Tutorial: 802.16 MAC Layer Mesh Extensions Overview

S802.16a-02/30

Date Submitted:

2002-3-11

Sources:

Dave Beyer Nokia Wireless Routers

Nico van Waes Nokia Wireless Routers

Carl Eklund Nokia Research Center

Venue:

IEEE 802.16, session 18, Monday 16:00 - 18:00

Base Document:

http://wirelessman.org/tga/contrib/C802.16a-02_30r1.pdf (and P802.16a/D2)

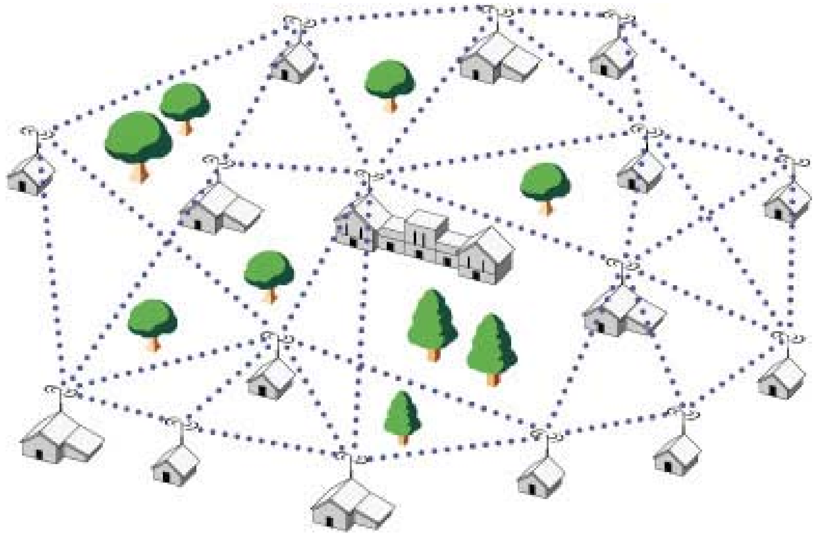
Purpose:

Information and discussion

Tutorial:

802.16 MAC-layer Mesh Extensions

(per P802.16a/D2, amended as suggested in C802.16a-02/30r1)



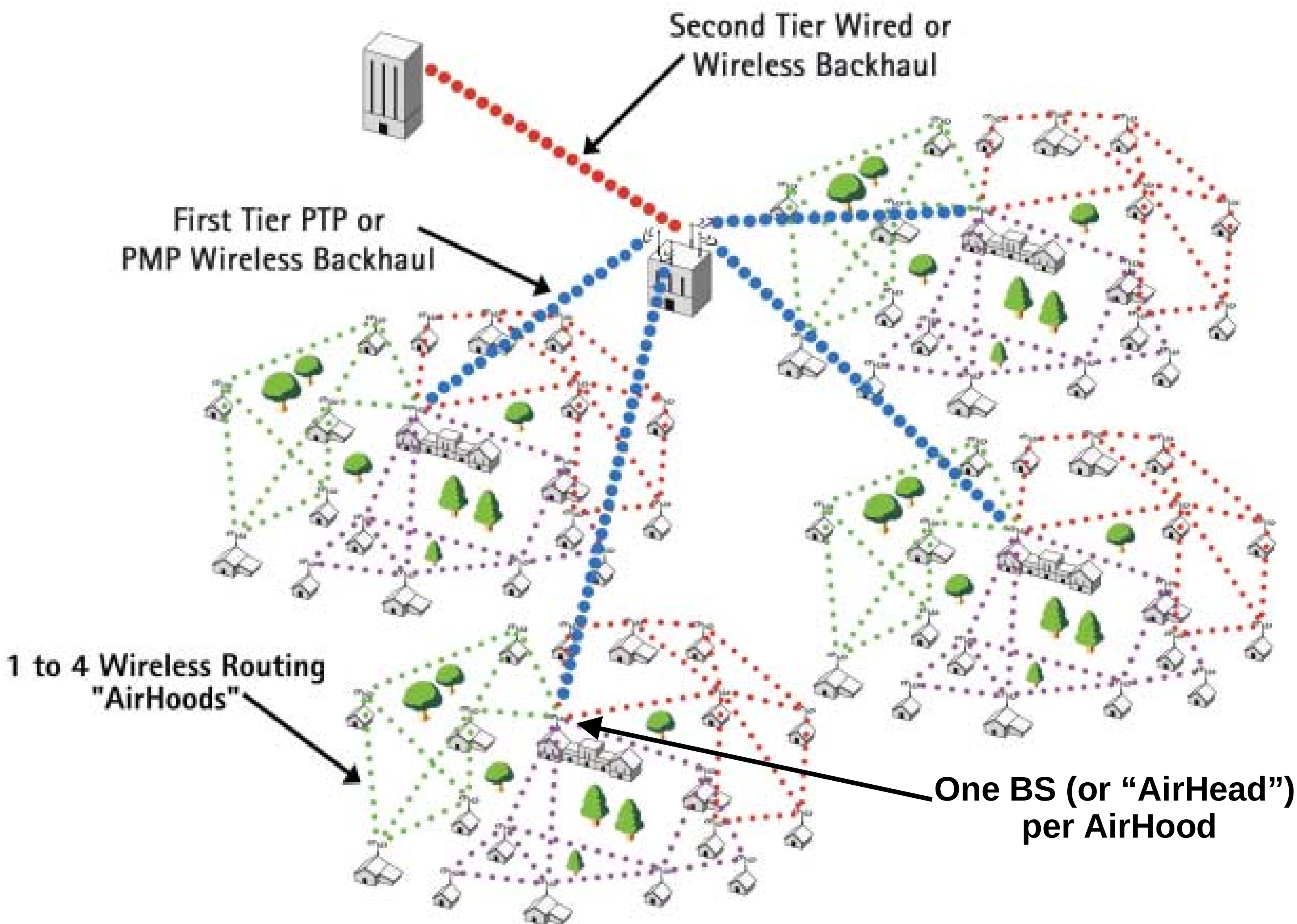
Dave Beyer, Nico van Waes, Carl Eklund
March 11, 2002
IEEE 802.16, St Louis

NOKIA

Mesh Tutorial -- Outline

- **Introduction – Dave Beyer**
  - Mesh Network Architecture
  - Fundamental Benefits of Mesh
    - Coverage
    - Scaling
    - Bandwidth
  - Tutorial Outline
  - Protocol & Frame Structure, Roles of BSs
- **Mesh MAC, Essential Functions – Nico van Waes**
  - Network Configuration
    - Mesh Distributed, Election-based Scheduling
  - Low-level Network Entry & Synchronization
  - Link Establishment
  - Data Scheduling
    - Centralized
    - Coordinated Distributed
    - Uncoordinated Distributed
- **Security, Configuration & Mechanics – Carl Eklund**
  - Upper-layer network entry
  - Neighbor authentication
  - Headers
    - Connection ID in Generic MAC Header
    - Transmitter Node ID in Mesh Subheader
  - SAP
    - Data & Connection Requests & Indications
    - Forwarding Tree Updates (for centralized scheduling)

Mesh-based Broadband Wireless Access



One BS (or "AirHead") per AirHood

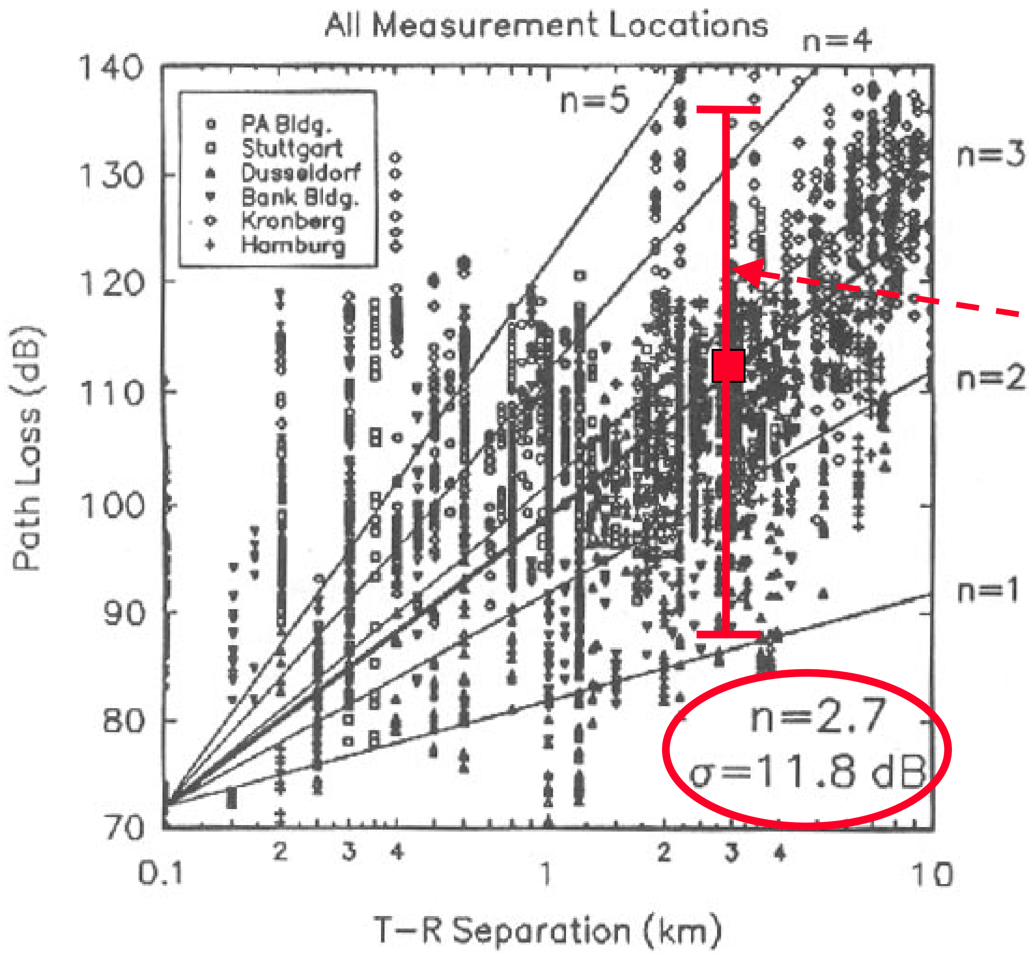
NOKIA

Mesh has more complex channel considerations than PMP

- avoiding “hidden terminal” collisions,
- selection of links,
- synchronization,
- power vs. data rate tradeoffs,
- greater routing--MAC interdependence,
- etc.

So why bother?

RF Path Loss Environment



2 standard deviations (contains 95% of points)

Measurement Data Source:
“Wireless Communications”
by Ted Rappaport, 1999

NOKIA

Solving Coverage
Path Loss is Highly Variable

Path loss is highly variable for wireless broadband

- Typically driven largely by obstacles
- Leads to the "Log-Normal" path loss model:

$$C + 10 \cdot n \cdot \log_{10}(\text{dist}) + X_\sigma$$

random variable X_σ with standard deviation σ

In PMP networks, large σ is bad

- Must design for worst-case; e.g., leads to $1/r^4$ or $1/r^5$ models

In mesh networks, for a given n, large σ is GOOD!

