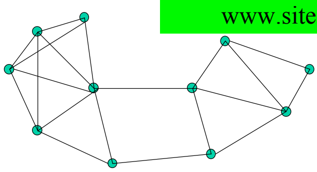
Routing in ad hoc and sensor networks with a realistic physical layer

Tutorial

Ivan Stojmenovic
Ivan@site.uottawa.ca
www.site.uottawa.ca/~ivan

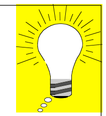


Is unit disk graph realistic ??

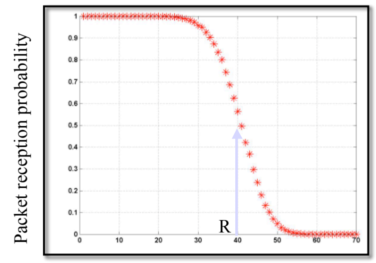
UDG
radius

•Routing: source → destination

Physical layer impact



Lognormal shadowing model



β =4
p(R)=0.5

Unit graph model:
Prp(x)=1, x≤R
Prp(x)=0, x>R

Distance between nodes

What is the transmission radius ?

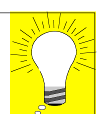
Simulation dillema

- Home-made simulator or one used by others (NS-2, Qualnet, J-sim,…)?
- Greedy routing uses hop count as measure
- NS-2 applies realistic physical layer, which mostly eliminates long hops →
- Why to use simulator that defeats the model, hides physical models and parameters which impact the data, impact comparison, and provide no explanation?
- Solution: build protocols and simulators in parallel, so that results can be explained and protocols improved →
- Network layer protocol need to be designed with more realistic physical layer, not with unit disk graph model

How to simulate ?

- Study one variable at a time, explain it fully
- Ideal MAC, no congestion
- If one routing A is on average better than one routing B, it should cause less congestion, thus show even more advantage at the transport layer
- Simulation to match 'ideal' assumptions
- Stable graphs first; localized design takes care of dynamics
- Independent variable is one that matters e.g. *density* (average number of neighbors per node), not transmission radius
- Compare against the best (e.g. shortest path), not against worst (e.g. flooding)

Physical layer impact



$p(x) \approx 1-(x/R)^{q\beta}/2$ for $x < R$

$\approx (2-x/R)^{q\beta}/2$ for $x \geq R$

q depends on L, packet length

- Signal strength is a random variable
- Transmission power is assumed fixed and same
- q=1 for L=1; q≈2 for L=120.
- β - The power attenuation factor $2 \leq \beta \leq 6$
- No coding schemes assumed currently
  - each bit is received or not
  - packet received correctly iff all bits received

Neighbors? Connectivity?

- Definition of q-connectivity:
- **Two nodes at distance *x* are neighbors if and only if *p(x)≥q*.**
- Definitions of q-connected graph?
- Properties and consequences of such definitions?

Acknowledgment based Algorithms

- HHR: hop by hop acknowledgement during the routing process
- Message is divided into packets
- Each packet and ack of same fixed size
- Expected Hop Count EHC:
  - Expected number of messages between sender and receiver (including retransmissions, acknowledgements etc)

MAC Layer between 2 nodes

- Sender: Repeat sending packet until acknowledgement received
- Receiver: Ack each received packet **u** times

Exp. hop count

$1/(p(x)(1-(1-p(x))^u))+u/(1-(1-p(x))^u)=f(x,u)$

Optimal u value depends on x, $u*p(x) \approx 1$

Reactive routing with physical layer

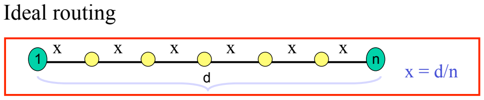
- In route discovery phase, forward the sum of expected hop counts along partial route, or
- timeout that corresponds to the expected hop count if all hops were of the same length as the last one
- Signal strength can be converted to distance, then apply distance based formula
- A single retransmission by a given node may not reach the best forwarding neighbor; tradeoff # of retransmissions and gains made
- Real traffic may not use routes created by control traffic – different packet lengths

Hello messages with physical layer

- *'fixed hello protocol'*
- Send hello messages fixed number of times, to increase the probability of reception by neighbors
- *'variable hello protocol'*
- Send hello packets until <u>sufficient</u> number of such packets from neighbors received
  (learn enough neighbors for desired density)

