loop Exploration* will make use of unidirectional links when bidirectional links are not available, it requires that the data link layer does not filter out unidirectional links.

Intuitively, the *Loop Exploration* (*LE*) mechanism relies on the idea that, even when no bidirectional path exists between two routers S and D in a network, bidirectional communication can be established if two (at least unidirectional) paths can be discovered from S to D, and from D to S – and the path from D to S can be discovered, if it exists, by traversing loops around unidirectional links detected in the path from S to D.

Formally, for a path p between S and D, $p = \{S \equiv x_0, x_1, x_2, ..., x_n \equiv D\} = \{(x_i, x_{i+1}) : 0 \leq i < n\}$ of n hops, the fact that a link $(x_i, x_{i+1}) \in p$ is unidirectional ($x_i \rightarrow x_{i+1}$) implies that path p is not bidirectional. In a path with unidirectional links, however, bidirectional communication between S and D is still possible if, for each unidirectional link $x_i \rightarrow x_{i+1}$, there exists a loop $\mathcal{L}_{i,i+1} = \{x_{i+1} \equiv y_0, y_1, ..., y_{l-1} \equiv x_i, y_0\}$, of l hops, such that x_{i+1} can communicate with x_i via $\mathcal{L}_{i,i+1}$.

In the example topology in figure 11, exploiting the loop $\mathcal{L}_{A,B} = \mathcal{L}_{B,F} = \{A, B, F, E, C, A\}$ would enable bidirectional communication between S and D, although no path with only bidirectional links between S and D exists.

This section proposes an algorithm that, upon detection of a unidirectional link in the path between S and D, allows finding a loop that can enable bidirectional communication between the endpoints of the detected unidirectional link. Exploring and identifying loops for each encountered unidirectional link in the path, allows to complete a `RREQ`/`RREP`, and thus to establish bidirectional communication between S and D, as long as there exist available (possibly non-bidirectional) paths from S to D, and from D to S.

The proposed algorithm is inspired by the approach taken by [7, 8], in which flooding is used to detect a loop for multicast in networks. This section further explores the loop exploration algorithm for standard reactive routing protocols.

Given a path p between S and D, and given a unidirectional link $x_i \rightarrow x_{i+1}$ in p, detected by router x_{i+1}, the algorithm consists of three steps:

1. *Detection of Unidirectional Link:* using the `RREP_ACK`-based unidirectional link detection mechanism (section 4), the end points of a unidirectional link (if encountered), $x_i \rightarrow x_{i+1}$, are identified.
2. *Detection of Loop and Anchor:* if a unidirectional link is detected, a (possibly localised) flooding procedure is triggered to search for an alternative (possibly non-bidirectional) path from x_{i+1} to x_i.
3. *Installation of Routing State:* if loop and anchor detection completes successfully, and a loop $\mathcal{L}_{x_i,x_{i+1}}$ is identified, the explored loop is traversed to install proper forward and reverse routes in the involved routers.

Each of these steps is detailed in the following subsections. Figure 11 depicts an example of the loop detection algorithm in a route request from S to D with two unidirectional links $A \rightarrow B$, $B \rightarrow F$.

5.1. Detection of a Unidirectional Link

The loop exploration algorithm is triggered when a router detects a unidirectional link in the path requested / discovered by a source S to a destination D. It is assumed that unidirectionality of link (x_i, x_{i+1}) is detected by router x_{i+1} by way of *RBC* (*i.e.,* because x_{i+1} fails to receive the acknowledgement from x_i to the unicast Route Request). This corresponds, in figure 11d, to F not receiving an `RREP_ACK` from B.

5.2. Detection of Loop and Anchor

When the link (x_i, x_{i+1}) has been detected as unidirectional by router x_{i+1}, router x_{i+1} stores a copy of the un-acknowledged `RREP`. It then performs a limited flooding of a special *Path Accumulation RREQ* (`RREQ-PA`) message with itself (x_{i+1}) as destination. In order to prevent excessive overhead, flooding is limited to *LES* (Loop Exploration Scope) hops from the triggering router. An `RREQ-PA` is exactly as any other `RREQ`, except that when an `RREQ-PA` is forwarded, the forwarding router records its own address in the message. Thus, the recipient of an `RREQ-PA` will know which path the message has followed[2].

The behaviour of a router y receiving an `RREQ-PA` depends on whether y is the destination of the `RREQ-PA`, or not:

[2] `RREQ-PA`-messages with behaviours as described in this paper exist as extensions for most reactive routing protocols, *e.g.,* for LOADng [38], for DSR [39], and for AODV [40].

