Candidate Selection Algorithms in Opportunistic Routing

Llorenç Cerdà-Alabern and Amir Darehshoorzadeh
Univ. Politécnica de Catalunya
Computer Architecture Dep.
Barcelona, Spain
Email: {llorenc,amir}@ac.upc.edu

Abstract—Opportunistic Routing (OR) has been investigated in recent years as a way to increase the performance of multihop wireless networks by exploiting its broadcast nature. In contrast to traditional routing, where traffic is sent along pre-determined paths, in OR an ordered set of candidates is selected for each next-hop. Upon each transmission, the candidates coordinate such that the most priority one receiving the packet actually forward it. Most of the research in OR has been addressed to investigate candidate selection algorithms. In this paper we compare a selected group of algorithms that have been proposed in the literature. Our main conclusion is that optimality is obtained at a high computational cost, with a performance gain very similar to that of much simpler but non-optimal algorithms. Therefore, we conclude that fast and simple OR candidate selection algorithms may be preferable in dynamic networks, where the candidate sets are likely to be updated frequently.

I. INTRODUCTION

Opportunistic Routing (OR), also referred to as diversity forwarding [13], cooperative forwarding [12] or any-path routing [11], has been proposed to increase the performance of multihop wireless networks by taking advantage of its broadcast nature. In OR, in contrast to traditional routing, instead of preselecting a single specific node to be the next-hop forwarder, an ordered set of nodes (referred to as candidates) is selected as the next-hop potential forwarders. Thus, the source can use multiple potential paths to deliver the packets to the destination. More specifically, when the current node transmits a packet, all the candidates that successfully receive it will coordinate such that the most priority one will actually forward the packet, while the others will simply discard it.

We shall refer to as candidate selection the algorithm which is run in the nodes addressed to choose and prioritize for every destination the list of neighbors (candidates) that can better help in the forwarding process. The aim of candidate selection algorithms is to minimize the expected number of transmissions from the source to the destination. Another key issue in OR is the candidate coordination, i.e. the mechanism used by the candidates to discover which is the most priority candidate that has received, and thus, must forward the packet. Coordination requires signaling among the nodes, and imperfect coordination may cause duplicate transmission of packets. However, given an equal number of candidates, coordination overhead is likely to be the same for different candidate selection algorithms. Therefore, for simplicity we shall compare the candidate selection algorithms assuming perfect coordination.

With perfect coordination among candidates, it is likely that the larger is the number of candidates (given that they approach to the destination) the lower is the expected number of transmissions from the source to the destination. However, increasing the number of candidates increases also the coordination overhead. Therefore, in practice, the maximum number of candidates that can be used is limited. This fact has been often neglected in candidate selection algorithms that have been proposed in the literature. I.e. the algorithms have been designed to select all candidates, without limit, that can reduce the expected number of transmissions, using the assumption of perfect coordination. Since we believe that this will be a practical constraint, we do consider it in our analysis. To do so, we propose modifications to the candidate selection algorithms under study that allow limiting the maximum number of candidates.

Previous works usually compare OR using the proposed candidate selection algorithm against a traditional uni-path routing algorithm. In our study we have compared four candidate selection algorithms that have been proposed in the literature. They range from non-optimum, but simple, to optimum, but with a high computational cost. We address the questions: Is there a big difference in performance between the simple and optimal algorithms? What is the computational cost in terms of maximum number of candidates? So, under which conditions it is worth using an optimal algorithm? Our main contribution is showing that optimal algorithms have a high computational cost, even for a small maximum number of candidates. Additionally, their performance gain is very similar to that of other non-optimal algorithms. Therefore, fast and simple OR candidate selection algorithms may be preferable in dynamic networks, where the candidate sets are likely to be updated frequently.

The remainder of this paper is organized as follows. Section II surveys the related work. In Section III we introduce the routing metrics which are usually used in OR. The description of the four candidate selection algorithms under study is done in Section IV. Section V introduces the methodology and the propagation model which we have used in our numerical experiments. The numerical results are presented and discussed in Section VI and concluding remarks are made in Section VII.
II. RELATED WORK

Biswas and Morris proposed ExOR [4], [5], one of the first and most referenced OR protocols. MORE [8] is a MAC independent protocol that used both the idea of OR and network coding. It avoids duplicate packet transmissions by randomly mixing packets before forwarding them. In [18], [17] Zhong et al. proposed a new metric – expected any-path random mixing packets before forwarding them. In [18], independent protocol that used both the idea of OR and most referenced OR protocols. MORE [8] is a MAC expected transmission count of the link is:

\[ ETX = \frac{1}{p_{ij}} \quad (1) \]

