EE597 Lecture 8/1/16

From the
Last Class:

- Backpressure Scheduling

- The Backpressure Collection Protocol (BCP)

(Max Weight)

→ An algorithm, that under idealized assumptions (such as: global sync, central monitoring, solving NP-hard max weight independent set problem, infinite buffers, etc.) guarantees throughput-optimal performance.

BCP is a practical protocol for collecting data from nodes in a wireless sensor network that is inspired by backpressure scheduling, and removes a lot of the idealized assumptions.

BCP

- does not require time sync
- ~~uses~~ randomized access (from L2 - CSMA layer)
- focuses only on letting nodes make forwarding decisions.
- doesn't assume perfect quality links: hence the ETX term in the weight calculation.

- Simplifies the problem to focus on a single-commodity flow, i.e. all data goes to one sink. (generalizing to handle more flows will require maintaining additional queues, one for each flow).

- does not assume infinite buffers

Recall BCP:

$$w_{ij} = Q_i - Q_j - V \cdot ETX_{ij}$$

find j^* that maximizes w_{ij}.

If $w_{ij^*} > 0$, then send next pkt from i to j.

If $w_{ij^*} \leq 0$, don't send pkts.

may happen temporarily at other nodes due to local congestion

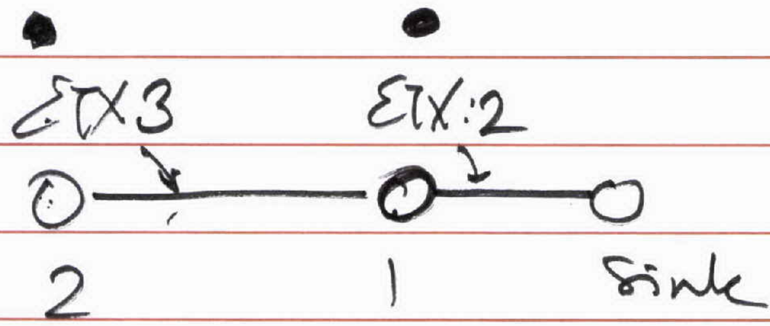
always happens at sink.

- Uses LIFO to serve pkts at each queue instead of FIFO
  - this improves the delay, particularly at low arrival rates

BUT means that some trapped pkts are not delivered (can be virtualized so are not "real" data

carrying pkts)

At steady state, BCP establishes gradients pointing to the sink node. If i^{th} node has shortest ETX path of cost C_i, the # of pkts in the queue establishing this gradient will be $V \cdot C_i$.

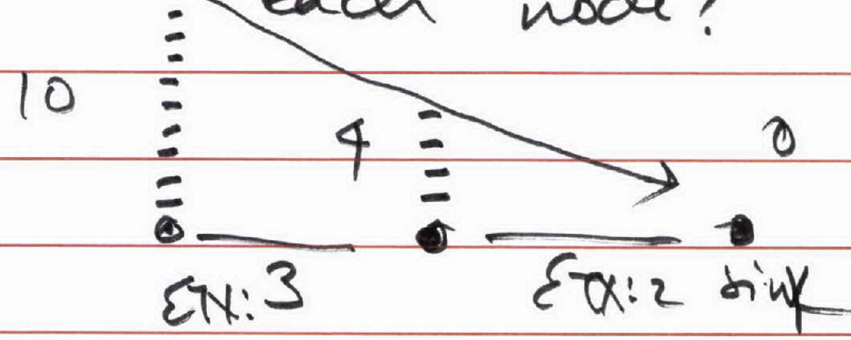


$$Q_2 - Q_1 - 2\cdot 3 = 1 - 0 - 6 = -5$$

$$Q_1 - Q_2 - 6 = -6 < 0.$$

$$Q_1 - Q_{sink} - 2\cdot 2 = 1 - 4 = -3 < 0$$

Steady state — if a large no. of pkts arrive in a burst, & are emptied out, which pkts remain, how many at each node?