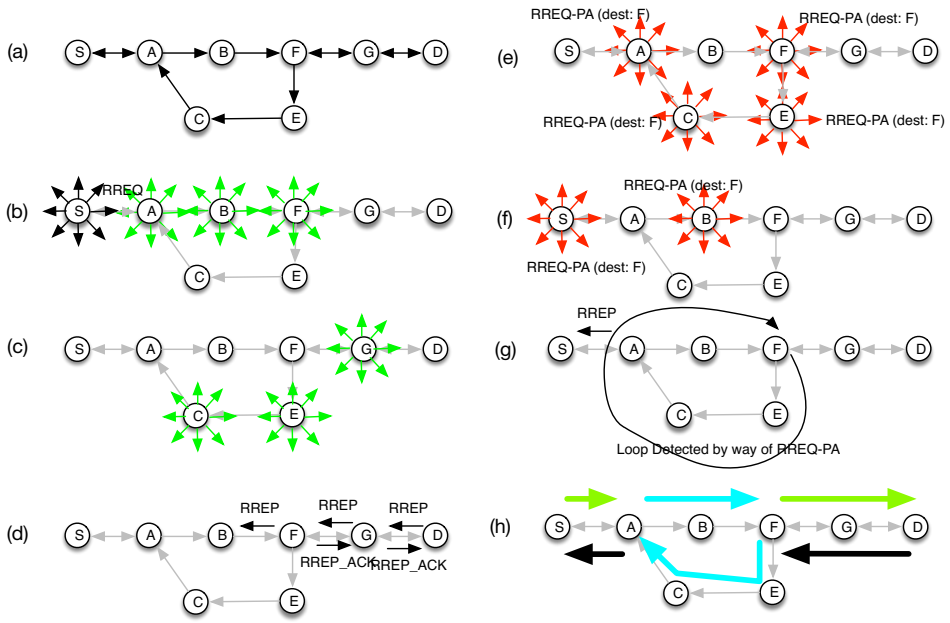

Figure 11: Route Discovery in a topology where bidirectional connectivity exists between S and D, but not through a bidirectional path. The underlying connectivity is as in (a), with the regular route discovery process depicted in (b), (c), and (d). When failing to receive an `RREP_ACK` from B, router F searches for an alternative path "to itself, via someone closer to S" by way of a `RREQ` with Path Accumulation, as described in this paperand as illustrated in (e), (f), and (g) – with (h) depicting the path elements installed by regular `RREQ` processing (black), regular `RREP` processing (green) and in blue the paths discovered and installed by the process documented in this section.

- if y is not the destination of the message ($y \not\equiv x_{i+1}$), then y records its own id and its own distance to S in the `RREQ-PA`, before re-flooding the message.
- if y is the destination (*i.e.,* $y \equiv x_{i+1}$), then the `RREQ-PA` has traversed a loop $\mathcal{L}$ consisting exactly of the routers recorded in the message.
  - If no router is recorded in the `RREQ-PA` with a smaller distance to S than that of x_{i+1} itself, to S, the `RREQ-PA` does not contain a useful loop and the `RREQ-PA` is discarded.
  - Otherwise, the router recorded in the received `RREQ-PA` with the minimum distance to S, *i.e.,* the router closest to S, is selected as the *anchor*, a, of the discovered loop (A in the example of figure 11).

Of course, if the anchor is the router triggering the loop exploration, *i.e.,* $a = x_{i+1}$, then the detected loop does not include router x_i, the detected loop $\mathcal{L}$ is not *valid*, and must be ignored, in as much as it does not allow bidirectional communication between routers x_i and x_{i+1}. Otherwise, router x_{i+1} proceeds to the step of *Installation of Routing State* (section 5.3).

To prevent deadlocks in the loop detection triggering router, x_{i+1}, a timer t_{loop} should be used to define the waiting interval for a `RREQ-PA` reporting a valid loop. Router x_{i+1} may receive several `RREQ-PA` packets during the t_{loop} interval, indicating different loops $\mathcal{L}$. The anchor with minimum distance to the route discovery source will be chosen. The longer t_{loop} and LES are, the longer will be discovered loops, and the longer the delay incurred in path discovery; for $t_{\text{loop}} \to \infty$ and no LES is set, any existing reverse path will be detected, at the cost of high discovery delay and associated overhead.

5.3. Installation of Routing State

Upon having identified a valid loop, x_{i+1} must send a (unicast) control message through the loop, so as to install necessary routing state. Specifically, this entails to:

1. instruct the anchor, a, to forward a `RREP` to S, so that S installs routes towards D via the anchor;
2. instruct routers *prior* to the anchor in the path from x_{i+1} (*i.e.,* $\{E, C\}$, in the example of figure 11) to install routes to D; and
3. instruct routers *after* the anchor in the path towards x_{i+1} (*i.e.,* B in the example of figure 11) to install routes towards S.

5.4. Discussion

Loop exploration using flooded `RREQ-PA` messages means that, potentially, multiple loops can return – and, thus, requires making a trade-off between the length of loops, and the length of time spent waiting for loops to be discovered. The longer x_{i+1} waits, the more likely it is that discovered loops are longer – and, consequently, that they offer anchors that are closer to the source S.

