A survey of QoS support for mobile wireless ad hoc networks

Ignacy Gawędzki[1] and Khaldoun Al Agha[1,2]

[1]LRI, University of Paris XI, F-91405 Orsay, France
[2]INRIA, Rocquencourt, France
Tel +33 (1) 69 15 66 17, Fax +33 (1) 69 15 65 86
{ig, alagha}@lri.fr

April 2003

Abstract

For the last several years, there has been a significant increase of interest in supporting quality of service (QoS) constraints in multi-hop mobile ad hoc networks (MANET). The specificity of MANETs make existing solutions for wireline networks little suitable and a broad range of novel approaches have been studied.

The goal of this paper is to outline the main features of emerging models and whether they can be applied on networks based on the widespread IEEE 802.11 family interface cards.

I. Introduction

Mobile ad hoc networks are distributed systems that comprise a number of mobile nodes connected by wireless links forming arbitrary time-varying network topologies (see Figure 1). Transmission range of each wireless network interface being limited, data flow targeted at a node that is out of reach has to be relayed by an intermediate node. Hence every node functions as a host as well as a router. Physical mobility of the nodes makes the topology change dynamically in an unpredictable fashion and require the use of specifically designed routing protocols. Initially, as reviewed in [1], these solutions provide only best-effort packet delivery and thus don't allow for an effective usage of the already scarce bandwidth.

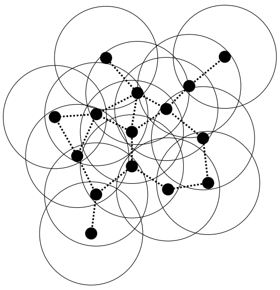


Figure 1: A wireless ad hoc network

Besides, real-time multimedia applications require more strict link quality constraints and/or link quality report. There already exist many solutions designed for wireline networks but inappropriate for MANETs [2], because of their specific features — namely low bandwidth, unpredictable packet loss and transmission delays.

First, in Section II, the general framework of QoS support is presented. In Section III, three existing models are discussed. Sections IV, V, VI, VII, VIII and IX present existing approaches to QoS support on MANETs, respectively what we will call DSDV+, CEDAR, Ticket-Based Probing, INSIGNIA, SWAN and QOLSR. Lastly, we conclude on future possible directions.

II. The big picture

As pointed out in [3], when considering QoS support, there are several aspects that must be taken care of.

First, one must choose a **QoS model**, i.e. the eventual goals one wants to achieve. There are tightly related to the applications that are to take advantage of the QoS support. Real-time audio/video applications require constant bandwidth and little delay, whereas FTP transfers can go with no more than classical best-effort. Moreover, the former can adapt its throughput to available bandwidth and sustain data loss to some extent, while the latter requires a lossless channel. Most of the time, the model must comprise bandwidth and delay constraints.

Second, a **QoS-aware routing protocol** should be used in order to find a suitable multi-hop path given some link constraints supported by the model.

Third, if reservation is to be made — which is the case for most QoS-requiring applications, though a strict reservation is not possibly satisfiable, because of the dynamic topology variations — a **signaling system** has to be chosen to communication between the source node and all the intermediates. Likewise, in case of constraint breakage, the signaling is used by the intermediate nodes either to repair the link — by re-establishing a bypassing path — or to report the failure to the source node. Lastly, when the connection ends, the reservation has to be torn down. Once more, because of the varying topology, the reservation is maintained in a soft state, i.e. it is never definitive and has to be renewed on a regular basis. This is due to the fact that a link failure may cause a partition of the network that would not allow to contact intermediates anymore. Signaling generally come in two flavors: in-band and out-of-band. The first consists of control information piggybacked to regular data packets (e.g. as options in the IP header) and has the advantage of not transmitting additional packets that would content with data packets and potentially waste bandwidth or delay. The second is a special kind of control packet that can be transmitted independently of any data flow.

Fourth, nodes must support some **packet scheduling** (fair queuing) algorithms in order to prioritize some packets at the expense of others as well as shape or police some flows[1].

Optionally, if the MAC layer provides some kind of support for QoS, it can be taken advantage of in upper layers. Support can exist in direct low-level delay respect or bandwidth allocation (e.g. in TDMA-based MAC protocols) or solely in link quality accounting capability.

III. Existing QoS models

A few models exist for the classical wireline networks. Since they are motivated by the same kind of applications as in MANETs, they are worth looking at. In this section, we also present an effort to combine these models into one that would be more suitable for MANETs.