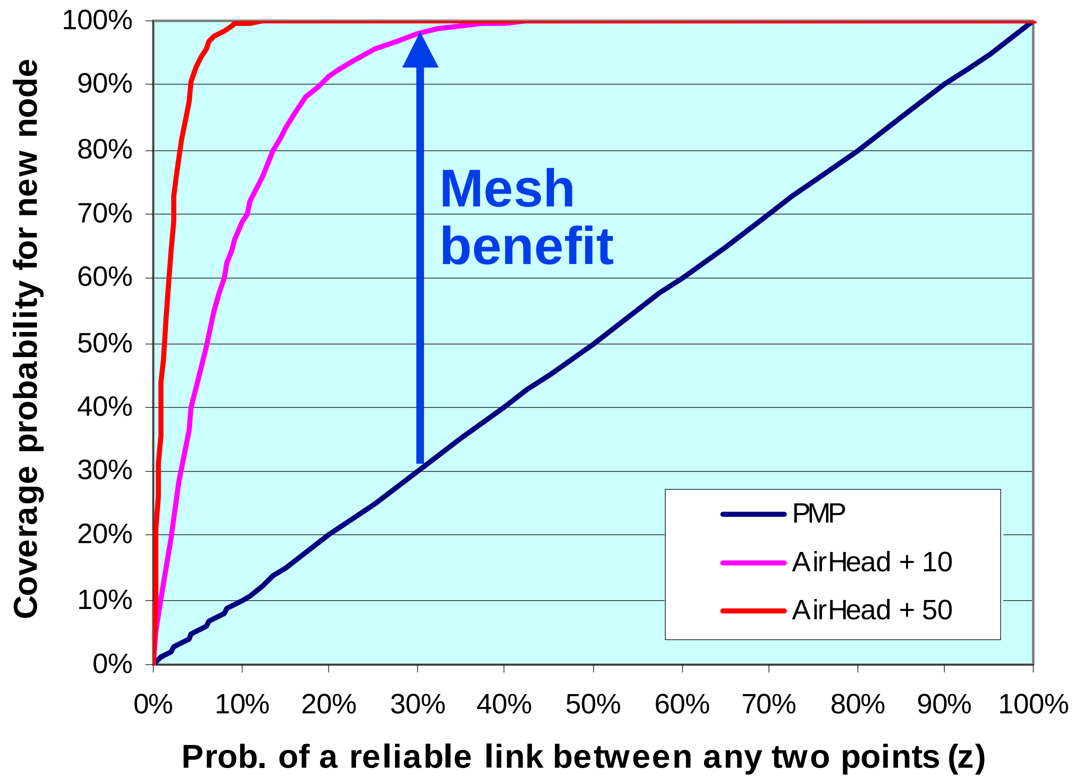
- Best-case links in the area are automatically identified & used.

NOKIA

Solving Coverage
Simplified Model

Assume standard deviation completely dominates

- Chance of a link between any pair of devices simplifies to a fixed link probability: *z*

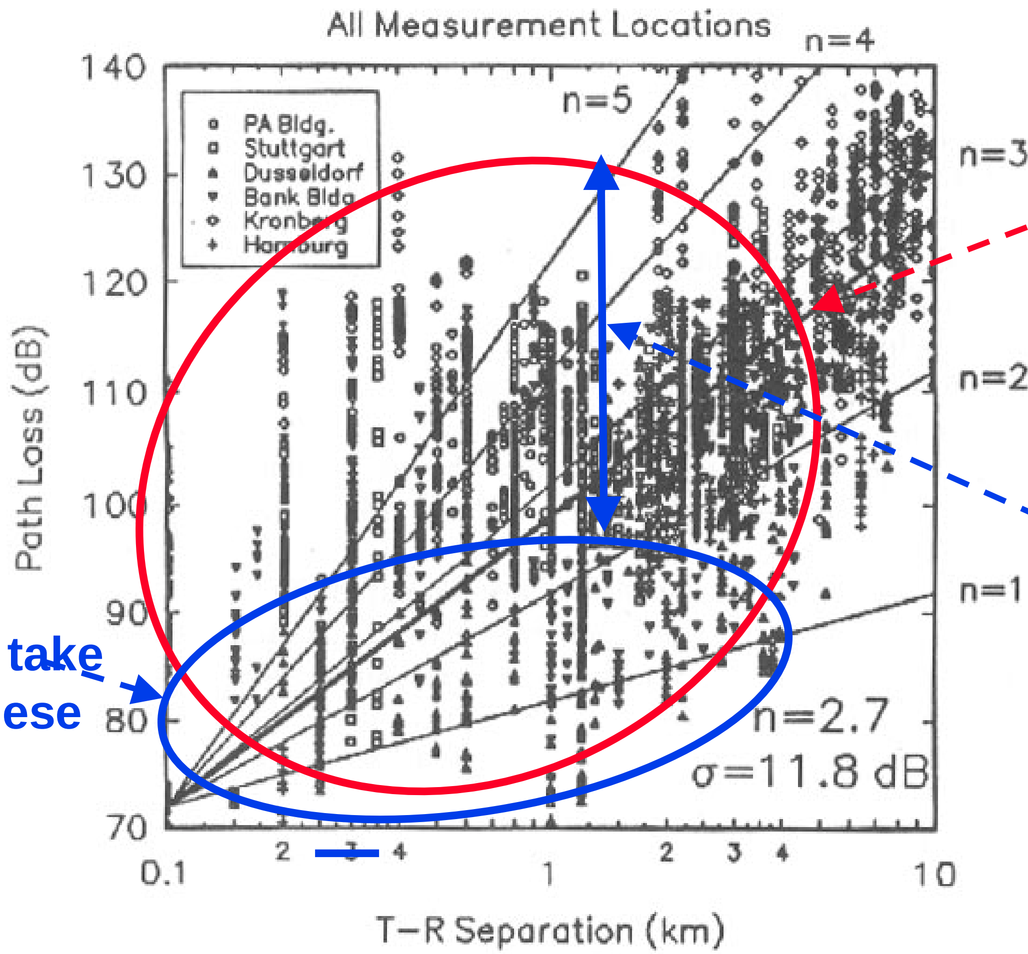


m-device mesh coverage probability
$= 1 - (1 - z)^{m}$

Mesh coverage & robustness improve *exponentially* as subscribers are added

NOKIA

RF Path Loss Environment



PtMP systems must plan for links which tolerate this kind of path loss

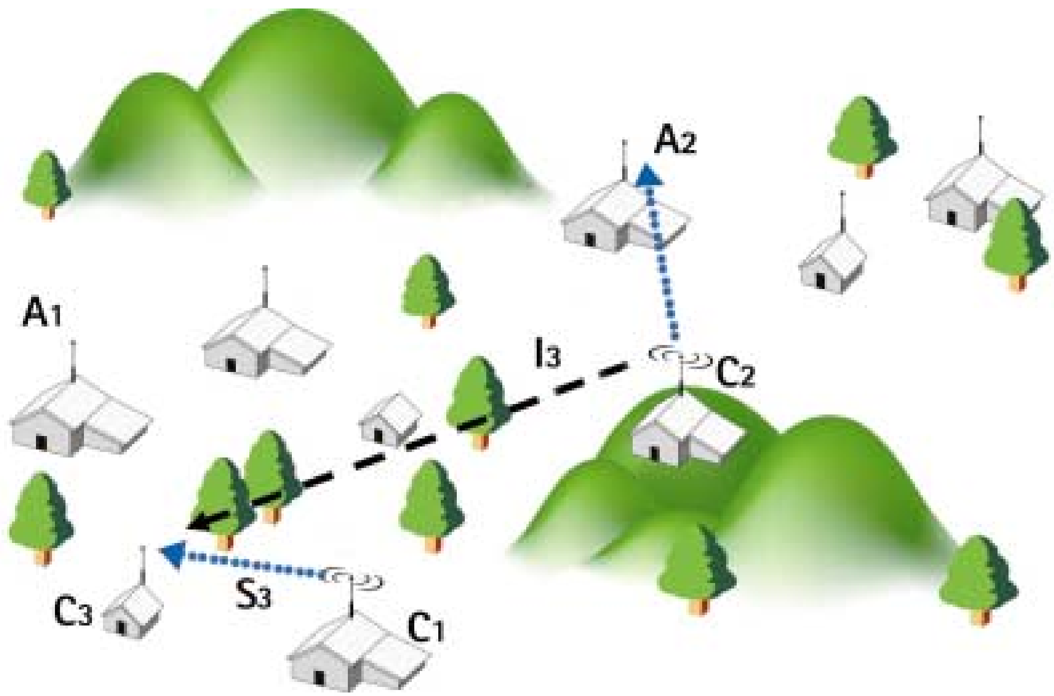
Mesh link budget advantage ~40 dB [50-device mesh, 95% coverage, 1-mile cells]

Mesh networks take advantage of these kind of links

Measurement Data Source: “Wireless Communications” by Ted Rappaport, 1999

NOKIA

Scaling with Mesh Networks



Inherently <u>advantageous</u> signal-to-interference relation

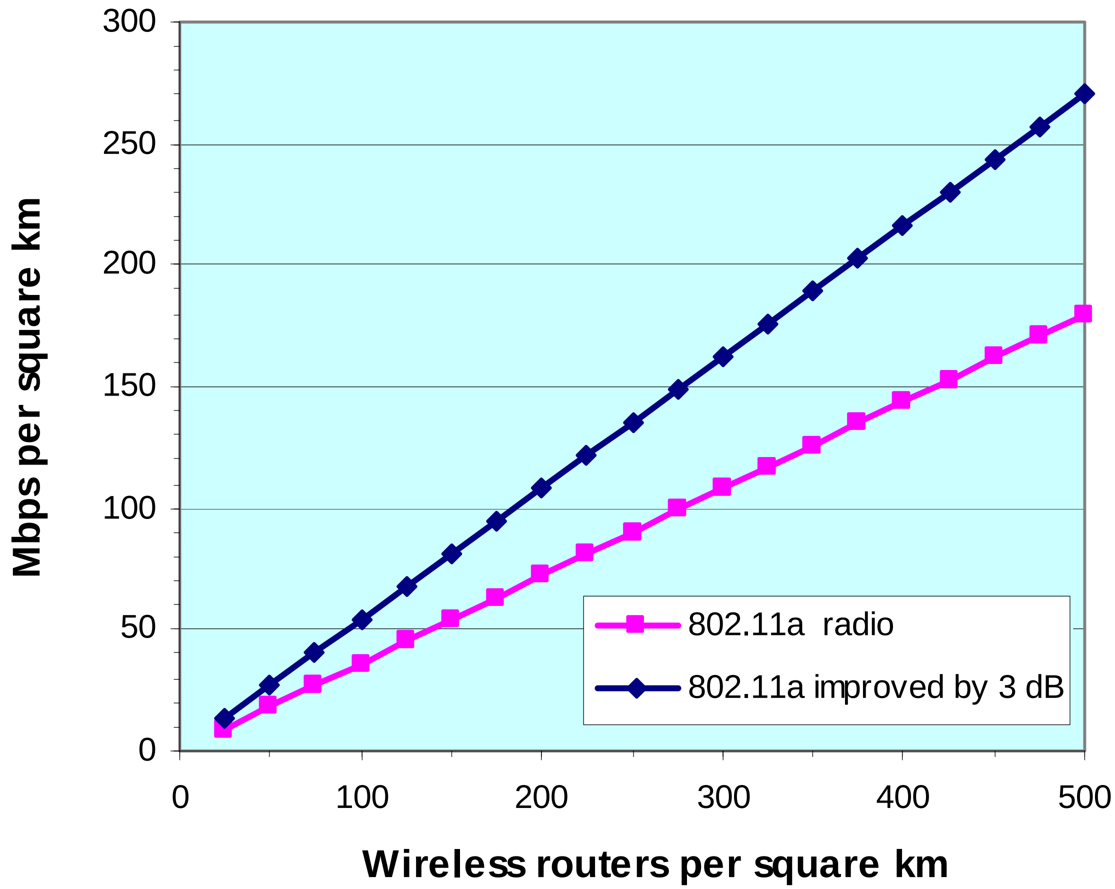
- E.g., ~1/r^{3} for interference versus ~1/r$^{2.5}$ for signal

Permits scaling to large, dense networks

- With adaptive power control & neighbor selection.