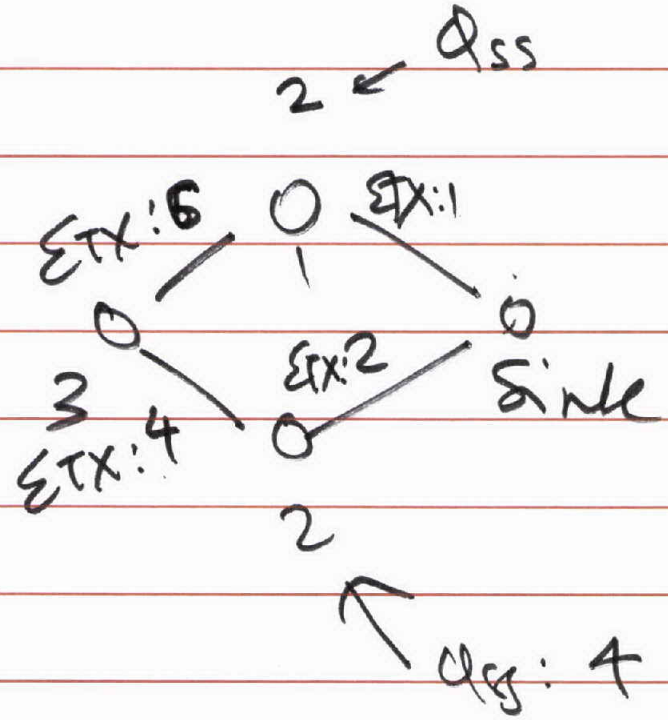


$$Q_i - Q_j - V\cdot ETX_{ij}$$

if $V=2$

assume:
$V=2$

$Q_{ss}:12$



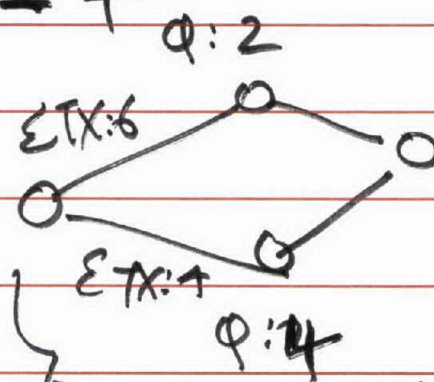
$Q_1 - Q_0 - 2 \geq 0$ for node 1 to move a pkt

$Q_1 - 2 \geq 0$

$Q_1 \geq 2$ for node 1 to move a pkt

$\therefore Q_{ss}^{1} = 2$

Similarly $Q_{ss}^{2} = 4$



$$\left.\begin{array}{l} \text{if } Q_3 - Q_1 - 2\cdot ETX_{31} > 0 \\ \text{OR if } Q_3 - Q_2 - 2\cdot ETX_{32} > 0 \end{array}\right\} \rightarrow \text{queue 3 will transmit a pkt}$$

At steady state: $Q_{ss}^2 = 4$

$Q_{ss}^1 = 2$

$Q_3 - 2 - 12 > 0$ — (1)

$Q_3 - 4 - 8 > 0$ — (2)

$Q_3 - 14 > 0$ — (1')

$Q_3 - 12 > 0$ — (2')

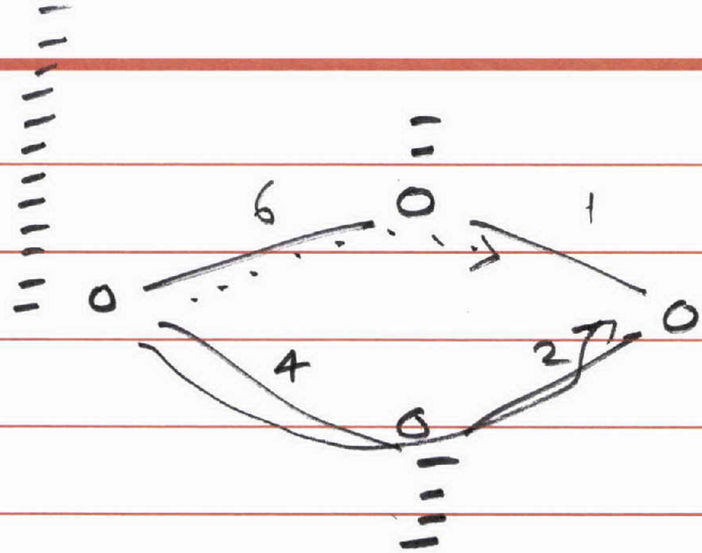
If $Q_3 \leq 12$, neither condition holds.

$Q_3 > 12$ the second condition holds & node 3 will transmit

$\therefore Q_{ss}^3 = 12$

note shortest ETX path for node 3 has ETX = 6.

$V \cdot 6 = 2 \cdot 6 = 12$



If all nodes are at steady state except 1 node which gets an additional pkt, this pkt will

flow to the sink along the shortest path.

When there is congestion along the path, other paths are followed for load balancing.