1. IntServ

IntServ, as described in [4], stands for “Integrated Services”. It has a predefined **set of flow types** that aim to satisfy most common requirements. But these types are tunable for specific flows and thus the QoS support has **per-flow granularity**. Resource reservation is supported by the use of RSVP, described in [5].

The design of IntServ has several flaws that make it not suitable for MANETs. First it has a scalability problem that already applies for wireline networks. Since it supports per-flow granularity, each intermediate node must store per-flow state information. Thus the more nodes one has in the network, the more potential flows are to be expected and the more information has to be kept in each node. Hence each node must provide support for RSVP, admission control, packet classification and scheduling. This problem is even more critical in MANETs given that the nodes are comprised of mobile systems with limited memory and processing power. Second, the use of RSVP for signaling has been shown to produce too much control overhead in the case of small time-varying network capacity.

2. DiffServ

DiffServ, described in [6], stands for “Differentiated Services” and come as an answer to the scalability problem of IntServ. It defines a limited **set of flow types** as its predecessor and maintains only **per-class granularity**. Each routing node discriminates incoming packets using the DSCP flags in the IP header. This way, there is no need to maintain flow-specific information and the solution is far more scalable.

The problem is that among the service classes offered by DiffServ, only a fraction can be proposed in MANETs because of the changing quality of links. Furthermore, the use of RSVP signaling still does not suit well rapidly changing network conditions of MANETs.

3. FQMM

FQMM stands for “Flexible QoS Model for MANETs” [7] and is an attempt to build a model explicitly for MANETs. It can be seen as a mix of IntServ and DiffServ in the way that it supports both **per-flow and per-class granularity**. Flows with borderline QoS requirements are granted per-flow processing while the others are aggregated in classes. As in DiffServ, there are three types of nodes: the sender (ingress node), the routers (interior nodes) and the receiver (egress node). The ingress node is required to police its outgoing traffic

[1] Shaping is packet delaying while policing is packet dropping, in case of excess of nominal bandwidth.

to meet a given traffic profile (currently only bandwidth requirements have been considered). Traffic profiles are expressed in terms of percentage of available bandwidth, to cope with changing link conditions. The routing algorithm is left unspecified and is assumed to be of multi-path kind (i.e. all routes to a target are known allowing for multiple choices with respect to some bandwidth requirements), though the use of some QoS-aware routing protocol is encouraged.

Many aspects remain unclear, namely the criterion for the choice between per-flow or per-class treatment of a given flow and to what extent the model applies to rapidly changing conditions (it is assumed in the initial paper, that conditions in MANETs remain stable over a long time-scale).

IV. DSDV+

In [8], Lin and Liu present a QoS routing and resource reservation protocol based on DSDV [9]. It is based on the **Time Division Multiple Access** (TDMA) MAC scheme which allows for pretty straightforward bandwidth computation. The initial idea was to allow ATM **virtual circuit** extension through multihop wireless networks, but can be used in the general case of an ad hoc network with no necessary ATM gateway. Thus, there is only a distinction between bandwidth constrained VC and plain datagram packets, other QoS parameters being ignored.

1. Frames and time slots

Communication between nodes is synchronized and divided in **time frames** which are themselves further divided in **time slots**. Each frame comprises a **control phase** and a **data phase** (as depicted in Figure 2). The purpose of the control phase is to negotiate time slot usage between adjacent nodes.

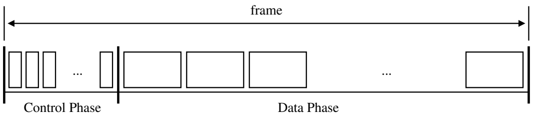


Figure 2: Frame structure

2. Path bandwidth calculation

The DSDV protocol is extended to allow nodes to advertise not only their distance to each node, but also their set of free time slots.

The available bandwidth on a link between two adjacent nodes is simply the set of commonly available time slots:

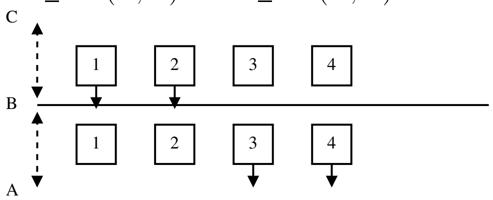
$$\text{link\_BW}(A, B) = \text{free\_slots}(A) \cap \text{free\_slots}(B)$$

A and B begin the link's endpoints.

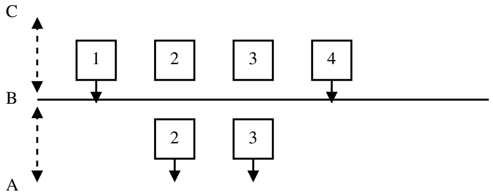
But in general, the computation of the available bandwidth for a path in a time-slotted network requires information not only about slot availability on individual links but also on the scheduling of free slots and is thus an NP-complete problem.

