Formal verification of distance vector routing protocols
Routing in a network

(Find the cheapest route from Source to Destination)

\[ L(i, j) = \text{Cost of direct link } i \rightarrow j. \]

\[ R(a, b) = \text{Cost of route from } a \text{ to } b. \]

\[ R(a, b) = \min \{ L(a, k) + R(k, b) \} \]
Outline

• RIP (Routing Information Protocol)
  – Internet routing protocol

• AODV (Ad-hoc On-demand Distance Vector routing)
  – Used for mobile ad-hoc networking.
Distance-vector routing in RIP

Initially
A: 0  B: 5  C: \(\infty\)
A: 5  B: 0  C: 7
A: \(\infty\)  B: 5  C: 0

After exchange
A: 0  B: 5  C: 12
A: 5  B: 0  C: 7
A: 12  B: 5  C: 0
Routing table: Each node maintains the cost of route to every other node
Initially: All nodes know cost to neighbors
Desired Final Goal: All nodes know cost to all other nodes

while(1)
{
    Nodes periodically send their routing table to every neighbor;
    R(a, b) = min{ L(a, k) + R(k, b) };
}
Count to Infinity

After exchange

A: 0  B: 5  C: 12
A: 5  B: 0  C: 7
A: 12  B: 5  C: 0

C: 12
C: 12+5=17
C: 17+5=22
Poisoned reverse

Works for loops of two routers (adds more cases for Verification)

RIP limitation: Doesn’t work for loops of three or more routers
Infinity = 16

• Since we can’t solve the loop problem
  – Set Infinity to 16
• RIP is not to be used in a network that has more than 15 hops.
Convergence

- Convergence:
  - All nodes eventually agree upon routes

- Divergence:
  - Nodes exchange routing messages indefinitely.

- Ignore topology changes
  - We are concerned only with the period between topology changes.
Some definitions

- Universe is modeled as a bipartite graph
  - Nodes are partitioned into routers and networks
  - Interfaces are edges.
  - Each router connects to at least two networks.
  - Routers are neighbors if they connect to the same network.
- Actually, we can do away with bipartite graph by assuming that router = network (i.e. each network has one router).
- An entry for destination $d$ at a router $r$ has:
  - $\text{hops}(r)$: Current distance estimate
  - $\text{nextR}(r)$: next router on the route to $d$.
  - $\text{nextN}(r)$: next network on the route to $d$. 
More definitions

• \( D(r) = 1 \) if \( r \) is connected to \( d \)
  \[ = 1 + \min \{ D(s) | s \text{ is a neighbor of } r \} \]

• \( k \)-circle around \( d \) is the set of routers:
  \[ C_k = \{ r | D(r) \leq k \} \]

• **Stability:** For \( 1 \leq k \leq 15 \), universe is **\( k \)-stable** if:
  (S1): Every router \( r \) in \( C_k \) has \( \text{hops}(r) = D(r) \)
  Also, \( D(\text{nextR}(r)) = D(r) - 1 \).
  (S2): For every router \( r \) outside \( C_k \), \( \text{hops}(r) > k \).
Convergence

- Aim of routing protocol is to expand $k$-circle to include all routers
- A router $r$ at distance $k+1$ from $d$ is $(k+1)$-stable if it has an optimal route:
  - $\text{Hops}(r) = k+1$ and $\text{nextR}(r)$ is in $C_k$.
- Convergence theorem (Correctness of RIP)
  - For any $k < 16$, starting from an arbitrary state of the universe, for any fair sequence of messages, there is a time $t_k$, such that the universe is $k$-stable at all times $t \geq t_k$. 
Tools

• HOL (higher order logic)
  – Theorem prover (more expressive, more effort)
• SPIN
  – Model checker (less expressive, easier modeling)
• Number of routers is infinite
  – SPIN would have too many states
  – States reduced by using abstraction
Lemmas in convergence proof

- Proved by induction on $k$.
  - Lemma 1: Universe is initially 1-stable. (Proved in HOL).
  - Lemma 2: Preservation of Stability. For any $k < 16$, if the universe is $k$-stable at some time $t$, then it is $k$-stable at any time $t' \geq t$. (Proved in HOL).
  - Lemma 3: For any $k < 15$ and router $r$ such that $D(r) = k + 1$, if the universe is $k$-stable at some time $t_k$, then there is a time $t_{r,k} \geq t_k$ such that $r$ is $(k+1)$-stable at all times $t \geq t_{r,k}$. (Proved in SPIN)
  - Lemma 4: Progress. For any $k < 15$, if the universe is $k$-stable at some time $t_k$, then there is a time $t_{k+1} \geq t_k$ such that the universe is $(k+1)$-stable at all times $t \geq t_{k+1}$. (Proved in HOL).
Abstraction

- To reduce state-space for SPIN
- Abstraction examples:
  - If property P holds for two routers, then it will hold for arbitrarily many routers.
  - Advertisements of distances can be assumed to be $k$ or $k+1$.
- Abstraction should be:
  - **Finitary**: should reduce system to finite number of states
  - **Property-preserving**: Whenever abstract system satisfies the property, concrete system also satisfies the property
Abstraction of universe

Concrete system with many routers

Abstract system with 3 routers

hops > k+1

hops = k+1

hops < k+1

Router processes Updates

Hop-count is \{LT, EQ, GR\}

Advertiser send updates
Bound on convergence time

- **Theorem:** A universe of radius $R$ becomes $15$-stable within time $= \min\{15, R\} \star \Delta$. (Assuming there were no topology changes).