In opportunistic routing, however, it is necessary to consider the fact that there are some candidates which can receive the packet, thus, a packet may travel along any of the potential paths. So using the ETX to compute the expected number of transmissions does not give an accurate metric for OR. Expected Any-path Transmission (EAX): This metric was defined by Zhong et al. in [18] to capture the expected number of transmissions taking into account the multiple paths that can be used under OR. Let \( s \) be a source node, with \( c_i \) the candidate with priority \( i \) (with \( i = 1 \) being the highest priority) and \( p_i \) the delivery probability between \( s \) and \( c_i \). Using the same assumptions as in equation (1), the expected number of transmissions to reach a destination node \( d \) is given by the recursive formula:

\[ EAX(C^{s,d}, s, d) = S(C^{s,d}, s, d) + Z(C^{s,d}, s, d) \quad (2) \]

\[ S(C^{s,d}, s, d) = \frac{1}{1 - \prod_{i=1}^{C^{s,d}} (1 - p_i)} \quad (3) \]

\[ Z(C^{s,d}, s, d) = \frac{\sum_{i=1}^{C^{s,d}} EAX(C^{c_i,d}, c_i, d) p_i \prod_{j=1}^{i-1} (1 - p_j)}{1 - \prod_{i=1}^{C^{s,d}} (1 - p_i)} \quad (4) \]

where \( C^{s,d} \) is the candidate set of node \( s \) to reach the destination \( d \), and \( |C^{s,d}| \) its cardinality; \( S(C^{s,d}, s, d) \) is the expected number of transmissions from \( s \) until at least one of the nodes in \( C^{s,d} \) receives the packet; and \( Z(C^{s,d}, s, d) \) is the expected number of transmissions to reach the destination \( d \) from one of the nodes in \( C^{s,d} \) which is responsible to forward the packet. Note that in equation (4) we take the product \( \prod_{j=1}^{i-1} \) equal to 1 for \( i = 1 \).

Equivalent expressions to the EAX formula (2) have been obtained by several authors using different approaches [10], [15]. In [6] equation (2) is straightforward from the Markov chain we propose to model OR. Although the ETX equation is much simple to compute, it does not accurately compute the expected number of transmissions under OR. Due to this reason, some authors have shown that using ETX may give suboptimal selection of candidates (see e.g. [10], [15]).

IV. CANDIDATE SELECTION ALGORITHMS

In this section, we briefly describe the candidate selection algorithms under study. These algorithms are: Extremely Opportunistic Routing (ExOR) [4]; Opportunistic Any-Path Forwarding (OAPF) [18]; Least-Cost Opportunistic Routing (LCOR) [10]; and Minimum Transmission Selection (MTS) [15].

ExOR is one of the first and most referenced OR protocols, it is based on ETX and is simple to implement. OAPF has an intermediate complexity: It uses the EAX metric but it does not guarantee to yield the optimum set of candidates. Finally, we have chosen LCOR and MTS because, as far as we know, they are the only two algorithms in the literature that select the optimum set of candidates (i.e. the candidates sets that minimize the expected number of transmissions).

Here we introduce some notations that we use throughout this paper:
Algorithm 1 Candidate selection ExOR($s$, $d$, $ncand$).

1: $G_{tmp} = \text{temporal copy of the network topology}$
2: $\text{cost}(s) \leftarrow \text{ETX}(s, d)$ in $G_{tmp}$; $C^{s,d} \leftarrow \emptyset$
3: while $|C^{s,d}| < ncand$ & $(s, d)$ connected in $G_{tmp}$ do
4: $\text{cand} \leftarrow \text{first node after } s \text{ in the SPF}(s, d)$ in $G_{tmp}$
5: if cond $\implies d$ then
6: $C^{s,d} \leftarrow C^{s,d} \cup d$
7: $\text{cost}(\text{cand}) \leftarrow 0$
8: else
9: $\text{cost}(\text{cand}) \leftarrow \text{ETX}(\text{cand}, d)$ in $G_{tmp}$
10: if $\text{cost}(\text{cand}) < \text{cost}(s)$ then
11: $C^{s,d} \leftarrow C^{s,d} \cup \text{cand}$
12: end if
13: end if
14: end while
15: $C^{s,d} \leftarrow C^{s,d}$ ordered by cost

- $ncand$ is the maximum number of candidates in each node. We shall refer as $ncand = \infty$ to the case when the maximum number of candidates is not limited. We shall also use the notation ExOR($n$) to refer to ExOR with $ncand = n,$ and similarly for the other algorithms under study (see the legend of figures 1-4).
- $\text{ETX}(v, d)$ is the uni-path ETX between two nodes $v$ and $d$.
- $\text{EAX}(C^{v,d}, v, d)$ is the EAX between node $v$ and $d$ by using $C^{v,d}$ as the candidates set of $v$ to reach node $d$. $|N(v)|$ is the set of all neighbors of node $v$.
- $|S|$ is the cardinality of the set $S$.