The article describes a heuristic that allows pretty good path bandwidth estimation, based on the assumption that otherwise, path bandwidth can be defined as the sum of the available bandwidth on the intermediate links. The general case is hence brought to one of the following three simple cases:

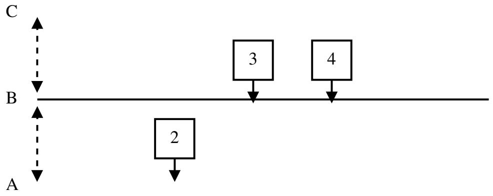
1. $\text{link\_BW}(A, B) = \text{link\_BW}(B, C)$



2. $\text{link\_BW}(A, B) \subset \text{link\_BW}(B, C)$
   or
   $\text{link\_BW}(B, C) \subset \text{link\_BW}(A, B)$



3. $\text{link\_BW}(A, B) \cap \text{link\_BW}(B, C) = \emptyset$



A, B and C being successive nodes in the path. In each case, the optimal slot assignment is straightforward and so a good estimation of the real path bandwidth can be computed.

3. Slot assignment

A new VC is set **hop by hop** from the source to the destination by reserving necessary time slots to satisfy the bandwidth requirement. The source, intermediate and destination nodes are treated in a different way, in order to reflect the fact that slots on one link interfere with those on the next link.

On the source node, enough slots have to be assigned among the ones belonging to the available bandwidth on the path.

On the intermediate nodes, the slots used by the incoming flow must be in the free set. Then incoming slots are **re-mapped** to remaining free slots to allow forwarding to the next node. If any of the two conditions does not hold, a RESET message has to be forwarded back to the source, freeing previously reserved slots along the way.

The destination node simply checks that the slots of the incoming flow are indeed in the free set and sends a REPLY or RESET message to the source indicating respectively success or failure of VC setting.

4. Route maintenance

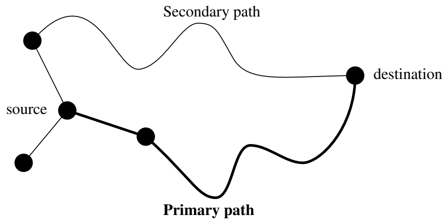


Figure 3: Primary and secondary paths

To cope with node mobility that can break an existing VC, the protocol maintains **secondary** paths to each node (Figure 3). Thus if the initial VC setting fails or is broken due to topology change, a new reservation is made along the secondary path. If this operations succeeds, the secondary path becomes primary and a standby route is computed and becomes the secondary path (figure 4). The DSDV protocol is very convenient to achieve secondary path computation as it does not imply any heavy modification. The secondary path is only the second shortest path according to neighbor advertisements.

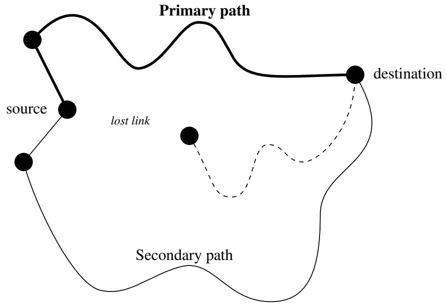


Figure 4: Route recovery

5. Pros and cons

The routing protocol performs very well in simulations. Bandwidth constraints satisfaction rate is high and resistance against failures in changing topology is strong.

A major drawback is that TDMA-based MAC layer is essential, hence widespread IEEE 802.11 NICs are not usable with it.

V. CEDAR

Stands for “Core-Extraction Distributed Ad hoc Routing” [10]. It is basically a reactive protocol[2] which optimizes routing request by using **core nodes**, i.e. nodes that belong to the dominating set of the network. Figure 5 shows the core nodes of a network and the core graph they form.