Scaling with Mesh Networks

- 4 channels per “cell” reused in each cell
- Average path loss $1/r^3$ active links $1/r^{2.5}$
- 802.11a-type radios



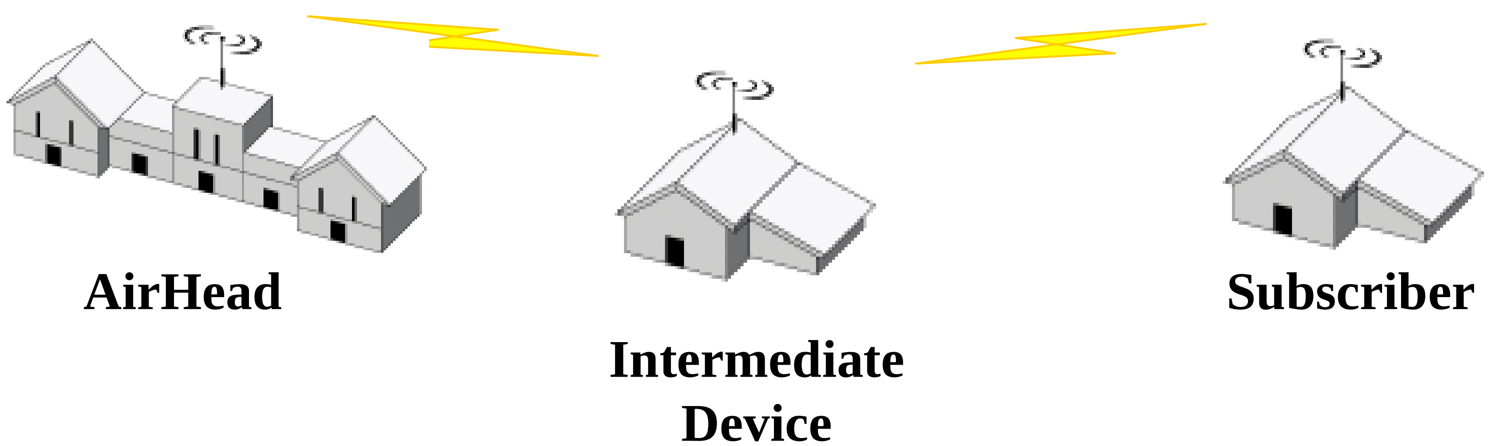
Mesh User Throughput Over Multiple-hop Paths

Simple example:

- $1/r^3$ path-loss & common noise environment
- Standard-compliant, .16a OFDM radios, 20 MHz BW

Which gives higher user throughput?

- Direct path, or
- Two-hop path?

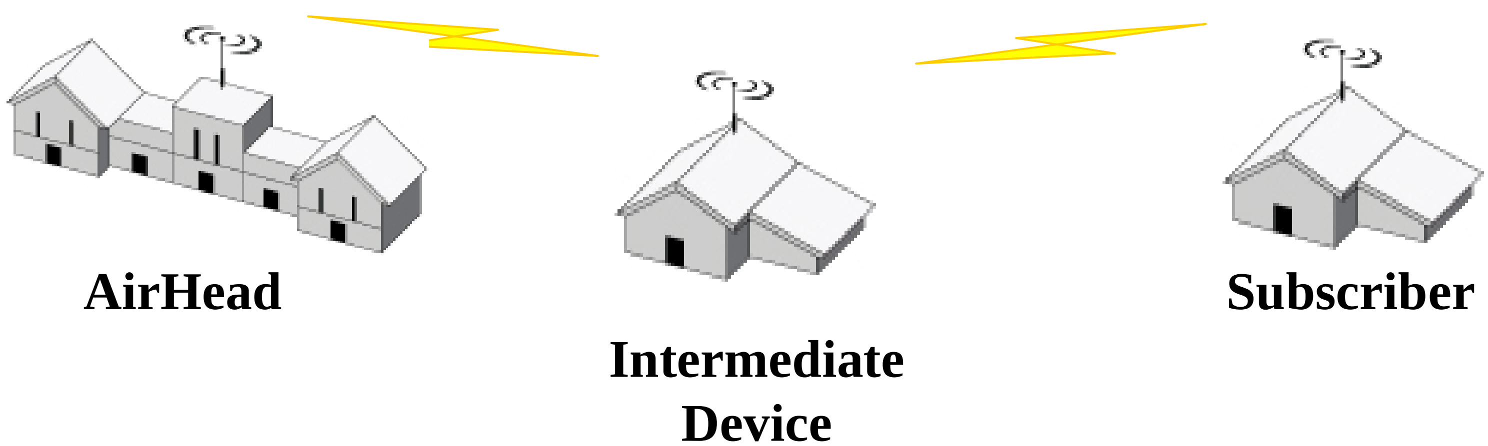


Mesh User Throughput Over Multiple-hop Paths

Mesh nodes adapt waveform on per-link basis

- If direct link supports QPSK ½ (16 Mbps) waveform, then
- Shorter links will use 16QAM ¾ (48 Mbps) due to 9 dB less path loss

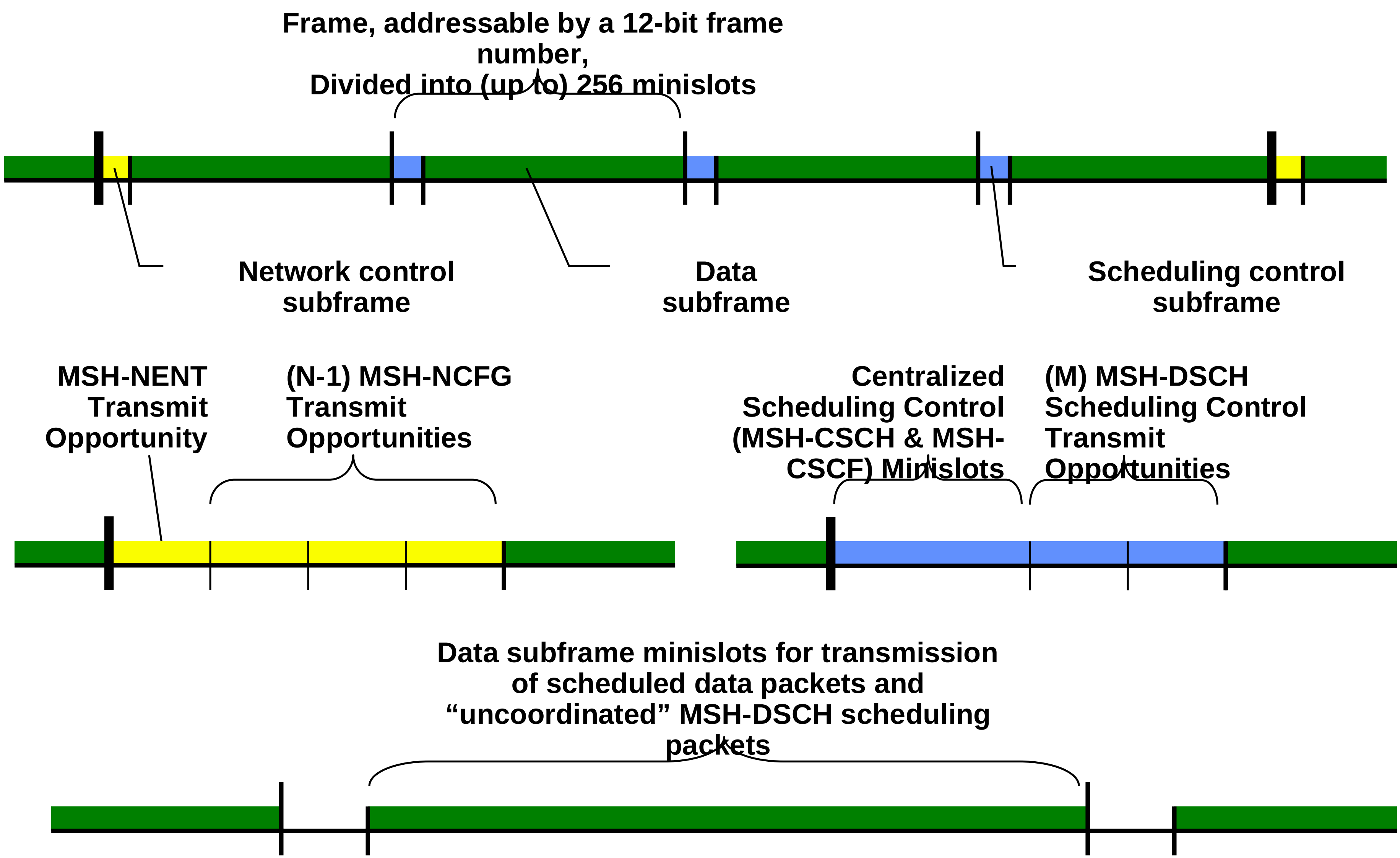
16 Mbps over direct path; 24 Mbps over two-hop path



…and

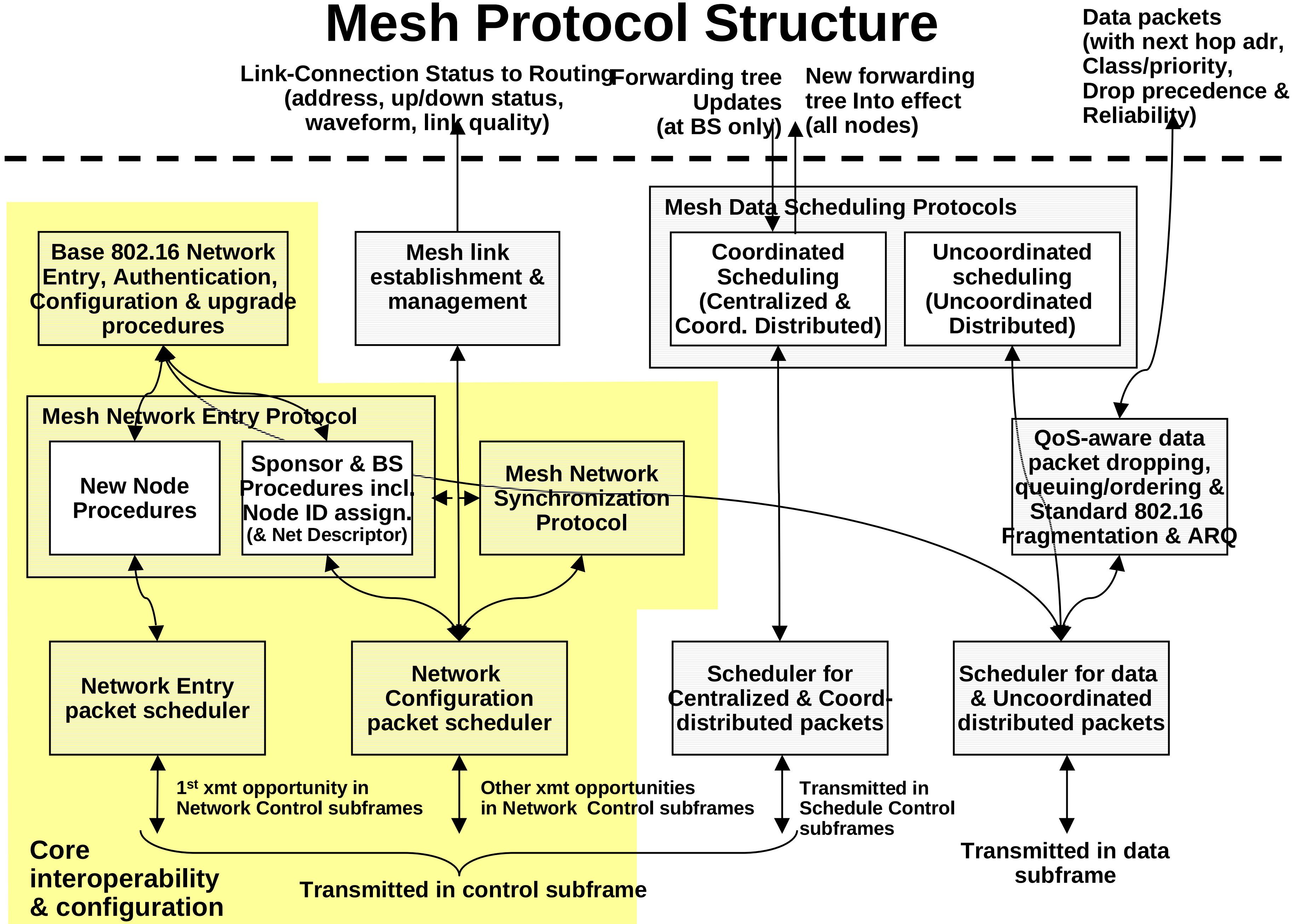
the complexities of the wireless mesh can be tamed.