Discovering anchors closer to the source, S, implies that potentially fewer total loop exploration processes need to be run – this, simply, because a detected loop may contain multiple unidirectional links (each of which otherwise would have required individual loop exploration).

6. Performance Quantification

To quantify the performance of the proposed *FBC* and *LE* mechanisms, in comparison to the *RBC* and *Blacklisting* methods as traditionally employed by reactive routing protocols, NS2 network simulations are conducted.

For the purpose of these simulations, the IEEE 802.11b data-link layer is used with the *TwoRayGround* propagation model and acknowledgement disabled. The radio transmission range is $r = 250$ meters. The data rate of 802.11b is set to 11 Mbps and a collision happens when a router receives two or more overlapping transmissions. The simulator will thus drop the involved packets. The network scenarios consist of $N = 125$ static routers, randomly positioned within a 1500m×1500m area. In order to reduce possible bias from a specific topology, 5 different random topologies (*i.e.,* 5 sets of router positions, selected randomly) are used in the simulations.

Simulations are run with different fractions of unidirectional links in the network, ranging from 0% to 70% (with reference to section 3: $\alpha = 0$ to $\alpha = 0.7$). For the sake of simplicity, the directionality of a link is randomly determined (unidirectional with probability α, bidirectional otherwise) per scenario, and does not change over time.

For each simulated fraction of unidirectional links, and for each of the 5 random topologies, 200 (non-overlapping) Route Discoveries between random (source, destination) pairs are performed – thus, each data-point corresponds to the average of 1000 (non-overlapping) route discovery operations with standard deviation as error bars plotted.

Figure 12 and figure 13 describe connectivity of the simulated networks and length of shortest available paths between tested sources and destinations. They provide the theoretical upper bound for the route discovery success ratio and the lower bound for source-destination path lengths. In figure 12, *bidirectional connectivity* does not require the forward and reverse paths to be the same; *bidirectional connectivity using the same path* implies, in contrast, that the discovered path is fully bidirectional, *i.e.,* its links fulfil the Reverse Bidirectionality Check (RBC) and the Forward Bidirectionality Check (FBC). In figure 13, the *forward path length* is the (shortest) path from source to the destination, bidirectional or not, while the *bi-directional path length* contains only bidirectional links.

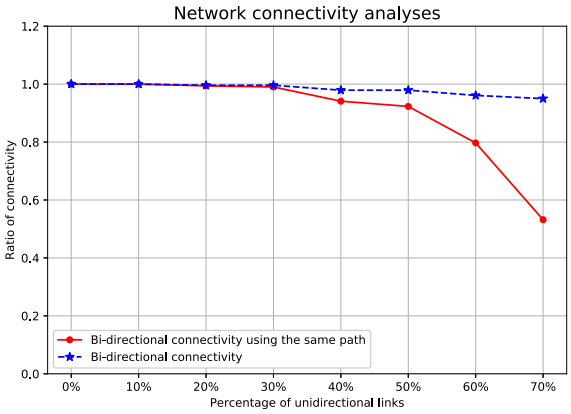

Figure 12: Connectivity of (source, destination) pairs in simulated networks, ideal case

6.1. Evaluation Metrics

For evaluating the behaviour of reactive routing protocols when faced with unidirectional links, the following metrics are of particular relevance:

- **Route discovery success ratio**, *i.e.,* the ratio of successful *vs.* total Route Discovery processes initiated in the network. Easily calculated, this is the ratio of Route Discoveries for which the originator of the `RREQ` receives an `RREP` from the intended destination.
- **Average route discovery delay,** *i.e.,* the average time (in seconds) needed for a route discovery to succeed. This metric considers only successful Route Discovery processes.
- **Overhead,** *i.e.,* the average amount of control traffic (in bytes/second) generated. This includes both successful and unsuccessful Route Discovery processes.
- **Average hop count,** *i.e.,* the average length of the paths discovered by the routing protocols (in hops).

Although the simulated networks are static rather than dynamic, the metrics used concerns only the route discovery process – the results will not have significant change in moderate mobile networks.