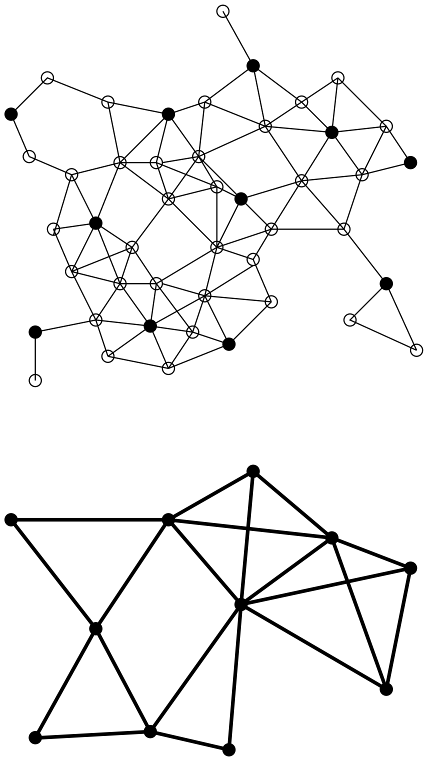


Figure 5: The network, the core nodes (upper) and the core graph (lower)

[2] A reactive protocol is one that does not maintain global topology information but initiates a route discovery mechanism if a route to an unknown destination is requested (the information is then kept in cache). A proactive protocol is one that maintains global topology information and provides routing information instantly.

1. Core maintenance

Each node broadcasts periodically BEACON messages which are used for one-hop neighborhood discovery.

In a first phase of the protocol, each node must elect its core node. The set of nodes that are the core node of a neighbor is the called the core of the network. The election is driven by a heuristic that ensures that the core is a dominating set of the network graph approximating the MDS[3].

Then, local topology information is disseminated — by means of piggybacking link information to BEACON messages — to the three-hop neighborhood of the nodes, allowing each core node to maintain **virtual links** to its neighboring core nodes. Thus core nodes maintain information about the local core graph.

2. Route calculation

When a node s needs a route to d, it requests a route calculation to its core node, $\mathrm{dom}(s)$. By means of some chosen reactive routing protocol, $\mathrm{dom}(s)$ finds a **core path**[4] to $\mathrm{dom}(d)$. After having learnt the route to $\mathrm{dom}(d)$, s sends packets to d by means of source-routing (Figure 6).

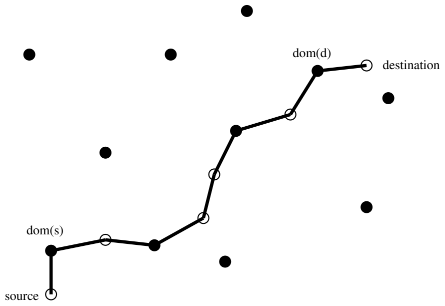


Figure 6: Route calculation

Now if QoS constraints are to be honored for a given route calculation, each core node must find a virtual link to the next core node towards $\mathrm{dom}(d)$ that satisfies them. The use of core nodes for route calculation assumes that the core path is a good hint towards the optimal route. But it does not ensures that the best route can be found.

3. Increase and decrease waves

The key idea for QoS routing in CEDAR is that link state broadcasts are bad and that a global topology information is not necessary to find optimal routes. The goal is to find a correct routes in highly changing topology conditions and to find near optimal in times the topology is stable.

Each physical link is **monitored** by its endpoints (i.e. bandwidth is monitored by means of interaction with the MAC layer, see [12]) and transmitted to their cores. Then this information is core broadcasted at some pace and to some limited hop distance. Information about stable links is disseminated further than that of others and information about high bandwidth is sent faster than that about low bandwidth.

This way, when the bandwidth of a given link goes higher than some threshold, an “increase wave” is generated. The higher the bandwidth, the farther the wave will propagate. When the bandwidth for the same link goes lower than some threshold, a “decrease wave” is generated. The decrease wave being faster than its dual, it kills any ongoing increase wave generated earlier. The direct consequence of it is that information about stable high-bandwidth links will be known to more remote core nodes that that about unstable or low-bandwidth links.

Each wave carries walked-so-far path information in it and thus allows core nodes to learn about interesting paths in terms of QoS.

4. Route maintenance

In case some link along the path towards the destination breaks, there are two possible strategies. Either the link is **repaired locally** by surrounding nodes, or the **source is notified** of the failure and recomputes a new route.

A combination of the two is chosen in that the first is suited for short-term measures, whereas the second is definitely better as a long-term solution.

5. Pros and cons

There are two important points that CEDAR addresses.

The first is that the control overhead of reactive protocols is great since the entire network is flooded with route probes. By the use of core nodes, CEDAR minimizes the flooding.

The second is that in spite of general belief, route computation based on neighborhood flooding broadcast is not reliable. This is due to the use of IEEE 802.11[13] MAC protocol which does not guaranty that broadcast messages are delivered. Thus in the case of multi-hop flooding, there are in fact many nodes which do not receive the data. On the other hand, unicast deliveries are ensured by the use of RTS/CTS plus ACK packets. Once more, the use

[3] Minimum Dominating Set. Finding the MDS is NP-hard and also hard to approximate [11].

