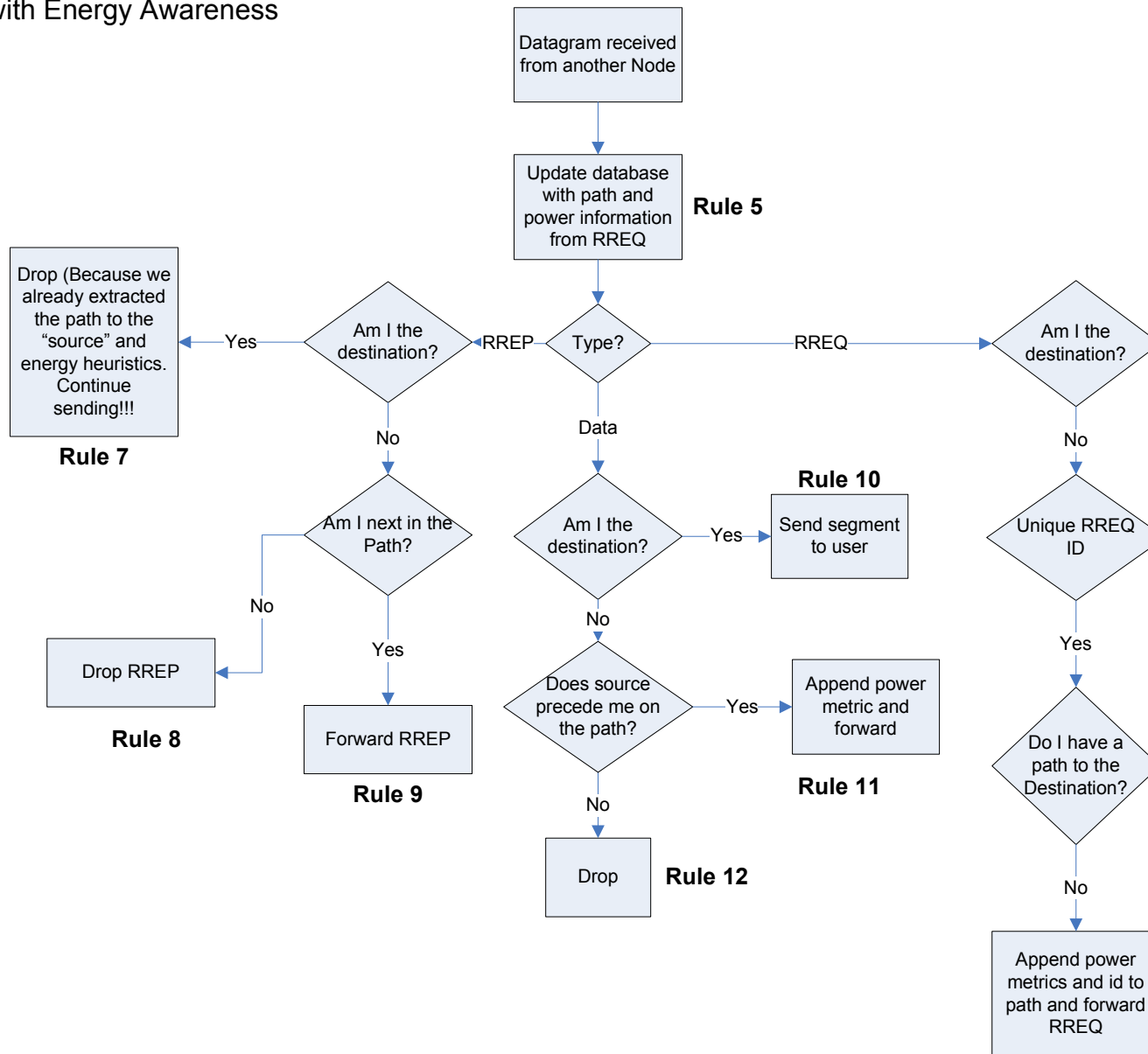


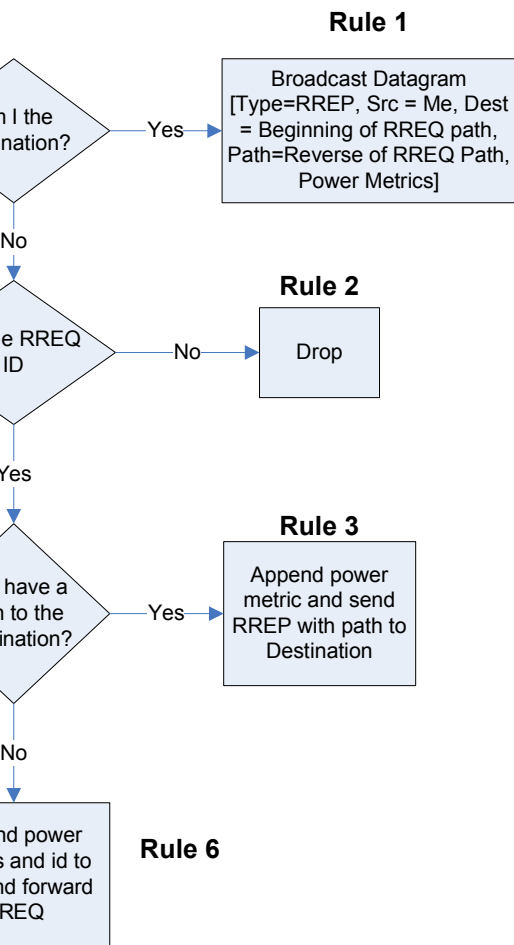
New Segment
(Destination and
Message) from
User

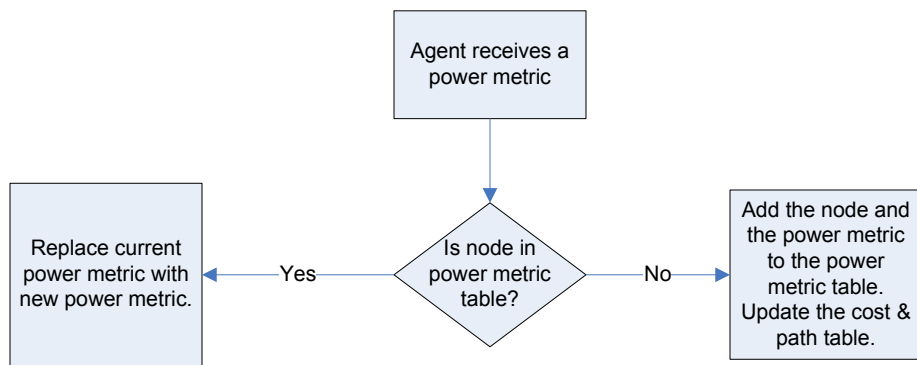


Create datagram
[Type=Data, Src, Dest,
Path, Segment, Power
Metric]
Broadcast Datagram

DSR Protocol with Energy Awareness