6.2. Routing Protocol Configurations

As discussed, reactive routing protocols may either: (i) assume that all links presented to them by an underlying data-link layer are bidirectional, and make no efforts at detecting or accommodating unidirectional links, (ii) assume that some links may be unidirectional and use various mechanisms (*i.e., RBC, FBC*) for excluding them

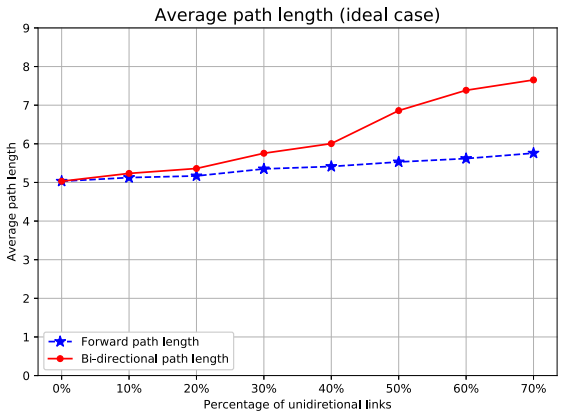

Figure 13: Average path lengths between sources and destinations in simulated networks, ideal case

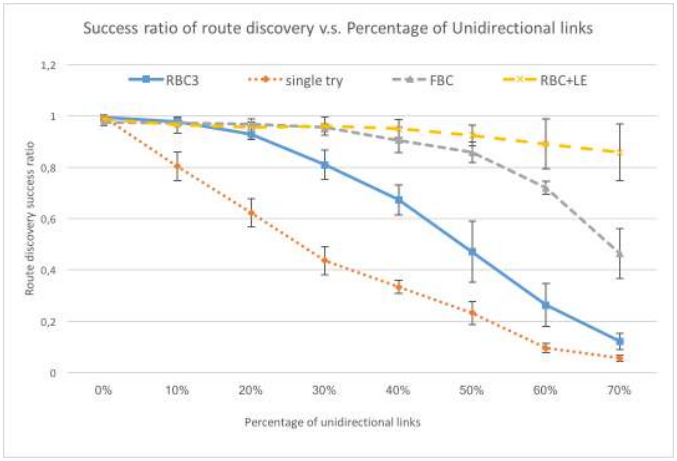

Figure 14: Average Route Discovery success ratio

from the routing topology, or (iii) assume that some links may be unidirectional, and attempt at utilizing unidirectional paths (*e.g., LE*) for forming bidirectional connectivity through the network. These policies can be implemented over any routing protocol, and the mechanisms discussed in this paper are indeed protocol-agnostic.

With this in mind, the following routing protocol configurations are considered, in this simulation study:

- **Single try.** For a given destination sought, a single Route Discovery is performed.

  This reflects the assumption that all links are bidirectional, *i.e.,* (i) in the above.

- **Reverse Bidirectionality Check with 3 retries (*RBC3*).** In this "basic" mechanism for handling unidirectional links, a router x *blacklists* a link when RBC over it fails, as per section 4.1, with the originator of the `RREQ` retrying Route Discovery thrice before considering the destination unreachable.

  This reflects the assumption that unidirectional links may be present in the network, and the policy of excluding these links from routing paths, *i.e.,* (ii) in the above.

  This is also the unidirectional link handling mechanism, common to the specifications of LOADng, AODV, and DSR. Using this as "default" seems to imply a further assumption, which is that unidirectional links are rare (with 3 retries, encountering more than 2 unidirectional links will make Route Discovery fail) in the network.

- **Forward Bidirectionality Check (*FBC*).** As described in section 4.2, `RREQ` messages are forwarded only after checking the bidirectionality of the link over which they were received. No retries are thus required.

  This, again, reflects the assumption that unidirectional links may be present in the network, and the policy of excluding these links from routing paths, *i.e.,* (ii) in the above. No assumptions are made about the density of unidirectional links in the network, *i.e.,* whether they are frequent or rare: *FBC* skips *all* paths with *any* number of unidirectional links, and Route Discoveries only succeed when a bidirectional path is found.

- **Loop Exploration after Reverse Bidirectionality check (*RBC+LE*).** As described in section 5, *LE* is triggered when *RBC* fails (first attempt), and allows the routing protocol to attempt at including unidirectional links in the routing topology, *i.e.,* (iii) in the above. The route discovery is considered failed after 3 loop exploration retries. The hop limit of loop exploration, or Loop Exploration Scope (LES), is $LES = 5$.

This paper uses LOADng as the base protocol, into which each of these are integrated and tested. This choice of routing protocol is *distinctly not* significant. While absolute values likely would change if any other reactive routing protocol was used (*e.g.,* due to control messages being of different sizes), the relative performance of the different options would be similar.

6.3. Simulation Results

The key success metric is the ability to discover paths, and figure 14 depicts the Route Discovery success ratio observed in performed simulations, with error bars (whiskers) corresponding to the standard deviation (σ). Unsurprisingly, when faced with an increasing fraction of unidirectional links α, *single try* yields the least successful route discoveries, whereas *RBC+LE* yields the most successful route discoveries.