[4] A core path is a route comprised of virtual links between core nodes.

of core nodes, hence core graph and core broadcast transforms such flooding into unicast deliveries and increases performance.

Furthermore, the model is open for additional optimizations like hop-by-hop routing — instead of source routing. Interaction with the MAC layer would allow for even more effective core path broadcasts with the monitoring of RTS/CTS packets.

The problem with CEDAR is that it relies on the core graph to find routes and does not take into account paths short-circuiting some core nodes. Moreover, a partition of the core graph implies momentary impossibility of find new routes.

VI. Ticket-based probing

The goal of TBP [14] is to achieve a near-optimal performance while avoiding flooding and accounting for link information imprecision.

QoS satisfying routes are found by sending **probes** carrying some limited total amount of **tickets** towards the destination using **distance vector** information [9]. At each intermediate node, a probe may be divided into several ones with the tickets distributed between them (Figure 7). The total amount of tickets is to remain the same along the propagation. The goal is to maintain a **compromise** between the odds of finding an optimal route (more tickets sent) and reduced control overhead (less tickets sent). If only one ticket is sent, then plain old shortest path is searched.

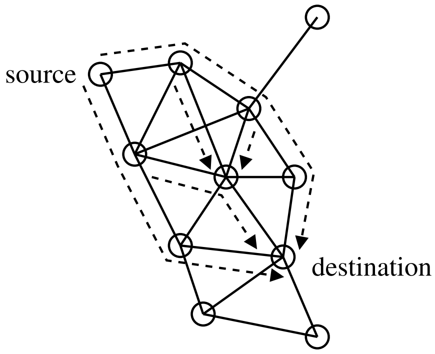


Figure 7: Probes sent from source to destination

TBP relies on the assumption that stable links tend to remain stable contrary to so called transient links. Each node i collects statistical information about delay $D_i(t)$, bandwidth $B_i(t)$ and cost $C_i(t)$ to each other node t in the network. It is thus not meant to be scalable, but is rather destined for moderate scale networks. Along with $D_i(t)$ and $B_i(t)$, each node maintains the association variation $\Delta D_i(t)$ and $\Delta B_i(t)$ by which the next reported value will differ with the current one.

The protocol makes a distinction between delay and bandwidth constrained paths. Thus, it either searches a route for least-cost delay-constrained connections, or least-cost bandwidth-constrained connections.

For each purpose, it defines two kinds of tickets — yellow ones and green ones — that make the probes either look for **feasible** paths or **least-cost** paths. The relative quantity of each kind governs which kind of path is going to be searched.

1. Delay-constrained path

The source node computes the number of yellow and green tickets based on the **imprecise** information is has gathered about delay and cost towards the destination. With the help of $\Delta D_i(t)$ and some additional system-wide constants, the algorithm ensures that if a feasible path does not exist, it will stop searching immediately. It issues yellow and green tickets depending on the degree of feasibility of finding a good path (Figure 8 shows the ticket distribution). It modulates the quantities in order to favor feasible and least-cost paths but favoring more feasible paths for tight requirements.

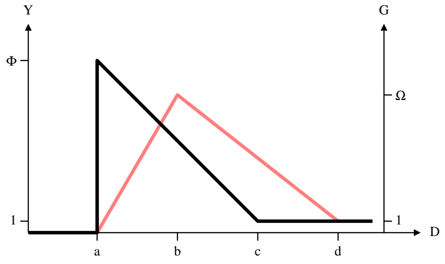


Figure 8: Distribution of yellow tickets (black, left) and green tickets (red, right) at the source node for delay-constrained path. *a, b, c and d are respectively $D_s(t) - \Delta D_s(t)$, $D_s(t)$, $D_s(t) + \Delta D_s(t)$ and $\theta(D_s(t) + \Delta D_s(t))$. Φ and Ω are respectively the maximum of yellow tickets and green tickets. θ is a threshold specifying the so-called* sufficiently large range *for the delay requirement.*

At each intermediate node, the tickets are redistributed among probes towards next-hop candidates with more tickets for candidates with looser link conditions, hence potentially more possible paths.

2. Bandwidth-constrained path

Analogously, the source node computes the number of yellow and green tickets but based this time on bandwidth information. The formulae are adapted to the convex nature of the bandwidth metric (Figure 9). The rule at each intermediate node for the forwarding of the probes is the same.

