DRoP: DNS-based Router Positioning

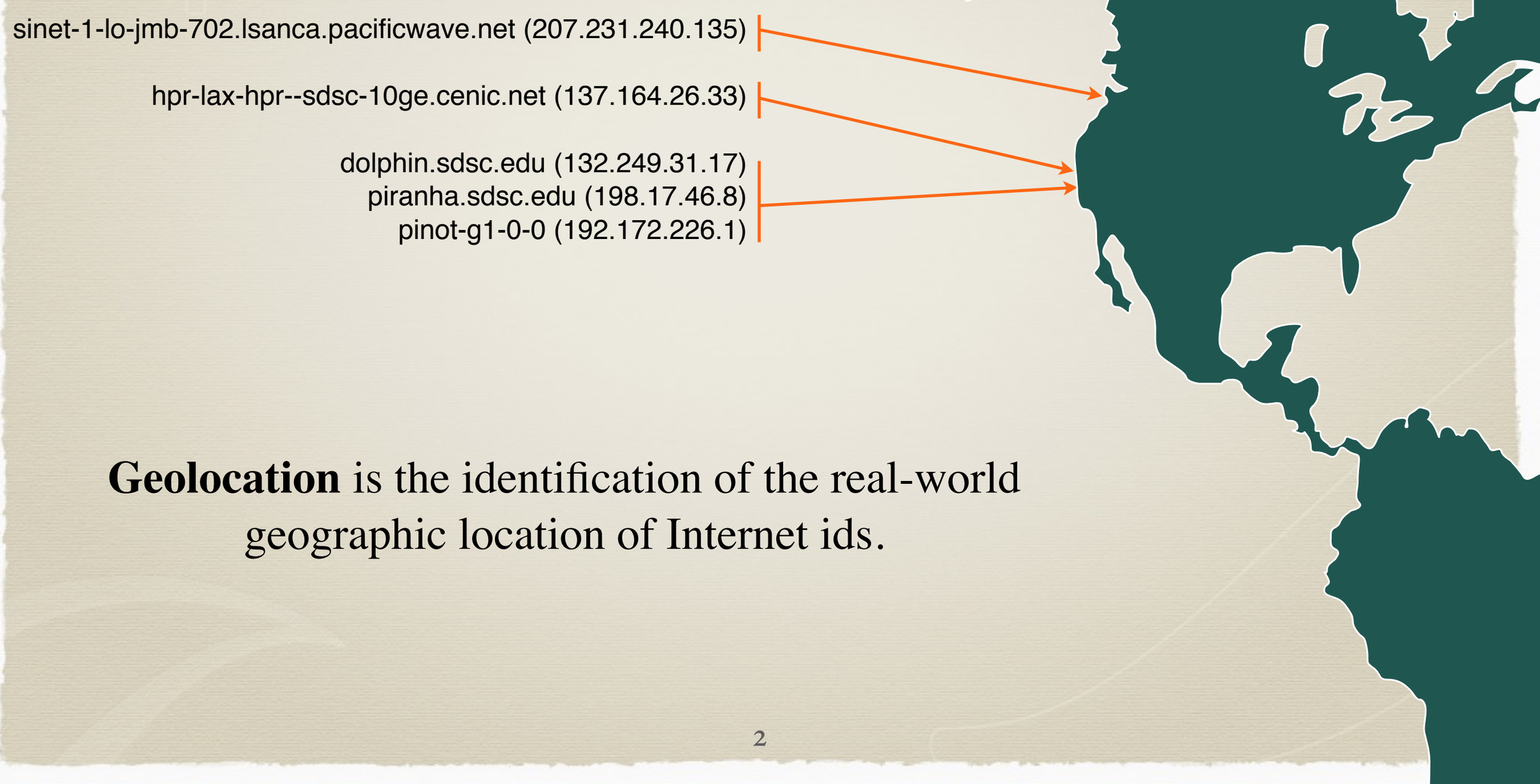
Bradley Huffaker, Marina Fomenkov, kc claffy

DDec: DNS Decoding

Ken Keys, Bradley Huffaker

DHS site vist
June 18, 2014

geolocation



Geolocation is the identification of the real-world geographic location of Internet ids.

geolocation solutions

Solutions	cost
undns$_f$	-
RIR$_f$	-
Software77$_f$	-
HostIP$_f$	-
IPligence	$
Cyscape	$$
MaxMind GeoIP	$$$
MaxMind GeoLite$_f$	-
IPInfoDB$_f$	-
Digital Envoy	$$$$

$_f$marks the free datasets

$ = $1-$300 $$ = $300-$90
$$$ = $900-$1800 $$$$ = $1800+

uses hostname hints

geolocation problems

common problems

* often inaccurate for routers
* often inaccurate outside the US
* better databases are more expensive

undns uses hostnames with geographic hints

sinet-1-lo-jmb-702.lsanca.pacificwave.net (207.231.240.135)

hpr-lax-hpr--sdsc-10ge.cenic.net (137.164.26.33)

dolphin.sdsc.edu (132.249.31.17)
piranha.sdsc.edu (198.17.46.8)
pinot-g1-0-0 (192.172.226.1)

