ALBA: a cross-layer integrated protocol stack for medium-large scale wireless sensor networks

Michele Nati, Chiara Petrioli[†]

[†] *Department of Computer Science – University of Rome "Sapienza" – Italy*

Authors: P. Casari, M. Nati, C. Petrioli, M. Zorzi

This work was partially supported by the European 6th Framework Programme through the Integrated Project (IP) ***e-SENSE****, contract number 027227.*

Outline

- Geographic routing concepts
- Handling dead ends: Related work
- Adaptive Load-Balancing Algorithm (ALBA)
- Rainbow
  - A node-coloring algorithm to route around dead ends
- Simulations settings
- Results for high and low nodal densities
- Impact of localization errors
- Conclusions and discussion

The geographic routing paradigm

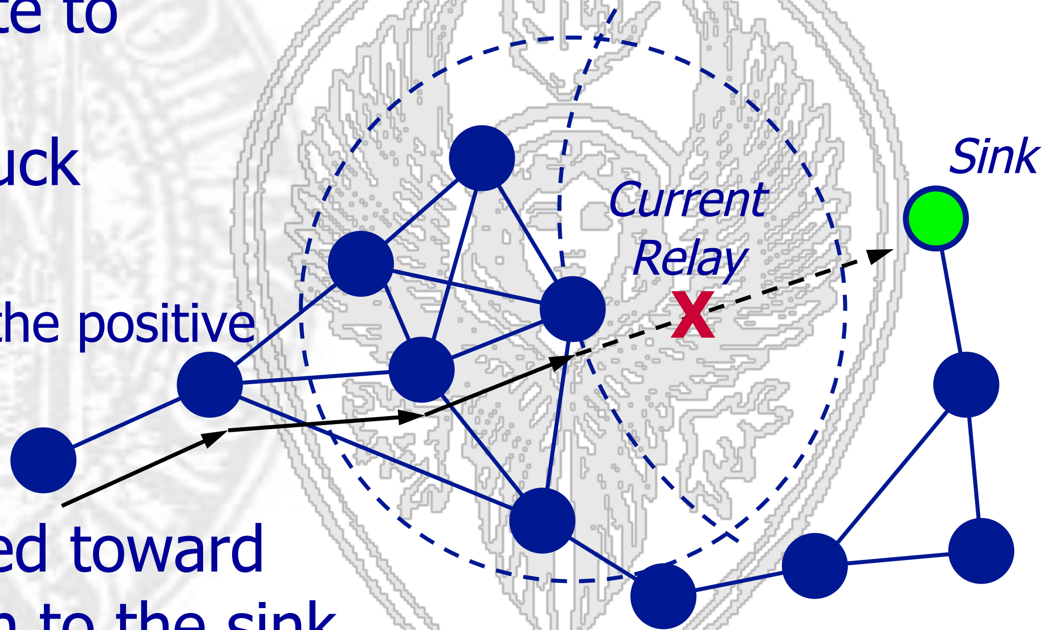
Geographic routing

"Forward the packet to a node that offers geographic advancement toward the destination"

- Pros
  - Virtually *stateless* (needs only knowledge of the source's and the destination's locations)
- Cons
  - Requires positioning estimation (BUT is it really critical?)
  - Requires mechanisms to route packets out of *dead ends*
    - ✓ The present relay is the closest to, yet not a neighbor of, the destination

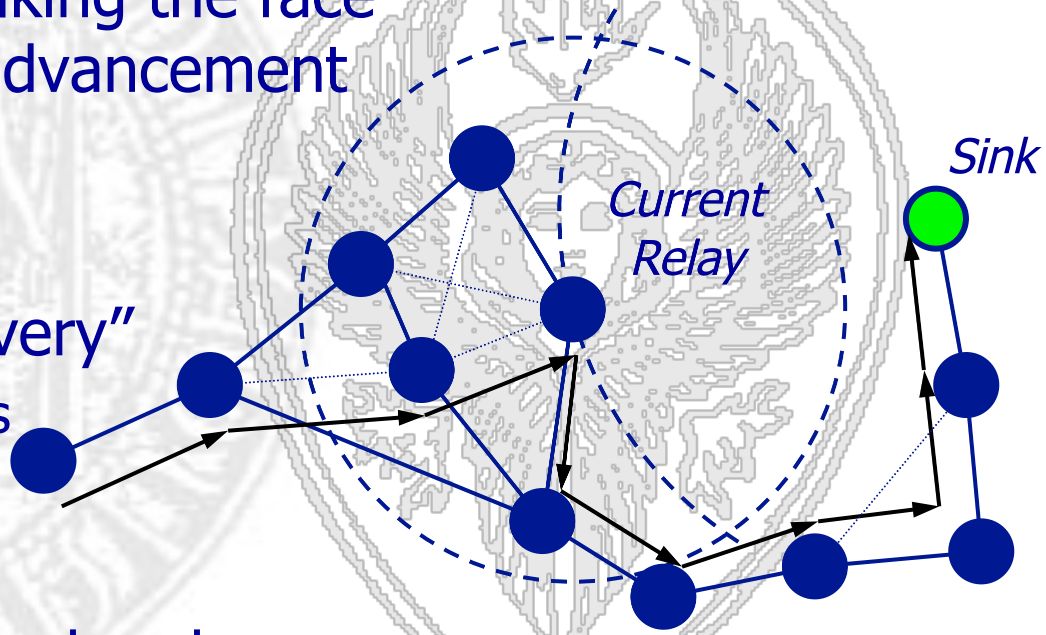
The dead end problem

- If the routing algorithm is tuned to achieve a positive advancement at each step, dead ends may occur
- In this example, a route to the sink is available but the packets get stuck at the current relay
  - There are no nodes in the positive advancement area
- Packet losses occur if data are not re-routed toward nodes that have a path to the sink



The dead end problem, 2

- Current approaches to dead end resolution include **planarizing** the network graph (the resulting graph has no cross links) and walking the face perimeters when the advancement area is empty

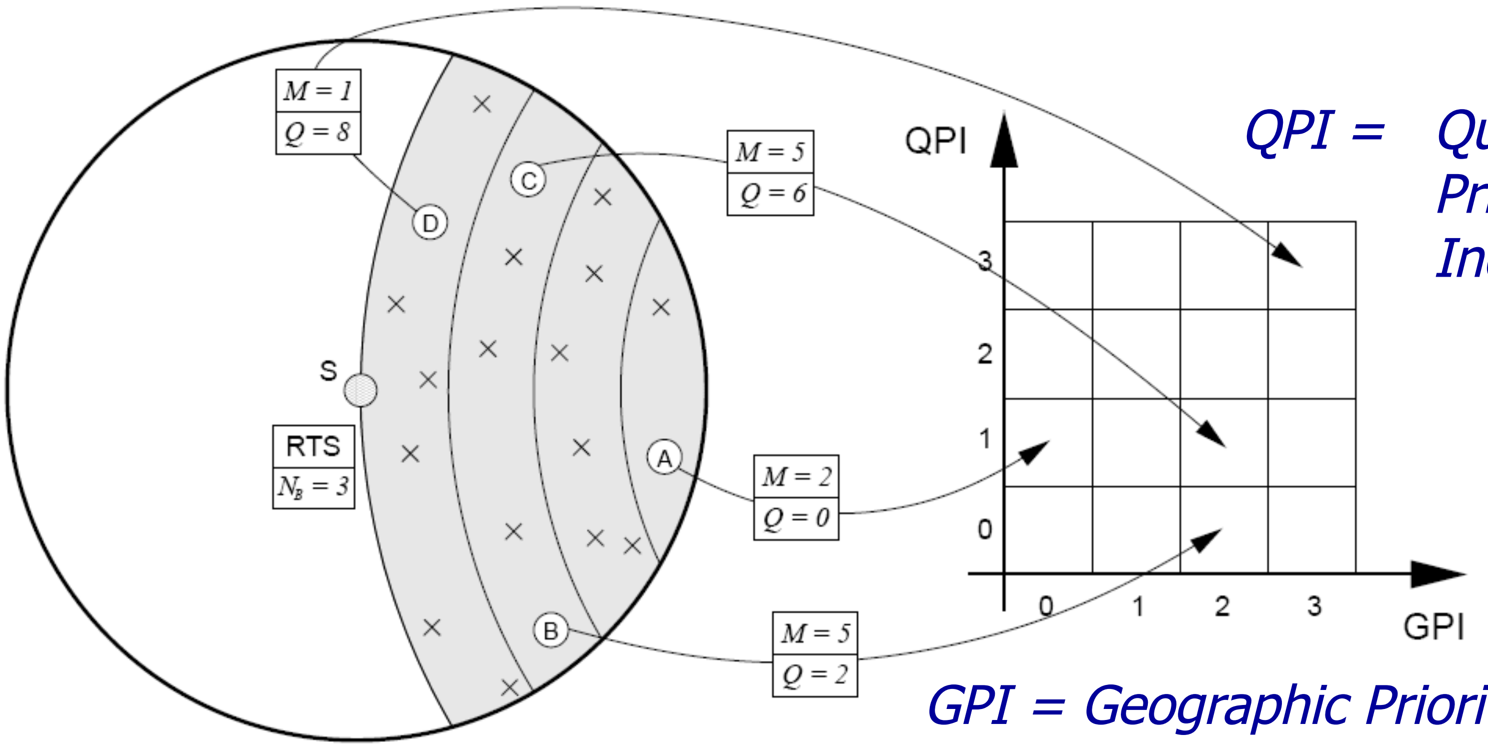


- **Pros**: "Guarantee delivery"
  - Planarization algorithms can be distributed
- **Cons**: planarization overhead, prone to location and channel errors

Our Approach: Basics

- ALBA → Adaptive Load-Balancing Algorithm
  - Integrates interest dissemination and converge-casting
  - Cross layer optimized converge-casting
    - ✓ MAC
    - ✓ Geographic Routing
    - ✓ Mechanisms to load balance traffic among nodes (to decrease the data funneling effect)
    - ✓ Schemes to distributely and efficiently deal with dead ends
- Operations:
  - Nodes forward packets in bursts (up to M_B packets sent back-to-back)
    - ✓ The length of the burst is adapted
  - Forwarders are elected based on
    - ✓ The ability to receive and correctly forward packets
      - The used metric involves the queue level, the past transmission history of the relay, and the number of packets the sender needs to transmit
    - ✓ The geographic proximity to the destination

Our Approach: Basics of the ALBA Protocol



QPI = Queue Priority Index

GPI = Geographic Priority Index

$$QPI = \lceil (Q + N_B)/M \rceil - 1$$

Queue level (Q)

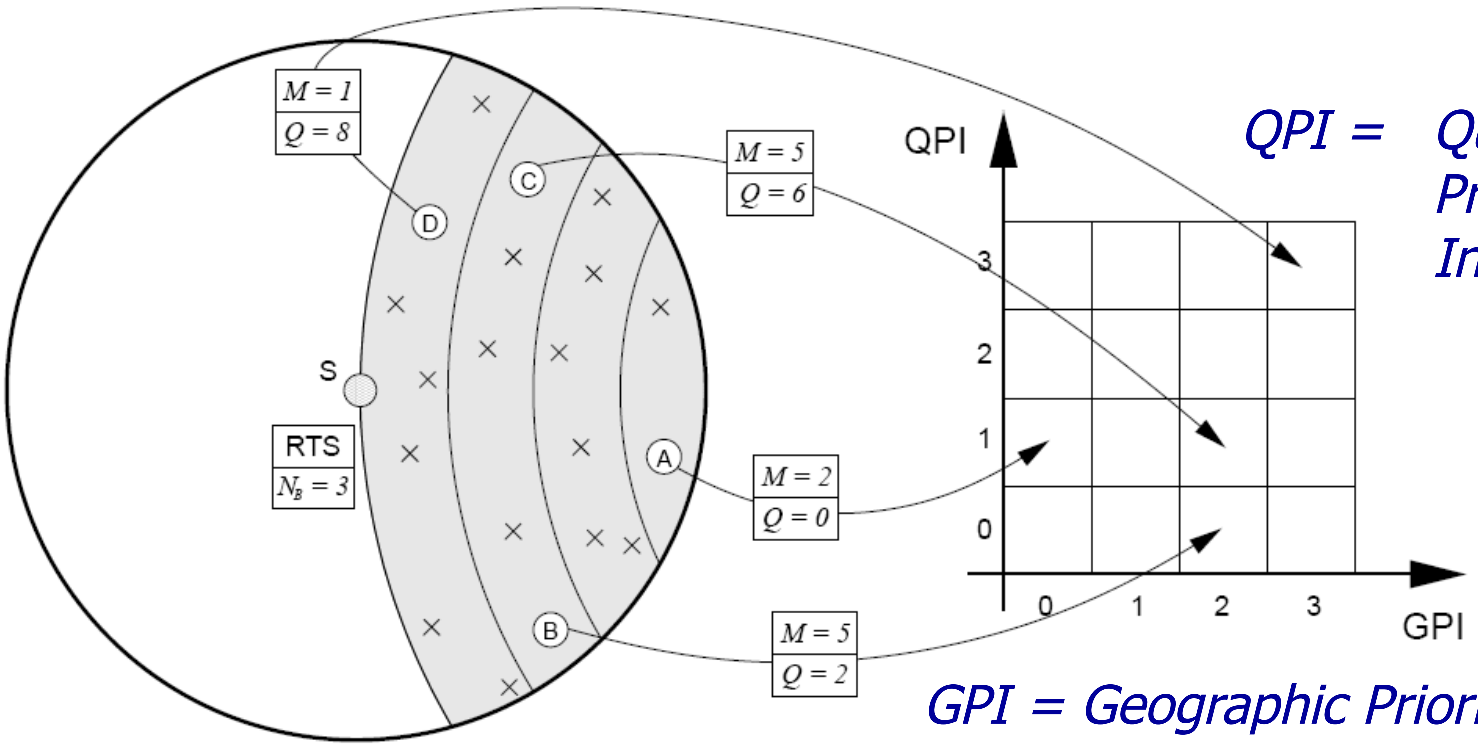
Requested length of the burst (N_B)

Average length of a burst the relay expects to transmit correctly (M)

ALBA Features

- The metric used for the choice of the relay ensures load balancing as it preferably chooses relays with
  - Low queue, especially if N_B is high
  - Good forwarding history (through *M*)
- Nodes employ duty-cycling to enforce energy saving
- The relay selection works in phases
  - Phase 1: Selection of the best QPI
    - ✓ Attempt 1 search for QPI=0, Attempt 2 for QPI=0,1, and so on
    - ✓ **Awaking nodes can participate in this selection phase**
  - Phase 2: Selection of the best GPI
    - ✓ Performed if more than one node with the same QPI was found
    - ✓ Awaking nodes cannot participate here (to speed up completion)
- **Still prone to dead ends**