Greedy routing is not hop count optimal

- Ideal routing

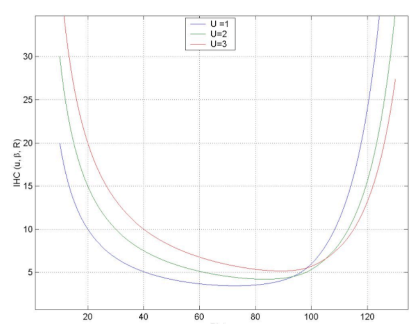


  - Place additional nodes between Source and Destination as required.
  - Ideal Hop count computed for different u and β values

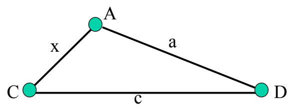
- Low values for 0.6R≤x ≤ 0.9R u=1
- 50% higher at x=R, very high x>R or x<0.1R

•Kuruvila, Nayak, Stojmenovic 2004

IHC for Different u Values (β=2)

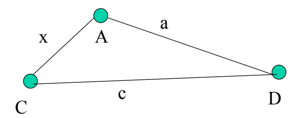


Ideal Hop Count Routing (IHCR)



- Node C will forward the message to a neighbor A that minimizes the sum of EHC(CA) + IHC(AD)
  - EHC – Expected Hop Count
  - IHC – Ideal Hop Count
- u depends on CA for EHC(CA) calculation, for IHC(AD) calculation u = 1

Expected progress routing



Progress: **c-a**

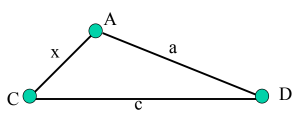
Expected hop count for u=1: **f(x,1) = 1/p²(x)+1/p(x)**

Best value of u: **u≈1/p(x)**

Forward to neighbor (closer to destination) that maximizes

(c-a)/f(x,1) (EPR-1) or (c-a)/f(x,u) (EPR-u)

Projection Progress Routing



The progress in forwarding from C to A is measured by the dot product |CD|.|CA|

1-Projection and u-projection variants are identical to EPR-1 and EPR-u except progress measure.

tR Greedy Algorithm

- The redefined notion of greedy routing.
- Current node S selects neighbor closest to D among all neighbors that are closer to D than itself, and which are at distance at most tR from S, for forwarding the message.
- Experiments for t = 1, 1.25 and 1.4377
- Threshold based greedy routing

Simulation Setup

- Simulation using MATLAB environment
- 250 nodes
- Randomly generated, *connected* unit graphs
- Density d is the average number of neighbors per node
- Neighborhood Criteria
  - Two nodes are considered neighbors, if and only if the distance between then is at most hR where p(R)= .5
  - A selection of p(hR) = 0.05 results in h = 1.4377
- Density is defined with respect to 1.4377R
- β value is fixed as 2

Performance Measures

- averages
  - Success rate in finding a route from the source to the destination
  - the expected hop count value of the found route.
- Hop count results presented as **dilation** ratios.
  - *Hop count dilation* is defined the ratio of the expected hop count performance of the specific algorithm to that of the *shortest weighted path algorithm (weight = EHC)*

Hop Count Dilations

Algorithm	Number of Nodes : 250							
	Density (with 1.4377R neighbors)							
	6	8	10	20	24	32	40	80
Ideal	0.555	0.591	0.651	0.831	0.855	0.887	0.910	0.946
Shortest Path	1	1	1	1	1	1	1	1
aEPR	1.335	1.356	1.355	1.123	1.069	1.065	1.049	1.038
aEPR-1	1.309	1.357	1.372	1.124	1.069	1.067	1.048	1.036
aEPR-u	1.362	1.392	1.426	1.145	1.093	1.077	1.057	1.037
IHCR	1.348	1.356	1.356	1.107	1.067	1.060	1.047	1.035
Proj Progress	1.343	1.344	1.347	1.123	1.071	1.075	1.060	1.062
1-Projection	1.320	1.348	1.341	1.119	1.069	1.073	1.059	1.063
u-Projection	1.343	1.373	1.380	1.129	1.084	1.074	1.062	1.064
1.4377R Greedy	3.576	3.701	4.140	5.477	5.827	6.250	6.715	7.316
1.25R Greedy	1.618	1.676	1.790	2.331	2.439	2.565	2.709	3.008
R Greedy	1.034	1.058	1.091	1.160	1.163	1.201	1.224	1.276

