CEG 7450:
(QoS Routing)

Internet Routing

- Focus: reachability only
- Unicast routing
- Multicast routing

QoS Constraints

- Link constraints: restriction on the use of a link, e.g. bandwidth requirements
- Path constraints: end-to-end QoS requirement on a single path, e.g. delay, cost, loss
- Tree constraints: QoS requirement for the entire multicast tree, e.g. delay

QoS Routing

- Feasible path (tree) is one that has sufficient residual resources to satisfy the QoS constraints of a connection
- QoS routing is to find a feasible path (tree), in addition, to optimize certain performance metric: e.g. optimization of resource utilization, cost
  - Find the least cost path (tree) among all feasible paths (trees)

Model

- Weighted graph G(*V*, *E*)
- *V*: vertices *E*: edges/links
- Link state: every link has state information, e.g. QoS metric
- Node state: measured independently or combined into the state of the adjacent links

Maintenance of State Info

- Local state info
  - Queueing and propagation delay
  - Residual bandwidth of outgoing links
  - Availability of other resources
- Global state info: the collection of local state info of all nodes
- Link state protocols:
  - broadcast local state of every node to every other nodes
  - each node has: network topology; state of every link

Maintenance of State Info

- Distance vector protocols:
  - Periodically exchange distance vectors among adjacent nodes
  - A distance vector includes:
    - cost of the best path;
    - the next node on the best path
- Approximation of the current network state info due to
  - Delay of propagating local state information
  - Aggregation of state information

Unicast QoS Routing Problem

- Given a source *s*, a destination *t*, a set of QoS constraints *C*, and possibly an optimization goal, find the best feasible path from *s* to *t* which satisfies *C*.

Classification of Unicast Routing

- Link optimization routing
  - Bandwidth optimization routing: find a path that has the largest bandwidth on the bottleneck link; widest path problem
- Link constrained routing
  - Bandwidth constrained routing: find a path whose bottleneck is above a required value
- Path optimization routing
  - Least cost routing: find a path whose total cost is minimized
- Path constrained routing
  - Delay-constrained routing: find a path whose delay is bounded by a required value

Classification of Unicast Routing

- Composite routing problems
  - Link-constrained path-optimization routing
    - Bandwidth constrained least delay routing
  - Link-constrained link-optimization routing
  - Multi-link-constrained routing
  - Link-constrained path-constrained routing
  - Path-constrained link-optimization routing

Classification of Unicast Routing

- NP-complete problems
- Path constrained path optimization routing
  - Delay constrained least-cost routing: find the least-cost path with a bounded delay
- Multi-path-constrained routing
  - Delay and delay-jitter constrained routing: find a path with both bounded delay and bounded delay jitter requirements

Hardness of QoS Routing Problem

- Multiple constraints often make the routing problem intractable
  - E.g. finding a feasible path with two independent path constraints is NP-complete
  - The QoS metrics are independent
  - Metrics allowed to be real numbers or unbounded integers

Hardness of QoS Routing Problem

- If all metrics except one take bounded integers, the problems are solvable in polynomial time by running an extended Dijkstra's algorithm
- If all metrics are dependent on a common metric, the problem is also solvable in polynomial time
  - E.g., The worst-case delay and delay jitter are functions of rate of a connection in networks using WFQ scheduling
  - The delay and delay jitter constrained routing problem is solvable in polynomial time in such networks

Source Routing

- Each node maintains complete state information
- A feasible path is locally computed at the source
- Control messages are sent to set up the path
- Link state protocols are used to update the global state at each node
- Strengths
  - Simplicity: avoid dealing with distributed computing
  - Guarantee loop-freedom
  - Easier to implement, evaluate, debug, and upgrade
  - Easier to design centralized heuristics for NP-complete problems than to design distributed ones

Source Routing

- Weakness
  - Has to frequently update global state → high overhead
  - Approximate global state only → QoS routing may fail to find a feasible path even if it may exist
  - High computation at source
  - Scalability problem: large network