In the following subsections we describe the implementation that we have done for each of the selection algorithms under study. For the sake of being precise, we shall give a pseudocode summarizing our implementations.

A. ExOR

ExOR [4] uses ETX as the metric for selecting candidates. Algorithm 1 shows our implementation of ExOR. Every node $s$ runs this algorithm for a destination $d$. The basic idea of ExOR is running the shortest path first (SPF) with weight $\frac{1}{p_{ij}}$, where $p_{ij}$ is the delivery probability between two nodes $i$ and $j$ (see section III). The first node after the source in this path is selected as candidate (cand). The $\text{ETX}(\text{cand}, d)$ (or 0 if the cand is the destination) is stored for the priority ordering. Then the link between $s$ and cand is removed, and the loop is repeated until no more paths to $d$ are available, or the maximum number of candidates is reached.

B. OAPF

This algorithm [18] is a hop-by-hop opportunistic routing which is based on ETX and EAX. The pseudocode of OAPF is shown in Algorithm 2. Assume that node $s$ wants to select its candidates set to reach the destination $d$. It creates an initial candidate set ($\hat{C}^{s,d}$). A neighbor $v$ of $s$ will be included in the initial candidate set only if $\text{ETX}(v, d) < \text{ETX}(s, d)$. Note that, all nodes in the initial candidate set must select their candidates set before $s$. The actual candidates set of $s$ will be a subset of the initial candidate set. After initiating the candidate set, $s$ selects the best candidate among the nodes in the initial candidate set. Here, the best candidate is the one that mostly reduces the expected number of transmission from $s$ to the destination. Node $s$ adds the best candidate to its actual candidate set ($C^{s,d}$) and removes it from its initial set. Node $s$ tries again to find the best node from its new initial candidate set. This process is repeated until there is not any other suitable node to be included in the candidates set of $s$, or the number of candidates in the $C^{s,d}$ reaches the maximum number of candidates ($ncand$). Finally, the candidate set is ordered by their EAX.

C. LCOR

The goal of this algorithm is to find the optimal candidates sets. LCOR [10] uses EAX as the metric to select candidates as shown in Algorithm 3.

The algorithm starts by initializing the cost (EAX) of each
The neighbors of node \( v_1 \) are the nodes that are closest to the destination and not neighbors of \( v \). The general idea of MTS consists of an exhaustive search over all possible subsets of \( N(v) \). The optimal candidate sets in the case of infinite number of candidates, and then we look for the best subset of candidate sets with at most \( ncand \) elements, the final candidate sets will be the optimal candidate sets.

**V. Methodology**

In order to compare the algorithms under study, and since we want to focus on the effect of candidate selection, we have considered the following scenario: (i) in the network there is only one active connection; (ii) perfect coordination between the candidates, i.e. the most priority candidate successfully receiving the packet will be the next forwarder; (iii) the nodes retransmit the packets until successful delivery. In the next section we describe the network topology.

To assess the delivery probabilities of the links we have used the shadowing propagation model (equation 5). The power received at a distance \( d (P_r(d)) \), in terms of the transmitted power \( (P_t) \) is given by:

\[
P_r(d)_{dB} = 10 \log_{10} \left( \frac{P_t \cdot G_t \cdot G_r \cdot \lambda^2}{L \cdot (4 \pi)^2 \cdot d^\beta} \right) + X_{dB} \tag{5}
\]

Where \( P_r(d) \) is the power received at a distance \( d \) and \( P_t \) is the transmitted power. The \( G_t \) and \( G_r \) are the transmission and reception antenna gains respectively, \( L \) is a system loss, \( \lambda \) is the signal wavelength \((c/\beta)\), with \( c = 3 \times 10^8 \) m/s, \( \beta \) is a path loss exponent and \( X_{dB} \) is a Gaussian random variable with zero mean and standard deviation \( \sigma_{dB} \). Larger \( \beta \), indicates more obstructions and hence, faster decrease in average received power as distance becomes longer. In our simulation we have used \( \beta = 2.7 \) and \( \sigma_{dB} = 6 \) dBs.

