QoS routing and CAC (Connection Admission Control)

Andrea Bianco
Telecommunication Network Group
firstname.lastname@polito.it
http://www.telematica.polito.it/

QoS routing and CAC

- Preventive traffic control technique (in principle it can become reactive)
- Permits to determine whether to accept or not a new incoming call
  - QoS routing selects a set (possibly one) of tentative paths
  - CAC checks whether enough resources are available over each link of each path
    - Cannot be done at the routing level because routing operates on less detailed info to ensure scalability
  - Resources are allocated to guarantee QoS
- The call is accepted if there are enough network resources to:
  - Satisfy the requested QoS
  - With the constraint of keeping at the same level the QoS offered to already accepted calls
- Can be applied to unicast and multicast calls
  - Multicast calls are routed over a tree rooted at the source and covering all receivers
- Call definition?
  - In ATM, each VPI/VCI
  - In Frame Relay each DLCI
  - In Internet? Flow identification problem

QoS routing

- Network modeled as a graph G(V,E)
  - Nodes represent switches, routers
  - Edges represent communication links
- Traditional routing problem
  - Call request from user a to user b (or to a set of users B)
  - Costs associated with edges
  - Find over G a path (tree) that minimize costs to route the call from a to b (or B)
    - If all edges have the same cost, shortest path “optimizes” network performance
- QoS routing problem
  - Call request from user a to user b (or to a set of users B) with a given set of QoS requirements
  - Nodes may have a state related to QoS metrics
  - Edges have a state, related to QoS metrics, associated
  - Find over G a feasible path (tree)
    - It must have enough residual resources to satisfy call QoS constraints
  - Among several feasible paths, it may choose the one which minimizes cost

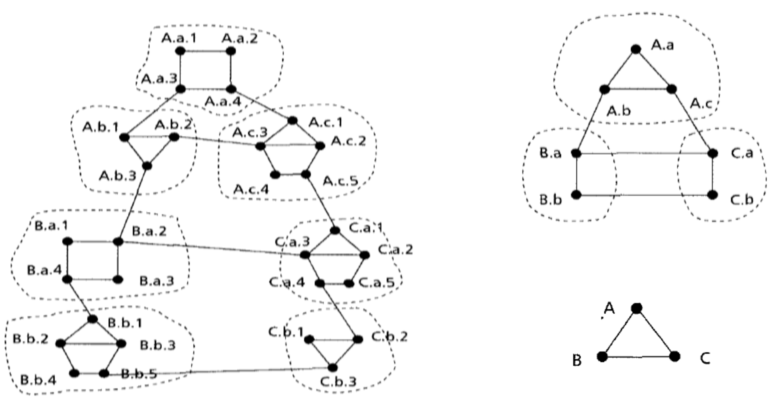
QoS routing

- Difficult problem
  - QoS constraints may be very diverse
    - Bit rate, delay, delay jitter, loss ratio
      - Additive constraints (hop count, delay)
      - Multiplicative constraints (loss ratio)
      - Concave constraints (bit rate)
    - Multiple constraints often make the QoS routing problem NP-hard
  - Integration with best-effort traffic
    - QOS traffic not affected, but best effort may suffer
  - Network state may change dynamically
    - Difficult to gather up-to-date state information
    - Performance may degrade dramatically if state information outdated

State information

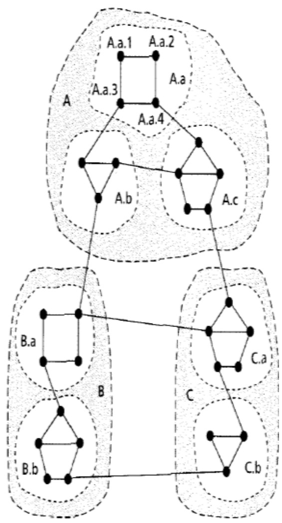
- Link state may be a triple
  - Bandwidth, Delay, Cost
- Node state may simply be a combination of its link state
  - CPU bandwidth may be taken into account
- Local state measured and kept by each node
- Global state exchanged through link-state or distance vector protocols
- Scalability may be achieved by information aggregation, exploiting the hierarchical structure of the network
  - Not only for link state info but also for addressing