Distributed Routing

- A path is computed by a distributed computation
- Control messages exchanged among nodes
- State information kept at each node is collectively used for the path search
- Use distance-vector / link-state protocols to maintain state information
- Strengths
  - Scalable
  - Shorter response time: path computation done collectively among the intermediate nodes between the source and destination
  - May search multiple paths simultaneously

Distributed Routing

- Weakness
  - Most existing distributed routing algorithms require each node to maintain global network state → high overhead
  - Algorithms that do not need global state information incur more control messages
  - Hard to design heuristics without complete topology and link state information
  - State information at every node may be inconsistent → loops

Source Routing Algorithms

- Wang-Crowcroft algorithm: find a bandwidth-delay-constrained path
  - All links with bandwidth less than the requirement are eliminated
  - The shortest path in terms of delay is found
  - The path is feasible iff it satisfies the delay constraint

Source Routing

- Ma-Steenkiste algorithms
  - Traffic source is constrained by leaky bucket
  - Link scheduling uses rate-proportional scheduling algorithms: WFQ
  - Delay and delay jitter are related to rate of the connection
- Delay bounded routing

Multi-constrained Path Problem (MCP)

- Given a directed graph G(V, E), a source *s*, a destination *t*, two weight functions $w_1 : E \rightarrow R_0^+$ $w_2 : E \rightarrow R_0^+$
- Two constraints: $c_1 \in R_0^+$ $c_2 \in R_0^+$
- Problem: MCP(G, s, t, $w_1 w_2 c_1 c_2$)
- Find a path *p* from *s* to *t* such that

$$w_1(p) \le c_1 \qquad w_2(p) \le c_2$$

if such a path exists

Multi-constrained Path Problem (MCP)

- Usually NP-complete: both weights are independent, real
- If one weight is integral, the problem is not NP

- Heuristic algorithm: convert the original problem to a new problem using a new integer weight function and a new constraint

- Problem: MCP(G, s, t, $w_1 w_2' c_1 x$)

$$w_2'(u,v) = \left\lceil \frac{w_2(u,v)x}{c_2} \right\rceil$$

Multi-constrained Path Problem (MCP)

- A solution for MCP(G, s, t, $w_1 w_2^{'} c_1 x$) must also be a solution for MCP(G, s, t, w_1, w_2, c_1, c_2)
- Let a path *P* be a solution for MCP(G, s, t, w_1, w_2, c_1, c_2) and l be the length of *P* if

$$w_2(P) \le (1 - \frac{l-1}{x})c_2$$

then *P* is also a solution for MCP(G, s, t, $w_1 w_2^{'} c_1 x$)

Extended Dijkstra's Algorithm

- For each vertex *v* in *V* and integer k in [0,x], a variable d[v,k] is maintained
- d[v,k] is an estimation of the smallest w_1-weight of those paths from *s* to *v* whose w'_2-weights are k
- pev[v,k] is the immediate predecessor of *v* on the path from *s* to *v*

MCP(G, s, t, w_1, w'_2, c_1, x)

Initialize (G, s)

```
Begin
    for each vertex v in V, each i in [0,x] do
     d [v,i] = infinity;
     pev [v,i] = nil;
    for each i in [0,x] do
      d [s,i] = 0;
End
```

Relax (u,k,v)

```
Begin
    k' = k + w'2(u,v);
    if k' ≤ x then
       if d [v,k'] > d [u,k]+w1(u,v) then
          d [v,k'] = d [u,k] + w1(u,v);
          pev [v,k'] = u;
End
```

EDSP (G,s)

```
Begin
    Initialize (G,S);
    N = empty;
    Q = {<u,k> | u in V, k in [0,x]};
    while Q is not empty do
       find <u, k> in Q such that d [u,k] = min {d[u', k'], for all <u', k'> in Q};
       Q = Q – {<u,k>};
       N = N + {<u, k>};
       for each outgoing edge of u, (u,v) in E do
          Relax(u,k,v);
End
```