Announcements:

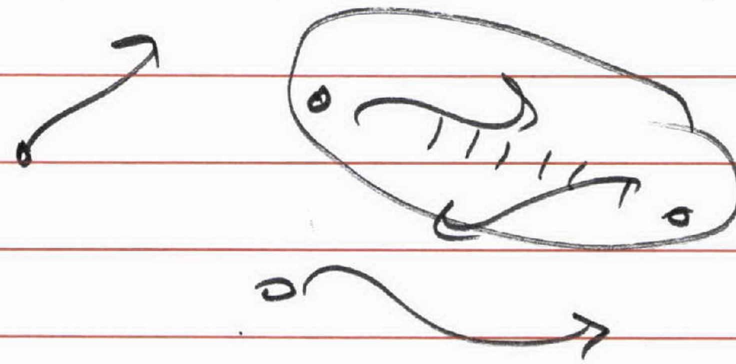
- Today extra pre-taped lecture from 3-4 pm in OHE 136.
  I am not here on Wednesday.
  No regular class on Wednesday.
- Assignment has been posted online, due friday.
- Sample final exam questions
- Exam is on monday, Apr 8 in class — will cover everything up to today's taped lecture, from after the midterm (not cumulative).

Topics:

- sleep cycling for MAC
- Any path routing
- ETX as a metric
- Cooperative Broadcast
- Backpressure scheduling
- Backpressure Collection Protocol
- AODV - read the paper!
- Intermittently Connected Mobile Networks
- TCP — loss notification
  — fairness for wireless
- Alternatives to TCP for Wireless

Intermittently Connected Mobile Networks:

mobile device P2P connectivity
vehicular networks.

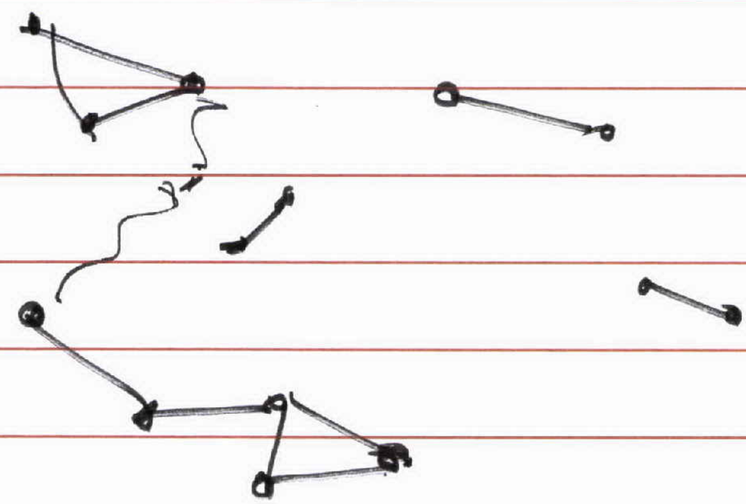


nodes are humans or cars or animals

they are mobile (random or predictable)
& two nodes
can connect to each other
whenever they are w/in
range of each other.

The network Graph is not necessarily connected at any moment in time.

def: $G=(V,E)$ is connected, if there exists a path between any pair of vertice $\in V$.



$\therefore$ standard shortest-path routing, broadcast routing, backpressure scheduling etc. will not work.

Instead, the system requires
store and forward routing
to move data through
the network.

Simplest mechanism: Wait and
deliver:
node A has data for node B.
If A - B are in range:

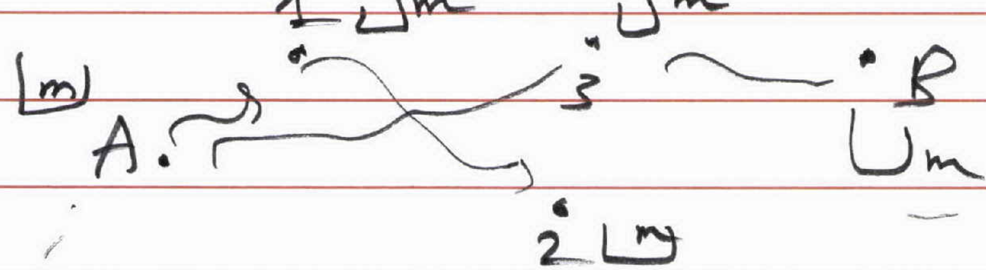
A transfers the data
else
A waits, storing the data.

incurs high latency, efficient in terms
of # of transmission

does not always work.
if A & B do not meet
or meet infrequently.

Alternative: epidemic flooding:

every node i, upon contact with a node j makes a copy of all its stored pkts & transmits them to j.



Some observations:

- if there is any way to get the data from A to B, flooding will deliver it (setting aside BW limits) because <u>every</u> transmission opportunity is used.

- In fact will deliver with <u>lowest</u> possible latency (i.e. along shortest <u>causal</u> path)

flooding	wait & deliver
· Robust	· not robust
· low latency	· high latency
· low efficiency (highest # transmission)	· high ~~efficiency~~ low efficiency (# transmission)

a Tradeoff between Latency & efficiency.