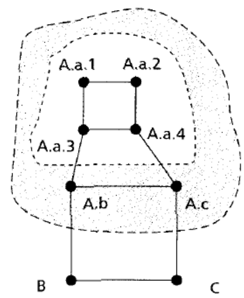
Hierarchical network model: network layers



Taken from Chen,Nahrstedt AN Overview of QoS Routing…", IEEE Network 1998

Hierarchical network model: hiding topological details





Unicast (Multicast) QoS routing

- Unicast (Multicast) QoS routing definition
- Given
  - A network topology
  - A source node s
  - A destination node d (set of destinations R)
  - A set of QoS constraints C
  - Possibly an optimization goal
- Find
  - The best feasible path from s to d (tree covering s and all nodes in R) which satisfies C
- Constraint
  - Algorithmic complexity
- Multicast routing is a generalization of unicast routing

Unicast QoS routing classification

- Link-Optimization (LO) or Link-Constrained (LC)
  - The state of a path is determined by the bottleneck link
    - Residual bandwidth and residual buffer space
  - Min-max operations on non additive metrics
  - Optimization:
    - Ex: find a path that has the largest bandwidth on a bottleneck link
  - Constrained
    - Ex: find a path whose bottleneck link is above a given value
  - Link-constrained can be mapped to link optimization
- Path-Optimization (PO) or Path-Constrained (PC)
  - The state of the path is determined by the combined state over all links of the path
    - Delay
  - Combinatorial operation over additive metrics
  - Optimization
    - Ex: find a path whose total cost is minimum
  - Constrained
    - Ex: find a path whose delay is bounded by a given value

Composite unicast routing problems

- Elementary routing problems can be combined to create composite routing problems
- LC-PO problem
  - Bandwidth constrained least delay routing
    - Can be solved by a shortest path algorithm on the graph obtained by removing links violating the bandwidth constraint
- LOLC, LCPO, LCPC, PCLO can be solved in polynomial time
- PCPO (find the least cost path with bounded delay) and Multi-Path Constrained (path with both bounded delay and jitter) are NP if
  - Two metrics are independent
  - Measured as real numbers or unbounded integers

QoS routing strategies

- Classification according to
  - how state information is maintained and distributed and
  - how the search of a feasible path is performed
- Strategies
  - Source routing
  - Distributed routing
  - Hierarchical routing
  - Can be combined

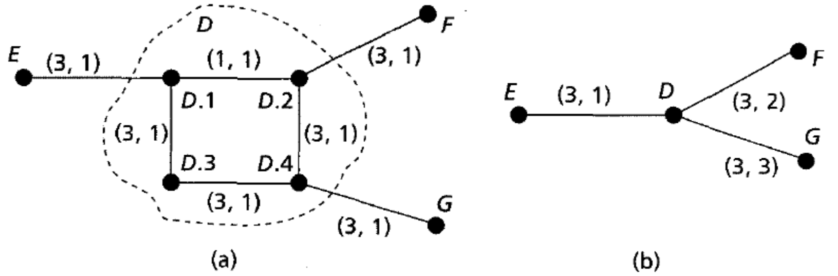
QoS routing strategies

- Source routing
  - Each source node
    - Maintains the complete global state (received by all other nodes)
      - Network topology, state information
    - Locally computes a feasible path
    - Sends a control message along the selected path to inform intermediate nodes of their precedent and successive nodes or insert the end to end path on each packet header
- Distributed routing
  - Path computed through a distributed computation
  - Each node keeps a partial (global) state
  - Routing done on a hop-by-hop basis
- Hierarchical routing
  - Nodes clustered into groups, further clustered in higher-level groups recursively, creating a multi-level hierarchy
  - Each physical node maintains an aggregated global state
    - Detailed information about the nodes in the same cluster and aggregated state information about the other groups