Packets are delivered correctly if the received power is greater than or equal to \( RXThresh \). Note that we shall no consider collisions in our model. Thus, the delivery probability at a distance \( d \) is given by:

\[
p(d) = \text{Prob}(P_r(d)_{dB} \geq 10 \log_{10}(RXThresh)) \tag{6}
\]

We have set the model parameters to the default values used by the network simulator (Ns-2) \([2]\), given in table I.

**VI. Performance Evaluation**

In this section we study the performance of the candidate selection algorithms described in section IV. We consider scenarios with different number of nodes \((10 \leq N \leq 50)\).
randomly placed in a square field with diagonal $D = 300$ m, except the source and the destination which are placed at the diagonal end points. Each point in the plots is an average of 10 runs with different random node positions. The delivery probabilities have been assigned with the shadowing model described in section V. We have assumed that a link between any two nodes exists only if the delivery probability between them is greater (or equal) than $\min dp = 0.1$. We have compared the algorithms for different maximum number of candidates: $\text{ncand} = 2, 3, 4, 5, \infty$. Recall that we refer as $\text{ncand} = \infty$ to the case when there is no limit on the maximum number of candidates.

We compare the performance of each algorithm in terms of the expected number of transmissions needed to send a packet from the source to the destination and the execution time of each algorithm. The expected number of transmissions has been obtained using the EAX formula (2). The execution times have been obtained running the algorithms on a PC with 2 processors Intel Xeon Dual-Core 2 with 4MB cache and 12 GB of memory.

### A. Expected number of transmission

First, we will do a detailed study using the $\text{ncand} = 3$, as shown in Figure 1. For each point in the figure we have added error bars at 95% confidence interval. For the sake of comparison, we have included the scenarios using uni-path routing and also the optimum candidate selection algorithm in the case $\text{ncand} = \infty$ (Opt($\infty$)). Note that uni-path routing is equivalent to use $\text{ncand} = 1$ in any of the OR algorithms under study. The curves have been obtained varying the number of nodes, but maintaining the distance $D = 300$ m between the source and the destination, thus, increasing the density of the network.

As a first observation in figure 1, we can see that using any OR algorithm outperforms the traditional uni-path routing. Regarding the optimum algorithms, LCOR and MTS, we have validated that they choose exactly the same candidates sets, and thus, the curves are the same. The same was true for $\text{ncand} = \infty$, so with Opt($\infty$) we show only one of the curves obtained with LCOR($\infty$) and MTS($\infty$). About OAPF, we can see that the expected number of transmissions is only slightly larger than those obtained with the optimum algorithms. Finally, we observe the expected number of transmissions required by ExOR is significantly larger than the other OR algorithms. The reasons that motivate this inferior performance of ExOR are the following: Recall that ExOR is a simple algorithm that looks for the candidates running SPF after removing the links to the nodes that have already been selected as candidates. By doing this, the candidates tend to be chosen close to each other. In [7] we have investigated the optimal position of the candidates and we have shown that they are not clustered, but distributed over distances that approximate to the destination. Therefore, we conclude that ExOR does a coarse selection of the candidates set. On the other hand, recall that OAPF incrementally adds the nodes to the candidates set that are most effective at reducing the expected number of transmissions (EAX). Although this does not guarantee choosing the optimum candidates sets, we can see from the figure that the results are very close to the optimum.

Regarding the scenario with $\text{ncand} = \infty$, figure 1 shows that achieves a noticeable reduction of the expected number of transmissions compared to the scenario with $\text{ncand} = 3$. However, as shown in figure 2, this is at cost of using a large number of candidates. Note that implementing an OR protocol with a high number of candidates is difficult, and possibly will introduce large signaling overhead and duplicated transmissions. Therefore, the differences obtained with $\text{ncand} = 3$ and $\text{ncand} = \infty$ in a real scenario, are likely to be much smaller than those shown in figure 1.
For other scenarios we have obtained similar results. For instance, figures 3 and 4 have been obtained respectively maintaining the total number of nodes equal to \(N = 10\) and \(N = 50\) (thus, representing a low and high density network), and varying the maximum number of candidates to: \(ncand = 1, 2, \cdots, 5\) and \(\infty\). Note that \(ncand = 1\) is equivalent to uni-path routing, thus, the expected number of transmissions obtained for \(ncand = 1\) is the same for all algorithms. In the case of \(ncand = \infty\) all algorithms have almost the same expected number of transmissions. This comes from the fact that in this case there is not any limitation on the maximum number of candidates. Therefore, all nodes which are closer to the destination than the source can be selected as candidates, and all of the algorithms have almost the same candidates sets.