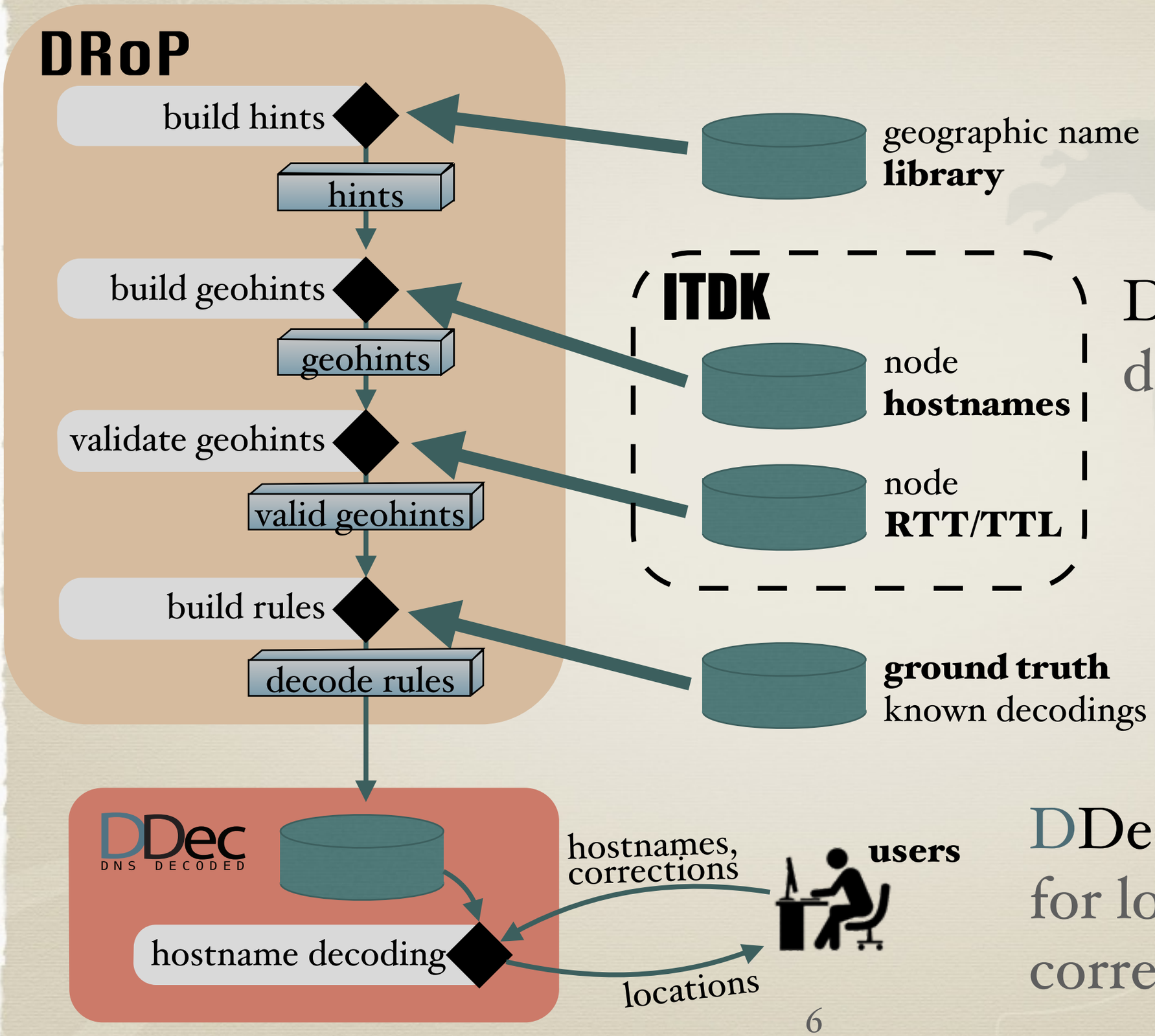
geographic hints

undns problems

- most inferences from 2002
- all entries require manual entry
- no validation

new systems

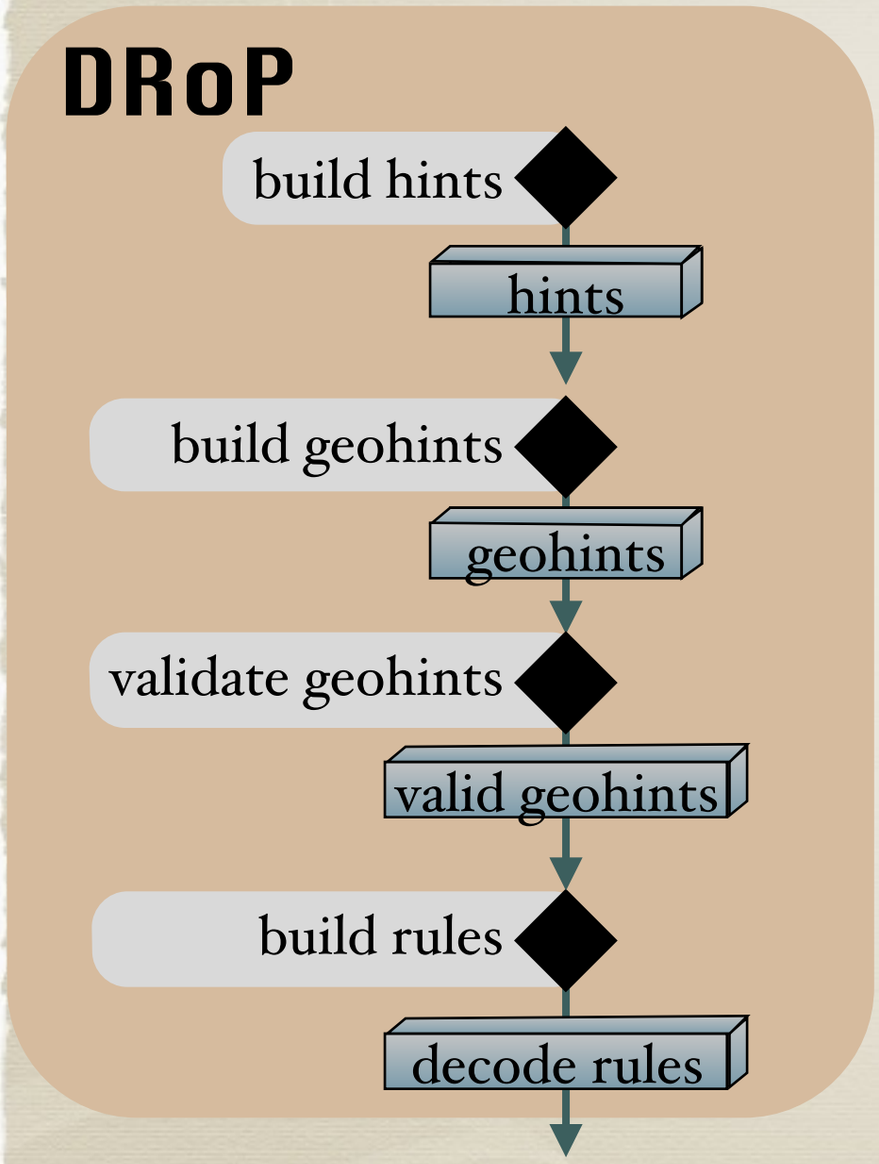
solutions



DRoP automated dns hint detection

DDec public interface for lookups and corrections

DRoP: automation



DRoP uses active **measurements** and a large geographic **library** to automatically **infer** the presence of geographic **hints in hostnames**.

DRoP: assumptions

- Operators use **common** geographic **hints** in their hostnames
- These hints will be the **same across** the same **domain**
- **Accurate hints** will group together routers with **similar** Round Trip Times (**RTT**) and Time To Live (**TTL**) values

DRoP: steps

* Construct **geohints** from hostnames with shared public suffixes, hints, and hint positions
* Create a 4-dimensional **vector** of active measurement data for each geohint
* **Classify** geohint for likely validity
* Derive general geolocation **rules**

DRoP: steps

* Construct geohints from hostnames with shared public suffixes, hints, and hint positions

* Create a 4-dimensional vector of active measurement data for each geohint.

* Classify geohint for likely validity

* Derive general geolocation rules

◆ DRoP: build geohint