ALBA: An example

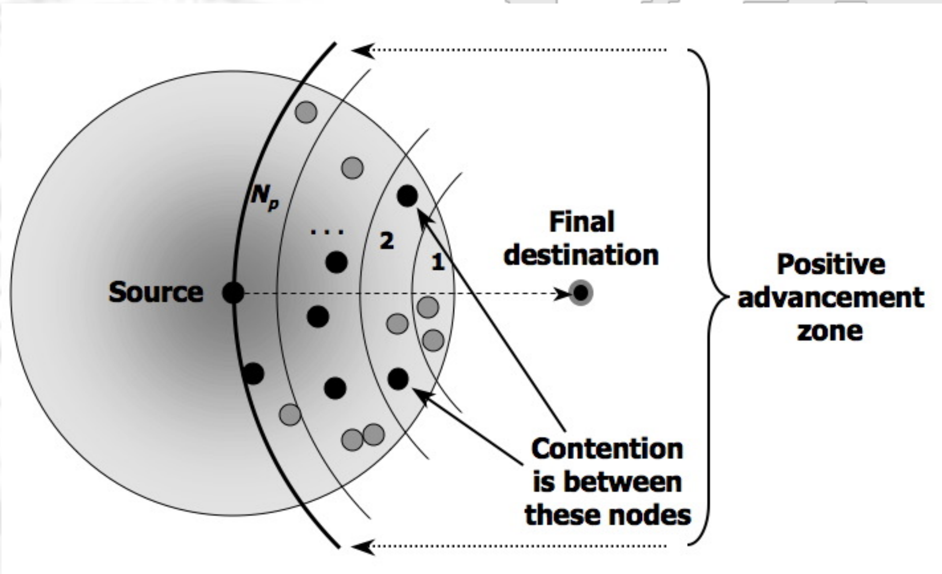


QPI = Queue Priority Index

GPI = Geographic Priority Index

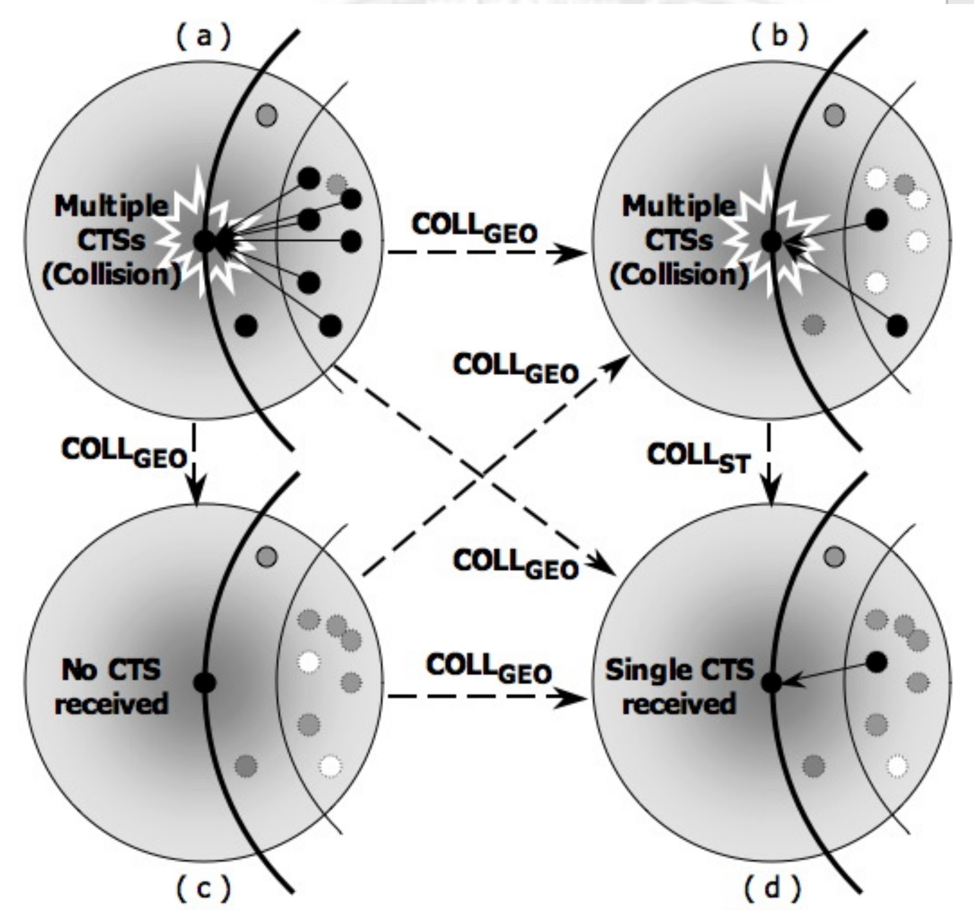
1. Node A is nearer to the sink (GPI =1) but has a low QPI (M=2); node B, is farther but has greater reliability (M) and comparable queue occupancy (Q); B has a greater QPI than A
2. In case of node B is sleeping at transmission time, node A is selected for its better GPI

Contention Mechanism



- Source nodes send a RTS msg to query relays. Relays respond with CTSs
- No response: a CONTINUE msg pings the following region
- Collision: a COLLISION msg starts the collision resolution algorithm
- CTS received: Burst of DATA transmission starts

Collision Resolution Algorithm



- Upon receiving a $COLL_{GEO}$ msg, nodes reply based on their GPI
- Upon receiving a $COLL_{ST}$ msg, nodes persist in sending CTSs with probability 0.5
- Should they all decide to stay silent, the following $COLL_{ST}$ msg enables a further decision
- Eventually, the process ends with a single valid relay being selected

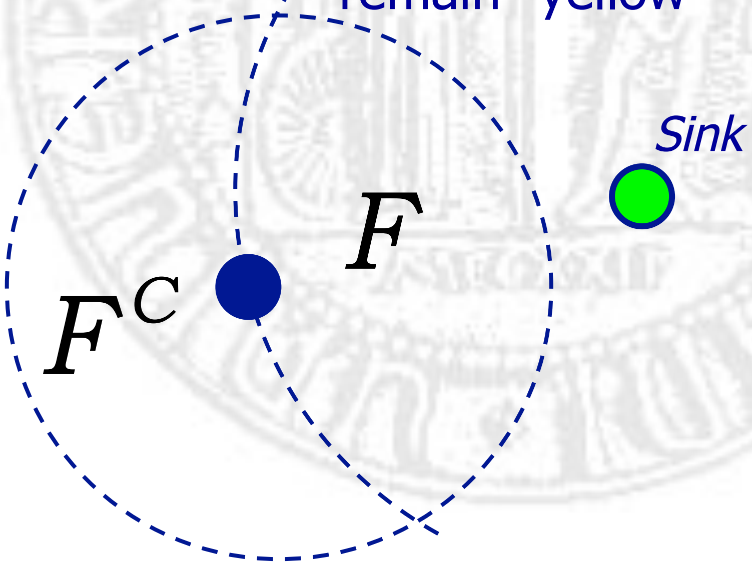
Rainbow

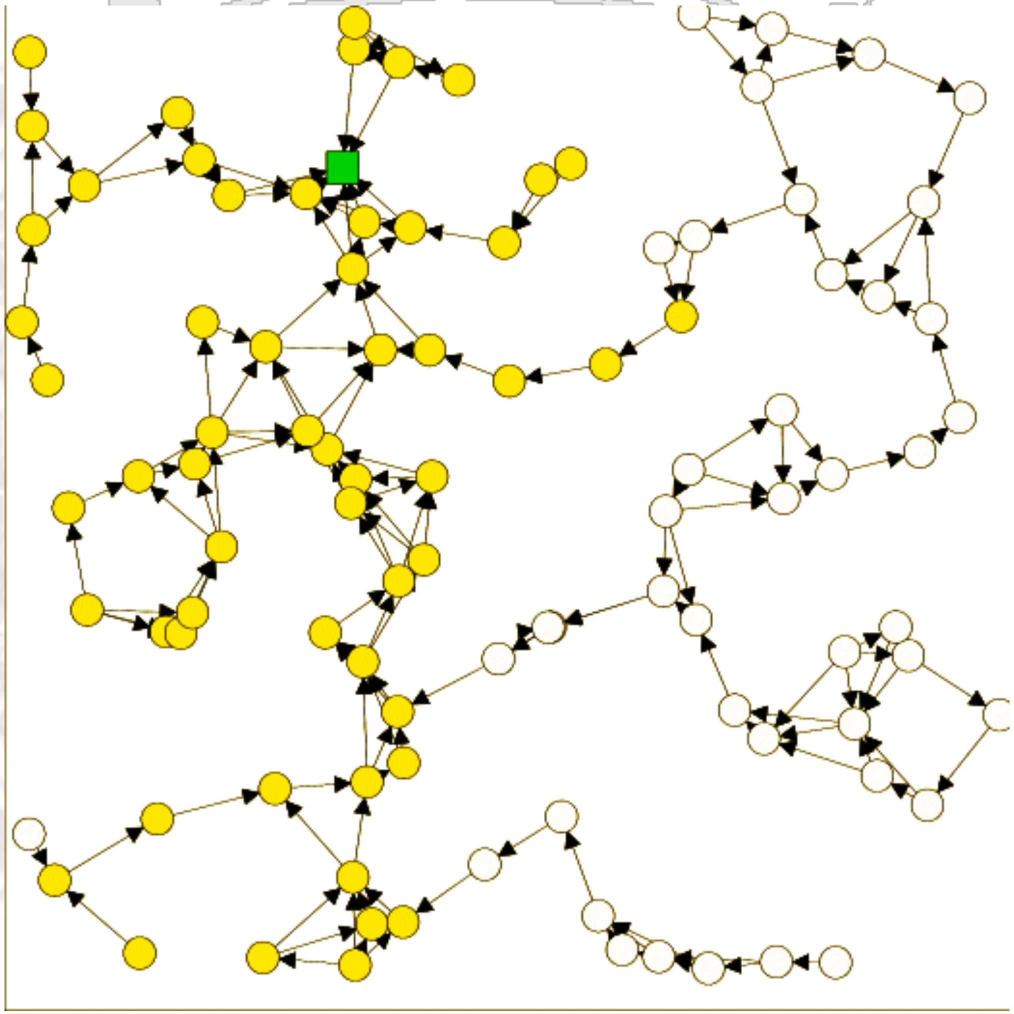
A node coloring algorithm for routing out of dead ends and around connectivity holes

- Concepts
  - In low density topologies, a method for routing around dead ends is needed
  - Nodes that recognize themselves as dead ends progressively stop volunteering as relays
  - To route traffic out of the dead end, they begin to transmit packets backward, in the negative advancement zone
  - Hopefully, a relay that has a greedy forwarding path to the sink can eventually be found
  - A recursive coloring procedure is used

Rainbow node coloring scheme – Yellow nodes

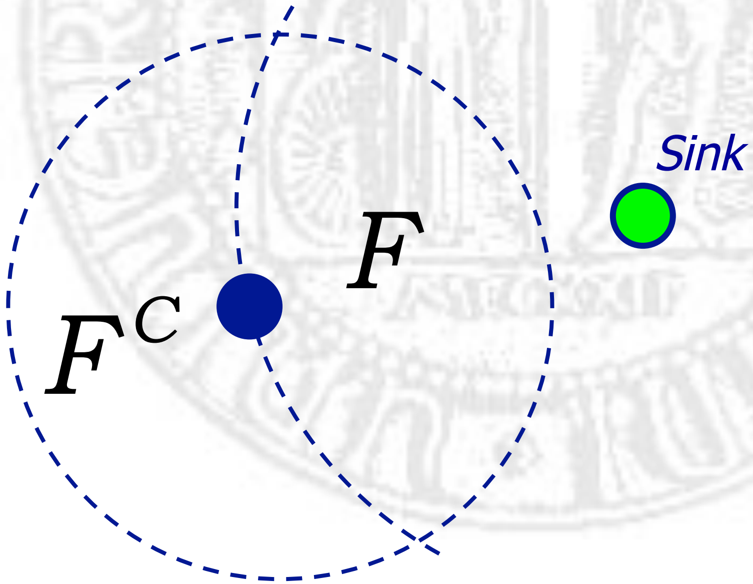
- *F* and *F* ^{C} are the positive and negative advancement areas, respectively
- Initially, all nodes are "yellow"
- All nodes that exhibit a greedy path to the sink remain "yellow"

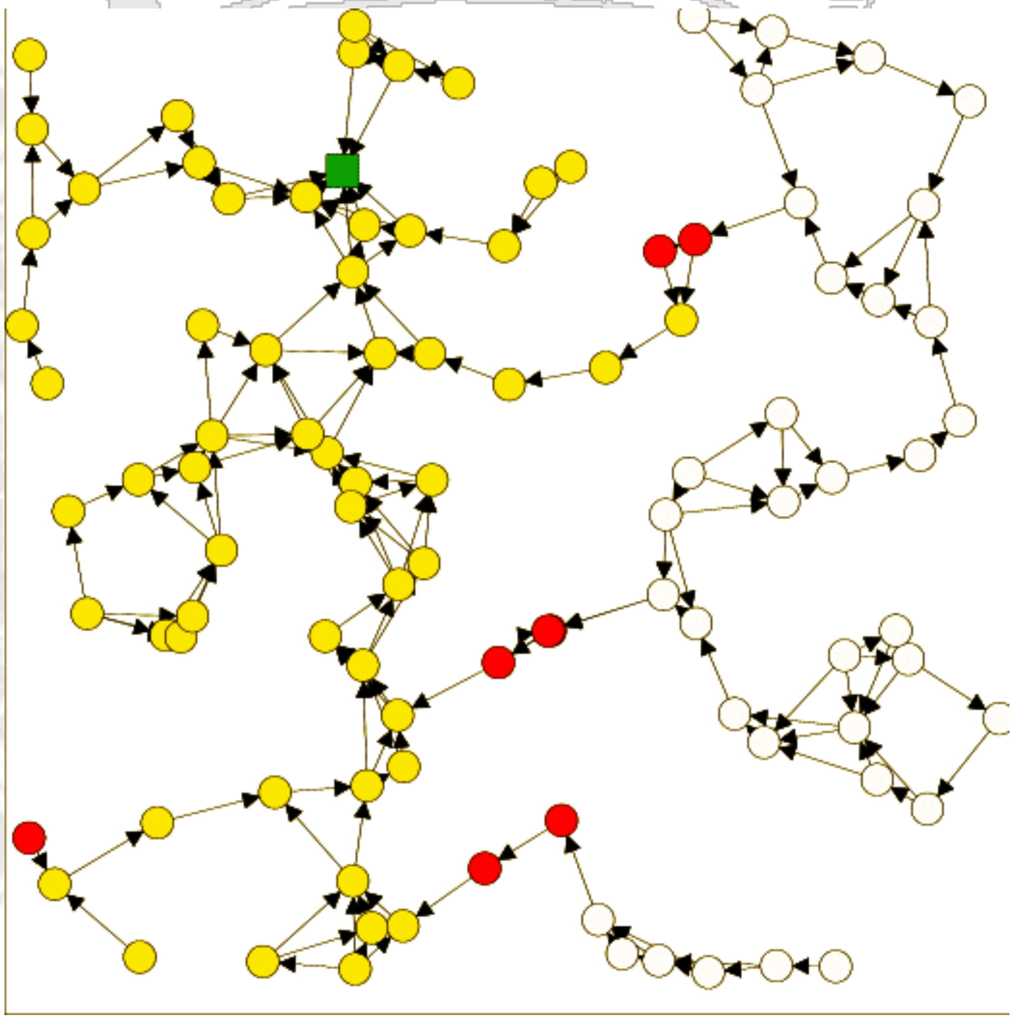




Rainbow Node Coloring Scheme: Red nodes

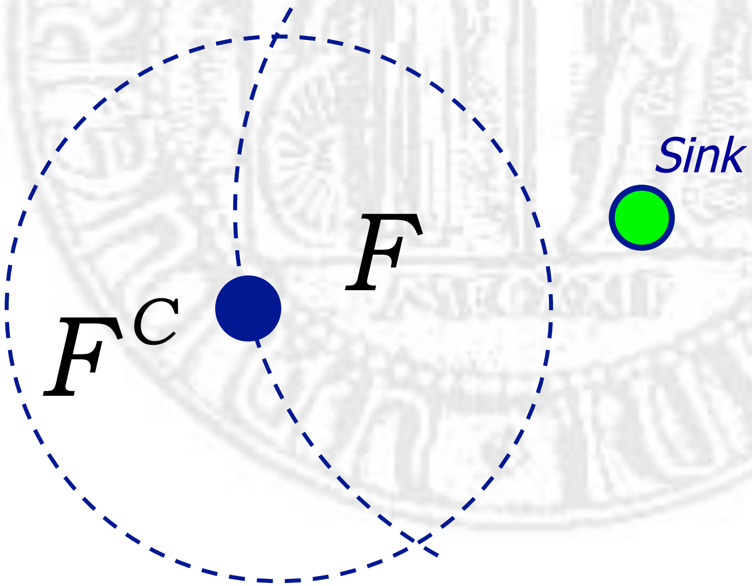
- If a yellow node cannot forward packets further, it switches to “red”
- From now, it looks for either “red” or “yellow” relays in F^C