As expected, *single-try* also sees the shortest amount of time to complete a Route Discovery (figure 15) – which,

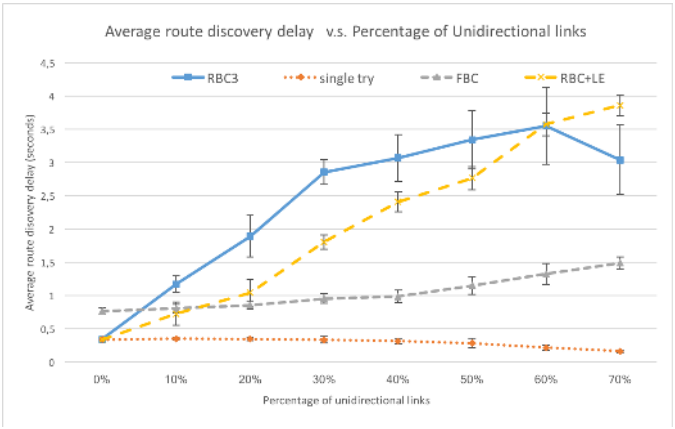

Figure 15: Average Route Discovery delay

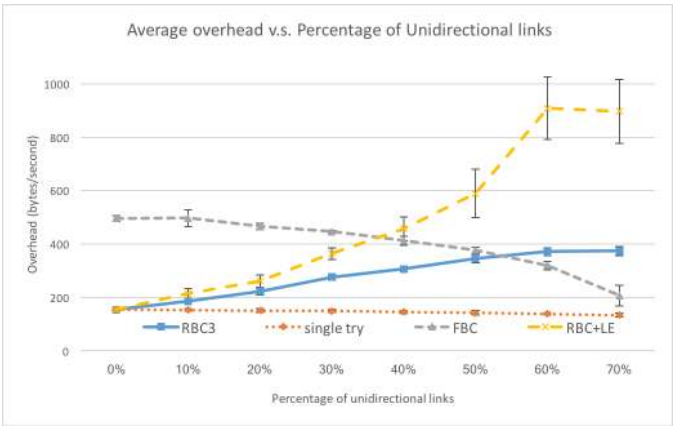

Figure 16: Average control overhead

of course, must be seen in light of the fact that very few Route Discoveries succeed (figure 14), and that those that succeed discover short paths with few links (figure 17). This is also visible in figure 16, where the incurred overhead is constant: *single-try* does not adapt its behaviour depending on the links directionality.

RBC3 manages to maintain a success rate close to 1 until roughly 20% of the links in the network are unidirectional (figure 14) – after which it exhibits the same tendency as *single-try*. This is rather intuitive: from their very definition, blacklisting and retries in *RBC3* allow Route Discoveries to eliminate up to 2 unidirectional links in a path, as detailed in section 6.2, and therefore to handle networks with relatively low fractions of unidirectional links. If paths are likely to contain more than 2 unidirectional links, *RBC3* performance degrades as in *single-try*. *RBC3* also sees a dramatic increase in Route Discovery delays, as per figure 15, plateauing when the fraction of unidirectional links reaches about 50%. This corresponds to a situation in which a substantial fraction of successfully discovered paths have required three retries. This is also visible in figure 16, where the overhead increases as the fraction of unidirectional links increases, and each successful route discovery necessitates retries.

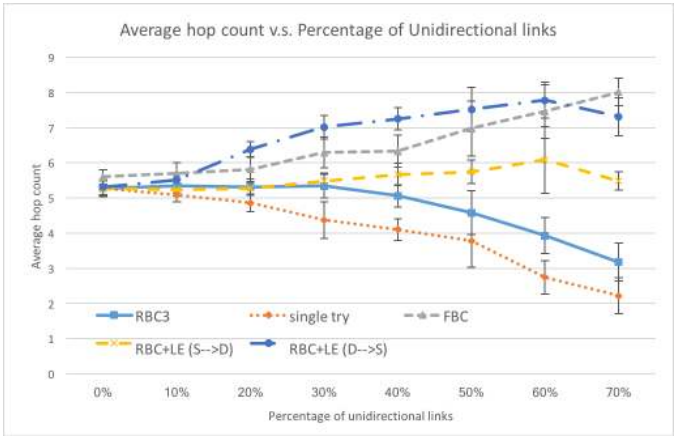

Figure 17: Average path length measured in hop counts

FBC attains (figure 14) a success ratio of > 0.8, as long as $< 50\%$ of the links in the network are unidirectional – after which the success ratio drops off rapidly. Thus, *FBC* does better than *RBC3* at finding bidirectional paths in the network, provided that bidirectional paths do exist. This is confirmed by figure 16, which shows that the overhead for *FBC* starts to drop off sharply when $> 50\%$ of the links are unidirectional: when an `RREQ` is received over a unidirectional link, it is dropped, not forwarded, and thus does not generate further overhead. Of course, Route Discovery cannot succeed in *FBC* when no path exists over only bidirectional links – therefore, the success ratio sharply drops for $> 50\%$ of unidirectional links in the network, when the probability of being bidirectional is $< 0.5^n$ for n-hop paths, *i.e.,* < 0.0078 for paths of ≥ 7 hops.

