D-CART: Dynamic Change Analysis with Routing Traces

Changements de routes

Pascal Mérindol, Pierre David, Jean-Jacques Pansiot: Université de Strasbourg
François Clad, Pierre François: Cisco Systems
Stefano Vissicchio: Université catholique de Louvain-la-Neuve

merindol@unistra.fr
http://icube-reseaux.unistra.fr/dcart
The practical problem

Design of the platform

Measurements & Analysis

Theoretical formulation of the solution

Problem definition

Greedy Backward Algorithm

\[ c := (\min_{x \in L} (\Delta(x)), \max_{x \in L} (\Delta(x))) \]
A Brief History of our Collaboration with RENATER

- **2011-14: Reveal, study and solve transient routing loops in LS-routing**
  - Ph.D. Thesis topic of François Clad (Unistra -> Post Doc Cisco)
    - Theoretic and incremental solutions: no protocol changes required!

A loopy illustration on RENATER:
the link Bordeaux-Nantes fails and the combination between pre- and post-routes triggers up to four transient forwarding loops for the pair Toulouse -> Quimper!

- **2014-16: Understand routing changes in general**
  - What are typical loss durations? their impacts? etc.
    - How minimize such periods?
Why D-CART?..and why this way?

- **Troubleshoot your IP network**
  - IS-IS configuration in particular & possible extensions for OSPF, MPLS, BGP, etc

- **Improve performances of your IP network**
  - modify configurations for improving the reachability and routing delays
    - avoid routing loops and reduce cut/blackhole periods due to routing changes
    - use better routing paths according to the traffic load (weight changes)

- **Develop specific monitoring primitives and re-use existing tools**
  - open software provided by the networking research community (GPL)!

- **Avoid measurement interferences**
  - dedicated but common hardware infrastructure at a (really) limited cost
D-CART: main characteristics

• **3 Sources of data that can be more or less correlated**
  - IS-IS routing states of all routers
  - Active directed ping-like measurements
    • + error messages
  - Maintenance and incident tickets

• **Open Platform Design**
  - using low cost Raspberry-PI hardware directly plugged to routers
  - targeting IPv4 intra-domain routing events in particular…
  - …but extensible in any directions in theory!
How D-CART works?.. The big picture!

- **Passive monitoring**
  - a silent IS-IS listener (an extension of Sprint’s work) and a tickets feed

- **Active monitoring**
  - a bunch of 16 Ras-PIs logically organized as a full-mesh (4 are located in PARIS)
How D-CART works?

- **Main Software Components**
  - **Probers Manager**
    - control/check probers and record their status
  - **Network Dynamicity Collector**
    - manage the listener output: filter and associate LSPs
  - **Measurements Collector**
    - manage most interesting routing events: *error messages* (TTL Excd. and Dest. Unreach.), *losses & de-sequencing*, *delay changes* and *path changes*.
  - **RDBMS (PostgreSQL)**
    - record events and ease correlations between them
  - **Post processor**
    - perform statistics about events: join distinct sources of data
Our set of specific measurement primitives

- **D-CART current design**
  - smart directed ping-like probing
    - -> get evidences and locations of transient routing loops!
  - Equal Cost Multi-Path aware probing
    - -> measure accurately all possible paths
      - need to use multiple IP address (load balancing performed at the IP level)
  - NTP synchronisation (10ms at worst, ~1ms in practice)
    - -> to compute one way delays and allows correlations among data sources

- **D-CART current calibration**
  - probing frequency: the highest possible -> 40 ms…mainly a hardware tradeoff
  - towards a low amount of logs: scalability (30GB for 4 months of overall data)
Listener vs. Losses

• What is the share of routing events triggering losses?

- a: adjacency down
- A: adjacency up
- l: (bi-dir) link down
- L: (bi-dir) link up
- r: router down
- R: router up
- W: weight change

Even Link up triggers many losses (and so micro-loops)!
why such a difference with r/Router and a/Adjacency changes?
• for a/A, we can’t observe all of them…(no probers for each leaf)
Listener vs. Losses (link focus)

- What is the distribution of worst losses (max durations) related to routing events?