- After $\Delta$, weakly 2-stable
- After $2\Delta$, weakly 3-stable
- After $3\Delta$, weakly 4-stable
- After $4\Delta$, weakly 5-stable
- …
- After $(R-1)\Delta$, weakly $R$-stable
- After $R\Delta$, $R$-stable
Weak stability

• Universe is weakly $k$-stable if:
  – Universe is $k$-1 stable
  – For all routers on $k$-circle: either $r$ is $k$-stable or $\text{hops}(r) > k$.
  – For all routers $r$ outside $C_k (\mathcal{D}(r) > k)$,
    $\text{hops}(r) > k$.

• By using weak stability, we can prove a sharp bound
Lemmas in Proof of timing bound

• **Lemma 5**: Preservation of weak stability. For any $2 \leq k \leq 15$, if the universe is weakly $k$-stable at some time $t$, then it is weakly $k$-stable at any time $t' \geq t$.

• **Lemma 6**: Initial Progress. If the topology does not change, the universe becomes weakly 2-stable after $\Delta$ time.

• **Lemma 7**: For any $2 \leq k \leq 15$, if the universe is weakly $k$-stable at some time $t$, then it is $k$-stable at time $t + \Delta$. 
Lemma 8: Progress. For any $2 \leq k \leq 15$, if the universe is weakly $k$-stable at some time $t$, then it is weakly $(k+1)$-stable at time $t + \Delta$. 
AODV

Routes are computed on-demand to save bandwidth.
AODV

- Each route request has a sequence number for freshness.
- Among two routes of equal freshness, smaller hop-count is preferred.
- Property formally verified is loop freedom
  - Above conditions mean a lot of cases need to be checked
Searching for loop formation

• The 3-node network shown previously, is run in SPIN.
  • $\Omega(!((next_D(A)==B) \land (next_D(B)==A)))$
• Four ways of loop formation are found.
• Standard does not cover these cases.
• Formal verification can aid protocol design.
Ways of loop formation

- To get an idea of case-analysis required, loops can be formed by:
  - Route reply from B to A getting dropped.
  - B deleting route on expiry.
  - B keeping route but marks it as expired.
  - A not detecting a crash of B.

- Loop was avoided by:
  - B keeping route as expired, incrementing the sequence number and never deleting it.
  - Is a good indicator of a loop-free solution.
Guaranteeing AODV loop freedom

- Based on the avoidance of loops for 3 nodes, we assume:
  - Nodes never delete routes, increment sequence number of expired routes, detect crashes immediately.
- Based on these assumptions, loop freedom is proved.
- Theorem: Consider an arbitrary network of nodes running AODVv2. If all nodes conform to above assumption, there will be no routing loops.
Abstraction

• Abstract sequence number is \{GR, EQ, LT\}
• Abstract hop count is \{GR, EQ, LT\}
• Abstract next pointer is \{EQ, NE\}
• Lemma 9: If \(t_1 \leq t_2\) and for all \(t\): \(t_1 < t \leq t_2\), \(\neg \text{restart}(n)(t)\), then:

\[
\text{seqno}_d(n)(t_1) \leq \text{seqno}_d(n)(t_2)
\]

• Lemma 10: If \(t_1 \leq t_2\) and \(\text{seqno}_d(n)(t_1) = \text{seqno}_d(n)(t_2)\), and for all \(t\): \(t_1 < t \leq t_2\), \(\neg \text{restart}(n)(t)\), then \(\text{hops}_d(n)(t_1) \geq \text{hops}_d(n)(t_2)\)
Adding to abstraction

• The following lemma involves two nodes.
• Abstract sequence number is \{GR, EQ, LT\} x \{EQ, NE\}
• Abstract hop count is \{GR, EQ, LT\} x \{EQ, NE\}
• Abstract next pointer is \{EQ, NE\} x \{EQ, NE\}
• Lemma 11: If next_d(n)(t)=n’, then there exists a time lut \leq t, such that:
  – seqno_d(n)(t) = seqno_d(n)(lut)
  – 1 + hops_d(n)(t) = hops_d(n’)(lut)
  – For all t’: lut < t’ \leq t . ¬ restart(n’)(t’).
Conclusion

• Specific technical contributions
  – First proof of correctness of the RIP standard.
  – Statement and automated proof of a sharp real-time bound on RIP convergence
  – Automated proof of loop-freedom for AODV.