Mesh MAC Frame Structure



NOKIA

Mesh Protocol Structure



Network Configuration, Entry and Synchronization

NetConfig (MSH-NCFG) Packet

Syntax	Size
MSH-NCFG_Message_Format() {	
Management Message Type = 42	8 bits
NumNbrEntries	5 bits
NumBSEntries	2 bits
Embedded Packet Flag	1 bits
Xmt Power	4 bits
Xmt Antenna	3 bits
NetEntry MAC Address Flag	1 bits
Network base channel	4 bits
NetConfig Count	4 bits
Timestamp Frame Number Network Control Slot Number in frame Synchronization hop count	 12 bits 4 bits 8 bits
NetConfig schedule info Next Xmt Mx Xmt Holdoff Exponent	 3 bits 5 bits
NetEntry MAC Address	48 bits
for (i=0; i< NumBSEntries; ++i) {	
MSH-BS_IE()	24 bits
}	
for (i=0; i< NumNbrEntries; ++i) {	
Nbr Node ID	16 bits
MSH-Nbr_Physical_IE()	16 bits
MSH-Nbr_Logical_IE()	16 bits
}	
MSH-NCFG_embedded_data()	variable
}	

- **MAC header**
  - **Broadcast Management Format**
  - **Net ID = 0x00: All-net broadcast**

- **Next Xmt Time** $> 2^{\text{Xmt Holdoff Exponent}} * \text{Next Xmt Mx}$
  $< 2^{\text{Xmt Holdoff Exponent}} * (\text{Next Xmt Mx} + 1)$
- **Earliest Subsequent Xmt Time** $= 2^{\text{Xmt Holdoff Exponent}+4} + \text{Next Xmt time}$

- **To announce presence or sponsorship of new node**
- **Identifies mesh BSs and “cost” to transfer data to it.**

NetConfig Packet - BS & NbrLink Info

Syntax	Size
MSH-BS_IE() {	
BS Node ID	16 bits
Number of hops	3 bits
Xmt energy/bit	5 bits
}	

- **Indicates minimum energy/bit needed to reach BS from this node in mWatts-secs * 2 $^{\text{XmtEnergyUnitsExponent} - 4}$**

"Distributed Bellman Ford Algorithm"

$$E_i = \min_{j \in N_i} [E_{j \to i} + E_j] \;\; mW \cdot \mu s$$

$$E_{i \to j} = \mathrm{P_{Tx}} / R_{i \to j}$$

Syntax	Size
MSH-Nbr_Physical_IE() {	
Logical Link Info Present	1 bit
Logical Link Requested	1 bit
Logical Link Accepted	1 bit
Hops to Neighbor	1 bit
Estimated propagation delay	4 bits
Nbr Next Xmt Mx	5 bits
Nbr Xmt Holdoff Exponent	3 bits
}	

- **Permits scheduling out to 3 hops**
  (if ExtendedNeighborhoodType ==1)

Syntax	Size
MSH-Nbr_Logical_IE() {	
Rcv Link Quality	3 bit
Nbr burst Profile	4 bit
Excess Traffic Demand	1 bit
Nbr Xmt Power	4 bits
Nbr Xmt Antenna	3 bits
Short Preamble flag	1 bit
}	

- **Related to PER % for NetConfig-size pkts**
- **Excess over current schedule**

NOKIA

NetConfig Embedded Packet

- **Used by protocols above (including higher-level MAC services) for communicating brief broadcast messages, or for brief communications to neighbors without requiring logical link establishment.**
- **All Embedded NetConfig Packets start with following 16-bit header:**

Syntax	Size
MSH-NCFG_embedded_data() {	
Extended embedded_data	1 bit
Reserved	3 bits
Type	4 bits
Length	8 bits
Embedded_data_IE()	variable
}	

- **Indicates whether another embedded NetConfig packet follows.**
- **Indicates which protocol module handles this packet type)**
  - **0x0 - Reserved**
  - **0x1 – Network Descriptor**
  - **0x2 – Network Entry Open**
  - **0x3 – Network Entry Reject**
  - **0x4 – Network Entry Ack**
  - **0x5 – Neighbor Link Establishment**

NetConfig Embedded Packet Types - I

- **NetEntryOpen**

Syntax	Size
MSH-NCFG_embedded_data_IE() {	
Minislot Start	8 bits
Minislot Range	8 bits
Frame number	12 bits
Channel	4 bits
Schedule validity	12 bits
Channel	4 bits
Estimated Propagation Delay	5 bits
Reserved	3 bits
}	

- **Purpose**
  - **Ranging / Fine synchronization**
  - **New Node to Sponsor schedule for upper-layer network entry**

- **NetEntryReject**

Syntax	Size
MSH-NCFG_embedded_data_IE() {	
Rejection Code	8 bits
Rejection Reason	160 bits
}	

- **NetEntryAck (No subfields)**

NOKIA

NetConfig Emb Packet Types - II

Syntax	Size
MSH-NCFG_embedded_data_IE() {	
Frame Length Code	4 bits
MSH-CTRL-LEN	4 bits
MSH-DSCH-NUM	4 bits
Scheduling Frames	4 bits
MSH-CSCH-slots	8 bits
Operator ID	16 bits
XmtEnergyUnitsExponent	4 bits
Channels	4 bits
Logical Network ID	8 bits
ExtendedNeighborhoodType	1 bit
MinCSForwardingDelay	7 bits
for (i=0; i< Channels; ++i) {	
Physical Channel code	8 bits
}	
Channel re-use	3 bits
Peak/Average flag	1 bits
Reserved	2 bits
NumChannelMaps	2 bits
for (i=0; i< NumChannelMaps; ++i) {	
Number of Channels	8 bits
Max. xmt power at antenna port	6 bits
Max. EIRP	6 bits
}	
}	

- **Number of Control Xmt Opportunties**
- **Number of MSH-DSCH transmit opportunities**
- **Number of scheduling control frames between network control frames**
- **Number of Minislots in data subframe allocated to centralized scheduling**

- **Used with BS Info IE to compute Xmt energy/bit to reach BS**
- **Number of logical channels**

- **2-hops (0); 3-hops (1)**
- **Mininimum MSH-CSCF Forwarding Delay (in OFDM symbols)**

- **Minimum number of hops of separation between links before a channel can be re-used by the centralized scheduling algorithm. Range: 1 hop to 7 hops.**
- **Regulatory rules in: peak = 0 , average = 1**

- **Number of regulatory rule sets**

- **Subsequent number of channels out of “Channels” above**
- **(dBm)**
- **(dBm)**

Mesh Distributed Election-based Scheduling

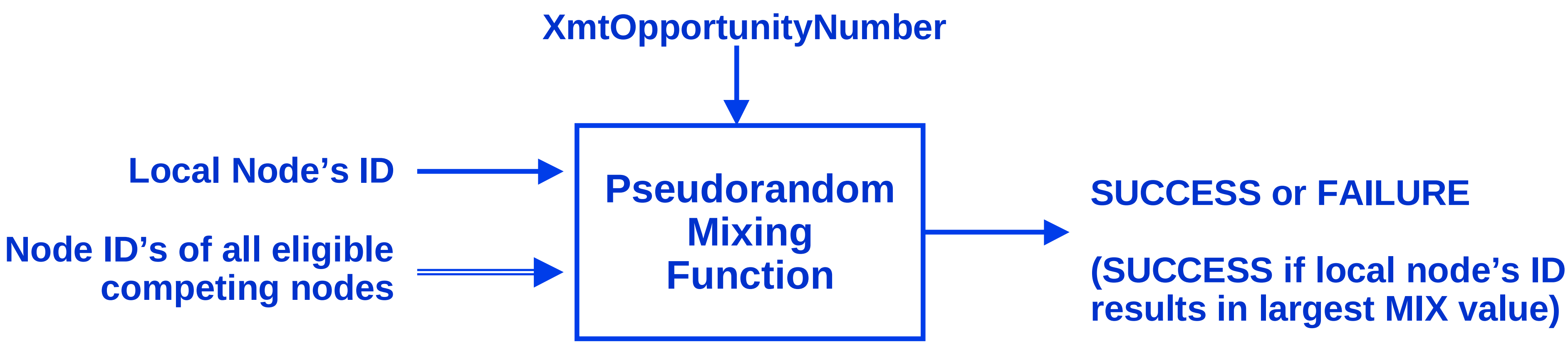
- **Features**
  - **No explicit negotiation:** Supports scheduling of regular broadcast transmissions in a multihop, mesh network without explicit schedule negotiation. So, useful for net configuration messages.
  - **Collision-free:** Scheduling is collision-free within each node's extended neighborhood.
  - **Distributed:** Algorithm is completely distributed requiring no central control.
  - **Fair:** Scheduling assignment treats all nodes equally.
  - **Robust:** In addition to being a distributed alg., the scheduling seed changes pseudo-randomly for each frame, so any collisions (e.g., caused by transient conditions) will not persist.
- **Use: For scheduling MSH-NCFG and coordinated MSH-DSCH messages.**
- **Algorithm Inputs**
  - The transmit opportunity number for the type of message being scheduled
    - {frame number and transmit opportunity within that frame}
  - The node identifiers for all nodes within the extended neighborhood
    - Communicated using the MSH-NCFG packets
    - Extended neighborhood can be defined as the 2- or 3-hop neighbors of the local node. (3-hop neighborhood used in environments which are closer to free-space.)
  - XmtHoldoffTime of the local node
    - Algorithm is run when it is the local node's turn to transmit; I.e., NextXmtTime == <now>
  - As many sets of {node ID, NextXmtTime, XmtHoldoffTime} of nodes within the extended neighborhood as have been received recently
    - Communicated within the message being scheduled
    - Sets with (NextXmtTime) + (XmtHoldoffTime) < the current time are no longer useful.
- **Algorithm Ouput**
  - New NextXmtTime of the current node

Mesh Distributed Election-based Scheduling

- **Distributed Election Algorithm**
  - For each CandidateXmtOpportunity, until local node's NextXmtTime is found
  - Determine set of eligible competing nodes
    - Initially, this will be all nodes within extended neighborhood
    - As {Node ID, NextXmtTime, XmtHoldoffTime} sets are learned, the eligible node list for any given transmit opportunity is reduced (detailed on next page).
  - For each eligible competing node, compute a pseudorandom MIX

    MIX(i) = {CandXmtOpportunityNum, Node ID of Ext. Neighbor Node(i)}
  - Compute the same MIX for the local node

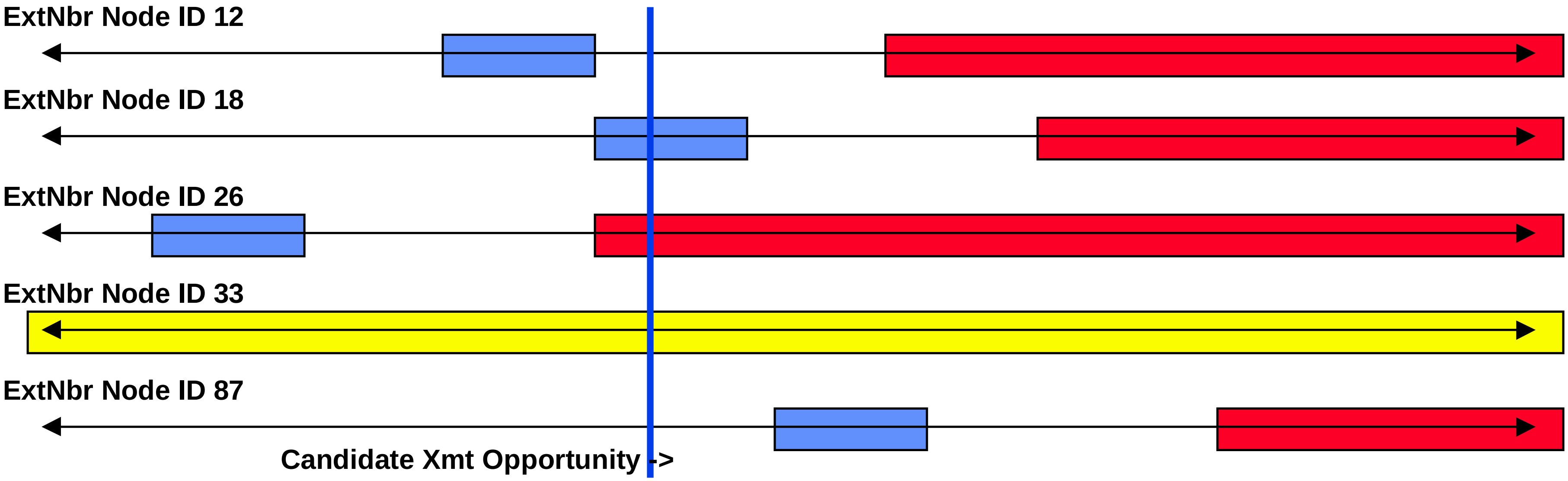
    MIX_local {CandXmtOpportunityNum, Local Node ID}
  - If MIX_local > MIX(i) for all eligible competing nodes for this candidate transmit opportunity, then
    - NextXmtTime for the local node is set to CandXmtOpportunityNum



NOKIA

Mesh Distributed Election-based Scheduling

- **Distributed Election Algorithm – Determining Eligible Competing Nodes**
  - For a given CandidateXmtOpportunity, the eligible competing nodes are all nodes within the local node's extended neighborhood for which:
    - The **NextXmtTime** interval includes the **CandidateXmtOpportunity**, or
    - The **EarliestSubsequentXmtTime** (equal to **NextXmtTime** + **XmtHoldoffTime**) is ≤ the **CandidateXmtOpportunity**
    - The **NextXmtTime** is not known.
  - Illustrated in figure below:
    - **Blue** rectangles indicate eligibility due to the **NextXmtTime** interval
    - **Red** rectangles indicate eligibility due to the **EarliestSubsequentXmtTime**
    - **Yellow** rectangles indicate eligibility due to lack of NextXmtTime info
    - For the example CandidateTransmitOpportunity, only nodes 18, 26, and 33 are eligible.