9

Success Rates

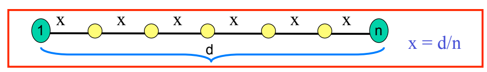
Algorithm	Number of Nodes : 250							
	Density (with 1.4377R neighbors)							
	6	8	10	20	24	32	40	80
Shortest Path	100%	100%	100%	100%	100%	100%	100%	100%
aEPR	36%	50.4%	74.4%	100%	100%	100%	100%	100%
aEPR-1	36.4%	52%	75.2%	100%	100%	100%	100%	100%
aEPR-u	37.6%	51.6%	75.6%	100%	100%	100%	100%	100%
IHCR	33.2%	47.6%	70.8%	100%	100%	100%	100%	100%
Proj Progress	34.4%	49.2%	73.2%	100%	100%	100%	100%	100%
1-Projection	35.6%	51.2%	75.2%	100%	100%	100%	100%	100%
u-Projection	36.8%	51.2%	75.6%	100%	100%	100%	100%	100%
1.4377R Greedy	45.2%	68.8%	81.2%	100%	100%	100%	100%	100%
1.25R Greedy	12%	26.8%	50.4%	98.4%	98.8%	100%	100%	100%
R Greedy	0.4%	1.2%	6%	81.6%	89.6%	99%	100%	100%

Performance summary

- Good performance for localized algorithms
- EPR, IHCR, Projection Progress have low hop count dilations for dense networks and 100% success rates
- tR-greedy are significantly inferior to all above methods: a choice of 'long' edge is quite likely on a route which then contributes to very high expected hop count measure

Non-ack based Algorithms

- Non-acknowledgement based algorithms
  - EER: end-to-end routing without acknowledgements
  - Consider hops at distances $x1, x2,..xn$
  - Probability that the message is received at destination is the product $p(x1)\ p(x2)...p(xn)$
- Ideal case

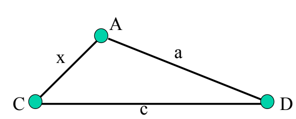


  - Place nodes at equal distances between the source and destination
  - The probability of delivery at destination is $(p(x))^n$

Ideal case … cont'd

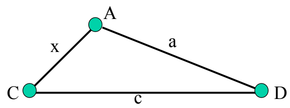
- Non acknowledgement based algorithms aim to maximize the probability of delivery of the message from the source to the destination
- By taking logarithm, we can write the probability of delivery as $n*\log(p(x))$, where $n=d/x$
- Using l'Hopital's rule, we can show that optimal case is when $x= 0$ (and ideal probability = 1)

EER Routing Algorithm



- Node C will forward the message to a neighbor A (closer to destination than itself) that maximizes the sum of logarithmic probability of delivery from C to A and the ideal probability of delivery from A to D
  - Same as delivering to the closest neighbor of node C with advance toward destination
- EER algorithm is same as the **NC (nearest closer)** algorithm

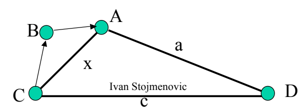
Expected Progress Routing (nEPR)



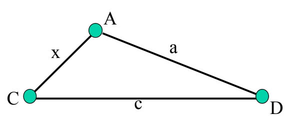
- Progress made in forwarding from C to A: **(c-a)**
- There are no acknowledgements. In nEPR, the node C which is holding the message will forward to a neighbor A (closer to destination than itself) which maximizes the product of the probability of delivery from C to A and the progress made (i.e,
- Maximize probabilistic progress p(CA)*(c-a))

Iterative EPR (InEPR)

- Algorithm Steps
  - Find neighbor A to forward from source C (as in nEPR)
  - Try to find an node B, common neighbor of C and A, closer to destination than C, that satisfies p(|CB|)p(|BA|) > p(|CA|) and has the maximum p(|CB|)p(|CA|) measure
  - If found, A=B,
  - Repeat the above step until no further improvements are possible.
  - C then forwards the message to A.
  - Repeat such forwarding until the destination is reached.