- ✳ Construct geohints from hostnames with shared public suffixes, hints, and hint positions
  - ❋ build library of common geographic hints
  - ❋ group hostnames by public suffix
  - ❋ break hostname into substring
  - ❋ search substrings for common geographic hints
  - ❋ place nodes into all geohints that have a matching hostname

building the library

	airport		telco	United Nation	population
	IATA	**ICAO**	**CLLI**	**LOCODE**	**major city**
number of hints	7,622 **1.1%**	6,402 **0.9%**	121 **0.0%**	134,106 **19.5%**	540,223 **78.5%**

common geographic hints

- **IATA** 3 letter airport codes ← SAN
- **ICAO** 4 letter airport codes KSAN
- **CLLI** 6 letter telecommunication codes SNDACA
- **LOCODE** 5 letter United Nations USSAN
- **PoP** city with largest population San Diego

***Not all hints are equally useful.**

geohints

hostname breakdown

				public suffix
hostname	ccr21.	par01.	atlas.	**cogentco.com**
position	2	1	0	
hint	ccr	par	atlas	
geolocation	Concord, CA	Paris, FR	Salas Atlas, ES	

- **public suffix** is not searched, although used for grouping
- hostname's nodes placed into **geohints**

[public suffix, hint + position, location]

[cogentco.com, ccr + 2, Concord, CA]

[cogentco.com, par + 1, Paris, FR]

[cogentco.com, atlas + 0, Salas Alta, ES]

are any geohints correct?