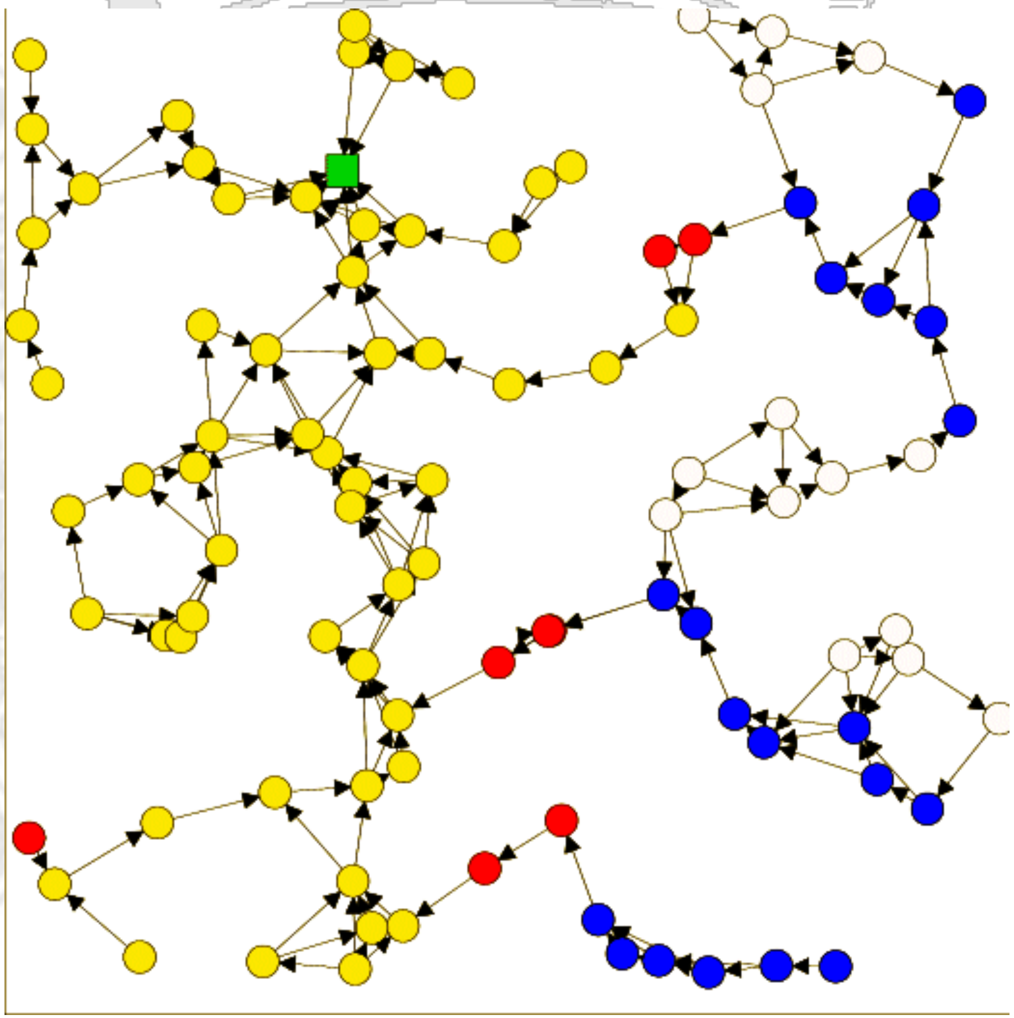




Rainbow Node Coloring Scheme: Blue Nodes

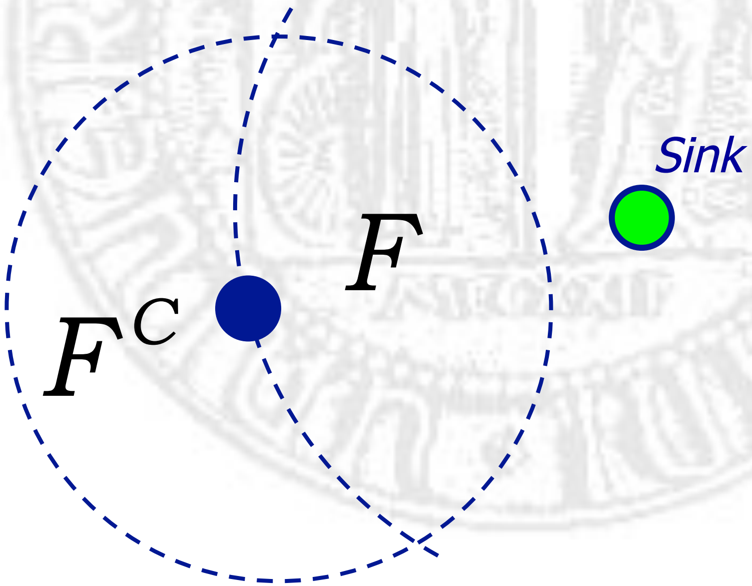
- If red nodes cannot advance packets, they turn to “blue”
- Again, they switch to look for relays in *F*
- They only look for “red” or “blue relays

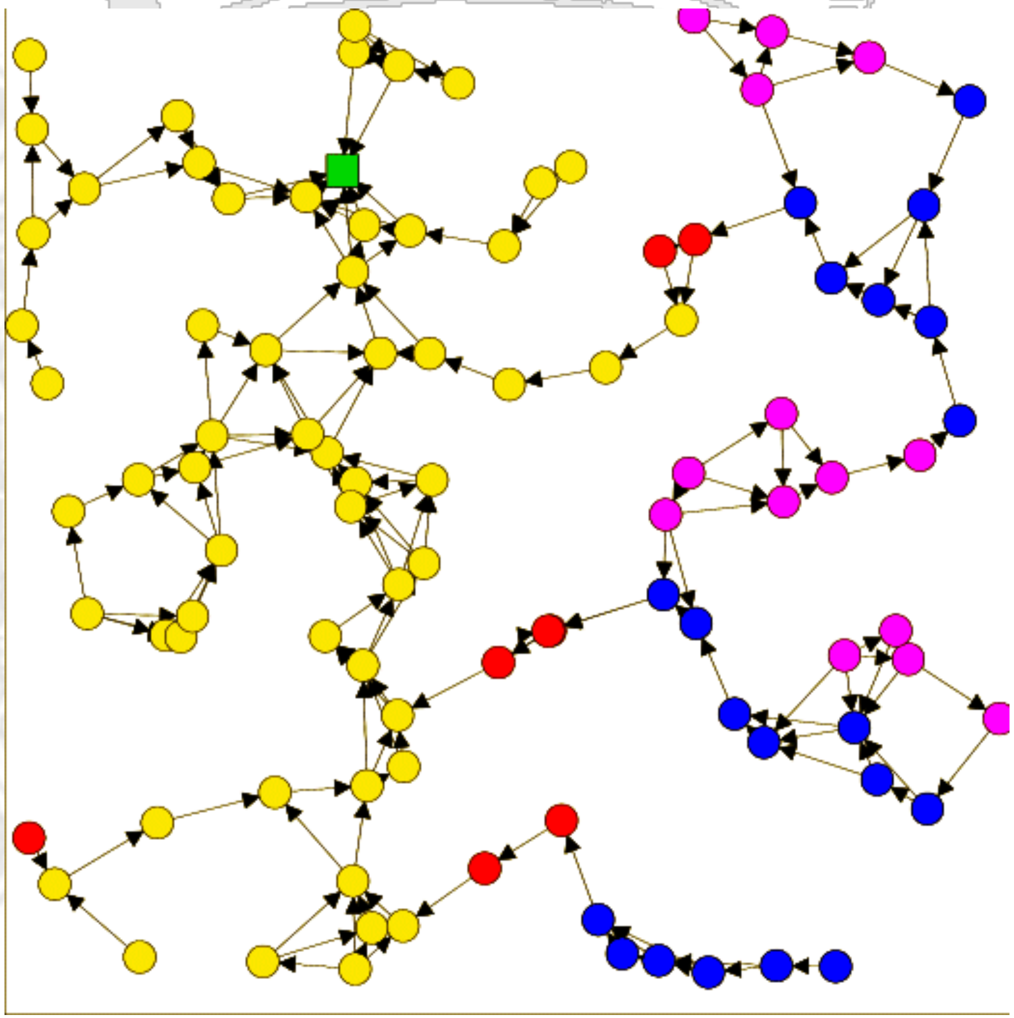




Rainbow Node Coloring Scheme: Violet nodes

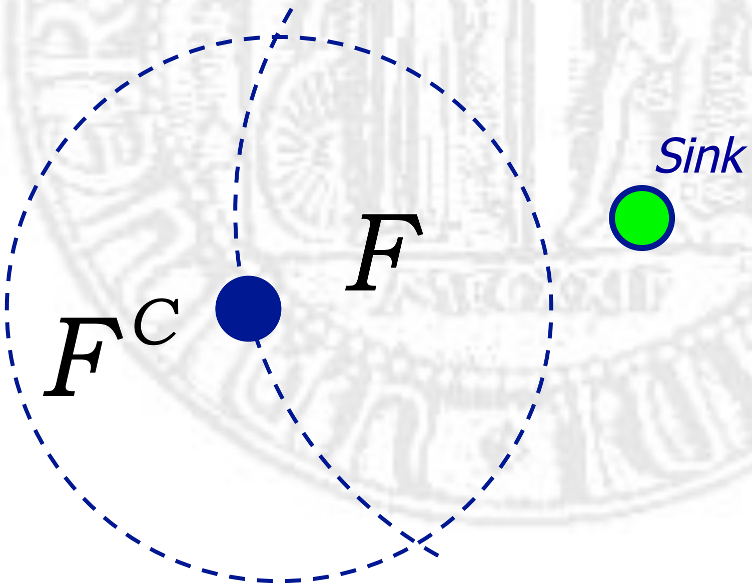
- If blue nodes still have problems finding relays they switch color again, to “violet”
- Like red nodes, they look for relays in F^C…
- …but only “blue” or “violet”





Rainbow Node Coloring Scheme: In general

- The number h of needed colors is fully general
  - The greater the number of colors, the more nodes can be connected to the converge-casting tree
- In general, given *h* labels $C_1, C_2, \ldots, C_h$ …
- The nodes switch from a label to the following one every time they perceive to be a dead end with their present label



- Nodes labeled C_1 are the only one with a greedy path to the sink
- Nodes with odd labels ($C_1, C_3, \ldots$) always look for relays in *F*
- Nodes with even labels ($C_2, C_4, \ldots$) always look for relays in F^C
- A node with label C_k always looks for C_k- or C_{k-1}-nodes, except C_1-nodes that always look for other C_1-nodes

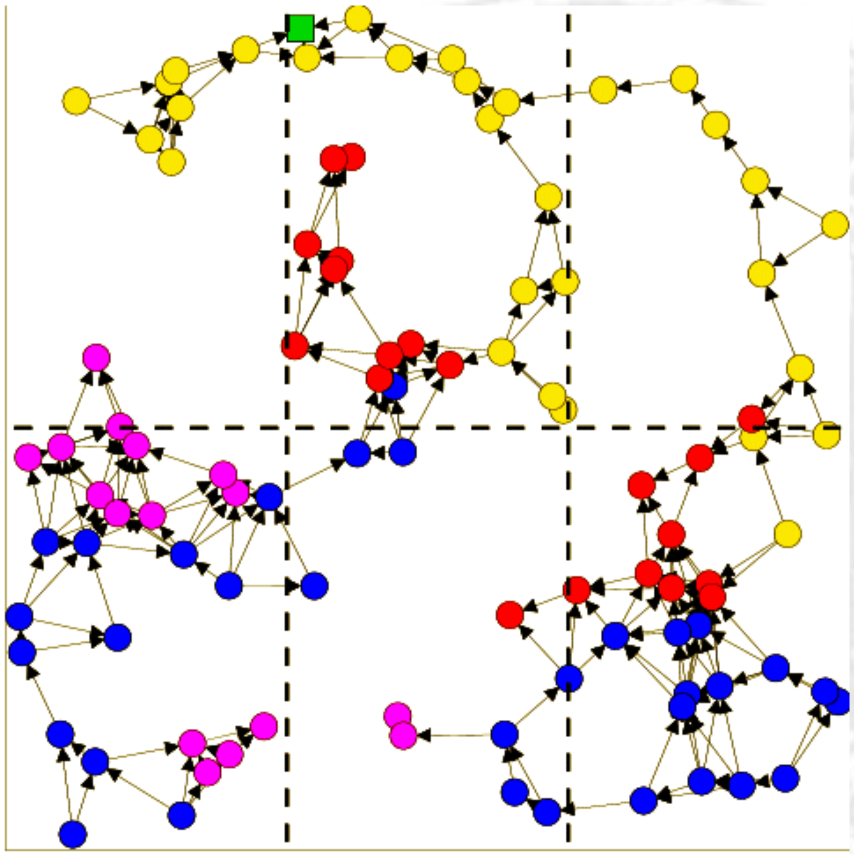
Rainbow: Wrap up

- **Concepts**
  - Nodes progressively realize to be dead end and automatically adapt to this condition
    - ✓ No abrupt changes in the color of a node (relays might be present but just *unavailable* for the moment)
  - More colors mean more nodes can successfully deliver packets
- **Pros**
  - Effectively routes around dead ends
  - Completely blind and distributed
  - Does not require planarization
  - The load-balancing features of ALBA are seamlessly used throughout
- **Cons**
  - The network requires some training for nodes to achieve the correct color