Guerin-Orda Algorithm

- Imprecise network states based on a probability distribution function
- Every node maintains, for each link *l*, the probability $p_l(w)$ of link *l* having a residual bandwidth of *w* units, $0 \le w \le c_l$
- Bandwidth constrained routing is to find the path that has the highest probability to accommodate a new connection with a bandwidth requirement of *x* units

$$\max \prod_{l=1}^{|P|} p_l(x)$$

$$\min - \sum_{l=1}^{|P|} \log p_l(x)$$

Multicast Routing and QoS Extension

- Weighted graph G=(V,E)
- V: set of nodes
- E: set of links
- $M \subset V$: set of nodes involved in a group of communication → multicast group

Multicast Communications

- One-to-many
- Many-to-many: a group member can be a source, receiver, or both

Multicast Routing

- A multicast tree is usually used to connect a source to all the multicast group members
  - Advantages
- A multicast tree *T*, a subgraph of *G*, spans all the nodes in *M*
- *T* may include relay nodes: non-group-member nodes along a path in the tree

Multicast Routing Problem

- Given a multicast group *M* and possibly a set of optimization objective functions, *O*, multicast routing is to construct, based on the network topology and network state information, a multicast tree *T* that spans all the members in *M* and optimizes the objective functions

QoS Driven Multicast Routing Problem

- Given a multicast group *M* and possibly a set of optimization objective functions, *O*, a set of constraints *C*, multicast routing is to construct, based on the network topology and network state information, a multicast tree *T* that spans all the members in *M*, satisfies the QoS constraints, and optimizes the objective functions
- QoS constraints:
  - end-to-end delay bound
  - inter-receiver delay jitter bound
  - minimum bandwidth
  - packet loss probability
  - combination of different constraints

Constraints

- Link constraints
  - Restrictions on the use of links for route selection
- Tree constraints
  - Bounds on the combined values of a metric along a path from a source to a receiver
  - Bounds on the difference of the combined values of a metric along the paths from the source to any two different receivers
- Classification of constraints
  - Additive
  - Multiplicative
  - Concave

Classification of Multicast Routing Problems

1. Link constrained problem, e.g., bandwidth constrained
2. Multiple link constrained problem: two or more link constraints imposed, e.g., bandwidth and buffer constrained
3. Tree constrained problem, e.g., delay constrained
4. Link and tree constrained routing problem, e.g., delay and bandwidth constrained
5. Link optimization routing problem, e.g., maximization of link bandwidth over the on-tree links

Classification of Multicast Routing Problems

6. Link constrained link optimization, e.g., bandwidth constrained buffer optimization problem
7. Tree constrained link optimization problem, e.g., delay constrained bandwidth optimization
8. Multiple tree constrained routing problem, e.g., delay and inter-receiver jitter constrained routing → NP
9. Tree optimization problem, e.g., Steiner tree problem → NP
10. Link constrained tree optimization problem, e.g., link constrained Steiner tree problem → NP
11. Tree constrained tree optimization problem, e.g., delay constrained Steiner tree problem → NP
12. Link and tree constrained tree optimization problem, e.g., bandwidth and delay constrained Steiner tree problem → NP

Multicast Routing Algorithms

- Centralized vs Distributed
- Source initiated vs Receiver initiated
- Source based vs Core based (shared tree)

Example Algorithms

- Shortest path tree: based on Dijkstra's or Bellman-Ford algorithms
- Minimum spanning tree: Prim's algorithm
- Steiner tree heuristics

Example Algorithm I

- Shortest path tree: based on Dijkstra’s or Bellman-Ford algorithms
  - Minimize the sum of the weights on the links along each individual path from the source to a receiver in the multicast group
  - Weight 1: then least hop tree

Example Algorithm II

- Minimum spanning tree: Prim's algorithm
- A tree that spans all the nodes in the network and minimize the total weight of the tree (i.e., sum of all link weights of the tree)
  - Tree construction starts from an arbitrary node and grows the tree spanning all the nodes in the network
  - At each step, a least cost link connecting an off-tree node to the partially constructed tree is added to the tree
  - examples