We now have three different locations for the same hostname. Are any of them correct?

ccr21.par01.atlas.cogent

[cogentco.com, ccr + 2, Concord, CA]

[cogentco.com, par + 1, Paris, FR]

[cogentco.com, atlas + 0, Salas Alta, ES]

are any geohints correct?

We now have three different locations for the same hostname. Are any of them correct?

We will check them with active measurements!

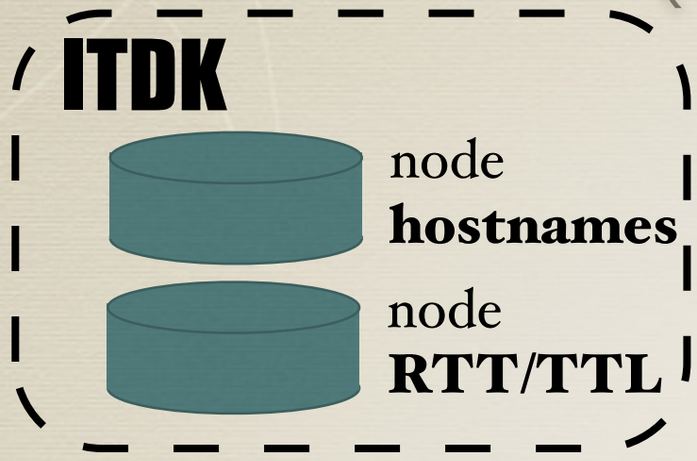
ccr21.par01.atlas.cogent

[cogentco.com, ccr + 2, Concord, CA]

[cogentco.com, par + 1, Paris, FR]

[cogentco.com, atlas + 0, Salas Alta, ES]

ITDK (active measurements)

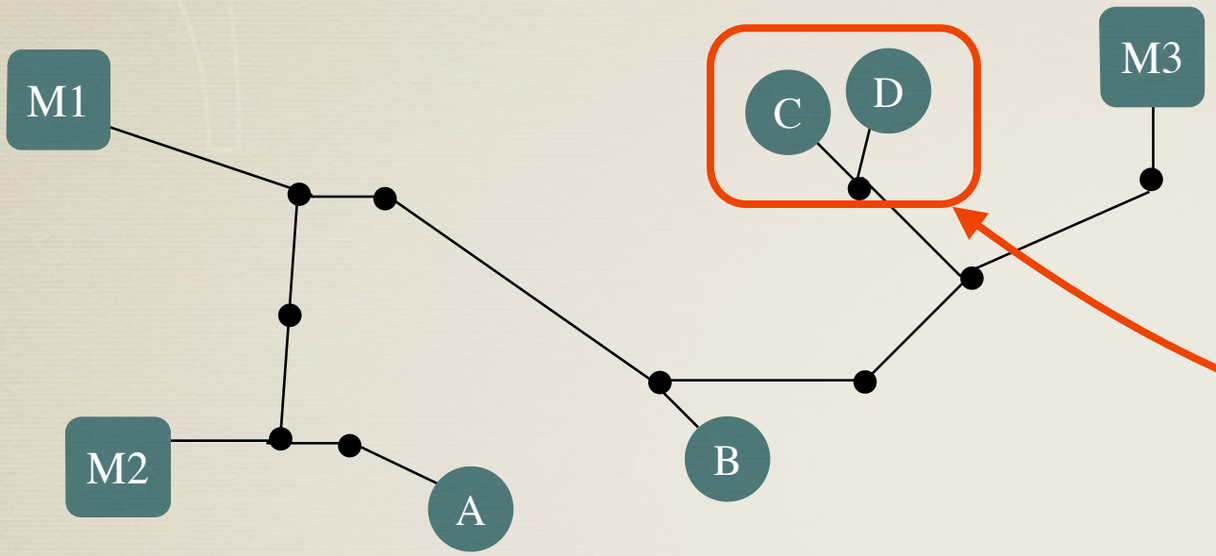


Internet Topology Data Kit (ITDK): is a large scale curated IPv4 Internet topology dataset

- periodically/curated (this study used July 2013)
- measurements
  - Round Trip Times (RTT)
  - Time To Live (TTL)
- collected from Ark (66 monitors on July 2013)
- router alias resolution (31,750k nodes*)

