The Gossip Objects (GO) Platform

- Ýmir Vigfússon
  IBM Research Haifa Labs
- Ken Birman
  Cornell University
- Qi Huang
  Cornell University
- Deepak Nataraj
  Cornell University
Gossip

- **Def:** *Exchange information with a random node once per round.*

- Has appealing properties:
  - Bounded network traffic.
  - Scalable in group size.
  - Robust against failures.
  - Simple to code.

- Per-node scalability?
  - When # of groups scales up, lose...
The GO Platform

- App
- App
- App
- App

- Rumor Queue
- Gossip Mechanism
- Event Loop
- GO Heuristic
- Neighbor Lists

Node

Network
Random gossip

• **Recipient selection:**
  – Pick node $d$ uniformly at random.

• **Content selection:**
  – Pick a rumor $r$ uniformly at random.
Observations

• **Gossip rumors usually small:**
  – Incremental updates.
  – Few bytes hash of actual information.

• **Packet size below MTU irrelevant.**
  – *Stack* rumors in a message.
  – But which ones?
Random gossip w/stacking

• **Recipient selection:**
  – Pick node $d$ uniformly at random.

• **Content selection:**
  – Fill packet with rumors picked uniformly at random.
Further ingredients

• **Rumors can be delivered indirectly.**
  – Uninterested node might forward to an interested one.
  – Could use longer dissemination paths.

• **Traffic adaptivity.**
  – Some groups have more to talk about than others.
  – Could monitor traffic and optimize to allocate bandwidth.
GO Heuristic

• **Recipient selection:**
  – Pick node $d$ biased towards higher group traffic.

• **Content selection:**
  – Compute the *utility* of including rumor $r$
    • Probability of $r$ infecting an uninfected host when it reaches the target group.
  – Pick rumors to fill packet with probability proportional to utility.
• Recipient selection:
  – Pick nodes towards higher group traffic.
• Content selection:
  – Compute the utility of including rumor $r$.
  – Compute the probability of $r$ infecting an uninfected host when it reaches the target group.
  – Pick rumors to fill packet with probability proportional to utility.

GO Heuristic

Include $r$?
Simulation

- Simulated but ‘clean’ topology shows benefit of the GO strategy.

```
       s

       d

```

- Individual rumors delivered
- Rumors delivered indirectly
Real-world Evaluation

- 55 minute trace of the **IBM WebSphere Virtual Enterprise (WVE) Bulletin Board layer.**
  - 127 nodes and 1364 groups

![Rumors generated per round in the trace](image)
Real-world Evaluation

- IBM WVE trace (127 nodes, 1364 groups)
Real-world Evaluation

- IBM WVE trace (127 nodes, 1364 groups)
Real-world Evaluation

- IBM WVE trace (127 nodes, 1364 groups)

Individual rumors delivered vs. messages sent
Conclusion

• GO implements novel ideas:
  – Per-node gossip platform.
  – Rumor stacking.
  – Utility-based rumor dissemination.
  – Traffic adaptivity.

• GO gives per-node guarantees.
  – Even when the # of groups scales up.

• Experimental results are compelling.
  – We plan to use GO as the transport for the Live Objects platform.