NOKIA

NetEntry Packet

- Mesh sub-header
  - Set to 0x0000 until node ID assigned
- Generic Header
  - Network ID = sponsor's network if known,
    = 0x00 if not known.

Syntax	Size
MSH-NENT_Message_Format() {	
Management Message Type = 43	8 bits
Type	3 bits
Xmt counter for this Type	3 bits
Reserved	2 bits
Sponsor Node ID	16 bits
Xmt Power	4 bits
Xmt Antenna	3 bits
Reserved	1 bits
if (Type == 0x2)	
MSH-NENT_Request_IE()	176 bits
}	

- 0x0 reserved, 0x1 NetEntryAck
- 0x2 NetEntryRequest, 0x3 NetEntryClose

- For Ack packets: acked sequence

Syntax	Size
MSH-NENT_Request_IE() {	
MAC Address	48 bits
OpConfInfo	64 bits
Operator Authentication Value	32 bits
Node serial Number	32 bits
}	

- Bottom 32 bits of HMAC[MacAddress | OpConfInfo | OpNetSecret], 0 if not attempting pre-authorization

NOKIA

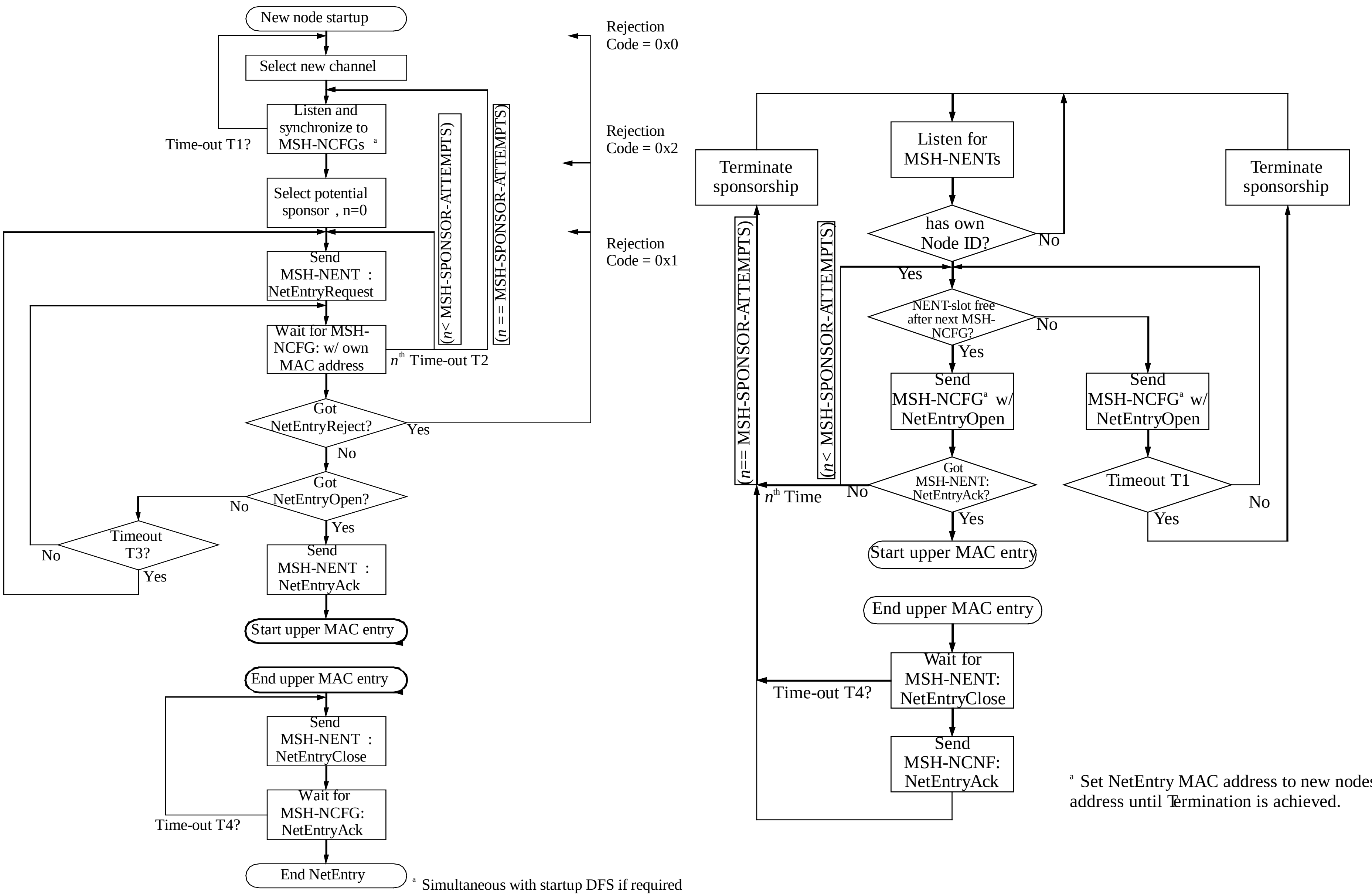
Network Entry – New Node Process

- **New node listens for MSH-NCFG with Network Descriptor**
  - **Learns network operational parameters**
  - **Chooses sponsor node**
  - **Does coarse synchronization**
  - **Does initial DFS**
- **New node sends MSH-NENT:Request to sponsor**
  - **Requests sponsor's attention**
  - **Includes provider configuration data and (optional) authentication code**
- **New node receives MSH-NCFG:NetEntryOpen from sponsor**
  - **New node's MAC address is advertised as being sponsored**
  - **New node can do fine synchronization**
  - **Message provides initial schedule**
- **New node sends MSH-NENT:Ack**

Perform higher-layer DHCP configuration & authentication (including Node ID and IP address assignment, upgrades & provider config files)

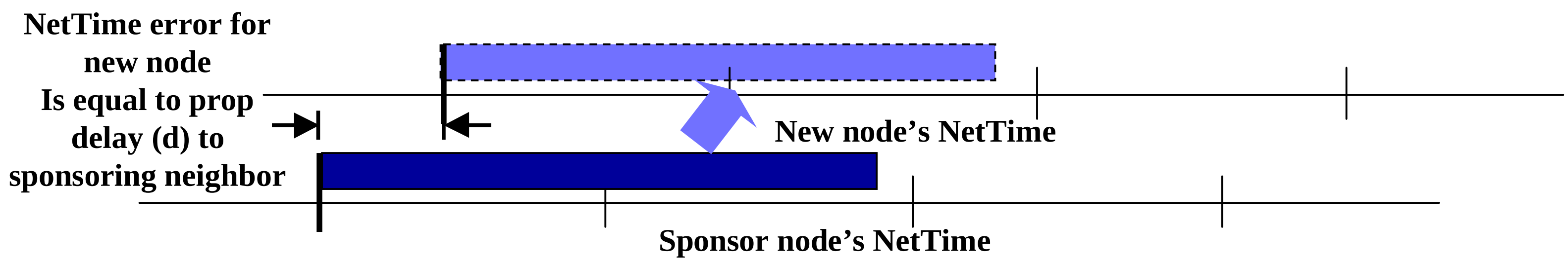
- **New node sends MSH-NENT:Close**
- **Sponsor node sends MSH-NCFG:Ack**

Network Entry flow diagrams

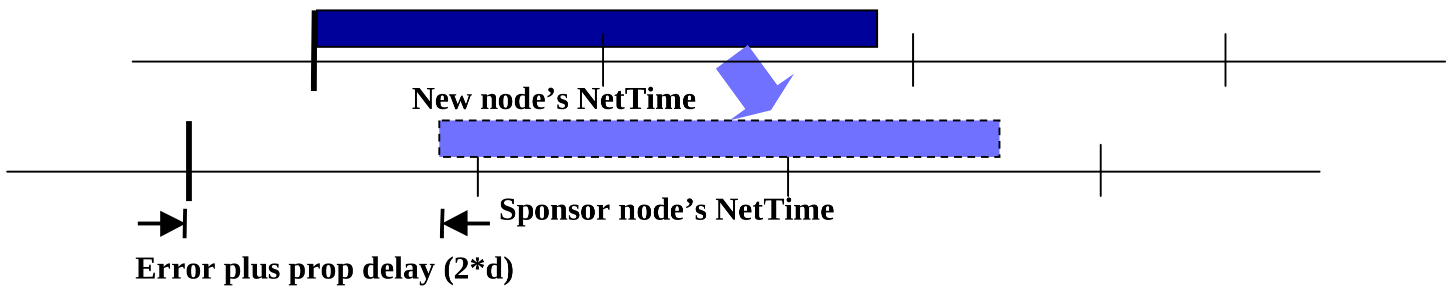


Net Time Synchronization (ranging) - I

- **New node acquires first order sync from MSH-NCFG packet received from "sponsor" node**
  - NetTime clock-rate acquired from first received packet
  - NetTime initialized to transmit time in first received pkt

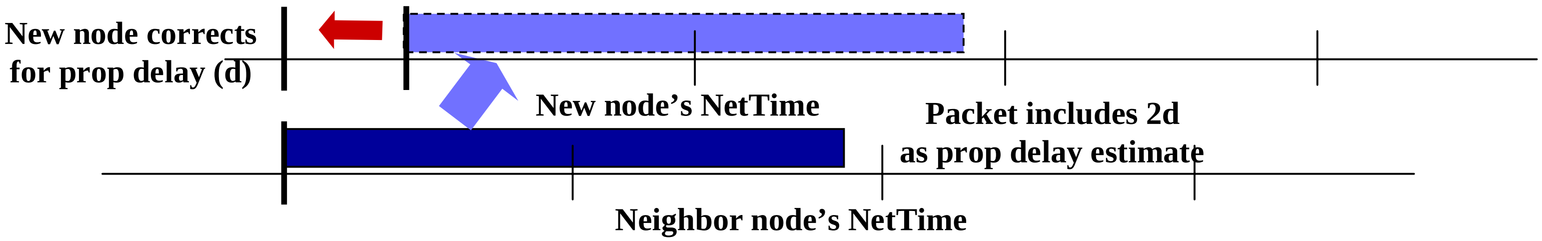


- **Timing of new node's MSH-NENT is off by error d.**
  - When received by same sponsoring neighbor, then error will equal prop delay (d), so packet will be received at an offset equal to 2*d



- **Sponsor includes the perceived prop. delay (2*d) in MSH-NCFG:NetEntryOpen**
- **NetTime correction made when new node receives packet from the sponsor node.**
  - Makes a negative correction to NetTime by half the 'prop delay' reported in the packet
- **New node now fully synchronized** (as long as prop delay correction wasn't max value of 0xF – if so, repeat initialization loop.)