Comparing figures 3 and 4 we can see that the difference between ExOR and the other algorithms is higher in a dense network (\(N = 50\)). This comes from the fact that in a dense network there is a larger number of possible choices of the candidates sets. Thus, limiting the maximum number of candidates makes the selection of the candidates sets more critical. However, we can see that the difference between OAPF and the optimum algorithms is kept small even in a dense network.

### B. Execution Time

In this section we evaluate the computational cost of the algorithms under study by measuring its execution time. This is shown in Figure 5, in logarithmic scale. These measures have been obtained averaging over the 10 runs of the corresponding points in Figure 1. We have selected \(ncand = 3\) as a sample case for our study. Recall that for MTS, we run first the algorithm without limiting the number of candidates (thus, equivalent to \(ncand = \infty\)), and then we do an exhaustive search over all subsets of the candidates sets with cardinality \(\leq ncand\). For this reason we have depicted the curve MTS(\(\infty\)), that corresponds to the first phase, and MTS(3) which shows the overall execution time.

As we expected, the fastest algorithm is ExOR, and LCOR has the longest execution time. For instance, when the number of nodes in the network is 50, LCOR needs about 3 hours and a half to finish. Obviously, with a maximum number of candidates larger than 5 the execution time will be much larger. OAPF is in the middle of the exhaustive search of the optimal algorithms and the simplicity of ExOR, and thus, has an execution time that falls in between these algorithms. E.g. it is 0.7 to 47 seconds for the low and high density networks, respectively.

From MTS(\(\infty\)) in Figure 5 we can see that MTS is rather fast obtaining the optimum candidates sets for \(ncand = \infty\) (if we compared it with LCOR). Recall that MTS(3) first looks for the optimal candidate sets without limiting the maximum number of candidates, and then the candidates sets are pruned to at most 3 elements. Therefore, the searching space for finding the optimal sets in MTS(3) is less than LCOR(3), which examines all the subsets of the neighbors of the nodes.

Comparing the two optimum algorithms that have been proposed in the literature, we can conclude that MTS outper-
forms LCOR in terms of the execution time. Additionally, it is possible to obtain candidates selection algorithms, as OAPF, that have a performance close to the optimum algorithms with a much lower execution time. With simple algorithms as ExOR, the performance may be significantly lower than the optimum.

VII. Conclusions

In this paper we have compared four relevant algorithms that have been proposed in the literature for the candidate selection in OR: Extremely Opportunistic Routing (ExOR); Opportunistic Any-Path Forwarding (OAPF); Least-Cost Opportunistic Routing (LCOR); and Minimum Transmission Selection (MTS). They range from the simplicity of ExOR, the intermediate computational complexity of OAPF, to the optimum but high computational cost of LCOR and MTS.

We have modified the algorithms such that the maximum number of candidates can be limited. The comparison of the algorithms has been made in terms of the expected number of transmissions needed to send a packet, and the execution time to compute the candidates sets.

Our results show that using any OR algorithm outperforms the traditional uni-path routing. Furthermore, if the maximum number of candidates is not limited, all of the algorithms have almost the same expected number of transmissions. This situation is not realistic, since the algorithms may choose a large number of candidates, which will introduce large signaling overhead and duplicated transmissions. When the maximum number of candidates is limited, our results show that the expected number of transmissions required by ExOR is significantly larger than the other OR algorithms. This is because of the coarse selection of the candidates sets of ExOR. However, the performance obtained with OAPF has always been very close to the optimal algorithms.

Regarding the execution times, we have obtained that MTS outperforms LCOR. However, both algorithms requires extremely large times to compute the candidates sets in a dense network (on the order of hours in a modern PC). On the other hand, OAPF is able to run the candidate selection with execution time orders of magnitude lower (on the order of minutes). Therefore, we conclude that a fast and simple OR candidate selection algorithm (as OAPF) may be preferable in dynamic networks, where the candidate sets are likely to be updated frequently.

VIII. Acknowledgments

This work was supported by the Ministry of Education of Spain under grants TSI2007-66869-C02-01 and TSI2007-66869-C02-02, and by the European NoE EuroNF.

References