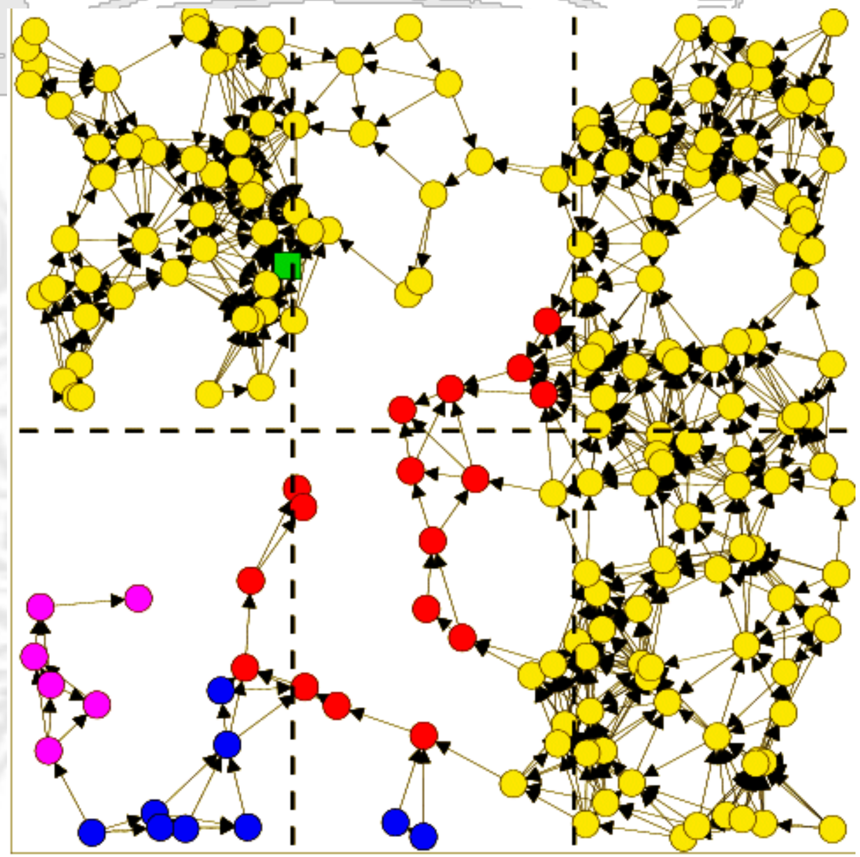
Results: Simulation Setting

- Simulation area: 320 m x 320 m
  - Random and uniform deployment
  - Non-uniform deployment
    - ✓ A more general case than uniform deployment
    - ✓ The area is divided in 3 high-density and 3 low-density zones
    - ✓ 75% of the nodes are randomly placed in high-density zones, the remaining 25% in low-density zones
- First set of results → Comparison
  - ALBA-R vs. GeRaF and MACRO
- Second set of results → High node densities
  - Show that Rainbow does not decrease performance if not used
  - N = 300, 600, 800, 1000 nodes
- Third set of results → Low node densities and different number of colors used in Rainbow
  - Used to show the effectiveness of Rainbow in rerouting packets

Sample non-Uniform Deployments



100 nodes



200 nodes

Results: Other Simulation Parameters

- Coverage range *r* = 40 m
- First-order energy model → $E_{TX}(r) = E_{TX_e} + E_{TX_a}(r)$

$$E_{TX_a}(r) = \varepsilon_a \cdot r^2$$

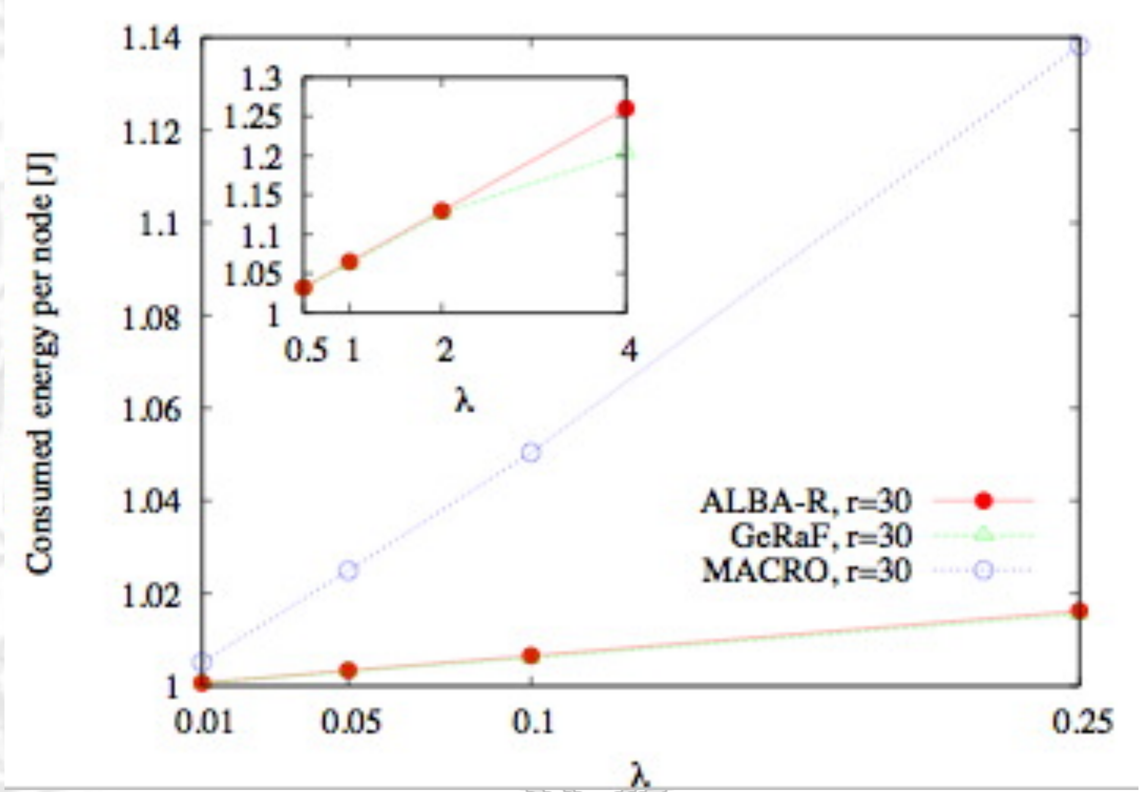
Energy for transmit circuitry

Energy to feed the transmit amplifier to cover a range *r*

- Duty cycle: 0.1
- Data pkt: 250 Bytes
- Signaling pkt: 25 Bytes
- Channel rate: 38.4 kbps
- Node queue: 20 packets
- Different values of the packet generation rate per node, λ
- ALBA–R parameters
  - Number of QPIs and GPIs: 4
  - Maximum length of a packet burst: 5

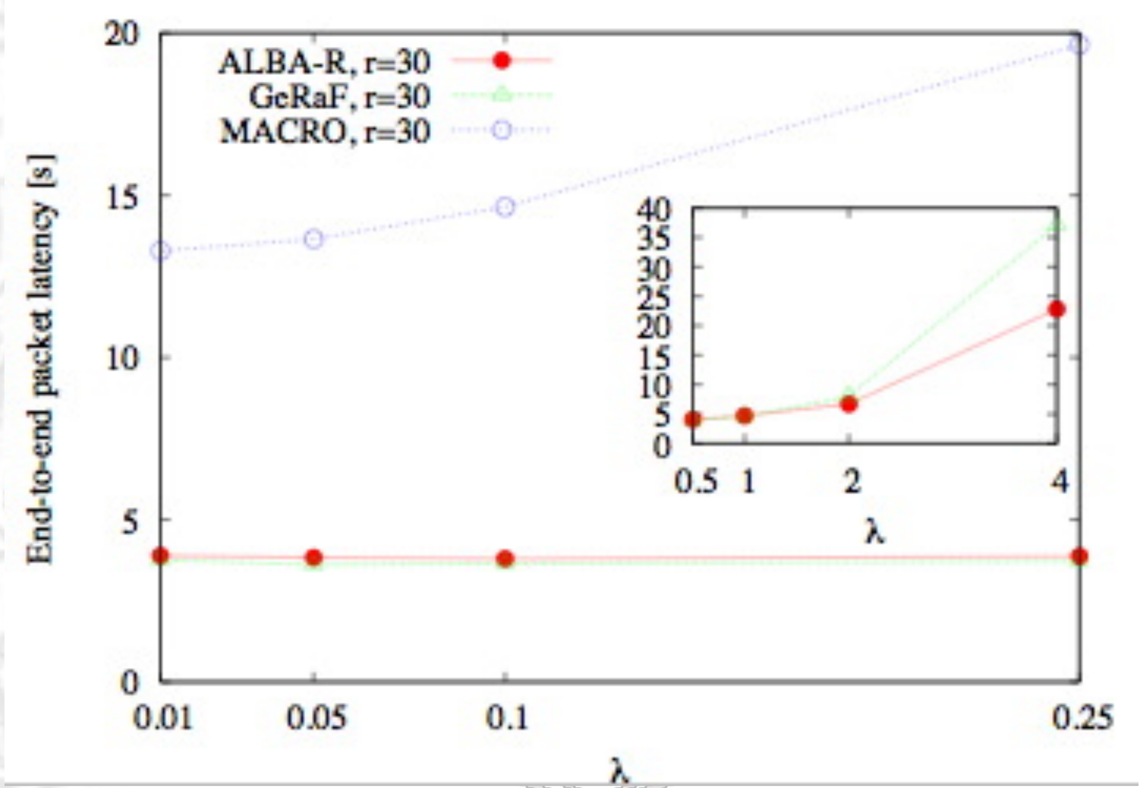
Results: ALBA vs MACRO and GeRaF. Energy consumption

- n = 600 nodes
- MACRO's energy consumption increases steeply due to expensive relay wakeup procedures
- In the considered traffic scenarios, ALBA-R continues to deliver the packets correctly
  - Energy increases accordingly
- GeRaF begins to suffer from excess backoffs at $\lambda = 2$ (energy decreases)



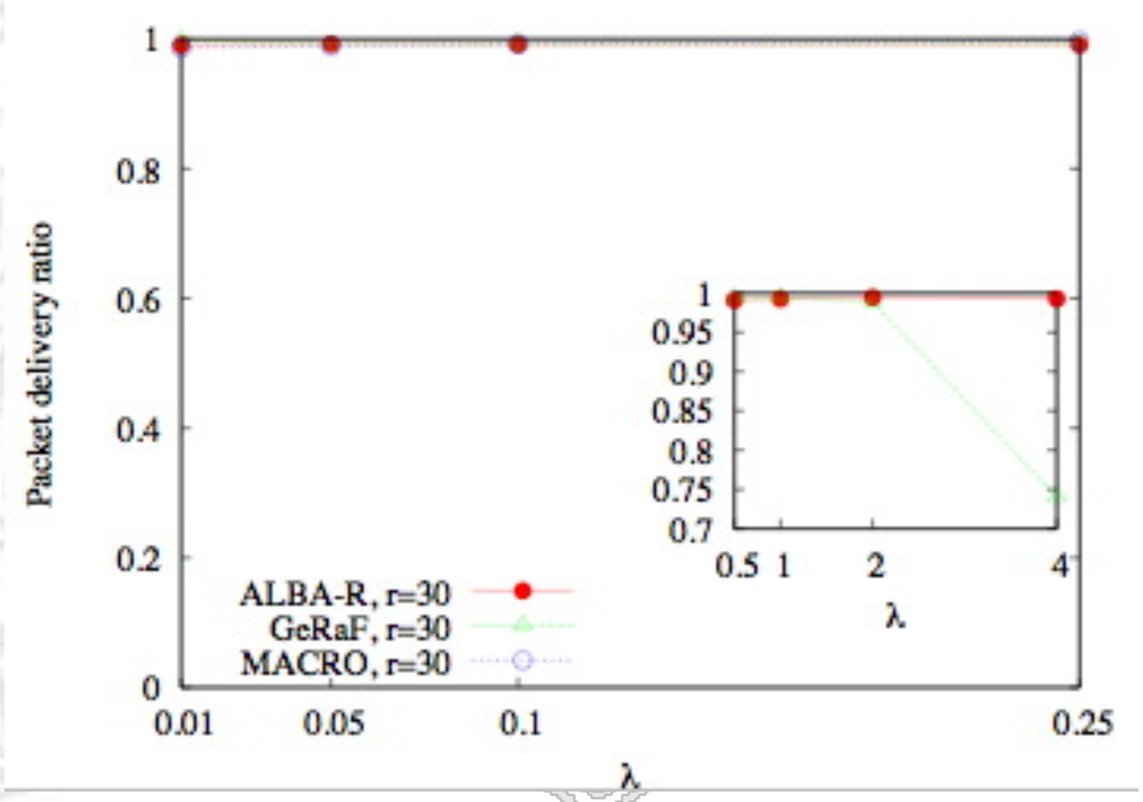
Results: ALBA vs MACRO and GeRaF. Average end-to-end delay

- n = 600 nodes
- ALBA-R scales better than the other schemes
- MACRO suffers from severe performance degradation due to overwhelming handshaking requirements
- ALBA-R works better thanks to load balancing, and back-to-back transmissions



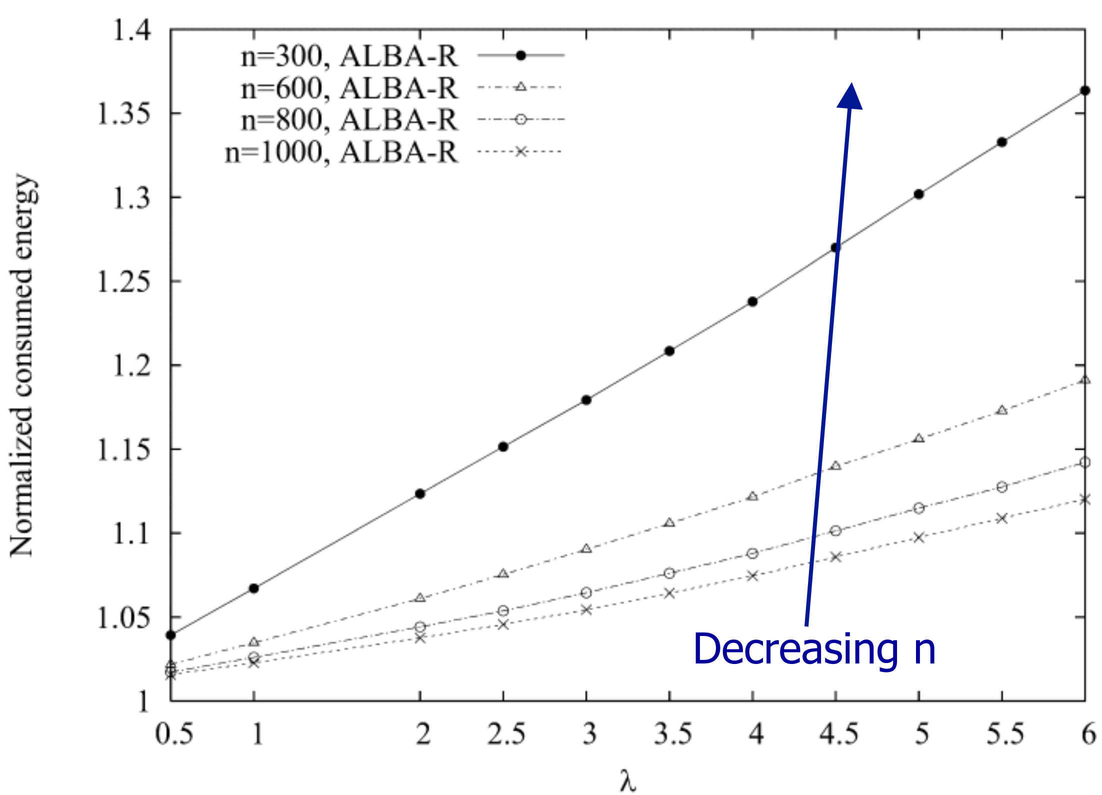
Results: ALBA vs MACRO and GeRaF. Delivery ratio

- n = 600 nodes
- Shows a similar trend with respect to latency
- MACRO does not scale beyond λ = 0.5
  - 57% deliveries for λ = 0.4 with 137s average latency
- ALBA-R distributes the traffic more evenly and thus achieves better delivery ratio than both GeRaF and MACRO