Net Time Synchronization (ranging) - II



- **MSH-NCFG packets will still include prop delay estimates in Nbr-Physical-IE to detect and correct any drift**
- When prop delay reported in each direction is different, then the two nodes' NetTimes have drifted apart.
  - Node with higher **Synchronization hop count** corrects time by half the difference
- **Maintaining synchronization throughout network.**
  - Nodes (typically BSs) may be assigned as "master" time keepers.
  - At least one node in each region should be assigned as a master.
  - If multiple nodes are assigned as masters, they must use an external, <u>implementation-dependent</u> method to synchronize between themselves.
  - Master time keeper(s) broadcast the frame number/transmit opportunity (time) and broadcast a "0" for the synchronization hop count .
- **All other nodes (including non-master BS's) adjust their time according to that broadcast by their direct (1-hop) physical neighbor(s) as follows:**
  - Neighbor with lower synchronization hop count takes precedence, other node adjusts both time and clock rate to match.
  - If sync hop counts are equal, then lower Node ID takes precedence.
  - Nodes broadcast a sync hop count of one plus the lowest hop count being broadcast by its physical neighbors.

Link & Neighbor Management

Mesh Neighbor/Link Selection

- **Overview of process**
  - 1. Select candidate link(s)
    - Mandatory
    - Based on simple determination of link qualities & transmit energy “distance” to BS/AirHeads
    - Uses the physical neighbor entries in the MSH-NCFG packets
  - 2. Verify provider pre-authorization & communicate link IDs
    - Mandatory (although pre-authorization is optional according to provider’s policy)
    - Uses embedded data in MSH-NCFG packets
    - Upon completion, distributed MAC scheduling for this link is enabled
  - 3. Authenticate node & communicate transmit encryption keys
    - Optional
    - Uses distributed MAC scheduling
      (Public certificates too large for embedded MSH-NCFG pkt, and unauthenticated links are not used in routing protocol due to lack of trust, so centralized scheduling can’t be used [furthermore, the link may not be in the current BS/AirHead scheduling tree].)

Mesh Provider Authorization & Link IDs

Syntax	Size
MSH-NCFG_embedded_data_IE() {	
Action Code	2 bits
Reserved	6 bits
if (Action Code == 0x0 or 0x1)	
Nbr Authentication value	32 bits
if (Action Code == 0x1 or 0x2)	
Link ID	8 bits
}	

- **0x0 = challenge, 0x1 = challenge response, 0x2 = Accept, 0x3 = Reject**

- **Purpose**
  - Mutual low-overhead initial authorization verification
  - link ID setup and exchange
- **Authorization key:**
  - Known to all nodes through pre-configuration or network entry (encrypted by node's public key during that process)
- **Link IDs**:
  - Selected randomly, among the Ids still available for the node.

- **Challenge/Response protocol**
  - Node A sends
    - Challenge
    - HMAC{Authorization Key, frame number, Node ID A, Node ID B}
  - Node B sends:
    - Challenge-reponse
    - HMAC{Authorization Key, frame number, Node ID B, Node ID A}
    - Node B's link ID for this link
  - Node A sends:
    - Accept flag
    - Node A's link ID for this link

NOKIA

Data Scheduling
For minislots in data subframe

Scheduling Overview

- Scheduling is performed by negotiating minislot ranges and associated channels within the data subframe
- Schedule is adaptive, based on the traffic demand for each link
- There are 3 scheduling mechanisms:
  - Coordinated centralized scheduling:
    - Coordinated: Uses scheduling packets transmitted in a collision-free way within scheduling control subframes
    - Centralized: coordinated by the mesh BS
    - Usage: Best for links supporting persistent traffic streams
  - Coordinated distributed scheduling:
    - Coordinated: Uses scheduling packets transmitted in a collision-free way within scheduling control subframes
    - Distributed: Uses the same distributed scheduling algorithm used for MSH-NCFG packets, (substituting in MSH-DSCH xmt opportunities).
  - Uncoordinated distributed scheduling:
    - Uncoordinated: performed in a partially, contention-based manner while avoiding any conflicts with the schedules established using the coordinated methods.
    - Distributed: Opportunity based scheduling between two nodes.
    - Usage: Best for scheduling over links with occasional or brief traffic needs.

NOKIA

Centralized Scheduling

Centralized Scheduling Config (MSH-CSCF) Packet

Syntax	Size
MSH-CSCH_Message_Format() {	
Management Message Type = 41	8 bits
Configuration sequence number	3 bits
Reserved	1 bit
NumberOfChannels	4 bits
for (i=0; i< NumberOfChannels; ++i) {	
Channel index	4 bits
}	
Padding Nibble	0 or 4 bits
NumberOfNodes	8 bits
for (i=0; i< NumberOfNodes; ++i) {	
NodeID	16 bits
NumOfChildren	8 bits
for (j=0; j< NumberOfChildren; ++j) {	
Child Index	8 bits
Upstream Burst Profile	4 bits
Downstream Burst Profile	4 bits
}	
}	
}	

- **Channels available for centralized scheduling**
- **4 bits Channel Identifiers**
- **indicates “index” of node**
- **Nodes are not allowed to participate in the centralized scheduling algorithm until they have received the current MSH-CSCF message containing their own Node ID.**
- **List specifies:**
  - **The bi-directional links in the current scheduling tree to & from the BS/AirHead.**
  - **Nodes ordered according to how their centralized control messages are scheduled.**

NOKIA

Centralized Scheduling Config (MSH-CSCH) Packet

Syntax	Size
MSH-CSCH_Message_Format() {	
Management Message Type = 40	8 bits
Configuration sequence number	3 bits
Grant / Request Flag	1 bits
Flow Scale Exponent	4 bits
Frame schedule Flag	1 bits
NumFlowEntries	8 bits
for (i=0; i< NumFlowEntries; ++i) {	
UpstreamFlow	4 bits
DownstreamFlow	4 bits
}	
}	

- **last MSH-CSCF sequence value**
- **[0 – Grant; 1 – Request]**
- **For downstream, this gives the absolute values of flow granted, so the total minislot range allowed for centralized scheduling need not be used if not needed, with the remainder set aside for distributed scheduling.**
- **For upstream, the lowest exponent possible is used at each hop, with quantization of forwarded requests rounded up (e.g., avoids reducing any requests to zero).**
- **[0 – sched. over single frame; 1 – schedule over two frames]**
- **ordered by MSH-CSCF**
- **Flow values computed as: 2**(FlowScaleExponent + 14) * <flow value>**

NOKIA

Centralized Scheduling
Control transmit order

- **Downstream MSH-CSCF or MSH-CSCH messages use the following transmission ordering (upstream MSH-CSCH uses reverse order):**
  - The mesh BS transmits first in a new frame.
  - Then, the eligible children of the mesh BS (i.e., nodes with hop count equal 1) in the most recent MSH-CSCF packet transmit next, ordered by their appearance in the MSH-CSCF packet, transmit.
  - Then, the eligible children of the nodes from step 2 (i.e., nodes with hop count equal 2), also ordered by their appearance in the MSH-CSCF packet, transmit.
  - …continue until all eligible nodes in the routing tree have transmitted.
  - Nodes shall fragment their message if it does not fit entirely before the end of the control subframe and at least the preamble and one data symbol fit.
  - All nodes are eligible to transmit the grant schedule, except those that have no children.
  - If a node's order requires it to transmit immediately after receiving, a delay of **MinCSForwardingDelay** is inserted.
- **For transmitting MSH-CSCH grant messages, all nodes with children are eligible.**
- **For transmitting MSH-CSCH request messages, all nodes, except the mesh BS are eligible.**
- **The transmission schedule for MSH-CSCF and MSH-CSCH messages is computed in a distributed fashion.**
  - All MSH-CSCH:Grant messages contain information about the entire AirHood (since all nodes need complete information for the schedule computation).

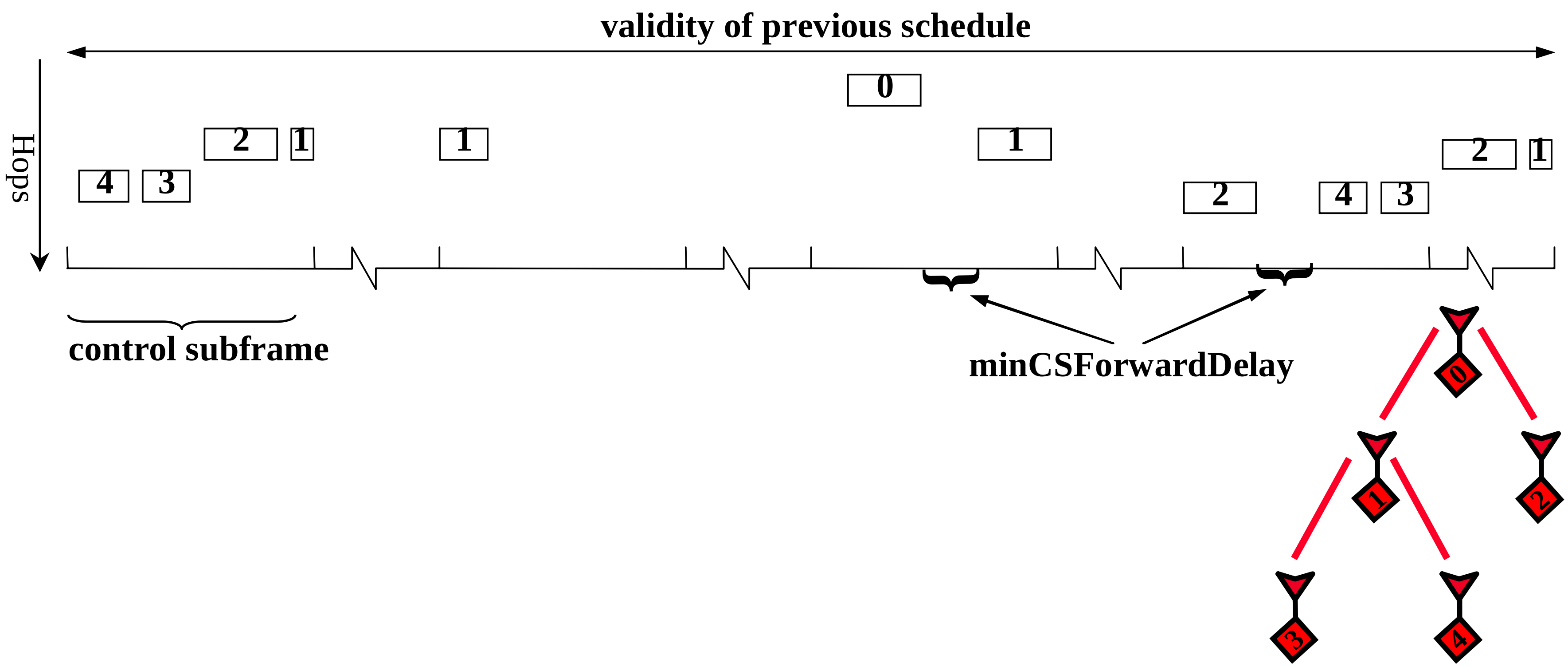
NOKIA

Centralized Scheduling Control

- Upon receiving any message in the current scheduling sequence, and assuming the node has up-to-date scheduling configuration information, a node will be able to predict all of the AirHood centralized scheduling transmissions, including its own.
- Besides the BS/AirHead, a node should not transmit any downstream centralized scheduling packet in a centralized scheduling sequence in which it hasn't yet received a MSH-CSCH message from a parent. A node should not sent any centralized scheduling packets if its MSH-CSCF information is outdated.
- **Implementation-dependent:** If a node fails to receive an upstream control packet from one or more of its children:
  - It may transmit its MSH-CSCH:Request message by reusing recently reported demand requests(s) from these nodes.
  - If this failure continues, the node should "age" these demands to reduce them to zero over time.
- **MSH-CSCH:Request messages**
  - Each node estimates and reports the level of its own upstream and downstream traffic demand to its parent, along with the demands reported by its children.
  - A node with neighbors not included in the scheduling tree may report the neighbors needs as its own.
  - The flow exponent used for each upstream transmission of the MSH-CSCH:Request message is set to the lowest exponent possible which still covers the value of the highest flow request. Any further flow request quantization required as a result is rounded up.
- **MSH-CSCH:Grant message**
  - The BS/AirHead determines the levels of flows to grant to each node in the AirHood, and issues a MSH-CSCH:Grant message, which then propagates down the tree.

