Generic emergent overlays in arbitrary peer ID spaces

Wojciech Galuba, Karl Aberer

Distributed Information Systems Laboratory, EPFL
Lausanne, Switzerland
http://lsirwww.epfl.ch/
The problem: scalable overlays

- Underlying blue network (e.g. TCP/IP)
- Red peers come and go
- Goal: allow the peers to communicate with each other
- Solution: interconnect the peers in an overlay (red links)
Structured vs. unstructured

- Unstructured overlays
  - More freedom in network formation
    - But too much freedom leads to power-law effects
  - Very simple protocol
  - No delivery guarantees, flooding necessary

- Structured overlays
  - Global, rigid topology assumed
  - Topology continuously maintained
    - To provide performance guarantees
  - Complex protocols, analysis hard

- Is there a middle ground? Can we have both:
  - the flexibility and simplicity of the unstructured overlays and
  - the performance of the structured ones
Structured overlay design pattern

- Each peer has an ID
- There is some notion of distance between the IDs
- Greedy routing: in each hop the message is brought closer to the destination
  - „closer” in terms of the ID distance
- Maintenance ensures that this greedy next hop is always possible
Our approach

- Find the minimal set of rules for running a robust and scalable overlay
- Abstract out the peer ID space
  - Now: ID space = any metric space
  - Application defines the ID space
- Greedy loop-avoiding routing
  - Loop-avoidance: each message remembers visited nodes
  - Robustness to routing problems
- Maintenance is lazy (reactive)
  - Only when routing problems occur
Routing convergence rate

- The rate \textit{cvg} at which messages approach their destinations

\[ \text{cvg} = \frac{A}{B} \]

- Application defines the minimum required \textit{cvg}
Maintenance

- Triggered when cvg is below the required level

1. cvg to small

2. send connection request, route in the overlay

3. request rcvd, establish new connection, cvg better for the new connection

- Connections opened only when there is overlay traffic that needs it
  - There is only as much maintenance as necessary
- Small-world topology emerges
  - Not hard-coded into the algorithm
Scaling

- Logarithmic scaling:
  - Average path length
  - Average node degree
  - Maximum node degree
- Independent of the ID space
Tolerance to churn

- Simple setup:
  - Poisson arrivals and departures at equal rates
  - 1000 nodes
- Tested with Kademlia and BitTorrent churn models [Stutzbach et al.]
  - Results similar
- Loop-avoidance is key to robustness
Degree vs. path length tradeoff

- Can be smoothly controlled by the application
  - Based on the „required cvg” setting

- This allows for easy control of the tradeoff between latency and maintenance bandwidth
The forging of the ring

- Structured overlays are locally tightly interconnected:
  - those links are crucial for routing reliability
  - e.g. the ring in Chord
- Locality – a measure of local interconnectedness
- Local structures emerge in our overlay:
  - even though they are not hard-coded in the algorithm
Watts-Strogatz model

Regular  Small-World  Random

Increasing randomness

This is what our maintenance algorithm does
The road ahead

- Required cvg can vary for:
  - Each node, each message type or depending on churn rate
  - Different topologies possible
- The ID space can vary over time (latency = distance?)
- "Power of k random choices"-type load balancing
  - Exploiting the inherent flexibility of connection opening
- Building a DHT on top of our overlay
  - Replication exploits small-world properties
  - DHTs with arbitrary key spaces
    - Obvious advantage for data storage
    - Range queries easier? Better than space-filling curves?
- First PlanetLab deployments
  - Precise measurement of overheads and maintenance traffic
Conclusions

- Two simple rules:
  - Greedy routing with loop-avoidance
  - Open new connections when $cvg$ too small
- Result: scalable and robust overlay
- Small-world topology is completely emergent
  - not hard-coded in the algorithm
  - emerges independently of the ID space
- Many knobs to turn to suit the application needs:
  - Works in any metric ID space
  - Control over the degree vs. path length tradeoff