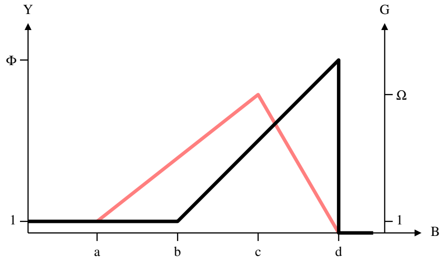


Figure 9: Distribution of yellow tickets (black, left) and green tickets (red, right) at the source node for bandwidth-constrained path. *a, b, c and d are respectively $\theta(B_s(t) - \Delta B_s(t))$, $B_s(t) - \Delta B_s(t)$, $B_s(t)$ and $B_s(t) + \Delta B_s(t)$. Φ and Ω are respectively the maximum of yellow tickets and green tickets. θ is a system parameter verifying* $0 < \theta < 1$.

3. Route selection and reservation

If at some point a requirement in a probe cannot be fulfilled in any of the next-hop candidate, the probed is marked as unsatisfied and send along towards the destination.

Each probe carries along the total number of tickets issues by the source and thus the destination awaits all the probes to arrive before selecting a route. Timeout conditions are set up to avoid stalling. When all probes have arrived at the destination, a route can be chosen among the satisfying ones — on probes that are not marked as unsatisfied — and a response is source-routed towards the source node and reservation is made at intermediate nodes.

4. Route adaptation

Several schemes are proposed for quick to slow adaptation to link failures. A route can either be repaired by neighboring nodes, thanks to the distance vector information or the source can be requested to issue a new route request, depending on jitter requirements.

5. Pros and cons

The model achieves routing quality comparable to **flooding** while keeping overhead close to **shortest-path** methods. The multi-path nature of TBP allows for **modularity** in link failure conditions with several levels of path redundancy.

The initial paper relies on a MAC layer providing resource reservation capabilities which seems not to be necessary if delay and bandwidth of local links can be computed easily and accurately; then the resource reservation can be made at a higher level.

VII. INSIGNIA

INSIGNIA is meant to provide a complete framework for QoS support in MANETs [15]. It is based on the assumption that applications requiring QoS support also provide a way to modulate this requirement and can sustain service degradation by the use of **adaptive services**. The model thus defines three operating modes for a flow: best effort (BE), base QoS (BQ) constraints with minimum bandwidth and enhanced QoS (EQ) with maximum bandwidth. It also assumes, quite rightfully, that the destination node is the best place to measure the QoS of a flow and make decisions about service degradation.

1. In-bound signaling

The framework provides a form of **in-band signaling** between source and destination by the addition of an option in the IP header of packets, the purpose of which is to attach QoS requirements to a flow as well as a means to alert the destination in case the requirements are not fulfilled at a bottleneck node.

2. No particular routing protocol

The framework is not specifically designed to operate with a given routing protocol and is rather meant to support any MANET routing protocol. Thus, it does not take any advantage of QoS-aware routing but rather relies on service adaptation on the application's side. Moreover, the framework remains independent of any MAC layer specificities which should ensure easy implementation with existing networking equipment.

3. QoS adaptation

The destination is required to report QoS level to the source in the **feedback** in order to modulate the flow to suit QoS conditions. Rerouting is provided by the routing protocol and quality is subject to degradation in case of failure to reserve resources along the new path. Old reservations are nevertheless supported by **soft state** and disappear if not regularly refreshed.

4. Pros and cons

The implementation seems simple, hence easy to implement, though not fully taking advantage of existing QoS-satisfying paths.

VIII. SWAN

Stands for “Stateless Wireless Ad hoc Network” and is more precisely a means to provide QoS support through service differentiation [16]. It is another framework independent of the routing protocol and the MAC layer, though it relies on the latter for bandwidth and delay measurement. The novelty is that routing nodes do not maintain any state information about the flows they forward — hence “stateless”.

1. Stateless routers

Each node differentiates two kinds of traffic through DSCP[5] flags in the IP header: real time (RT) and best effort (BE) (see Figure 10 for a general view of the model). RT flows are forwarded normally, whereas BE are fed to a **traffic conditioner**[6]. The node actively monitors RT bandwidth usage and sets the conditioner’s parameters accordingly to **shape** BE traffic to the residual bandwidth so that packets queues are maintained short to avoid delay increase[7].