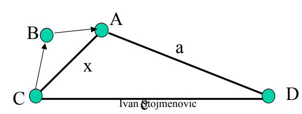
Projection Progress Algorithm



- Progress measured differently compared to EPR algorithms. The progress in forwarding from C to A is measured by the dot product of CD·CA
- In Projection Progress Algorithm, node C which is holding the message will forward it to a neighbor A (closer to destination than itself) which maximizes p(CA)* (CD·CA)

Iterative Projection Progress

- Algorithm Steps
  - Find node A to forward from source C (from Projection progress)
  - Try to find node B, common neighbor of C and A, closer to destination than C, that satisfies p(|CB|)p(|BA|) > p(|CA|) and has the maximum p(|CB|)p(|CA|) measure
  - If found, A=B.
  - Repeat the above step until no further improvements are possible.
  - C then forwards the message to A.
  - Repeat such forwarding until the destination is reached.



tR Greedy Algorithm

- The redefined notion of greedy routing, which allows flexibility of neighborhood definition.
- In tR greedy routing, a source node S considers all neighbors (that are closer to destination than itself) which are at distance at most tR from S, for forwarding the message.
- Select neighbor closest to D among them
- Experiments for tR greedy with t = 1, 1.25 and 1.4377

Simulation: Performance Measures

- Performance Measures (averages)
  - Success rate in finding a route from the source to the destination
  - the product of probabilities of successful delivery along all hops of the computed route
- Probability results presented as **dilation** ratios.
  - *Probability dilation* is defined the ratio of the probability of successful reception at the destination of an algorithm to that of the *shortest weighted path algorithm*

Shortest Weighted Path Scheme

- Log (p(x1)…p(xn)) = log(p(x1))+… +log(p(xn))
- Each edge of length x is assigned weight –log(x)
- Run shortest weighted path algorithm with such weights
- Finds route with maximal probability of delivery

Probability dilations

Probability of Successful Delivery

Algorithm	Number of Nodes : 250							
	Density (with 1.4377R Neighbors)							
	6	8	10	20	24	32	40	80
Shortest Path	1	1	1	1	1	1	1	1
EER (NC)	0.082	0.123	0.213	0.470	0.559	0.741	0.829	0.949
I nEPR	0.189	0.202	0.183	0.467	0.552	0.721	0.806	0.939
I Proj Progress	0.186	0.200	0.205	0.474	0.567	0.722	0.805	0.938
nEPR	0.010	0.019	0.024	0.059	0.077	0.114	0.151	0.267
Proj Progress	0.016	0.020	0.030	0.061	0.078	0.119	0.155	0.267
1.4377R Greedy*	1.372	1.381	1.439	26.51	39.5	114.9	351.1	5618.2
1.25R Greedy**	3.155	81.23	86.55	199.8	315	621	951	4329
R Greedy	0	0.023	0.074	0.0438	0.044	0.051	0.062	0.103

* All numbers to multiplied by E-7 ** All numbers to be multiplied by E-6

Success Rates

Algorithm	Number of Nodes : 250							
	Density (with 1.4377 Neighbors)							
	6	8	10	20	24	32	40	80
Shortest Path	100%	100%	100%	100%	100%	100%	100%	100%
EER (NC)	35%	49.6%	73.6%	97.6%	99.6%	100%	100%	100%
I nEPR	38%	52.8%	76.4%	98.4%	100%	100%	100%	100%
I Proj progress	38%	52.2%	76%	98.4%	100%	100%	100%	100%
nEPR	38%	54%	76%	99.2%	100%	100%	100%	100%
Proj Progress	40%	52.4%	73.6%	99.2%	100%	100%	100%	100%
1.4377R Greedy	46%	68%	85.2%	99.6%	100%	100%	100%	100%
1.25R Greedy	12%	29.2%	54%	97.6%	98.4%	100%	100%	100%
R Greedy	0%	1.2%	8.4%	79.6%	87.2%	97.2%	100%	100%

Performance summary

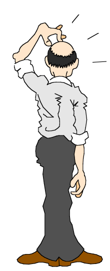
- Good performance for localized algorithms
- NC, InEPR, I Projection Progress have high probability dilations for dense networks and 100% success rates
- nEPR and Projection Progress have significantly worse data
- tR-greedy are significantly inferior to all above methods and remain unsuccessful for dense networks: a choice of ‘long’ edge is quite likely on a route which then contributes to very low probability of delivery

Routing without ack’s: summary

- Find a route that maximizes the probability of receiving a packet
- Theoretically, maximum 1 achieved for infinite number of tiny hops toward destination
- Practically, send to the nearest neighbor closer to destination, or better apply *expected progress routing*:
- Send to neighbor *A* that maximizes $p(SA)(|DS|-|DA|)$
- Iterative expected progress routing:
- Improve probability via common neighbors

•Kuruvila, Nayak, Stojmenovic 2004

Future work



- Other coding schemes?
- Variable packet length ?
- Variable bit rates ?
- Other physical layer models ?
- Adjusted transmission powers ?
- Ack’s via forwarding ?