Steiner tree heuristics I

- Not the same as minimum spanning tree
- Minimize the total cost of a multicast tree
- Steiner tree heuristics:
  - add one multicast group member at a time;
  - use the least cost path to the partial tree

Steiner tree heuristics II

- INPUT: An undirected graph $G = (V, E)$ and a set of Steiner Points
- $S \subseteq V$
- OUTPUT: A Steiner Tree T_H for G and S
- Step 1: Construct the complete undirected graph $G_1 = (V_1, E_1)$ from G and S, in such a way that $V_1 = S$, and for every $\{v_i, v_j\} \in E_1$, the weight (length/cost) on the edge $\{v_i, v_j\}$ is equal to the length of the shortest path from v_i to v_j
- Step 2: Find the minimal spanning tree T_1 of G_1. (If there are several minimal spanning trees, pick an arbitrary one.)
- Step 3: Construct the subgraph G_s of G by replacing each edge in T_1 by its corresponding shortest path in G. (If there are several shortest paths, pick an arbitrary one.)
- Step 4: Find the minimal spanning tree T_s of G_s. (If there are several minimal spanning trees, pick an arbitrary one.)
- Step 5: Construct a Steiner Tree T_H, from T_s by deleting edges in T_s, if necessary, so that all the leaves in T_H are Steiner points

Constrained Steiner Tree

- Unconstrained Steiner tree algorithm can be used to solve tree optimization problems
- They do not attempt to fulfill the tree constraints on an end-to-end basis
- Constraints: e.g., delay, delay jitter, degree constraints

Constrained Steiner Tree

- G(V,E)
- Two weight functions C(e) and D(e) on edge e
- C(e): positive real cost function on e
- D(e): positive integer delay function on e
- s: source
- S: destination node set
- Δ: delay bound integer value
- Constrained spanning tree T: a tree rooted at s, spans the nodes in S such that for each node v in S

$$\sum_{e \in P(s,v)} D(e) < \Delta$$

$$\sum_{e \in T} C(e)$$ is minimized

Constrained Cheapest Path

- $C_d(v,w)$: the cost of the cheapest path from v to w with delay exactly d

$$C_d(v,w) = \min_{u \in V}\{C_{d-D(u,w)}(v,u) + C(u,w)\}$$

$$P_c(v,w) = \min_{d<\Delta} C_d(v,w)$$

- $P_d(v,w)$ is the delay on the path from v to w and is determined by the constrained cheapest path that corresponds to $P_c(v,w)$
- Compute the all-pairs constrained cheapest paths using a dynamic programming approach

- Find a constrained cheapest path between v and w:
  - i.e., the least cost path from v to w that has a delay less than Δ
- $P_c(v,w)$: the cost on such a path
- $P_d(v,w)$: the delay on such a path
- Build a closure graph G' on a set of nodes N:
  - i.e. a complete graph on the nodes in N with edge cost between nodes v, w in N equal to $P_c(v,w)$ and delay $P_d(v,w)$

Constrained Steiner Tree Algorithm

- Step1: determine the constrained cheapest paths between all pairs of nodes in the set of nodes that includes s and S subject to delay constraint Δ
  - Can be done in polynomial time
- Step 2: construct a closure graph G' on nodes {s} and S
- Step 3: construct a constrained spanning tree of G' using a greedy approach to add edges to a subtree of the constrained spanning tree until all the destination nodes are covered

- Assume v is in the tree constructed thus far
- Consider whether to include some edge adjacent to v
- Use selection functions

$$f_{CD}(v,w) = \begin{cases} \dfrac{C(v,w)}{\Delta - (P(v) + D(v,w))} & \text{If } P(v) + D(v,w) < \Delta \\ \infty & \text{otherwise} \end{cases}$$

$$f_C = \begin{cases} C(v,w) & \text{If } P(v) + D(v,w) < \Delta \\ \infty & \text{otherwise} \end{cases}$$

- P(v) is the delay on the path from s to v in the spanning tree constructed so far

- Step 4: expand the edges of the constrained spanning tree into the constrained cheapest paths they represent and remove any loops that may be caused by this expansion