Compared to *RBC3*, the bidirectionality check of all links in FBC incurs an additional control traffic overhead of around a hundred bytes/second in total over 125 routers (figure 16) – or, roughly, 7 bytes/second per router in the network.

Of note also, *FBC* yields an only slightly longer delay than does *single-try* (figure 15): this is due to the extra required message exchange to verify bidirectionality of each link before forwarding an `RREQ`. The delay incurred slightly increases as the fraction of unidirectional links increases. This is unsurprising, and explained by the fact that longer paths result (and thus, more links require bidirectionality check, see figure 17) as the fraction of unidirectional links in the network increases.

When paths with only bidirectional links become rare, all of *single-try*, *RBC3*, and *FBC* perform poorly. *RBC+LE* yields, as does *FBC*, a success ratio > 0.9 for as long as $< 50\%$ of the links are unidirectional. For $> 50\%$ unidirectional links, the success ratio of both *FBC* and *RBC+LE* drop off – but with *RBC+LE* dropping off much slower: when 70% of the links are unidirectional, *FBC* attains a success ratio of only about 0.46, whereas *RBC+LE* still attains a success ratio of about 0.85.

The delay (figure 15) and overhead (figure 16) incurred by *RBC+LE* both increase steadily, as the fraction of uni-

directional links increases. This is because LE is invoked more frequently when (i) the number of paths with unidirectional links increases, and (ii) the number of unidirectional links on each path increases.

Specifically for LE, the paths $S \rightarrow D$ and $D \rightarrow S$ are not necessarily identical in length, as depicted in figure 17. The path $D \rightarrow S$ is *at least as long* (in hops) as the path $S \rightarrow D$: equal when all links are bidirectional, but with $D \rightarrow S$ being longer when LE is employed to overcome unidirectional links.

It is worth to note that for the delay and path length, only the successfully discovered paths are counted – thus in some scenarios where the ratio of unidirectional links is high, the delay and path length might decrease because only shorter paths are discovered.

7. Conclusion

This paper has studied the challenge of adapting reactive routing protocols for use in networks where a substantial, or even preponderant, portion of links are unidirectional.

This paper provides an analytical model, allowing understanding of the impact of unidirectional links in reactive routing protocol performance, and quantifying potential performance improvements. Subsequent analysis demonstrated that a common mechanism, *Reverse Bidirectionality Check* (*RBC*) (*e.g.,* discovery of only bidirectional paths based on *blacklisting* upon non-reception of an `RREP_ACK`, and timer-driven route discovery retries), which restrict path discovery, can result in both a significant underutilization of communication resources, and a substantial degradation of routing performance, even when the fraction of unidirectional links is relatively small. This paper also identifies situations where *RBC* fails to properly exclude unidirectional links from being included in routing paths, causing *False Forward Routes* (*i.e.,* routes with unidirectional links in the "wrong direction", rendering them useless) to be installed.

This paper has proposed two mechanisms, for better adapting reactive routing protocols to networks with unidirectional links: *Forward Bidirectionality Check* (*FBC*), and *Loop Exploration* (*LE*). These mechanisms have been compared to *RBC3* (*i.e., RBC* with 3 route discovery retries – a common default mechanism in reactive routing protocols [2, 3, 1]).

FBC is a proactive mechanism in which an `RREQ` is only forwarded once the receiving router has established, through a local message exchange, that the link over which the `RREQ` was received is bidirectional. Since bidirectionality is checked, systematically and for every link, before the `RREQ` is forwarded, a possible objection to *FBC* would be that it may cause undue Route Discovery delays, as compared to *RBC3*, in networks with few unidirectional links. However, simulation studies using NS2 in random topology networks with a variable ($0\% - 70\%$) fraction of unidirectional links showed that even in networks with relatively few unidirectional links ($\sim 5\%$), the delay of *RBC3* exceeded that of *FBC*. These simulations also showed that as the fraction of unidirectional links increased, so did the benefit of *FBC*.

The simulations reveal that for an additional overhead (as compared to *RBC3*) of < 7 bytes/second per router in the network, *FBC* enables reactive routing protocols to be resilient, even when faced with 50% of the links in the network being unidirectional.

LE is a reactive mechanism which safely allows a reactive routing protocol to use unidirectional links, when encountered, for constructing (possibly non-bidirectional) paths providing bidirectional communication between a source and a destination. The Route Discovery success ratios for *FBC* and *LE* are comparable as long as $\leq 50\%$ of the links in the network are unidirectional – with *LE* incurring a significantly higher control traffic overhead than *FBC* already when $\geq 10\%$ of the links in the network are unidirectional.

In conclusion, this paper has established that *Forward Bidirectionality Check* yields a considerably lower Route Discovery delay, and a considerably higher Route Discovery success ratio, than does the mechanism for handling unidirectional link included "by default" in reactive routing protocols (*i.e., RBC3*) – and this, for a wide range of fractions of unidirectional links present in the network. In the topologies tested in this paper, this range was from 15% to 50% of the links.