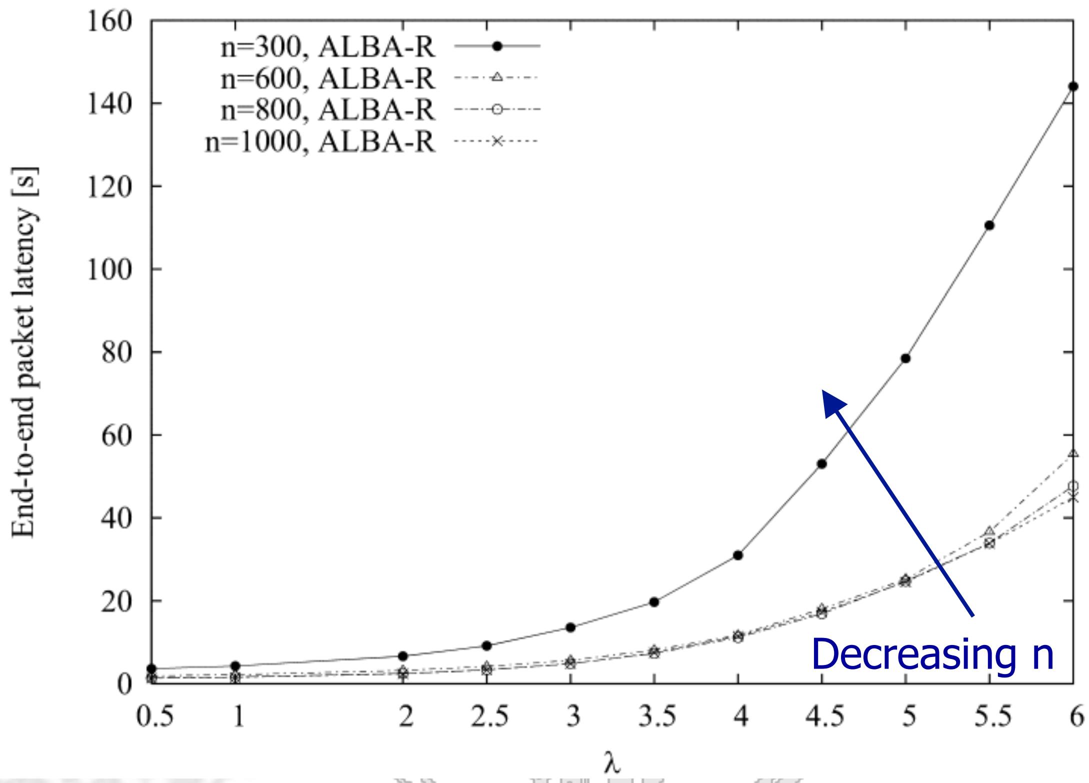
Results: High density. Energy consumption

- Energy normalized to that of a node that follows the duty cycle
- *h* = 4 colors here
- Due to high density, rainbow is almost never applied
- The energy consumption increases for decreasing number of nodes because it becomes harder to find relays
- The increase over the energy expenditure due to the duty cycle is very small if the node density is sufficiently high



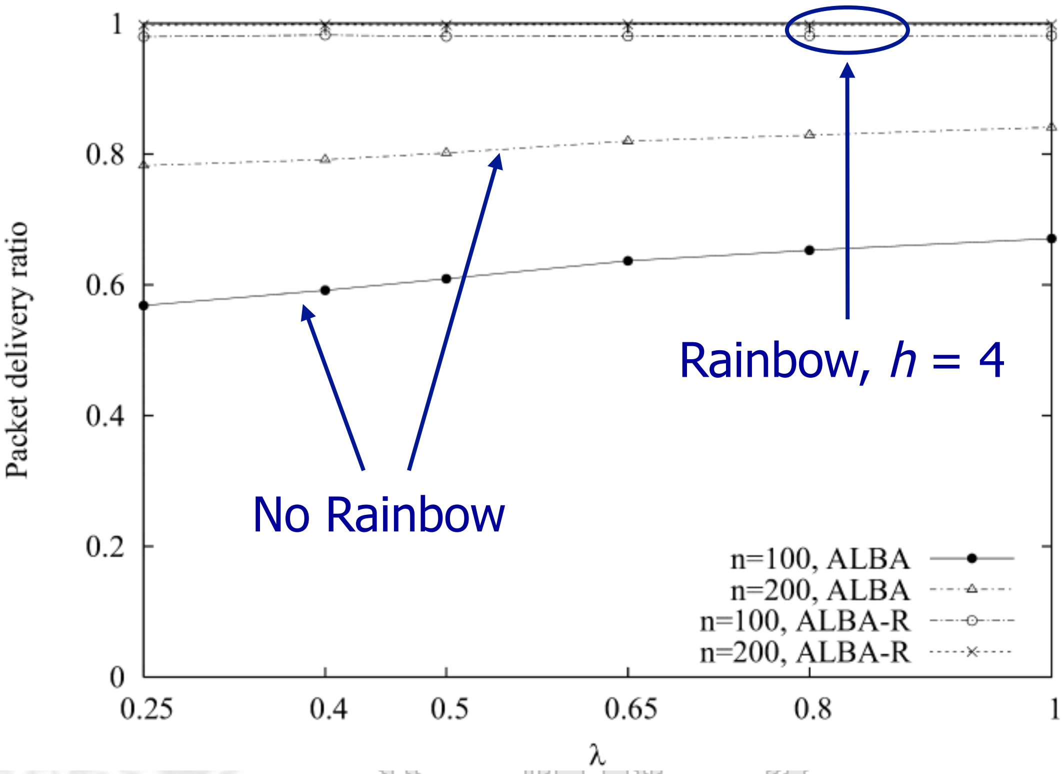
Results: High density. Average end-to-end delay

- Same setting as before
- The latency is quite stable for decreasing number of nodes, until the density becomes critically low at $n = 300$
- Expected behavior at very high traffic, quite stable behavior at low to medium traffic



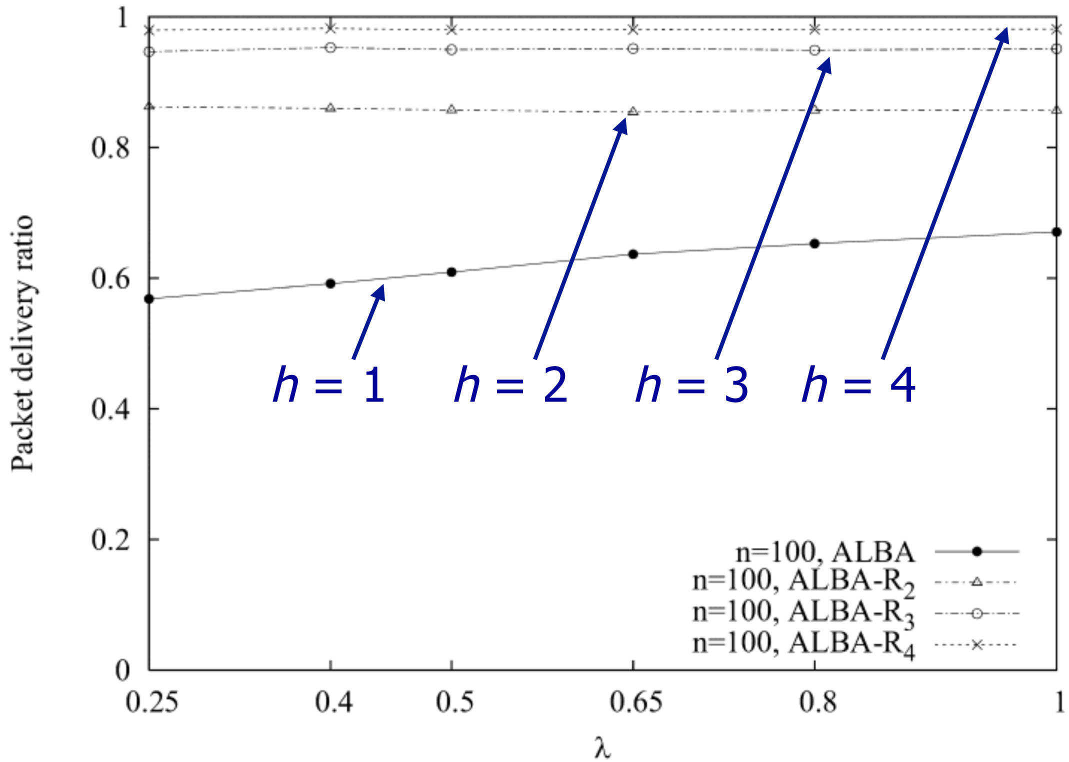
Results: Low density. Delivery ratio

- ALBA vs. ALBA–R
- *h* = 4 colors
- ALBA–R allows almost 100% of the packets to be delivered, with respect to the version without the Rainbow algorithm



Results: Low density. Delivery ratio (varying h)

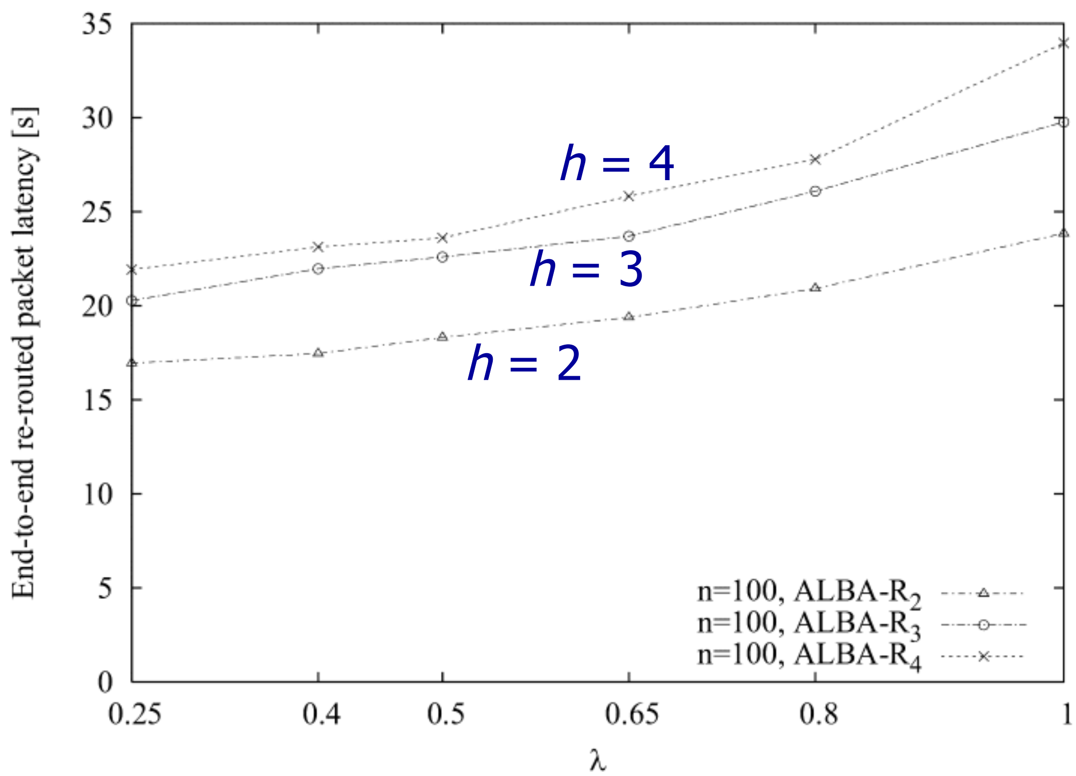
- From: *h* = 1
  to: *h* = 4 colors
- Increasing the number of colors connects more nodes to the converge-casting tree
- The average delivery ratio increases



- Note: after all nodes are connected with the used number of colors, the residual errors are due to packet losses caused by channel impairments or by the difficulty to find relays

Results: Low density. End-to-end delay (varying h)

- End-to-end delay for back-tracked packets only
- From: $h = 2$
  to: $h = 4$ colors
- Higher number of colors also means greater time required to backtrack
- The average end-to-end
  - delay increases
  - longer backtracking routes



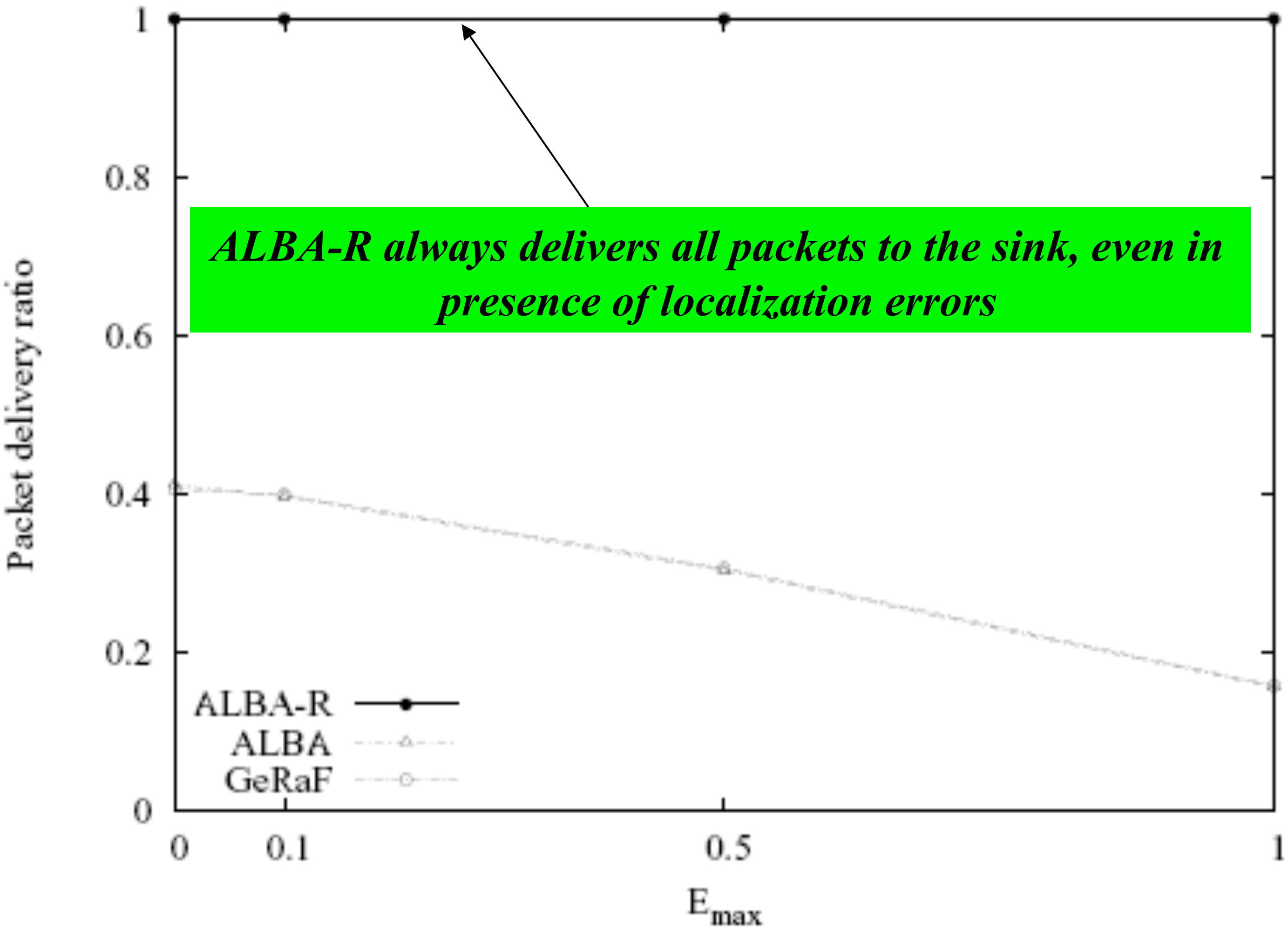
In addition it is possible to show that ALBA is resilient to localization errors (works independetly of localization errors)→No significant performance degradation in case it is integrated with localization errors of the orders of the transmission range