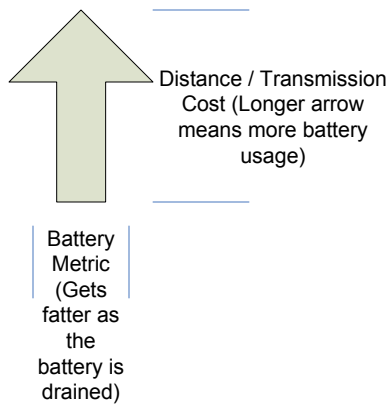




Evaluating Power Metric

Battery	Power Metric
100 - 90%	1
89 – 70%	2
69 – 50%	3
49 – 25%	4
24 – 0%	5

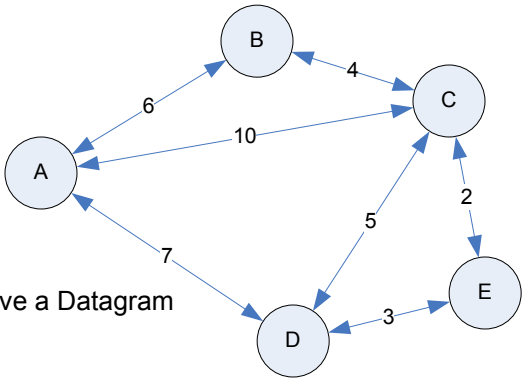
We want to use a system with a Lower Power Metric more often. As a battery on a system gets used up, we want to use it less.