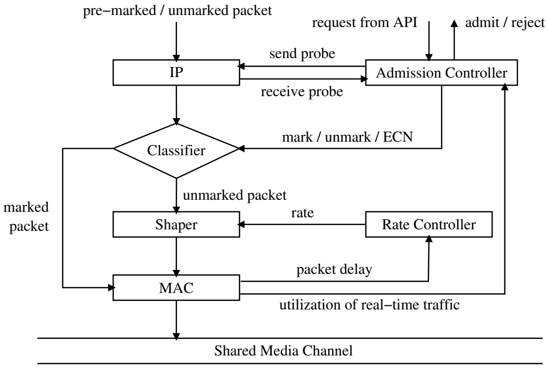


Figure 10: SWAN Model

2. Source-based admission control

When a node intends to open a QoS constrained connection, it sends a probe to measure **bottleneck bandwidth** and **available delay**. Here it seems that any routing protocol may be used, especially a QoS-aware one comes in handy. While traversing the network towards the destination, the probe is marked by the routing nodes with the measure actual link delay and minimum bandwidth. The destination copies this gathered information in the reply packet and sends it back to the source which can then decide whether to admit the flow or not. Once the flow is admitted, it has to be routed through the chosen path.

3. False admissions and network conditions change

If two source nodes simultaneously send a probe prior to open a new connection, they can read the same “availability” in the network which does not hold anymore when the connections are actually opened. This is called **false admission** and can generate congestion.

Thus, independently of the reason of congestion, routing nodes use the ECN[8] bit provided by the IP header. When an RT packet arrives at a node in excess of the link’s capacity, it is marked with CE[9] and forwarded in BE mode towards the destination. Then the destination can command the source to either find a new route or drop the connection. This approach has nevertheless two problems: it cannot differentiate between a false admission condition and a topology change that made two flows interfere. There are two possible answers.

4. SWAN-1

The first idea is that a source node that is commanded to re-admit the flow waits a random backoff delay before doing so to avoid synchronized re-admission which would make the source nodes “see” overcrowded routers and drop their flows (in case of false admission). To make a distinction between false admission and regulation due to node mobility, source nodes should be able to distinguish between newly admitted flows and others. It then would re-admit flows preferably.

5. SWAN-2

The second answer is to make routing nodes mark the exceeding packets selectively for a subset of the existing flows. In addition, source nodes should use an additional DSCP/TOS bit to mark old flows and thus allowing routing nodes to smartly select flows for marking in order to re-establish correct network operations.

6. Pros and cons

The model is pretty simple and requires not much in the intermediate nodes. It seems easy to implement and can be associated with almost any QoS-aware or not routing protocol[10].

But it relies on the MAC layer for link quality measurements which cannot be realizable in every

[5] Differentiated Services CodePoint[17].

[6] Usually a token bucket filter.

[7] This implies the use of AIMD control, i.e. additive increase multiplicative decrease, to adapt rates.

[8] Explicit Congestion Notification, see [18] for details.

[9] Congestion Experienced.

[10] Though a QoS-aware routing protocol would make a better use of the network capacity.

case.

IX. QOLSR

QOLSR [19] is an enhancement of the OLSR routing protocol to support multiple-metric routing criteria.

1. OLSR

The Optimized Link-State Routing (OLSR) protocol [20], is a **proactive** routing protocol based on the **link-state** scheme. It introduces an **optimized network flooding** method by the use of **multi-point relay** nodes (MPRs). Each node in the network elects some of its one-hop neighbors to be in its MPR set, so that any node in its two-hop neighborhood is reachable either directly, or by relaying through an MPR. Figure 11 illustrates the benefit of using MPRs.

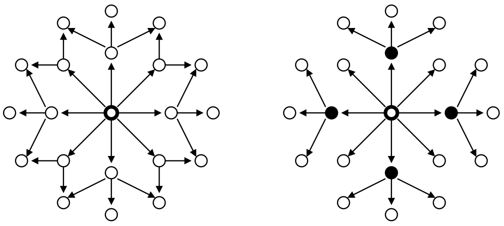


Figure 11: Pure vs. MPR flooding

To achieve this, each node advertises its one-hop neighborhood by broadcasting HELLO messages to its one-hop neighbors in UDP packets. Based on the information contained in its neighbors' HELLO messages, a node maintains its MPR set and generates Topology Control (TC) messages. The TC messages are broadcasted through the MPRs to every node and their purpose is the advertising of a node's MPR Selector set (i.e. the set of neighbors that have chosen it as one of their MPRs). The TC messages provide necessary link-state information to allow any node to compute a route to any node in the network.

Thanks to the TC messages, each node has a global view of the network and computes routes by the application of the Dijkstra or Bellman-Ford algorithms. Basically, every known links are of equal weight and thus these methods allow for shortest-path best effort route computation.

2. Delay and Bandwidth constraints

TC messages can be augmented with quality information about each link (between a node and its MPR selectors). A minimal delay route can be found using plain Dijkstra or Bellman-Ford algorithm, since the delay is an additive metric. A maximum bandwidth route can be found using a modified Dijkstra or Bellman-Ford variant, because bandwidth is a convex metric. But finding a route with maximal bandwidth and minimal delay is difficult or may be even impossible, for such a route may not even exist.