* each node roughly a router

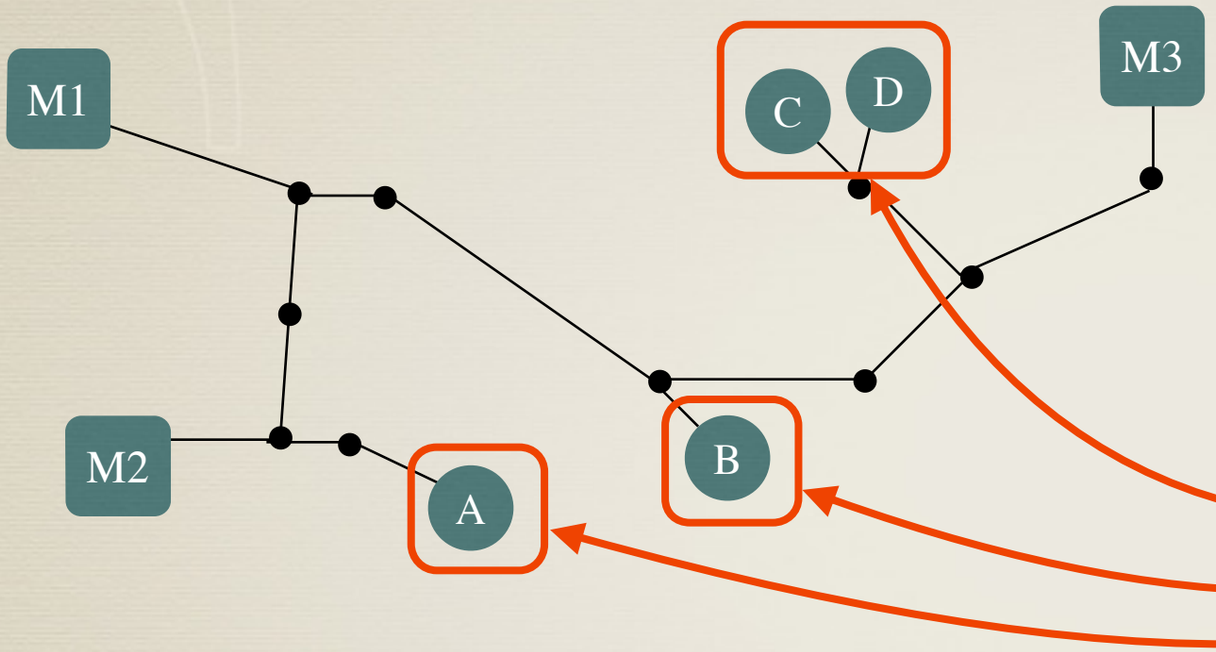
checking same location



Nodes in the same location should have similar RTT and hop counts for each monitor.

		ccr + 2 Concord	par + 1 Paris	M1		M2		M3	
				RTT	hop count	RTT	hop count	RTT	hop count
A	**ccr**01.san01.altas.cogentco.com	**X**		23	4	26	3	112	10
B	**ccr**02.was02.altas.cogentco.com	**X**		42	4	49	6	76	5
C	**ccr**04.**par**01.altas.cogentco.com	**X**	**X**	97	8	98	9	33	4
D	**ccr**02.**par**02.altas.cogentco.com	**X**	**X**	93	8	101	9	35	4

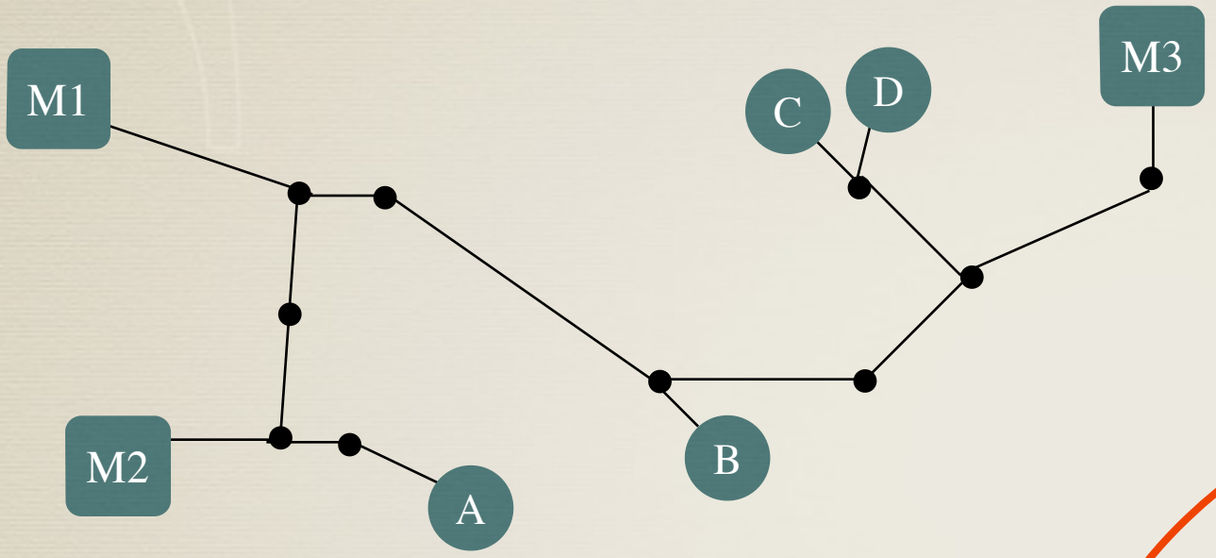
different locations



Nodes in different locations should have dissimilar RTT and hop counts for each monitor.