- For sure, it is worst for link down than for Link up…

- mainly due to blackhole periods occurring for down events
Losses vs. Listener (link focus)

- What are the distributions and shares of routing events related to losses?

- The vast majority of short-time losses are due to congestions…
- …long ones are “hopefully” mainly related to routing events!
An illustration of listener vs. probing correlation

- About the origin of routing change related losses

**The “setup”: a link is flapping between Nantes and Vannes**

- 3 potential loops in a row

The longest cut occurs at the last link **UP**: only due to micro-loops!
Zoom on the last link up

The “setup”

Timelines for the 3 destination prefixes (the probers) and loop locations analysis

More than one second of traffic interruption! <- FIB update order
Simple comparison between listener and tickets #events?

- The listener provides a better granularity than tickets
  - and it is automatic!

<table>
<thead>
<tr>
<th>type</th>
<th>nevents</th>
<th>rel_freq</th>
<th>freq_per_day</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>239</td>
<td>0.20568</td>
<td>2.096</td>
</tr>
<tr>
<td>L</td>
<td>872</td>
<td>0.75043</td>
<td>7.649</td>
</tr>
<tr>
<td>R</td>
<td>39</td>
<td>0.03356</td>
<td>0.342</td>
</tr>
<tr>
<td>W</td>
<td>12</td>
<td>0.01033</td>
<td>0.105</td>
</tr>
</tbody>
</table>

Listener events

<table>
<thead>
<tr>
<th>ntick</th>
<th>freq_per_day</th>
<th>type</th>
<th>subtype</th>
</tr>
</thead>
<tbody>
<tr>
<td>52</td>
<td>0.29050</td>
<td>incident</td>
<td>link</td>
</tr>
<tr>
<td>12</td>
<td>0.06704</td>
<td>incident</td>
<td>router</td>
</tr>
<tr>
<td>72</td>
<td>0.40223</td>
<td>maintenance</td>
<td>link</td>
</tr>
<tr>
<td>33</td>
<td>0.18436</td>
<td>maintenance</td>
<td>router</td>
</tr>
</tbody>
</table>

Tickets events

2.36 link up/down per day
aggregation per link per day // flapping
What are next plans?

- Recalibrate probers on the fly!
  - e.g. modify probing frequency of a subset of probers according to the listener
  - e.g. reprogram probers with specific measurement mechanisms for planned events
About the D-CART platform: summary and future works

• Basically, it is an open software monitoring platform
  • the hardware doesn’t matter that much…
    • …as long as it is not specific and powerful enough
  • it is generic enough to support all kind of specifics measurements
    • it ensures flexibility, scalability and extensibility
  • we get numerous loop evidences and even more!
    • automatic ticketing system, failure prediction (flapping), bugs detection, etc.

• We envision to extend it in several directions
  • across multiple IGP networks…Geant and more?
  • comparing IPv4 and IPv6 traffic forwarding performance differences
  • TCP-like measurements: routing changes side effects on real applications
  • focus on BGP modifications effects, etc.
D-CART

- The practical problem
- Design of the platform
- Measurements & Outcomes
- Problem definition

Theoretical formulation of the solution

Greedy Backward Algorithm

Constraint $c$ associated with a loop $L$
$$
c := \left(\min_{\forall x \in L} (\Delta(x)), \max_{\forall x \in L} (\Delta(x))\right)
$$
About our solution

• **Objective: get rid of transient forwarding loops**
  • dealing with all kind of routing changes: up/down/weight changes * link/router
  • at least for planned events (but works well in any cases in theory)

• **Constraint: design a practical solution**
  • No protocol changes
    • incremental solution
  • No explicit synchronisation among routers required
    • ≠ oFIB or other schemes
  • Efficient and scalable at all levels
    • ≠ ships in the night that works and is designed for the whole network
D-CART -> GBA

- But first, let us recall the problem on a detailed but simple illustration...