Above that range, and with $\sim 50\% - 70\%$ of the links in the network being unidirectional for the topologies tested in this paper, *Loop Exploration* offers an unparallelled ability to maintain bidirectional communication – with a Route Discovery success ratio of as high as 0.65 when 70% of the links in the network are unidirectional.

References

[1] D. A. Maltz and D. C. Johnson, "The Dynamic Source Routing Protocol (DSR) for Mobile Ad Hoc Networks for IPv4," RFC 4728, mar 2013. [Online]. Available: https://rfc-editor.org/rfc/rfc4728.txt

[2] C. Perkins, E. Belding-Royer, and S. Das, "Ad hoc On-Demand Distance Vector (AODV) Routing," July 2003, Experimental RFC 3561.

[3] T. Clausen, J. Yi, and A. C. de Verdiere, "LOADng: Towards AODV Version 2," in *VTC Fall*. IEEE, 2012, pp. 1–5.

[4] A. Dunkels, "The contikimac radio duty cycling protocol," December 2011.

[5] M. Buettner, G. V. Yee, E. Anderson, and R. Han, "X-mac: A short preamble mac protocol for duty-cycled wireless sensor networks," in *Proc. of ACM SenSys'06*. ACM, 2006.

[6] S. Duquennoy, A. Elsts, B. Al Nahas, and G. Oikonomou, "Tsch and 6tisch for contiki: Challenges, design and evaluation," in *Proc. of IEEE DCOSS'2017*. IEEE, 2017.

[7] A. Colin de Verdiere, "Multicast and flooding in ad hoc networks," 2014. [Online]. Available: http://www.escholarship.org/uc/item/4f65h7sx

[8] M. Gerla, Y.-Z. Lee, J.-S. Park, and Y. Yi, "On demand multicast routing with unidirectional links," in *IEEE Wireless Communications and Networking Conference, 2005*, vol. 4, March 2005, pp. 2162–2167 Vol. 4.

[9] T. Clausen, C. Dearlove, and J. Dean, "Mobile Ad Hoc Network (MANET) Neighborhood Discovery Protocol (NHDP)," Std. Track RFC 6130 (Proposed Standard), The Internet Engineering Task Force (IETF), April 2010. [Online]. Available: http://www.ietf.org/rfc/rfc6130.txt

[10] R. Prakash, "Unidirectional links prove costly in wireless ad hoc networks," in *Proc. DIAL 99*, vol. 4, 1999.

[11] M. K. Marina and S. R. Das, "Routing performance in the presence of unidirectional links in multihop wireless networks," in *Proc. MobiHoc'02*, 2002.

[12] J. G. Jetcheva and D. B. Johnson, "Routing characteristics of ad hoc networks with unidirectional links," *Ad hoc Networks*, vol. 4, no. 3, pp. 303–325, May 2006.

[13] N. Baccour, A. Koubaa, L. Mottola, M. A. Zuniga, H. Youssef, C. A. Boano, and M. Alves, "Radio link quality estimation in wsns: A survey," *ACM Transactions on Sensor Networks*, vol. 8, no. 4, pp. 34:1–34:33, Sept. 2012.

[14] L. Sang, A. Arora, and H. Zhang, "On link asymmetry and one-way estimation in wireless sensor networks," *ACM Transactions on Sensor Networks*, vol. 6, no. 2, Feb. 2010.

[15] R. Prakash, "A routing algorithm for wireless ad hoc networks with unidirectional links," *Wireless Networks*, vol. 7, no. 6, pp. 617–625, Nov. 2001.

[16] V. Shah and S. Krishnamurthy, "Handling asymmetry in power heterogeneous ad hoc networks: A cross layer approach," *Computer Networks*, pp. 2594–2615, 2007.

[17] R. Karnapke, "Unidirectional links in wireless sensor networks," Ph.D Thesis, 2012.

[18] E. Duros, W. Dabbous, H. Izumiyama, N. Fujii, and Y. Zhang, "A Link-Layer Tunneling Mechanism for Unidirectional Links," RFC 3077, Mar. 2001. [Online]. Available: https://rfc-editor.org/rfc/rfc3077.txt

[19] S. Nesargi and R. Prakash, "A tunneling approach to routing with unidirectional links in mobile ad-hoc networks," in *Proc. ICCCN'2000*, 2000.

[20] D. Kim, C.-K. Toh, and Y. Choi, "On supporting link asymmetry in mobile ad hoc networks," in *Proc. GLOBECOM'01*. IEEE, Nov. 2001, pp. 2798–2803.

[21] J. G. Jetcheva and D. B. Johnson, "On-demand multicast routing in ad hoc networks with unidirectional links," 2004.