		ccr + 2 Concord	par + 1 Paris	M1		M2		M3	
				RTT	hop count	RTT	hop count	RTT	hop count
A	**ccr**01.san01.altas.cogentco.com	**X**		23	4	26	3	112	10
B	**ccr**02.was02.altas.cogentco.com	**X**		42	4	49	6	76	5
C	**ccr**04.**par**01.altas.cogentco.com	**X**	**X**	97	8	98	9	33	4
D	**ccr**02.**par**02.altas.cogentco.com	**X**	**X**	93	8	101	9	35	4

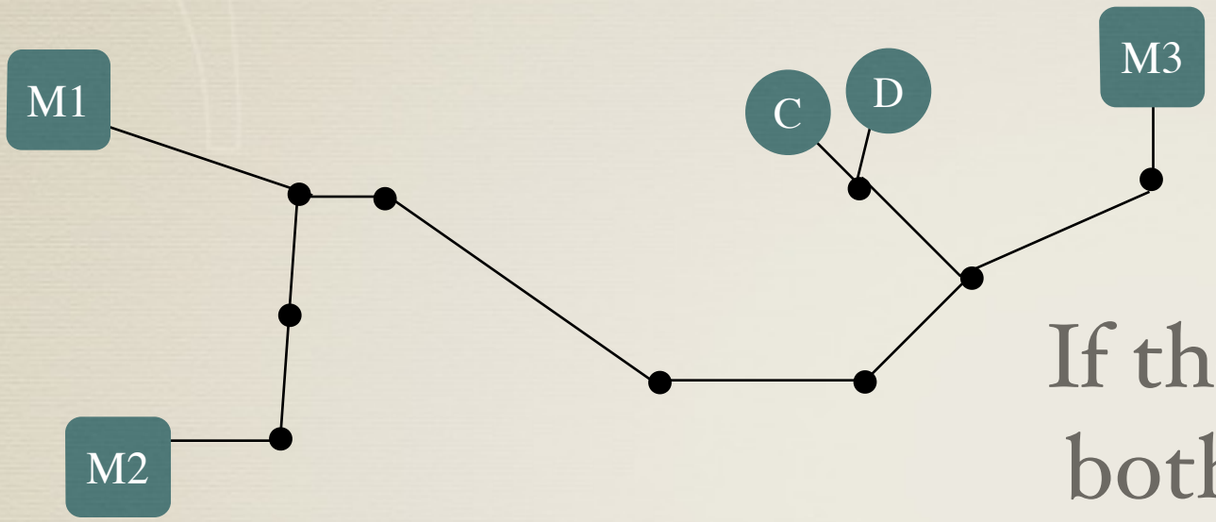
inferring valid geohint



so we infer that **par**+1 is likely a valid geohint

		ccr + 2 Concord	par + 1 Paris	M1		M2		M3	
				RTT	hop count	RTT	hop count	RTT	hop count
A	**ccr**01.san01.altas.cogentco.com	**X**		23	4	26	3	112	10
B	**ccr**02.was02.altas.cogentco.com	**X**		42	4	49	6	76	5
C	**ccr**04.**par**01.altas.cogentco.com	**X**	**X**	97	8	98	9	33	4
D	**ccr**02.**par**02.altas.cogentco.com	**X**	**X**	93	8	101	9	35	4