Results: Simulation Setting

- Simulation area: 320 m x 320 m
  - Random and uniform deployment
- 100 nodes scattered on the area → average nodal degree=5 (sparse scenario)
- Traffic
  - Poisson arrivals with λ = {0.25, 0.5, 1.0}
- Transmission range r=40
- Duty-cycle d = 0.1
- Data rate 38400kpbs, EYES nodes energy model
- Presence of localization errors
  - nodes believe to have a position which is randomly selected in a circle centered at the real position and of radius equal to 0.1R,0.5R,R

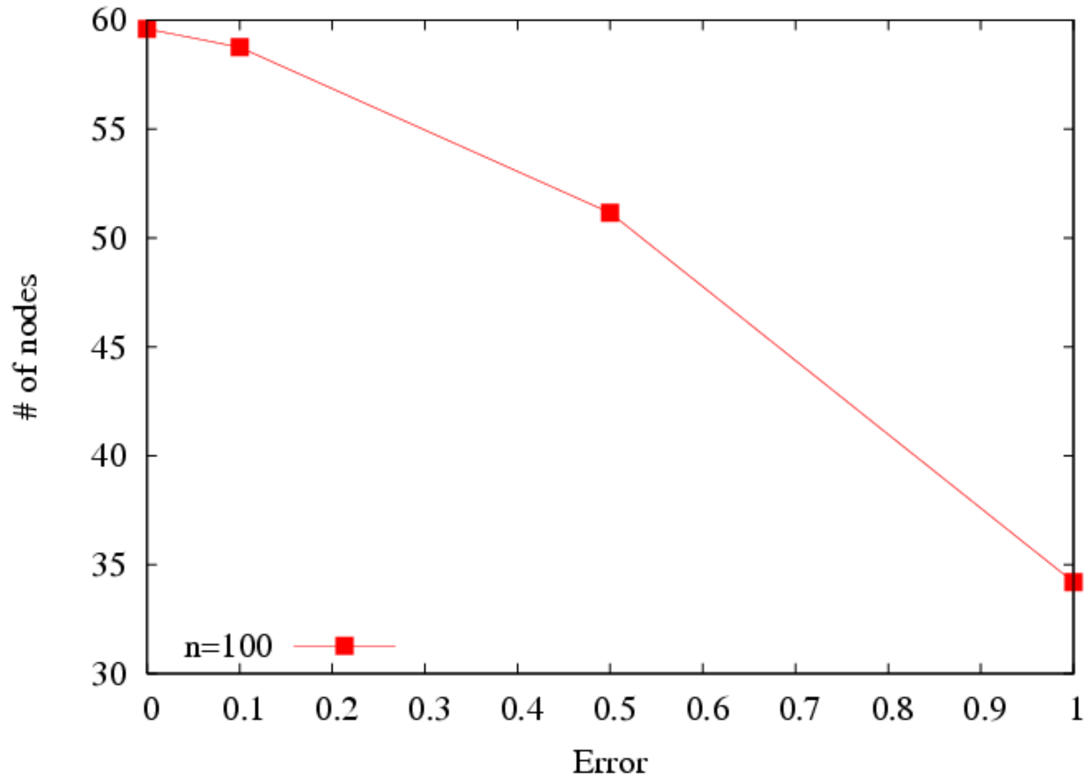
Resilience to localization errors

Packet delivery ratio



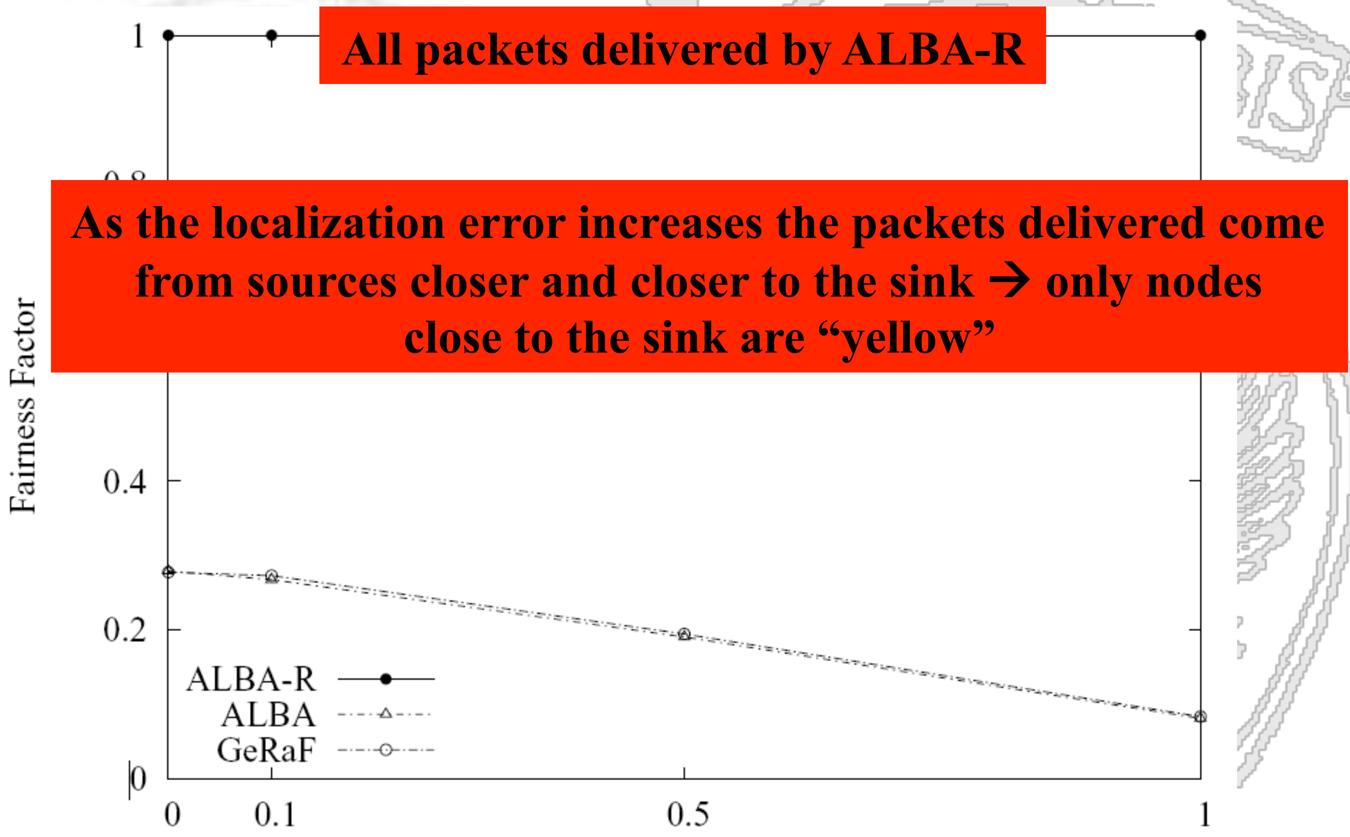
Resilience to localization errors

Number of "yellow nodes"



Decreases with the increase of localization error. This motivates the worst ALBA and GeRaF performance. Being able to reroute packets ALBA-R performance is unaffected in terms of packet delivery ratio.

Resilience to localization errors
Where the "yellow nodes" are located



"Fairness factor"= ratio between the distance from the sink of the sources of the delivered packets AND the average distance from the sink of all sources

Resilience to localization errors

Where the "yellow nodes" are located

MAX ERROR=R

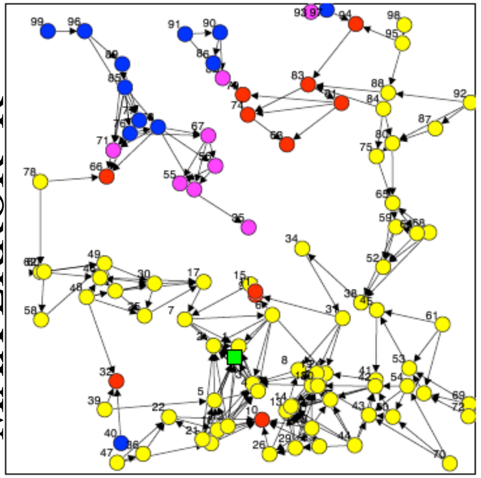


Figure 1: nodes=100 topology=1 localization_error=1.0.

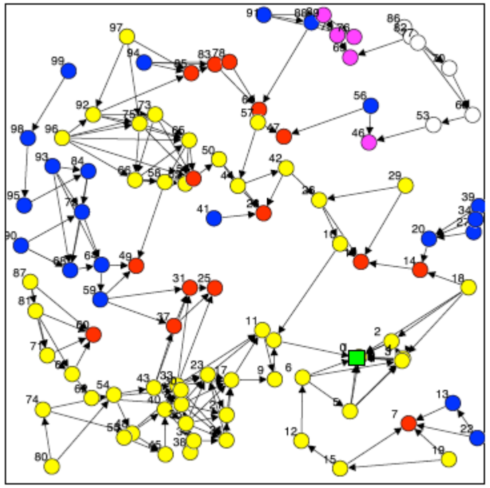


Figure 5: nodes=100 topology=5 localization_error=1.0.

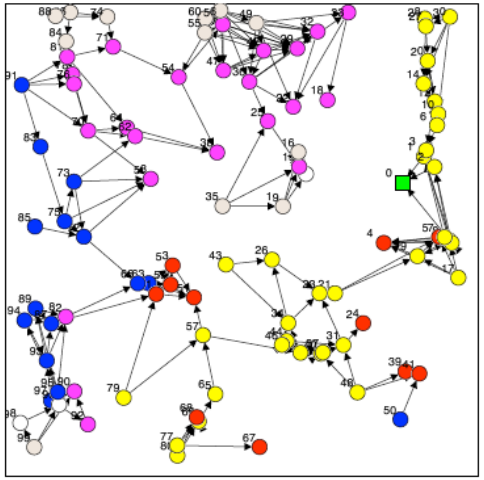


Figure 7: nodes=100 topology=7 localization_error=1.0.

NO ERROR

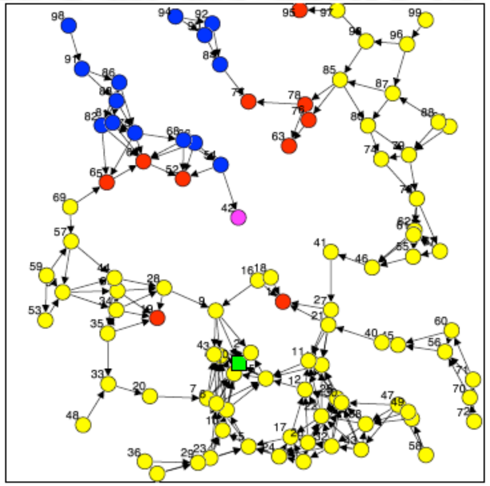


Figure 1: nodes=100 topology=1 localization_error=0.0.

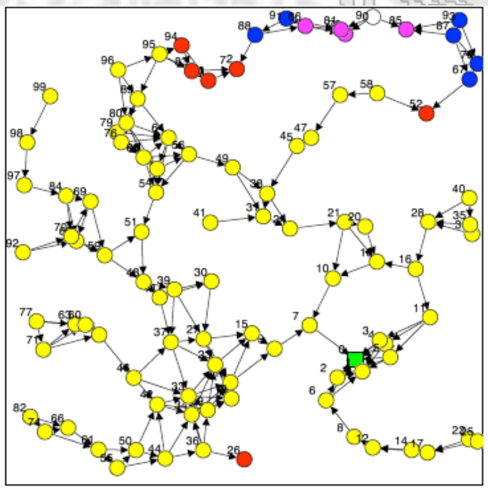


Figure 5: nodes=100 topology=5 localization_error=0.0.

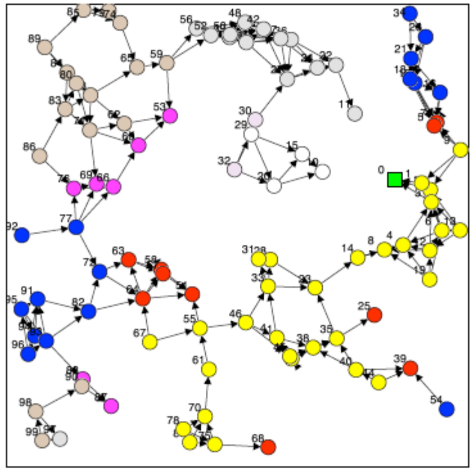


Figure 7: nodes=100 topology=7 localization_error=0.0.

Resilience to localization errors

Where the "yellow nodes" are located

MAX ERROR=R

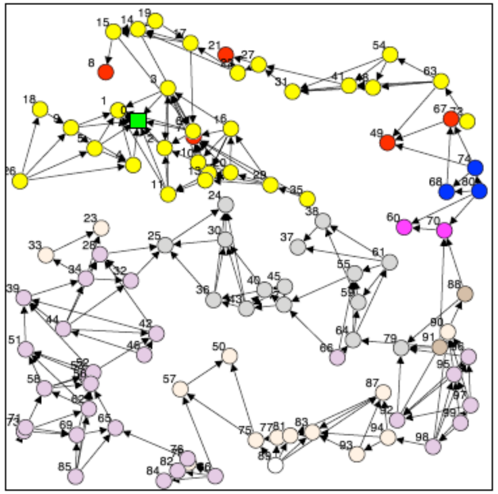


Figure 17: nodes=100 topology=17 localization_error=1.0.

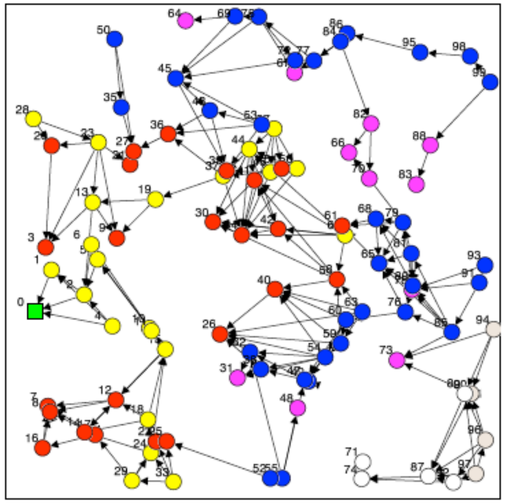


Figure 21: nodes=100 topology=21 localization_error=1.0.

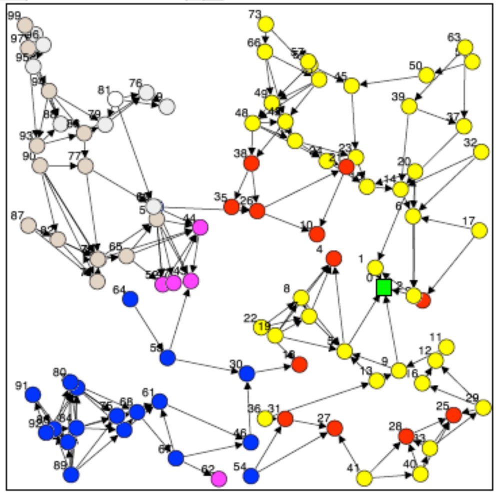


Figure 22: nodes=100 topology=22 localization_error=1.0.

NO ERROR

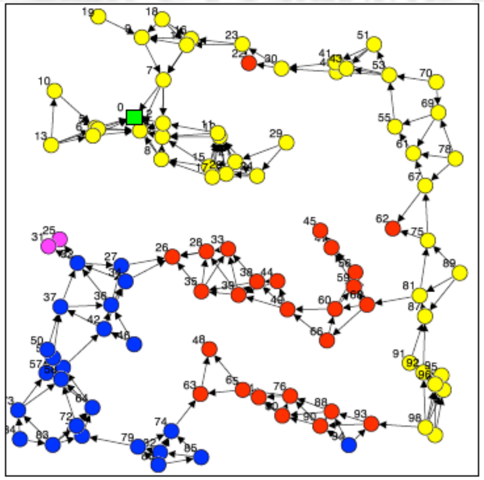


Figure 17: nodes=100 topology=17 localization_error=0.0.