Centralized Scheduling
Schedule validity

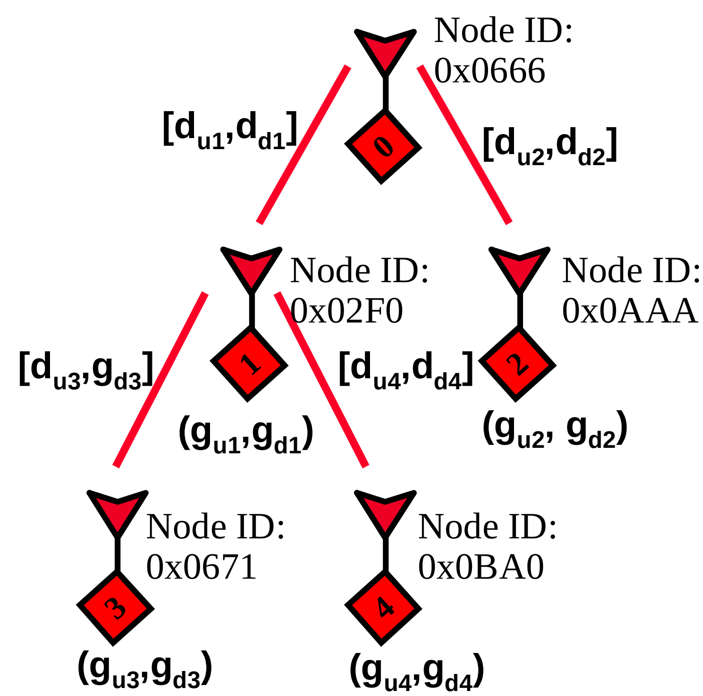
- From the information contained in the MSH-CSCF and MSH-CSCH packets, each node can compute the validity of the latest schedule.
- The time between the first frame in which a node sends the request schedule and the last frame in which a node receives the new grant schedule marks the validity of the previous grant schedule.



Centralized Scheduling
Schedule Computation – Example 1

- **5-node network (BS/AirHead plus 4 subscriber nodes)**
- **Single channel, no reuse within AirHood**

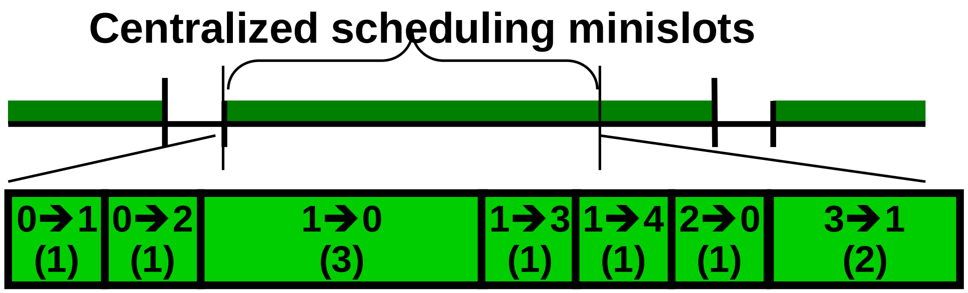
i	Node ID	Child Indexes
0	0x0666	$\{1 : [r_{u1}, r_{d1}], 2 : [r_{u2}, r_{d2}]\}$
1	0x02F0	$\{3 : [r_{u3}, r_{d3}], 4 : [r_{u4}, r_{d4}]\}$
2	0x0AAA	$\{\}$
3	0x0671	$\{\}$
4	0x0BA0	$\{\}$



MSH-CSCH Flowlist	Upstream link fraction from	Downstream link fraction to
1: (g_{u1}, g_{d1})	1: $(g_{u1}+g_{u3}+g_{u4})/r_{u1}$	1: $(g_{d1}+g_{d3}+g_{d4})/r_{d1}$
2: (g_{u2}, g_{d2})	2: g_{u2}/r_{u2}	2: g_{d2}/r_{d2}
3: (g_{u3}, g_{d3})	3: g_{u3}/r_{u3}	3: g_{d3}/r_{d3}
4: (g_{u4}, g_{d4})	4: g_{u4}/r_{u4}	4: g_{d4}/r_{d4}

Centralized Scheduling
Schedule Computation – Example 1

- **Split up time according to link time fractions within slots dedicated for centralized scheduling**
  - Group together all transmissions by each node Ordered by listing in CSCF, with AirHead 1st

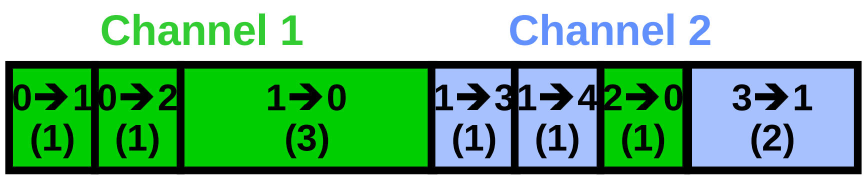


- **Assign actual minislot ranges**
  - Insert guard times
  - Decrease time fractions proportionately if sum > 100% while minimum allocation at least preamble+ 1 data OFDM symbol.
  - If still over 100%, round up to 200%. The schedule then spans two frames (**FrameScheduleFlag**=1)

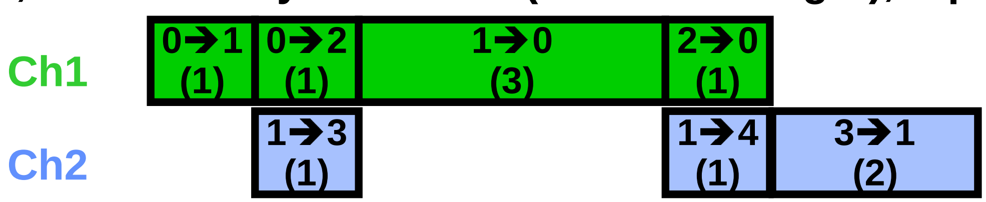
Not in draft or proposal (yet).

Centralized Scheduling Schedule Computation – Example 2

- **Same as Example 1 but with 2 channels**
- **Split up time in same manner as for single channel case**
- **Assign channels to links according to the number of hops to the AirHead**
  - Except that if channel reuse is permitted, first assign channel 1 to links according to hops from AirHead that can reuse, then channel 2, etc.



- **For time allocations with the same channel:**
  - If they are separated by the required hop count separation, treat as different channels
  - Otherwise, append together in time
- **Then, allocation by allocation (from left to right), reposition as far left as possible.**



- **Assign actual minislot ranges as in Example 1**

Centralized Scheduling Schedule Computation

Not in draft or proposal (yet).

- **De-centralized computation**
  - The actual schedule is computed in a decentralized manner, where all nodes use the following inputs to compute the new schedule:
    - Node IDs included in the schedule, along with their ordering by MSH-CSCF.
    - Schedule tree links and data rates from MSH-CSCF.
    - List of available channels from the MSH-CSCF packet.
    - Assignments from MSH-CSCH:Grant message.
- **First, the demands are accumulated on the links of the routing tree**
  - down_demand(i,j) = total_down_demand(j)
    total_down_demand(i) = granted_down_demand(i) +
    SUM{over all downstream nbrs j} [down_demand(i,j)]
  - up_demand(i,j) = total_up_demand(j)
    total_up_demand(i) = granted_up_demand(i) +
    SUM{over all downstream nbrs j} [up_demand(i,j)]
- **Each node then runs the scheduling computation algorithm to assign non-conflicting minislot ranges and channels to the links among the available CentralizedSchedulingSlots and channels to the links, using the following principles:**
  - All grants are assigned slots proportionately
    - But not all CentralizedSchedulingSlots will be used if the requests are below this capacity.
  - Channels are assigned by the distance from the BS
  - Ordering is per the ordering in the MSH-CSCF packet.

NOKIA

Distributed Scheduling

Distributed Scheduling (MSH-DSCH) Packet Format I

Syntax	Size
MSH-DSCH_Message_Format() {	
Management Message Type =39	8 bits
Coordination Flag	1 bits
Grant/Request Flag	1 bits
Sequence counter	6 bits
No. Requests	4 bits
No. Availabilities	4 bits
No. Grants	6 bits
Reserved	2 bits
if (Coordination Flag == 0) {	
MSH-DSCH_Scheduling_IE()	variable
}	
for (i=0; i< No_Requests; ++i) {	
MSH-DSCH_Request_IE()	16 bits
}	
Padding nibble	0 or 4 bits
for (i=0; i< No_Availabities; ++i) {	
MSH-DSCH_Availability_IE()	32 bits
}	
for (i=0; i< No_Grants; ++i) {	
MSH-DSCH_Grant_IE()	40 bits
}	
}	

- **Usage: 0 – Uncoordinated; 1 – Coordinated**
- **Usage:**
  - **0 – coordinated distributed scheduling**
  - **0 – Uncoordinated "Request" message**
  - **1 – Uncoordinated "Grant" message**
- **Counter**
  - **Independent counters are used for the coordinated & uncoordinated messages.**
  - **The uncoordinated MSH-DSCH:Grant messages use the count from the associated MSH-DSCH:Request.**

Distributed Scheduling Packet Format II

Syntax	Size
MSH-DSCH_Scheduling_IE() {	
Next Xmt Mx	5 bits
Xmt holdoff exponent	3 bits
No. SchedEntries	8 bits
for (i=0; i< No_SchedEntries; ++i) {	
Neighbor Node ID	16 bits
Neighbor Next Xmt Mx	5 bits
Neighbor Xmt holdoff exponent	3 bits
}	
}	

- **Coordinated DSCH use only**
- **Note that the number of hops to this (extended) neighbor etc.. can be found in the Physical Neighbor Table updated by the NetConfig packets.**

Syntax	Size
MSH-DSCH_Request_IE() {	
Link ID	8 bits
Demand Level	8 bits
Persistence	3 bits
Reserved	1 bit
}	

- **of the node transmitting this packet**
- **in minislots (under current burst profile)**
- **Usage:**

0: cancel	**4: 8 frames**
1: 1 frame	**5: 32 frames**
2: 2 frames	**6: 128 frames**
3: 4 frames	**7: infinite till cancel**

Distributed Scheduling Packet Format III

Syntax	Size
MSH-DSCH_Availability_IE() {	
Start Frame number	8 bits
Minislot start	8 bits
Minislot range	7 bits
Direction	2 bit
Persistence	3 bits
Channel	4 bits
}	

- The LSB of the indicated frame number
- (≥ 2 entries are needed to cover the entire data subframe)
- Usage
  - 0 = Minislot range is not available
  - 1 = This node is available for transmissions in this minislot range
  - 2 = This node is available available for receptions in this minislot range
  - 3 = Available for either transmission or reception

Syntax	Size
MSH-DSCH_Grants_IE() {	
Link ID	8 bits
Start Frame number	8 bits
Minislot start	8 bits
Minislot range	8 bits
Direction	1 bit
Persistence	3 bits
Channel	4 bits
}	