checking valid location

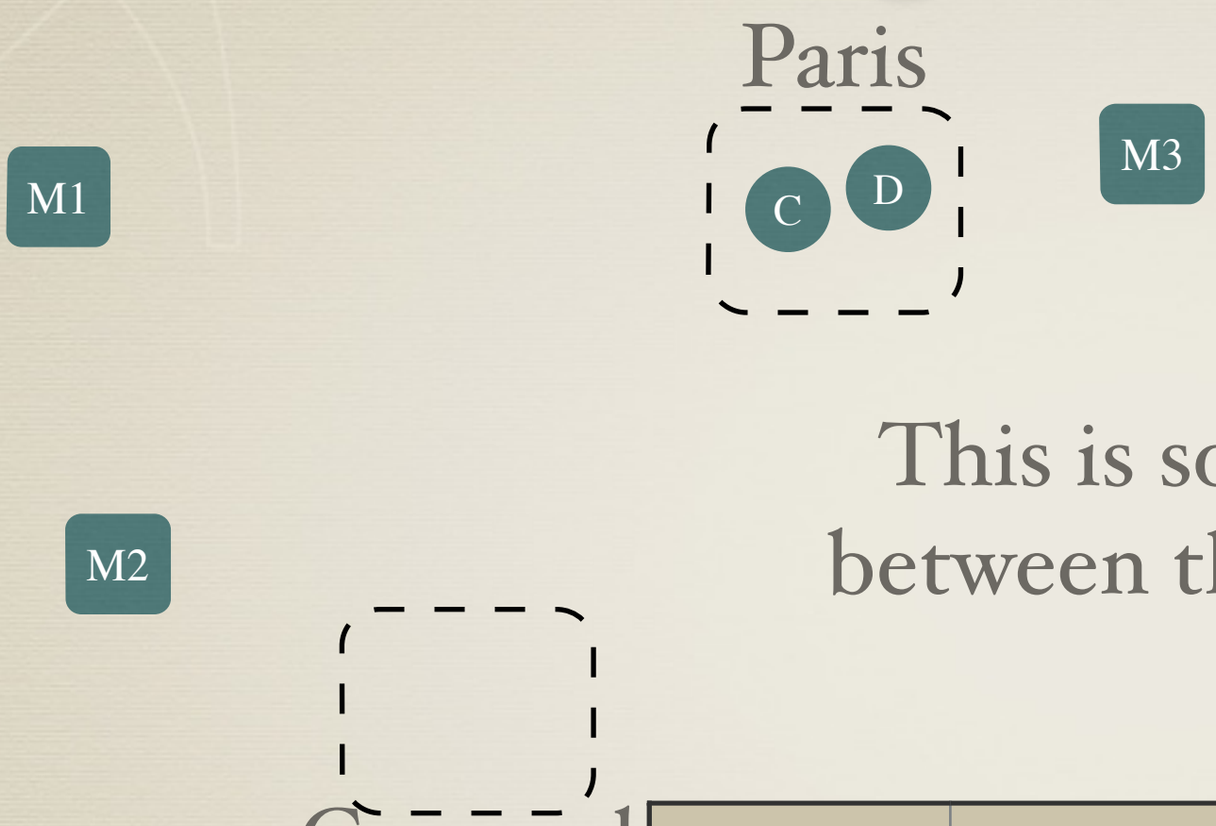


If the network only had C and D, both ccr+2 and par+1 look valid.

C and D are in the same location.

		ccr + 2 Concord	par + 1 Paris	M1		M2		M3	
				RTT	hop count	RTT	hop count	RTT	hop count
C	**ccr**04.**par**01.altas.cogentco.com	**X**	**X**	97	8	98	9	33	4
D	**ccr**02.**par**02.altas.cogentco.com	**X**	**X**	93	8	101	9	35	4

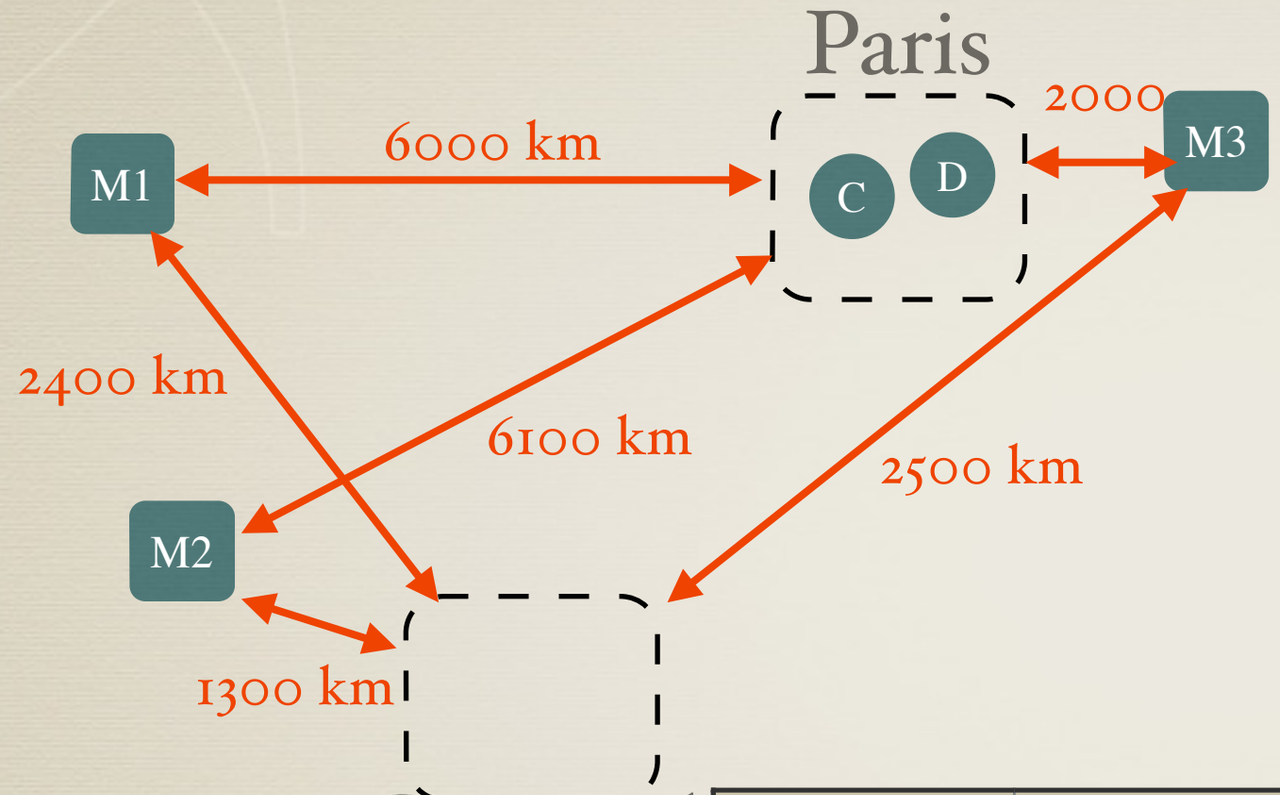
checking valid location



This is solved by inferring the speed between the monitors and the inferred location.