- Can we play with link weights to progressively shift the traffic and update router’s FIB in the right order?
The “Delta”: equilibrium values at which routes change

For a given destination $d$, we define for each router a pivot increment, denoted $\Delta_d(x)$:
\[ \forall x \in N, \; \Delta_d(x) = C'(x, d) - C(x, d) \]

<table>
<thead>
<tr>
<th>$x$</th>
<th>$C(x)$</th>
<th>$C'(x)$</th>
<th>$\Delta_{SEAT}(x)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEAT</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>LOSA</td>
<td>1342</td>
<td>1342</td>
<td>0</td>
</tr>
<tr>
<td>SALT</td>
<td>913</td>
<td>913</td>
<td>0</td>
</tr>
<tr>
<td>HOUS</td>
<td>3047</td>
<td>3047</td>
<td>0</td>
</tr>
<tr>
<td>KANS</td>
<td>2242</td>
<td>2242</td>
<td>0</td>
</tr>
<tr>
<td>CHIC</td>
<td>2931</td>
<td>5477</td>
<td>2546</td>
</tr>
<tr>
<td>ATLA</td>
<td>3976</td>
<td>4432</td>
<td>456</td>
</tr>
<tr>
<td>WASH</td>
<td>3836</td>
<td>6176</td>
<td>2340</td>
</tr>
<tr>
<td>NEWY</td>
<td>3931</td>
<td>6453</td>
<td>2522</td>
</tr>
</tbody>
</table>

General definitions:

<table>
<thead>
<tr>
<th>$G(N, E, w)$</th>
<th>Directed weighted graph representing the network</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C(x, d), C(x)$</td>
<td>Cost of a shortest path (distance) from $x$ to $d$ before the change</td>
</tr>
<tr>
<td>$C'(x, d), C'(x)$</td>
<td>Cost of a shortest path (distance) from $x$ to $d$ after the change</td>
</tr>
</tbody>
</table>
Problem definition (system constraints)

- Notion of sequence of increments

**Theorem**

A monotonic weight update sequence $S$ prevents a transient loop $L = \{x_1, x_2, \ldots, x_t\}$ for a destination $d$, if and only if there exists $e \in S$ such that:

$$\text{MIN}_{x \in L}(\Delta_d(x)) < e < \text{MAX}_{x \in L}(\Delta_d(x))$$

The sequence must contain a weight update that makes one router involved in the loop to completely reroute, while another is still in its initial routing state.

- $e < 456$ Nothing happens...
- $456 < e < 2546$ Atlanta reroutes.
- $e > 2546$ Chicago reroutes.
Several destinations, several loops: how to treat them?

- Intervals intersections for all destinations and loops

Minimum loop-free sequence for the link \((CHIC, KANS)\): \(S = \{470\}\)
And with more dimensions?

- What about a router-wide operation? Here with simply 2 outgoing links.

Constraints associated with a loop $L$

$$c := \left( \min_{x \in L} \Delta(x), \max_{x \in L} \Delta(x) \right)$$

$$c_1 = \left( \left( \frac{5}{3} \right), \left( \frac{7}{5} \right) \right)$$
A greedy algorithm that works in reverse fashion

- because it does not (always!) work in a forward fashion...

**Greedy Backward Algorithm (GBA)**

At each step, retrieve the maximum value on each index among the lower bounds of the remaining constraints.

\[ S_{GBA} = \{ (9, 4), (13, 20) \} \]

**Theorem**

Given a set of loop-constraints, GBA computes a minimal sequence of weight updates preventing all associated convergence loops.
GBA/D-CART summary

- Transient loops impact evaluation
  - Loops do occur and impact the traffic in ISP networks

- Improvement of the existing approach
  - Sequence minimality with polynomial time algorithms
  - Efficient implementation

- Generalization to node-wide operations
  - Practical solutions to deal with routing instabilities

=> To be tested in RENATER soon!
http://icube-reseaux.unistra.fr/dcart

• We get GEANT GN4 Open Call funds to make the story continue…
  • …we need manpower for software development and platform management!
  • and more robust hardware (SD cards of R-PI are not!)
    • currently D-CART is down except the listener (it is running on a real server)

• Are you interested in collaborating with us?
  • or just discuss…

• Have you any suggestions?
  • or simply questions?

merindol@unistra.fr