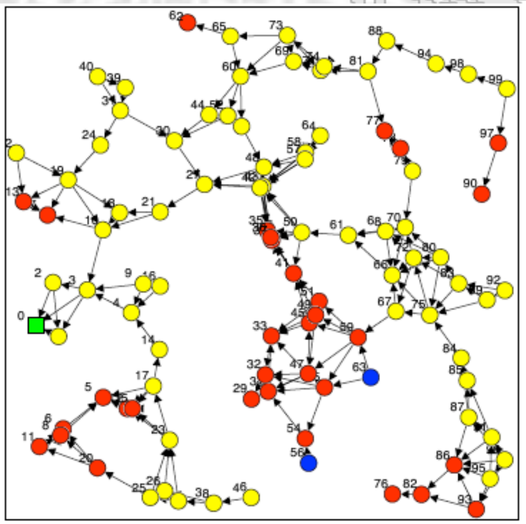


Figure 21: nodes=100 topology=21 localization_error=0.0.

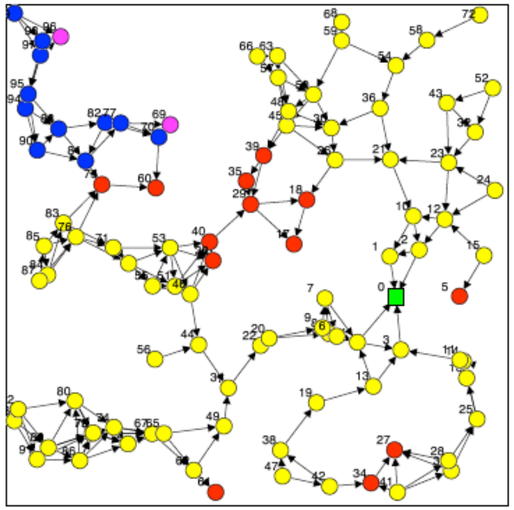


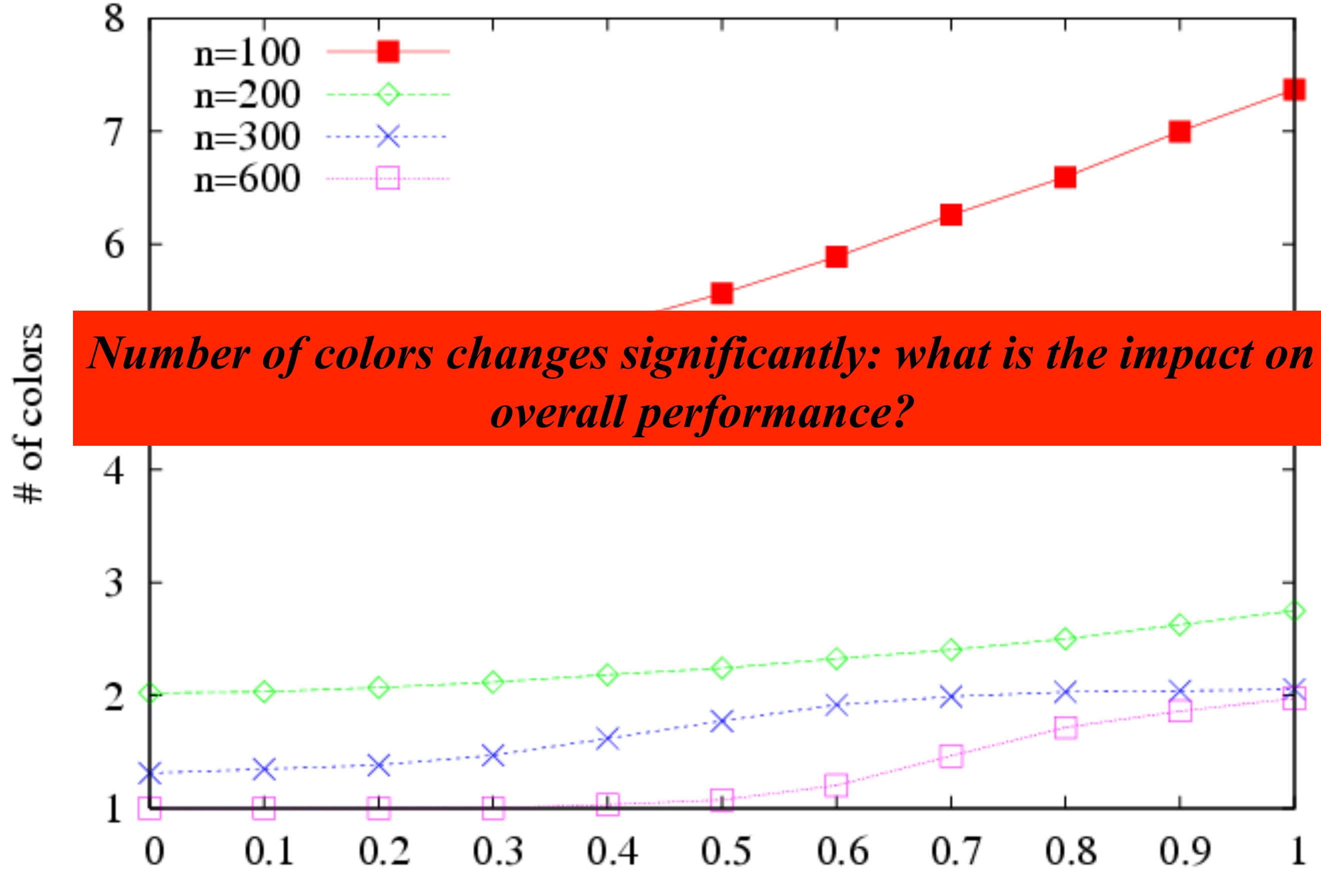
Figure 22: nodes=100 topology=22 localization_error=0.0.

Resilience to localization errors

Number of colors

What does change?

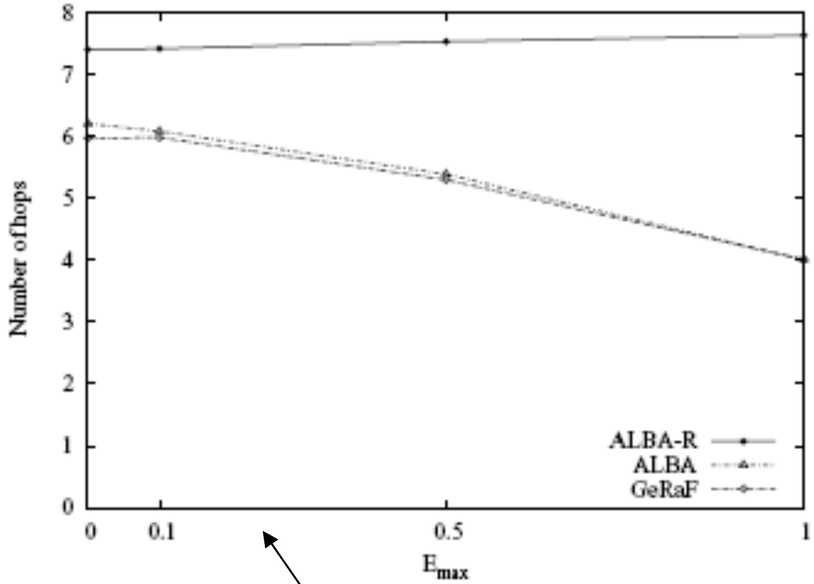
The number of colors needed for all nodes to be able to deliver all packets to the sink

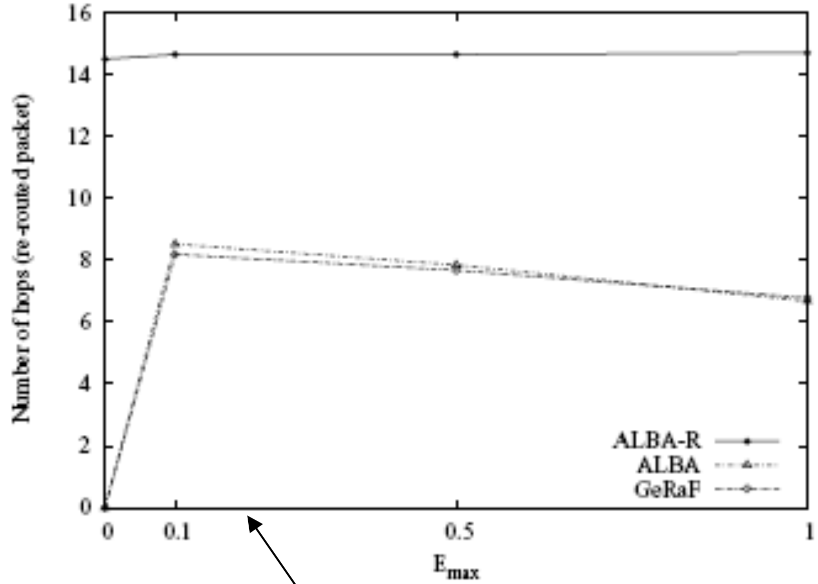


Number of colors changes significantly: what is the impact on overall performance?

Resilience to localization errors
Route length

Only slight increase in ALBA-R with localization error





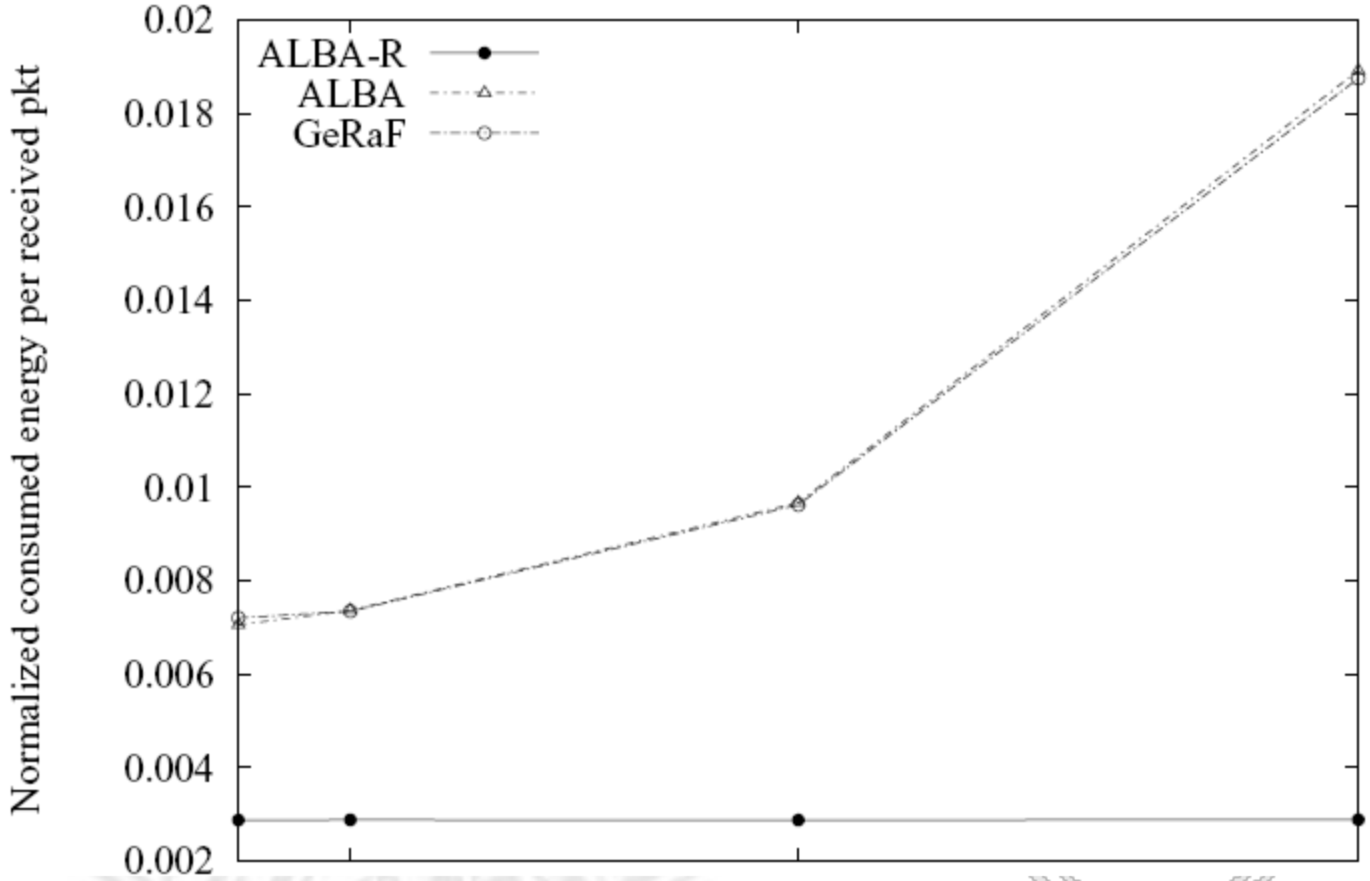
Yellow Nodes (in the original network topology)

Non- Yellow Nodes (in the original network topology)

Exactly same trends for the latencies

Resilience to localization errors

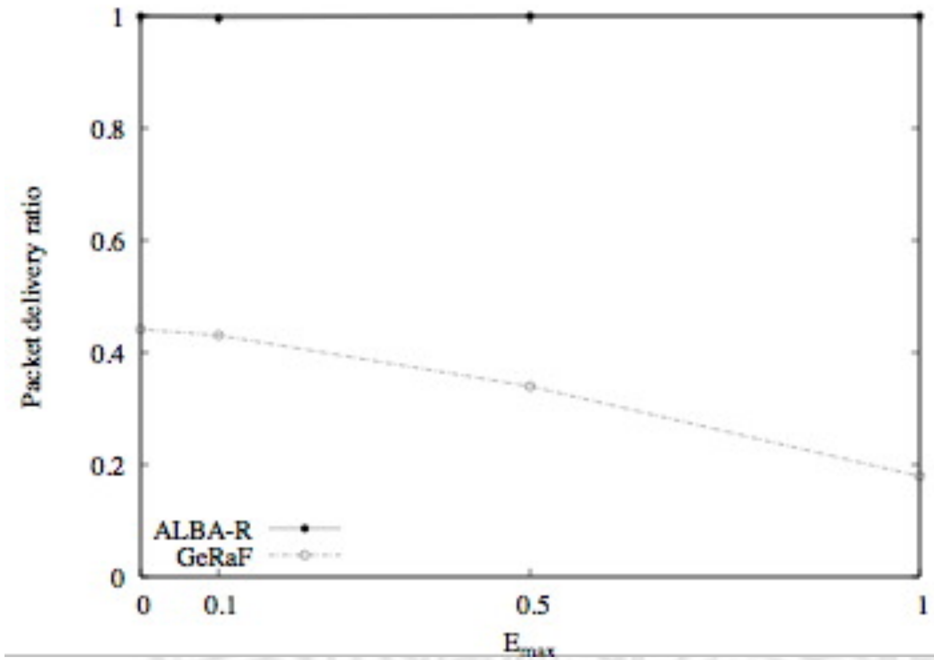
Energy consumption

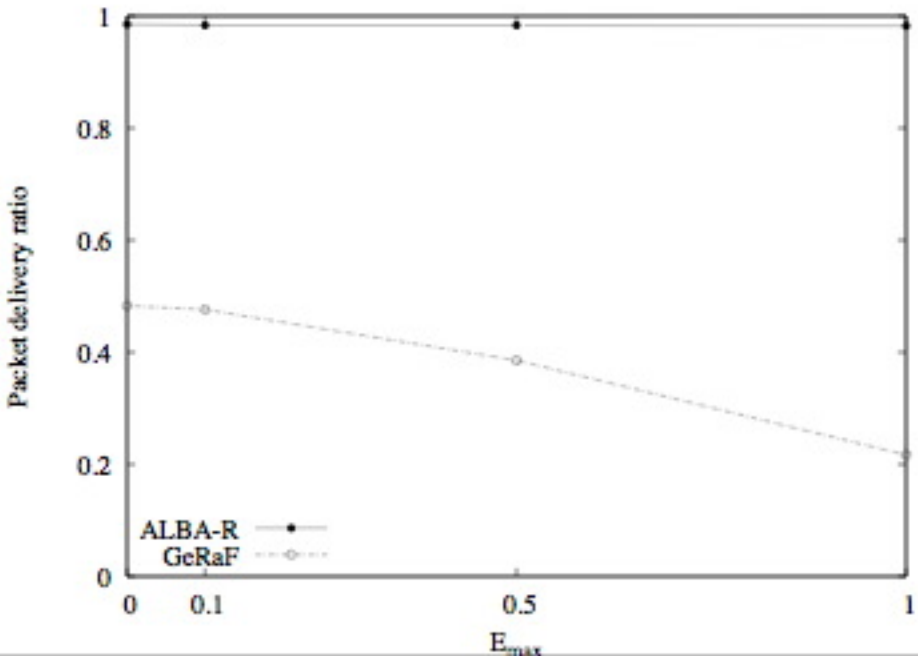


Reduced energy consumption over ALBA and GeRaF. ALBA-R energy consumption does not increase with localization error