QoS routing strategies

- Source routing
  - Centralized solution
  - Avoids problem with distributed solutions (deadlock, distributed terminations, loops)
  - Large communication overhead to update state
    - Imprecision in the global state information
  - Large computation overhead
- Distributed routing
  - More scalable
  - Parallel search possible
  - Loop due to inconsistencies
  - Large communication overhead
- Hierarchical routing
  - Often used in conjunction with source routing
  - Routing computation shared by many nodes (source and border nodes)
  - Adds imprecision due to aggregation (mandatory to scale)

Hierarchical aggregation: loss of detailed info



Link state = (bandwidth, delay)

Unicast QoS routing: examples

- Examples of proposed distributed algorithms
  - Widest Path
    - Path with the maximum bottleneck bandwidth
  - Shortest Path
    - Path with smallest delay
  - Shortest-Widest Path
    - Among widest paths, select the one with smallest delay
  - Widest-Shortest Path
    - Among shortest paths, select the one with the maximum bottleneck bandwidth
  - Delay constrained least-cost routing
    - Each node keeps a cost and a delay vector for the best next hop for any destination
    - A control message is sent from the source to construct a delay-constrained path
    - Any node con select one of two alternative links (least cost path or the least delay path)
    - Least cost path has priority as long as the delay constraint is not violated
    - Loops detected if control messages seen twice
    - Roll back until reaching a node who chooses the least cost path

Unicast QoS routing: examples

- Examples of proposed source routing algorithms
  - Bandwidth-delay constrained
    - All links with not enough bandwidth are eliminated, then the shortest path is searched for
  - Transform delay, jitter and buffer space bounds in bandwidth bounds when traffic is token bucket controlled and nodes are running proper scheduling algorithms

QoS routing: issues

- For high loads, maximum throughput is provided by the minimum hop
- For medium-low loads algorithm performance depend on network topology and traffic pattern
- Some algorithms may be implemented only in a centralized way
  - Hop-by-hop decisions may be sub-optimal
- The more complex the link/node metric used
  - Increase in signaling bit rate to distribute status
- The more dynamic the link/node metric used
  - Increase in the frequency of status update
  - Need to re-run the routing algorithm

Multicast QoS routing classification

- Similar to the unicast QoS case, but optimization or constraints must be applied to the full tree
  - Link optimization or constrained
  - Tree optimization or constrained
- Steiner tree problem (tree optimization) is to find the least-cost tree
  - Tree covering all destinations with the minimum total cost over all links
  - It is NP-hard
  - If destination set includes all network nodes, the Steiner tree problem reduces to the minimum spanning tree problem which can be solved in polynomial time

Composite multicast routing problems

- Elementary multicast routing problems can be combined to create composite routing problems
- LCLO, MLC (Multi-link constrained: Bandwidth buffer-constrained), LCTC, TCLO can be solved in polynomial time
- LCTO, TCTO, and MTC (Multi-tree constrained: delay-delay jitter constrained) are NP if
  – Two metrics are independent
  – Measured as real numbers or unbounded integers

Issues in multicast traffic

- Multicast trees are dynamic
  – Users leave and join
    - Maintain or update the tree while the call is on
    - Pruning easier than extending
    - Heuristic: Adding to the current tree via a minimum cost path
    - Periodic tree re-design possible
- Receiver heterogeneity
  – Allocate for the most demanding user but only if using hierarchical coding at the source
  – Generate a set of flows at different rate
  – Run the application according to the minimal capabilities
- ACK explosion for reliable multicast

CAC algorithm

- INPUT DATA
  - Traffic characterization at network ingress
  - Call QoS requirements
  - Path(s) selected by (QoS) routing algorithms
  - Network status (available bit rate, buffer occupancy, …) and data traffic already accepted in the network
- OUTPUT
  - Accept (if QoS requirements can be satisfied) or refuse the call
- CONSTRAINTS
  - Not violate QoS requirements of already accepted calls