[22] S.-J. Lee, M. Gerla, and C.-C. Chiang, "On-demand multicast routing protocol," in *WCNC. 1999 IEEE Wireless Communications and Networking Conference (Cat. No.99TH8466)*, 1999, pp. 1298–1302 vol.3.

[23] J. G. Jetcheva, "Adaptive demand-driven multicast routing in multi-hop wireless ad hoc networks," Ph.D. dissertation, Pittsburgh, PA, USA, 2004, aAI3141082.

[24] L. Bao and J. J. Garcia-Luna-Aceves, "Link-state routing in networks with unidirectional links," in *Proc. ICCCN'99*, 1999.

[25] J. Wu and H. Li, "Domination and its applications in ad hoc wireless networks with unidirectional links," in *Proc. International Conference on Parallel Processing (ICPP)*, 2000, pp. 189–197.

[26] J. Wu, "Extended dominating-set-based routing in ad hoc wireless networks with unidirectional links," *IEEE Transactions on Parallel and Distributed Systems*, vol. 13, no. 9, pp. 866–881, Sept. 2002.

[27] Y. bae Ko, S. ju Lee, and J. beom Lee, "Ad hoc routing with early unidirectionality detection and avoidance, personal wireless communications," in *Proc. of International Conference on Personal Wireless Communications (PWC 04)*. Springer, 2004.

[28] D. M. Shrestha and Y. bae Ko, "A new routing protocol in ad hoc networks with unidirectional links," in *Proc. IWDC 2005*. Springer, 2005, pp. 287–292.

[29] R. Karnapke and J. Nolte, "Unidirectional link counter-a routing protocol for wireless sensor networks with many unidirectional links," in *Ad Hoc Networking Workshop (MED-HOC-NET), 2015 14th Annual Mediterranean*. IEEE, 2015, pp. 1–7.

[30] R. Alexander, A. Brandt, J. Vasseur, J. Hui, K. Pister, P. Thubert, P. Levis, R. Struik, R. Kelsey, and T. Winter, "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks," RFC 6550, Mar. 2012. [Online]. Available: https://rfc-editor.org/rfc/rfc6550.txt

[31] O. Iova, F. Theoleyre, and T. Noel, "Using multiparent routing in rpl to increase the stability and the lifetime of the network," *Ad Hoc Networks*, vol. 29, pp. 45–62, 2015.

[32] Y.-C. Tseng, S.-N. Ni, Y.-S. Chen, and J.-P. Sheu, "The broadcast storm problem in a mobile ad hoc network," *Wireless Networks*, vol. 8, no. 2, pp. 153–167, Mar. 2002.

[33] J. A. Cordero, J. Yi, and T. Clausen, "An adaptive jitter mechanism for reactive route discovery in sensor networks," *Sensors*, vol. 14, no. 8, pp. 14 440–14 471, 2014. [Online]. Available: http://jiaziyi.com/wp-content/uploads/2016/08/An-Adaptive-Jitter-Mechanism-for-Reactive-Route-Discovery-in-Sensor-Networks.pdf http://www.mdpi.com/1424-8220/14/8/14440/htm

[34] V. Ramasubramanian, R. Chandra, and D. Mossé, "Providing a bidirectional abstraction for unidirectional ad hoc networks," in *Proc. INFOCOM'02*, 2002.

[35] T. Clausen, A. C. de Verdiere, J. Yi, A. Niktash, Y. Igarashi, H. Satoh, U. Herberg, C. Lavenu, T. Lys, and J. Dean, "The Lightweight On-demand Ad hoc Distance-vector Routing Protocol – Next Generation (LOADng)," Internet Draft, work in progress, draft-clausen-lln-loadng, IETF, July 2015.

[36] T. Clausen, C. Dearlove, J. Dean, and C. Adjih, "Generalized manet packet/message format," Std. Track RFC 5444 (Proposed Standard), The Internet Engineering Task Force (IETF), February 2009. [Online]. Available: http://www.ietf.org/rfc/rfc5444.txt

[37] "Wireless lan medium access control (mac) and physical layer (phy) specifications," IEEE 802.11-2012, 2012.

[38] T. Clausen and J. Yi, "Path accumulation extensions for the loadng routing protocol in sensor networks," in *Internet of Vehicles ? Technologies and Services*, ser. Lecture Notes in Computer Science, R.-H. Hsu and S. Wang, Eds., vol. 8662. Springer International Publishing, 2014, pp. 150–159.

[39] D. B. Johnson and D. A. Maltz, "Dynamic source routing in ad hoc wireless networks," in *Mobile Computing*. Kluwer Academic Publishers, 1996, pp. 153–181.

[40] S. Gwalani, E. M. Belding-Royer, and C. E. Perkins, "Aodv-pa: Aodv with path accumulation," in *Communications, 2003. ICC '03. IEEE International Conference on*, vol. 1, May 2003, pp. 527–531 vol.1.