If the data rate of arrivals is high, the low efficiency of flooding means a lower throughput.

Spray and Wait

Initially the source node has a certain no. of "copies" of the pkt : K.

at each encounter, a node that has ~~the~~ k copies, hands over k/2 copies to the encountered node, if ~~that~~ the latter has no copies, so long as $k > 1$.

If $k = 1$, no more copies are made by the node.

increasing K gives lower efficiency & lower latency

decreasing K increases efficiency but also latency.

BWAR - backpressure w/ adaptive redundancy

---

move pkts between nodes i & j only if $Q_i \geq Q_j$

& if queues are small, then copy packets.

gives same latency performance as schemes like Spray & Wait but gives higher throughput

Spray & Wait

first of all, note total
# of transmissions in
network $\leq K-1$

as $K \uparrow$ we are
creating a hybrid between
wait & deliver & flooding.

$K=2 \iff$ wait & deliver.

$K=\infty \iff$ flooding.
(large K)

---

let's try $K=2^{n+1}$, $n=$ # of nodes.

will guarantee each node
always has ≥ 2 copies.

$\Rightarrow$ behaves exactly as
flooding.

A:16

eg. K = 16. say src : A

A - 1 A:8 1:8

1 - 2 A:8 1:4 2:4

A - 3 A:4 1:4 2:4 3:4

2 - 3 A:4 1:4 2:4 3:4

2 - 5 A:4 1:4 2:2 3:4 5:2

5 - 4 A:4 1:4 2:2 3:4 4:1 5:1

4 - 6 A:4 1:4 2:2 3:4 4:1 5:1 6:0

5 - 6 A:4 1:4 2:2 3:4 4:1 5:1 6:0

A - 6 A:2 1:4 2:2 3:4 4:1 5:1 6:2

3 - B A:2 1:4 2:2 3:2 4:1 5:1 6:2 B:2

delivered

note that sharing k/2 copies only requires transmitting one copy & updating counters on each node

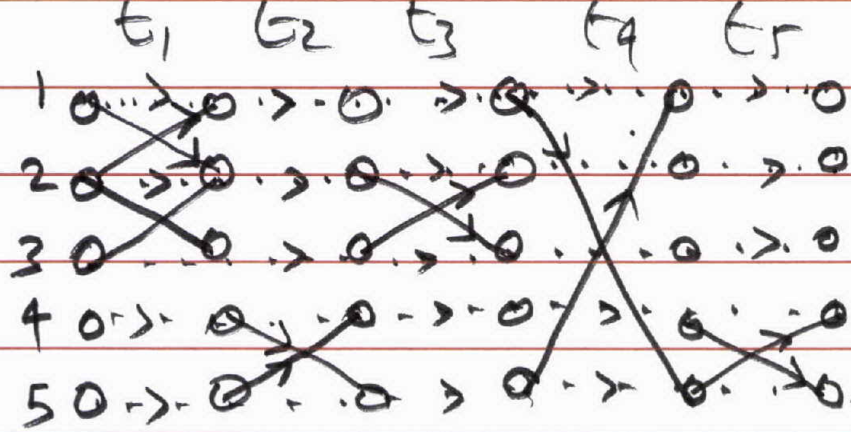
How to
model & analyze the mechanisms for DTN / ICMN ?

Delay-tolerant networks

intermittently connected mobile networks.

deterministic, known encounters:

trellis diagram showing connectivity over time:



direction of links show causality

Can reason about path formation, deliverability, routing, delay, efficiency etc. by studying this path.

Termination Criterion :

even if destination gets the message, due to the distributed nature of ICMN, other nodes may continue to fwd / make copies.

- either do nothing
- can set timer.

- send an ack flood to suppress further ~~[illegible]~~ transmissions.

A simple mobility model / encounter model to analyze ICMN protocols: <u>i.i.d encounter model.</u>

At each time, any pair of the $\binom{n}{2}$ pairs of nodes are selected at random to meet each other

<u>Illustration:</u> Let's use this model to compute the expected time <u>to delivery</u> for flooding. Simpler yet, time till all nodes get the pkt.

The process starts w/ one node having the pkt

$$1 \xrightarrow{T_{12}} 2 \longrightarrow 3 \longrightarrow \cdots \; k \longrightarrow k+1 \; \cdots \longrightarrow n$$

random time $\longrightarrow T_{k,k+1}$

$$E[T] = \sum_{k=1}^{n-1} E[\underbrace{T_{k,k+1}}]$$

total time to deliver to all nodes

to analyze this, we construct a geometric r.v. X w/ some success prob. p,

$$E[X] = \frac{1}{p}$$

TA continued 3-4 pm today, OHE136.