CAC algorithm

- Algorithm executed
  - In all network nodes through which the call is routed
- It is possible to envision QoS parameters re-negotiation in case of negative answer
- Main CAC methodologies
  - Parameter based admission control
    - Peak rate, average rate
    - Worst case analysis
    - Equivalent bandwidth
  - Measurement based admission control

Peak rate CAC

- Peak rate allocation
  - Call k is accepted if available bandwidth is larger than the peak bandwidth of call k:

$$B_P^{(k)} \leq C - \sum_{i\,acc.} B_P^{(i)}$$

- Rationale
  - Worst case dimensioning
- CBR traffic
  - Bit rate guarantees
  - Delay guarantees as a function of the number of accepted calls
  - Zero losses if buffer size proportional to number of accepted calls
- VBR traffic
  - Same guarantees as of CBR traffic
  - Link utilization proportional to: $\frac{B_M}{B_P}$

Peak rate CAC

- Simple
- Does not exploit potential benefits of statistical multiplexing
- Very good QoS guarantees
- Transmission link capacity may be largely under-utilized for VBR traffic
- Network behaves very similarly to circuit switched networks
  - Bit rate guaranteed, loss probability negligible or null
  - Data transmission is not synchronous
  - Delay guarantee depends on other user behavior
- Many multiplexing stages could increase B_P over a short time interval, thus partly worsening QoS guarantees

Average rate CAC

- Average rate allocation
  - Call k accepted if:

$$B_M^{(k)} \leq C - \sum_{i\,acc.} B_M^{(i)}$$

- Rationale
  - Over a long period of time the network is never overloaded

Average rate CAC

- Simple
- Very high link utilization
- Zero loss only with infinite buffer
- With finite buffers
  - Congestion (link overload proportional to source burstiness)
    - Uncontrolled losses
    - Uncontrolled delays
    - Unless more tight constraints on the traffic source
- Network behaves similarly to packet switched networks with datagram service
  - But permanent overload is avoided
- May take some safety margin from 100% utilization to statistically control losses and delays

An example

- Focus on a single node
  - Focus on an output link with capacity 100Mbit/s
- Incoming calls are VBR with peak rate 10Mbit/s and average rate 1Mbit/s (burstiness **10**)
- If using peak rate CAC
  - At most 10 calls are accepted
    - Average output link utilization **10**%
- If using average rate CAC
  - At most 100 calls are accepted
    - Worst case overload is 1Gbit/s (**10** times larget than link speed)

Worst-case analysis: examples

- Suppose a source is constrained by a token bucket
- Can accept calls when
  - The summation of token rates is smaller than link capacity
  - The summation of token depth is less than available buffer space
- Properties
  - Zero losses
  - Delay guarantees depending on number of calls and token depth
  - Low utilization
- If used scheduler is WFQ (see slides on scheduling)
  - Can allocate bandwidth to
    - Satisfy the worst case delay along the path
    - Bound the buffer size to avoid packet losses

Example of statistical guarantees

- Adopting statistical instead of deterministic guarantees provides singificant benefits
- Example
  - 10 identical sources with rate 1.0
  - Each source active with probability 0.1
- What is the probability of overloading a link of capacity 8.0?
- If sources are independent, probability of having n active sources

$$\binom{10}{n} 0.1^n 0.9^{10-n}$$

- Probability of overloading smaller than 10^{-6}
- By allowing a very small overflow probabilty, resource requirements are reduced by 20%

Equivalent bandwidth CAC

- DATA:
  - Traffic characterization (peak rate, average rate, burst duration,...)
  - QoS requirements (mainly cell loss)
  - Traffic behavior of other calls
- OUTPUT:
  - Equivalent bandwidth (bandwidth needed to satisfy call QoS requirements)
- Call k is accepted over a link with capacity C if:

$$B_{eq}^{(k)} \leq C - \sum_{i\,acc.} B_{eq}^{(i)}$$

How to compute equivalent bandwidth: traffic model

- To compute B_{eq} a traffic model must be used:
  - Define the source stochastic behavior
  - Emulate (or solve) the system under study, which comprises all previously accepted calls plus the new call
  - Determine the bit rate that should be allocated to the new call to satisfy the QoS needs
- Several models were proposed
  - Some take into account even buffer size
- B_{eq} often assumes a value ranging between B_M and B_P
  - B_{eq} can be larger than B_P if delay constraints are very tight
  - B_{eq} is never smaller that B_M