Resilience to localization errors

Medium-High Traffic





λ = 0.5 (Medium Traffic) **λ = 1.0 (High Traffic)**

- *ALBA-R continues to work properly also in presence of medium and high traffic*
- Very few losses are due to congestion at critical nodes

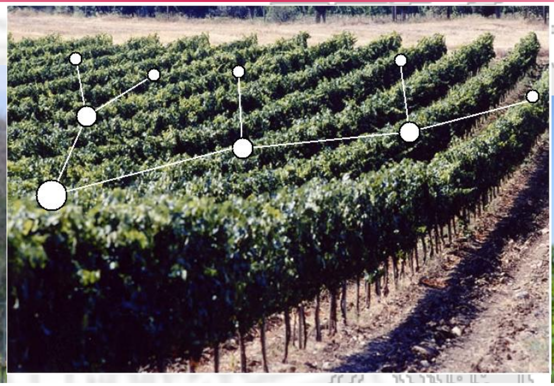
Resilience to localization errors

- ALBA-R is resilient to localization errors
  - Not only it is possible to find a route to the sink, BUT ALSO

↓

  - Performance (in terms of energy consumption, route length, latency) does not significantly decrease in presence of localization errors
  - DESPITE the increase in the number of colors

Testing ALBA-R
Agricoltura di precisione



Monitoraggio:

Temperatura, umidità,

luce, quantità di fertilizzanti

nel terreno

Problematiche

Localization & synchronization	⇒	Risolvibile: ALBA-R (funziona con errori di localizzazione arbitrari)+ prototipo con buona precisione dell'orologio
Processing delays	⇒	Aggiungono ritardo al funzionamento simulato del protocollo
Real life propagation	⇒	I link sono instabili e i messaggi devono essere ritrasmessi
Device unreliability	⇒	Malfunzionamenti dei nodi
Transceiver features	⇒	Overflow coda FIFO, malfunzionamenti del transceiver

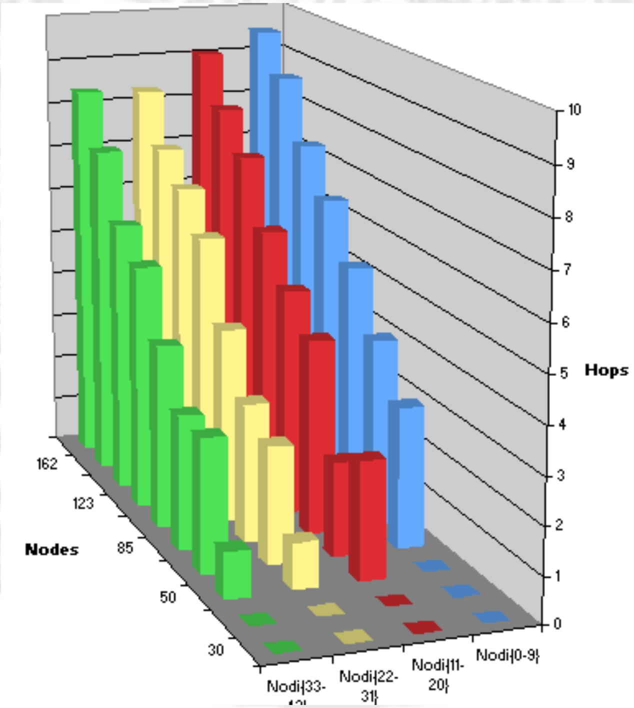
Riprogettazione del protocollo

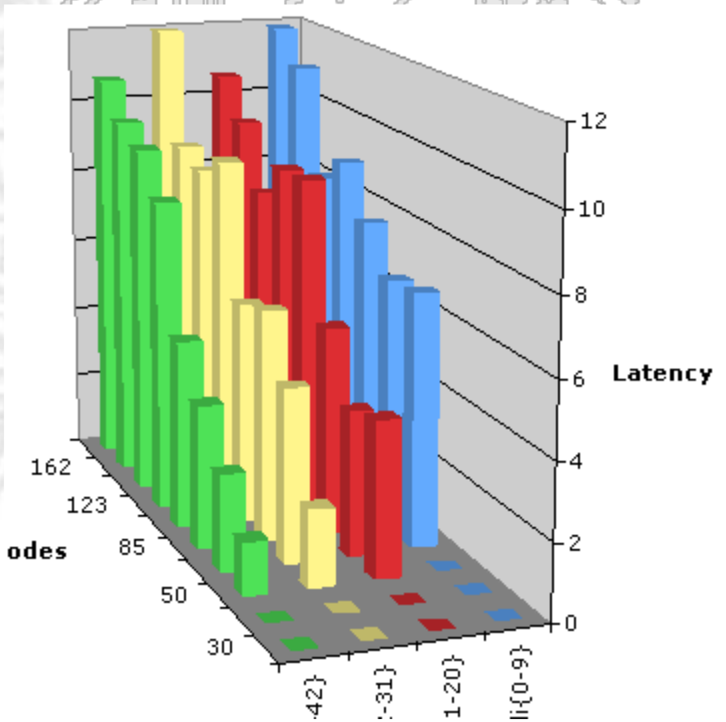
Riprogettazione HW

Scenario di test

- Deployment
  - 49 nodi disposti su una griglia. Il sink è al centro di uno dei lati;
  - Ogni 3 secondi (in media) la rete genera un nuovo pacchetto;
  - I nodi seguono un duty cycle pari a .3
  - Durata dell'esperimento: 1 h (circa 30 pacchetti generati per nodo)
- Metriche
  - Percentuale di pacchetti consegnati con successo
  - Latenza end to end
  - Lunghezza media delle rotte
  - Valori di temperatura misurati
  - Energia consumata per nodo
  - Percentuale di handshake falliti una volta che il nodo ha trovato il relay
  - Durata media delle contese

- I pacchetti sono consegnati con successo
- Consumo energetico dominato dal duty cycle (<18% più alto del duty cycle nominale)





- Green sensor networks
  - Usare l'energia ambientale (energy scavenging) immagazzinata in supercapacitori [+ batterie ricaricabili] per
    - ✓ consentire alle reti di sensori di operare per periodi di tempo lunghisssimo
    - ✓ a basso impatto ambientale
- Andare oltre il concetto di duty cycle, realizzando nodi e reti che consumino solo quando realmente c'è una necessità di comunicazione

ROME: The Protocol

Building blocks

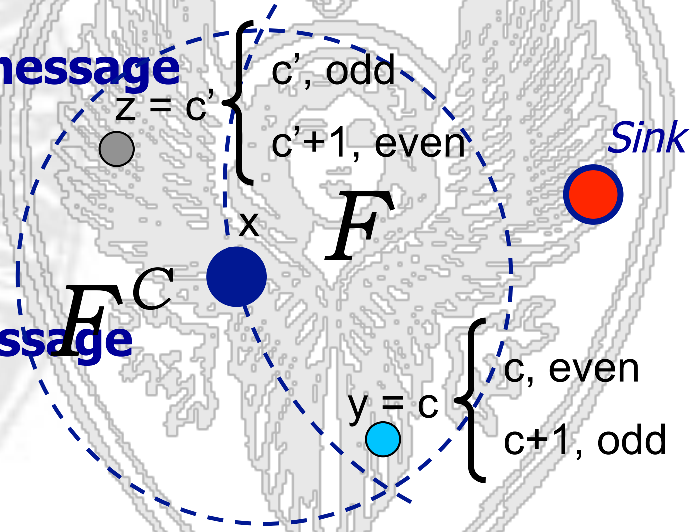
- **Assumptions:**
  - Nodes may detect motion (accelerometers) and re-compute their position
  - The sink is always AWAKE and colored C_0
  - Static nodes have C_{max} as initial color
  - During movement nodes have no color; they select as relay the awake neighbor with the best (i.e., lowest) color
- **Integrated MAC & Routing**
  - If the channel is sensed free the source node x starts looking for a relay (*scanColor()*)
  - If a relay is found:
    - ✓ Data packets are sent back-to-back (ACK is required)
    - ✓ If x is not moving it updates its color based on the relay found
  - If a relay is not available:

findRelay()

scanColor() procedure

Differently from static scheme nodes with any smaller color than the sender can answer the RTS

- **The *scanColor()* procedure**
  - **Return the ID and color of the best relay available**
  - **Permits to upgrade the sender color and adapts to topology changes**
- **The source node broadcasts a RTS message**
  - **Desired range of colors**
  - **Sender and sink positions**
- **Feasible relays reply with a CTS message**
  - **Required color**
- **Three events may occur**
  - **No relay is available**
  - **A relay is found**
  - **Collision among multiple relays**
    - ✓ **A relay with the lowest color is selected (Binary Search)**

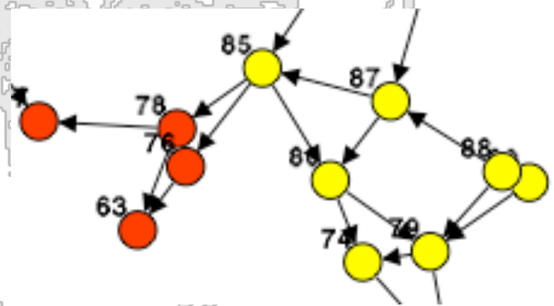


Fast convergence due to limited neighborhood colors variability

ROME: Colors Update

Adding nodes

- ***Moving mobile node***
  - Node has no color ⇒ it does not reply to any RTS
  - Node search for a relay in the range $C_0, \dots, C_{max}$
  - Color is updated when node stops

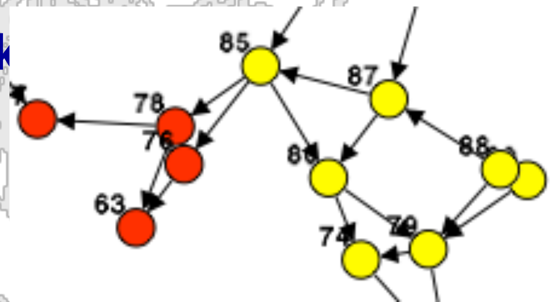


- ***A new node added***
  - Node sets its color to $C_{max} + 1$
  - Node looks for a color in the set $C_0, \dots, C_{max}$
  - A static node replies to the RTS
  - Node participates to upcoming contentions
  - Node provides color update to its neighbors only if it remains

ROME: Colors Update

Removing nodes

- ***A mobile node leaves a location***
  - When a node moves, it looses its color
  - A mobile node never affects the color of the region it is in
    - ✓ When it leaves the region other routes will be used by the nodes which were using it as relay
    - ✓ No overhead and time consuming re-coloring is needed
  - Each neighbors left behind continues to work



- ***Nodes death or failure***
  - A static node *x* is removed from the network
  - Its neighbors need to update their color

Simulation settings

- **Simulation area: 320 m x 320 m**
- **N = 300 nodes randomly and uniformly deployed**
- **Static nodes = p_s * N ($p_s \leq 1$), mobile nodes = p_m * N ($p_m = 1 - p_s$)**
- **p_m varies in the set {0.2, 0.5}; Mobility model: Random way-point**
- **ROME duty cycle d = 0.1 - other solutions do not provide duty cycle**
- **Source randomly selected, poissonian traffic generation with λ = {1, 2, 4} pkt/s**
- **Comparison with:**
  - Greedy Perimeter Stateless Routing (GPSR)
  - Probabilistic flooding
- **Metrics:**
  - Packet delivery ratio
  - Average route length

ROME performance

- **ROME p_m = {0.2, 0.5}, p_t = {60s, 900s}**

p_m	0.2		0.5	
Metrics ↓ | Pause time p_t →	$900s$	$60s$	$900s$	$60s$
Packet Delivery Ratio	1	1	1	1
End-to-end Latency (s)	$4.83(s), 4.9(m)$	$5.62(s), 5.55(m)$	$7.9(s), 7.96(m)$	$11.25(s), 11.26(m)$
Route Length (hops)	$8.35(s), 8.57(m)$	$8.34(s), 8.48(m)$	$9.57(s), 9.87(m)$	$9.66(s), 10.02(m)$
Normalized Energy Consumption (mJ/bit)	0.30	0.27	0.29	0.21
Overhead	13.61	13.7	15.52	16.03

  - **All packets successfully delivered**
  - **No performance degradation for mobile nodes**
  - **Reduced number of eligible relays for aggressive mobility**
  - **Mobility increases the energy savings**
  - **Static network is still able to efficiently manage more traffic**

- **Mobile nodes improve the performance**
  - **With p_t = 900 and p_m = 0.5, latency is 34% lower than when only static**

Results

ROME vs GPSR and Probabilistic Flooding

- **ROME vs GPSR, $p_m = 0.2$, $p_t = 240s$**

λ	1.0		2.0		4.0	
Metrics ↓ | Protocols →	ROME	GPSR	ROME	GPSR	ROME	GPSR
Packet Delivery Ratio	1	0.98	1	0.78	1	0.32
End-to-end Latency (s)	5.42	1.97	7.29	7.7	17.71	29.2
Route Length (hops)	8.42	7.23	8.51	10.7	8.83	18.9
Normalized Energy Consumption(mJ/bit)	0.29	2.93	0.15	2.00	0.09	2.70
Overhead	13.75	15.13	13.5	44.05	13.33	139.5

- **ROME v**

Metrics ↓ | Forwarding probability →	1.0	0.8	0.7	ROME
Packet Delivery Ratio	0.94	0.93	0.9	1.0
End-to-end Latency (s)	3.73	3.6	3.6	5.42
Route Length (hops)	9.2	9.1	9.0	8.42
Normalized Energy Consumption (mJ/bit)	3.32	3.31	3.38	0.29
Overhead	303.3	244.4	215.7	13.75

  - **All packets delivered, effective relay selection scheme (no beacons, congestion mitigation)**
  - **Better energy performance (more aggressive mobile nodes duty cycle)**