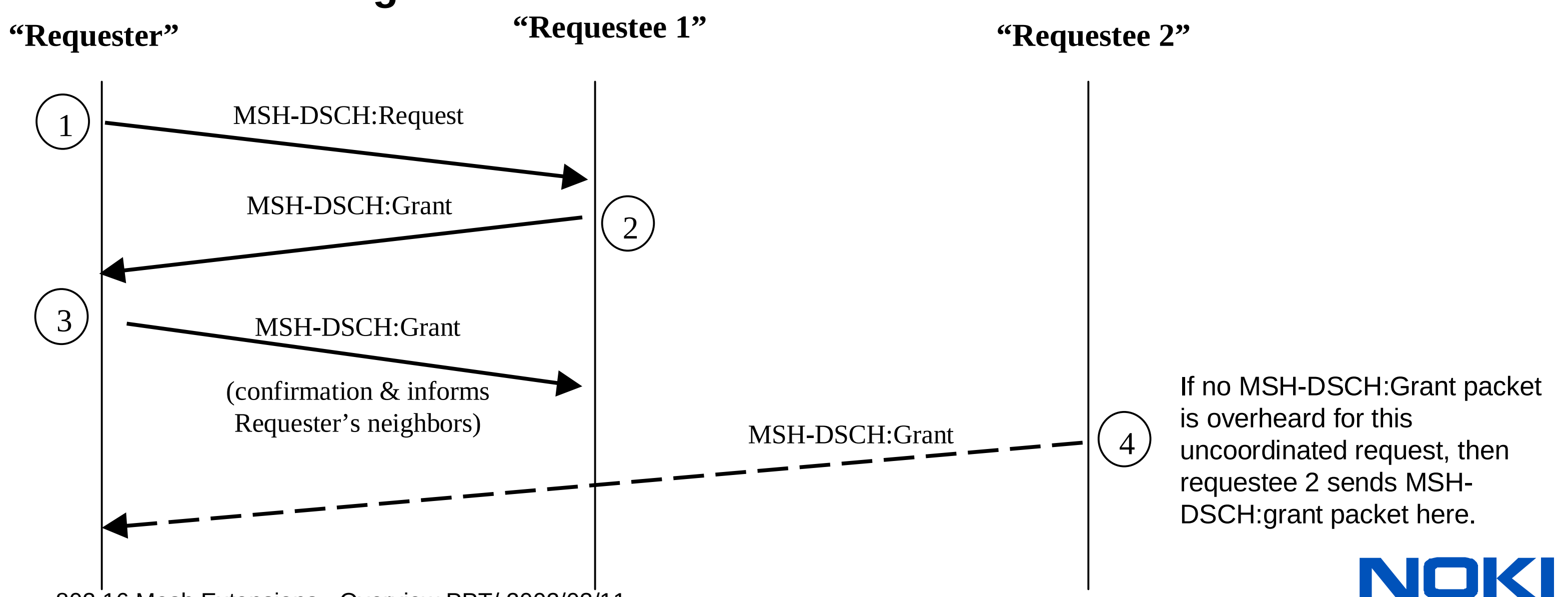
- ID (of the node transmitting this packet)
- The LSB of the indicated frame number
- Usage:
  - 0 = This node receives from this neighbor
  - 1 = This node transmits to this neighbor

(Coordinated) Distributed Scheduling

- **Implementation-dependent: Method for determining when and where to grant requests.**
- **Only permitted to schedule slot ranges in the data subframe from among those minislots not included by CentralizedSchedulingSlots (reported in the NetConfig:NetDescriptor).**
- **The relevant "confirmation Grant" must be transmitted/received before a node can start transmitting into a negotiated slot range.**
  - I.e., it's not enough to Grant a Request (or receive the Grant for a Request) to start using it.
  - The confirming Grant must be first transmitted before either node can start using the new slot ranges.

Uncoordinated Distributed Scheduling I

- **Used for fast setup of a new, temporary slot range “bursts” between a pair of neighboring nodes. Can be useful to handle:**
  - Transient traffic requirements that are in excess of what can be handled by existing schedule.
  - Traffic over links which are not included in the current centralized and/or coordinated distributed schedule.
  - Traffic during network initialization or after network changes.
- **Unlike all other MAC control packets, Uncoordinated scheduling control transmissions are sent during the data subframe.**
  - Uncoordinated scheduling transmissions shall not conflict with the existing schedule.
  - Uncoordinated MSH-DSCH packets are always transmitted on the “base” channel.
- **Uncoordinated scheduling requests use the handshake illustrated below to establish new slot ranges:**



Uncoordinated Distributed Scheduling II

- **MSH-DSCH : Request**
  - **Transmission is scheduled using a random-access algorithm among the "idle" slots of the current schedule**
    - "Idleness" is according to the requester's view of the schedule throughout its extended neighborhood.
  - **Random backoff used for scheduling after an unsuccessful attempt, or after completion of a nearby "burst."**
  - **Lists one or more neighbor nodes being solicited, prioritized by their ordering in the MSH-DSCH:Request entries.**
    - The "demand" and "persistence" fields indicate the level of traffic demand this node has for this neighbor, or if demand is set to 0, then this neighbor is only included in the list due to excess traffic demand reported previously by the neighbor for this node.
  - **Lists the "idle" slots (in Availability IE) to be used both for:**
    - Scheduling the immediately following MSH-DSCH:Grant transmissions (all on the base channel), and
    - The total candidate minislots for the slot range to be negotiated (any channel).

Uncoordinated Distributed Scheduling III

- **MSH-DSCH : Grant (from requestee)**
  - Each neighboring node listed in the MSH-DSCH:Request may issue a grant.
    - The first requestee node can start its Grant transmission in the immediately following base-channel idle minislot as listed in the MSH-DSCH:Request.
    - The $(n-1)^{th}$ requestee can attempt its Grant transmission n * ‘grant transmission duration’ idle minislots later.
    - Requestee must remain silent if transmitting a Grant would cause a collision in its neighborhood.
  - Node determines jointly available minislots from MSH-DSCH:Availabilities and its own schedule
  - Node may add reverse traffic to its grant.
  - The MSH-DSCH:Grant message can include one or two sets of grant entries to indicate the slot ranges used for both:
    - the “forward” portion of the burst (from the requester to the requestee), &
    - The “reverse” portion of the burst (from the requestee to the requester).

Uncoordinated Distributed Scheduling IV

- **MSH-DSCH : Grant (confirmation from requester)**
  - The MSH-DSCH:Grant from the requester simply repeats the two Grant entries listed in the MSH-DSCH:Grant from the requestee, for the benefit of the requester's neighbors.
  - The Grant confirmation is sent in the first available minislots following the minislots reserved for the Grant opportunity of the last potential requestee.
- **Valid Period for negotiated slot-range**
  - The number of frames that the negotiated slot-range (burst) is valid is according to what was reported in the persistence field of the MSH-DSCH:Grant packets.
  - However, if coordinated scheduling is in effect, then regardless of the persistence field in the slot-range (burst), any minislots within the agreed slot range that are in conflict with a new schedule must be terminated.

Basic Capability Negotiation

- **Capability negotiation between Sponsor and Candidate exactly as in PMP (SBC-REQ, SBC-RSP)**

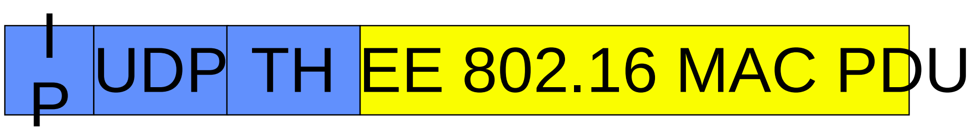
- **The other neighboring nodes learn the PHY capabilities via the MSH-NCFG protocol**

Mesh initialization

- **Protocol peers may be separated by multiple hops!**
- **The MAC layer forwarding defined**
- **Security issues not present in PMP e.g. “Man in the middle”**
- **Still same need to prevent theft of service, cloned nodes etc.**

MAC PDU tunneling over MESH

- **MAC PDUs are tunneled over UDP**
- **Use a well known UDP port (to be registered with IANA)**
- **Introduce a 1 byte tunnel header after the UDP header**



TH=Tunnel header

Authorization

- **Authorization:**
  - Candidate sends and receives MAC messages (PKM-REQ, PKM-RSP)
  - Sponsor tunnels the MAC messages to the Authorization node over UDP
  - Sponsor de-tunnels the replies from the authority and sends PKM-RSP messages (Auth Reply, Auth Reject) to the Candidate
  - The Sponsor has learned the IP address of the node performing authorization from the configuration file

Authorization messages

- **Message contents:**
  - PKM-REQ:Auth Info : X.509 certificate of CA issuing terminal certificate
  - PKM-REQ:Auth Req : X.509 certificate, security related capabilities
  - PKM RSP:Auth Reply : AK encrypted with public key+lifetime and sequence number, list of Security associations and their parameters, **operator shared secret**
  - Operator Shared Secret = 128 bit long key known to all authorized nodes in a mesh network
- **Main use of the Operator Shared Secret is to authenticate message exchanges between nodes**
- **Maintained as the AK**

Initial Transmission Key exchange

- Candidate sends PKM-REQ:Key Request to Sponsor including Node Certificate authenticated using the operator shared secret
- Upon reception Sponsor generates TEK
- Sponsor replies with PKM-RSP:Key Response containing the transmission key encrypted with the Candidate nodes public key and a message digest over the key reply calculated with the operator shared secret
- Halfway through the key lifetime a new key exchange is performed. Transition between keys exactly as in base protocol
- Transmission rules as in PMP with the node that generated the key using the BS rules and the peer using the SS rules
- Security association IDs unique per link

- TEKs with the other neighbors are established the same way once the node is operational

Registration

- **The Candidate node sends a REG-REQ to the Sponsor with the following parameters**
  - Node capabilities including IP version support
  - Version
  - HMAC digest
- **The sponsor replies with a REG-RSP containing**
  - Node ID
  - Status (the parameter is called Response)
  - IP version
- **The Sponsor tunnels the REG-REQ message on top of UDP to the node handling registration**
- **The Sponsor de-tunnels the REG-RSP and sends it to the Candidate as is**

Establish IP address

- **Candidate sends DHCP discover**
- **Sponsor relays this to DHCP server**
- **Candidate receives a DHCP offer from the server de-tunneled by the Sponsor**
- **Candidate sends DHCP request and expects a DHCP response**

Exactly as for PMP!

ToD and TFTP

- **Time is established using the Time Protocol (RFC-868) using server returned by DHCP**

- **Address of TFTP server is returned by DHCP**

Setting up service parameters

- The Candidate sends a DSA-REQ to the Sponsor containing optionally the Operator information
- The sponsor tunnels the the DSA-REQ over UDP
- The Sponsor replies with the de-tunneled DSA-RSP setting up the appropriate service parameters
- The Candidate ends the protocol by sending a DSA-ACK to the sponsor that tunnels it back to the originator of the DSA-RSP

Connections in Mesh versus PMP

- **Connections are not explicitly set up in the Mesh as this would swamp the network with signaling traffic**
- **Instead the first byte the CIDs are used to:**
  - Convey priority information
  - Determine if ARQ is enabled or not
  - Determine what is carried in the PDU (management or user data)
- **The second byte determines the receiver of the packet**
- **CIDs 0xXXFF are reserved for Broadcast and always contain MAC Management messages**
- **The SAP primitive is identical**

NOKIA

Mesh Connection ID

Unicast

Type (2)	Rel (1)	Prio (3)	DP (2)	Xmt Link ID (8)

Broadcast

NetID (8)	OxFF (8)

<u>Type</u>:
0x00 MAC management
0x01 IP
0x02-0x03 Reserved

<u>Rel</u>: Packet reliability requirements
<u>DP:</u> Drop precedence
<u>Prio</u>: class/priority
<u>XmtLinkID</u>: Transmitter's Link ID
0xFF is management bcast

<u>NetID</u>: Provider's Network ID
(In broadcast packets only)
0x0000 is reserved for
all-net broadcast.

Connection ID used to indicate:

- **Link to intended neighbor, or broadcast for intended network**
- **Service to be applied**
- **Whenever a new link is created (given by Xmt Link ID), 64 "connections" are automatically created over that link, for the different services, indicated by {Rel, DP, Priority}**

Mesh MAC Sub-header

Xmt Node ID (16)

QoS

- **Both in PMP and Mesh the QoS is done based on the CID!**
- **The scheduling services are different but so is the scheduling and the topology**

Encryption in Mesh

- **User traffic is encrypted**
- **The security associations are over a single hop (no MAC level forwarding!)**
- **Traffic is decrypted and re-encrypted in each node**

- **TEK key exchange using RSA**
- **Traffic mapped to the SAs based on the CID in the MAC header and the Node ID in the Mesh Sub-header**