=== Every time we receive a new path, we update the power, cost & path table ===

---- Path Listing Table ---- Potentially updated when we read a path in the datagram
(Unlimited Paths to a Node)

Dest	Path
A	Null (Us)
B	
C	
D	
E	A-D-E: (1*7+4*3=19)
E	A-B-C-E: (1*6+6*4+5*2=30)



---- Battery Metric Table --- Updated when ever we receive a Datagram

Node	Battery Metric
A	1
B	6
C	5
D	4
E	1

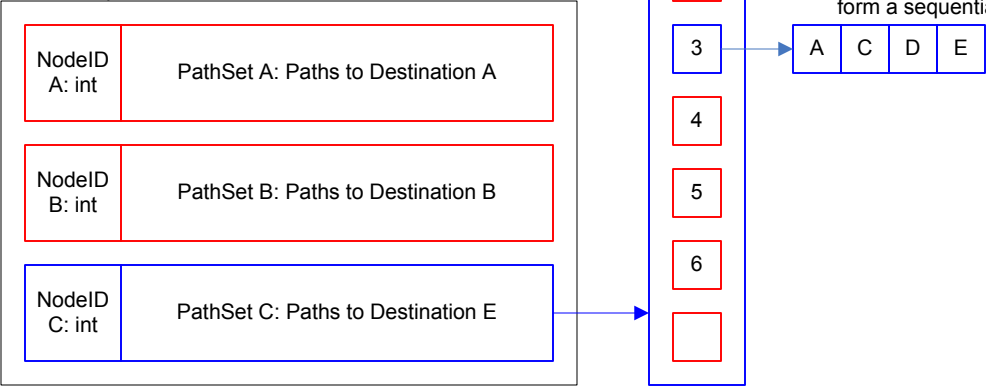
---- Transmission Cost ----
(Based on distance between any two nodes)

Hop	Cost				
	A	B	C	D	E
A	∅	6	10	7	∞
B		∅	4	∞	∞
C			∅	5	2
D				∅	3
E					∅

PathTable <NodeID,PathSet> : Link
a list of paths with a destination

PathSet<Path>:
All Path Options

PathArray<NodeID> : NodeID
form a sequential path



Energy Aware Routing Protocol UML

Network

-List<Nodes> allNodes
-HashMaps<Node,Point> geography

AddNode(): Node
AddNode(Node n, Point p)
Broadcast(Datagram d, EnergyLevel e) – Network decides which nodes can receive the broadcast. Can send a route error if the next “node” on the path cannot receive a message.
Connect(Node n)
Disconnect(Node n)
MoveNode(Node n, Point p)

Datagram

-Type
-Source
-Destination
-Segment
-Route: Path
-BatteryMetricValue(s): Path

getLastPath()
getPowerMetric()
appendPowerMetric()
addIPToPath(Ip addr)
addPowerMetricToPath(IP)

Segment

-Message
-Destination

Message

-Payload

Node

-Agent
-Energy
-NodeID

Node(Energy)
AssignNetwork(Network)
Receive(Datagram)
Send(Datagram)

NodeID

-ID: Number
-HashCode()
-Equals()

Agent

-PathTable paths
-PowerMetric(s)
(Rules In Jess)

Battery

-Capacity: Int
-PowerLevel: Int

BatteryMetric

-Value: Double (Cost)
CalculatePowerMetric(Battery): Double

Path Table

(Jess Template. List of Destinations)
-HashMap(NodeID, DestinationPaths)
-Exists()
AddPath(NodeID dest, Path pt)
RemovePath(NodeID dest, Path pt) “Not used”

DestinationPaths

(Possible Paths to a Destination)
-List<Paths> paths

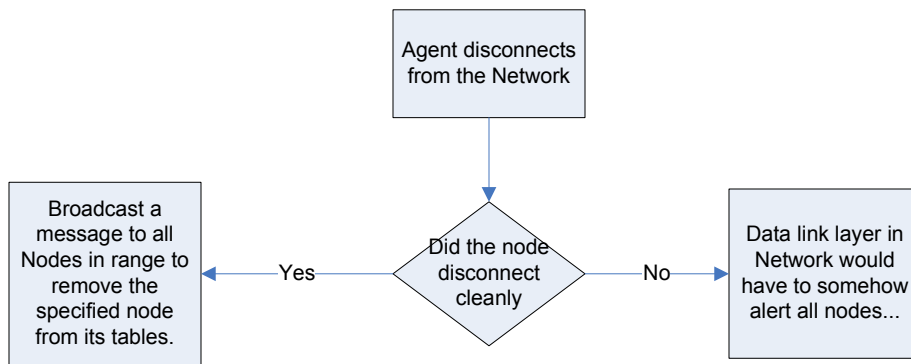
Path

-Cost
-List<NodeID> path

A node connects
to the network



Node and Network
does nothing



Datagram

Type	Source	Destination	Segment	Path	Transmission Cost(s)	Battery Metric(s)
------	--------	-------------	---------	------	-------------------------	----------------------