Equivalent bandwidth: an example

- Assume a fluid approximation
  - Buffer size B
  - Buffer is drained at a constant rate e
  - Worst case delay B/e
  - The equivalent bandwidth is the value of e that makes the loss probability smaller than a given value
  - Jointly provides bandwidth, loss and delay guarantees

Equivalent Bandwidth CAC

- Permits to compute a service rate adequate to guarantee call QoS
  - This rate can be used to allocate bit rate resources within nodes
- The method works properly if the traffic model is realistic, i.e. if the traffic generated by the call is similar to the one defined by the model
- Difficult to extend to sequence of links
  - Multiplexing effect modifies traffic shape
- Can be computation intensive to solve the model on-line, i.e. for each new incoming call

Equivalent Bandwidth CAC

- As an alternative, it is possible to define a (small) set of traffic classes, where each class is identified by the same
  - Traffic characterization
  - QoS requests
- If the traffic classes are known a-priori, it is possible to pre-compute (off-line)
  - B_{eq} required by each call of each class, therefore the number of calls acceptable on each link for each class
  - Since it is off-line, it is also possible to use more complex (and hopefully more efficient) models

Equivalent Bandwidth

- The off-line approach constraints user traffic generation and QOS requirements to simplify the on-line CAC procedure
- Traffic classes are derived from applications run by the users
  - Applications development much faster than network standard modification
- Mix the off-line and the on-line approach?
  - Not easy
  - Can be done by statically partitioning link bandwidth
    - Create two virtual infrastructures and manage them separately

Measurement based CAC

- Normally used with a very simple traffic characterization
  - E.g., call peak rate B_P
- Basic idea
  - Measure the traffic load on each link in real time
    - this is normally done anyway in network devices
  - This measure, performed over a pre-defined measurement interval, permits to compute the residual available bandwidth
  - Call k is accepted if: : $B_P^{(k)} \leq B_{measured\_available\_bit\_rate}$
- Note that after acceptance, calls are accounted for their real traffic, not on the basis of declared parameters
- Useful if traffic characterization parameters or network status are unknown or known with a large error
- Normally leads to high link utilization
  - Difficult to guarantee QoS

Measurement based CAC

- Disadvantages/problems:
  - Measurement parameter setting (e.g., measurement window duration)
    - Window too large implies more stable but less reactive estimate
    - Window too short may provide unreliable estimate
  - Implicit assumption that accepted call behavior is similar during a measurement interval
  - Measurement errors
  - If too many calls arrive during a measurement period
    - Many calls are rejected, since they are accepted on the basis of their peak rate
  - Useful for CAC only, but no information on the bit rate that should be allocated to calls to guarantee QoS
    - Very difficult to predict call QoS a priori

CAC issues

- Un-fairness for calls requiring higher bit rate in saturated conditions
  - Resource partitioning
- Difficult to extend algorithms to several consecutive links
  - Users are interested in end to end quality, non in single hop behavior
- Renegotiation may be interesting?

References

- S.Chen, K.Nahrstedt, "An Overview of Quality of Service Routing for the Next Generation High-Speed Networks: Probems and Solutions", IEEE Network, 1998
- H.Perros, K.Elsayed, "Call Admission Control Schemes: a Review", IEEE Communications Magazine, Nov. 1996, pp. 82-91
- H.Salama, D.Reeves, Y.Viniotis, "Evaluation of Multicast Routing Algorithms for Real-Time Communication on High-Speed Networks", IEEE JSAC (Journal on Selected Areas in Communications), vol.15, n.3, April 1997