The idea is to prioritize the bandwidth constraint, hence if more than one route with maximum bandwidth exist, choose the one with minimum delay (i.e. find the shortest-widest route). In [19], the method uses the **Lagrange Relaxation-based Hop** algorithm (LRH) which solves the **Delay and Bandwidth Constrained Least Hop** problem (DBCLH) in polynomial time and finds a path of minimum hop-count while keeping delay, resp. bandwidth, below, resp. above, given bounds.

3. Pros and cons

QOLSR, as its predecessor OLSR, is independent of the MAC layer, but supposes the ability of the MAC drivers to report enough information to compute a link quality (delay and available bandwidth).

QOLSR is compatible with OLSR as it uses the same message format. Additional information is carried by means of optional fields in the messages. If a given MPR node is only OLSR-aware, it will forward the packet in best-effort mode, but will otherwise not interfere with his QoS-aware siblings.

Work on QoS-aware OLSR is promising, simulation results show that it make efficient use of available resources. But it is still in a early stage of development and requires extensive testing to prove its applicability.

X. Further works

Actually, there are several attempts to solve the problem of QoS support using different approaches. Some employ an existing best effort routing protocol and augment it with QoS link information to make it QoS-aware and others try to start from scratch. In each case, extensive simulations and experimentations are essential to validate its usability.

References

[1] E. Royer and C. Toh, "A review of current routing protocols for ad-hoc mobile wireless networks," 1999.

[2] "Mobile Ad-hoc Networks." http://www.ietf.org/html.charters/manet-charter.html.

[3] K. Wu and J. Harms, "QoS support in mobile ad hoc networks."

[4] R. Braden, D. Clark, and S. Shenker, "Integrated services in the Internet architecture: an overview," Tech. Rep. 1633, IETF, 1994.

[5] L. Zhang, S. Deering, and D. Estrin, "RSVP: A new resource ReSerVation protocol," *IEEE network*, vol. 7, pp. 8–?, September 1993.

[6] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An architecture for differentiated services," 1998.

[7] H. Xiao, W. Seah, A. Lo, and K. Chua, “A flexible quality of service model for mobile ad-hoc networks.”

[8] C. R. Lin and J.-S. Liu, “QoS routing in ad hoc wireless networks,” *IEEE Journal on Selected Areas in Communications*, vol. 17, pp. 1426–1438, August 1999.

[9] C. Perkins and P. Bhagwat, “Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers,” in *ACM SIGCOMM’94 Conference on Communications Architectures, Protocols and Applications*, pp. 234–244, 1994.

[10] P. Sinha, R. Sivakumar, and V. Bharghavan, “CEDAR: a core-extraction distributed ad hoc routing algorithm,” in *INFOCOM (1)*, pp. 202–209, 1999.

[11] S. Guha and S. Khuller, “Approximation algorithms for connected dominating sets,” in *European Symposium on Algorithms*, pp. 179–193, 1996.

[12] M. Kazantzidis, “End-to-end versus explicit feedback measurement in 802.11 networks.”

[13] “IEEE 802.11 Wireless.” `http://standards.ieee.org/getieee802/802.11.html`.

[14] S. Chen and K. Nahrstedt, “A distributed quality-of-service routing in ad-hoc networks,” *IEEE Journal on Selected Areas in Communications*, vol. 17, August 1999.

[15] S.-B. Lee, G.-S. Ahn, X. Zhang, and A. T. Campbell, “INSIGNIA: An IP-based quality of service framework for mobile ad hoc networks,” *Journal of Parallel and Distributed Computing*, vol. 60, no. 4, pp. 374–406, 2000.

[16] G.-S. Ahn, A. T. Campbell, A. Veres, and L.-H. Sun, “SWAN: Service differentiation in stateless wireless ad hoc networks.”

[17] K. Nichols, S. Blake, F. Baker, and D. Black, “Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 headers,” 1998.

[18] K. Ramakrishnan, S. Floyd, and D. Black, “The addition of explicit congestion notification (ECN) to IP,” 2001.

[19] H. Badis, A. Munaretto, K. A. Agha, and G. Pujolle, “QoS routing in OLSR: multiple-metric enhancement,” 2003.

[20] T. Clausen, P. Jacquet, A. Laouiti, P. Muhlethaler, a. Qayyum, and L. Viennot, “Optimized Link State Routing protocol,” in *IEEE INMIC Pakistan*, 2001. Best paper award.