		ccr + 2 Concord	par + 1 Paris	M1		M2		M3	
				RTT	hop count	RTT	hop count	RTT	hop count
C	**ccr**04.**par**01.altas.cogentco.com	**X**	**X**	97	8	98	9	33	4
D	**ccr**02.**par**02.altas.cogentco.com	**X**	**X**	93	8	101	9	35	4

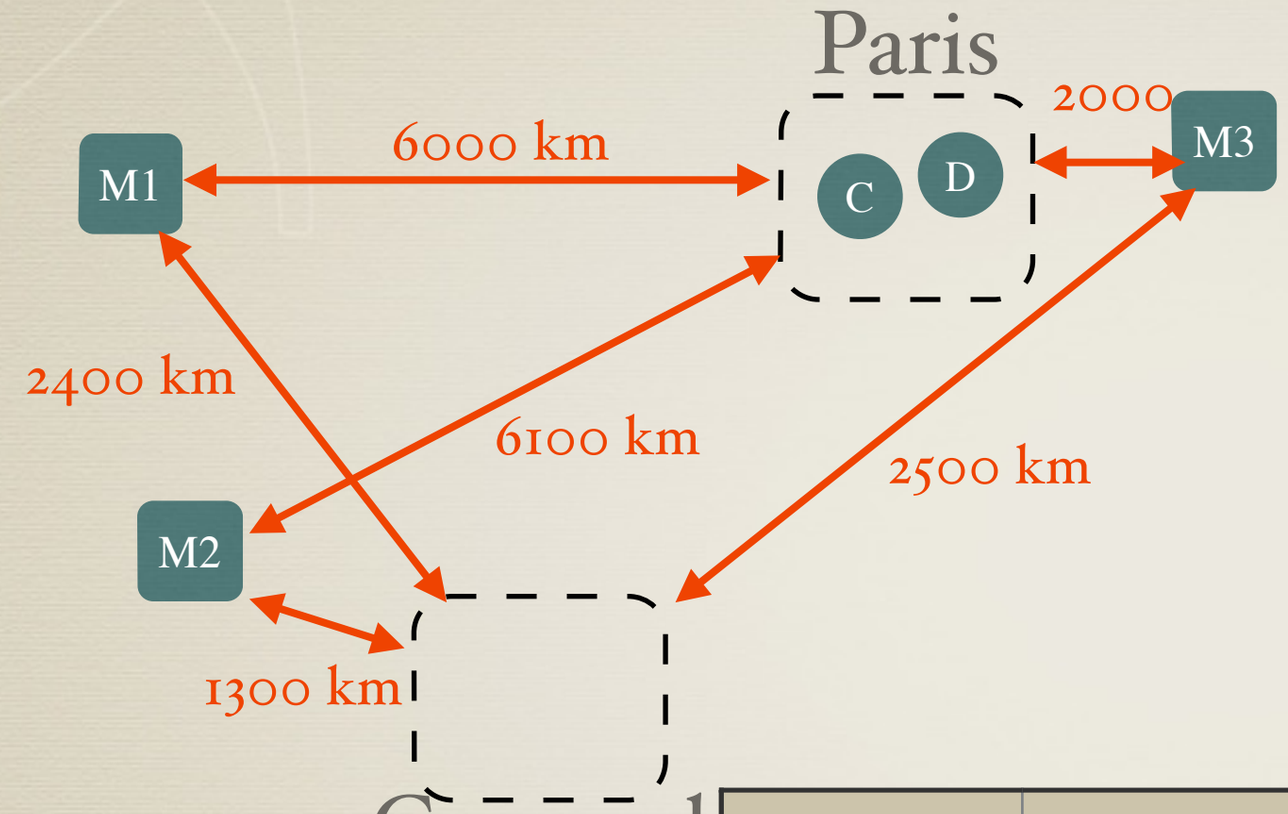
estimating speed



The speed is calculated using the distance between the infer location and the monitors, and the measured RTTs.

		ccr + 2 Concord	par + 1 Paris	speed (km/ms)					
				M1		M2		M3	
				ccr+2	par+2	ccr+2	par+2	ccr+2	par+2
C	**ccr**04.**par**01.altas.cogentco.com	**X**	**X**	46	123	26	124	151	120
D	**ccr**02.**par**02.altas.cogentco.com	**X**	**X**	51	129	25	121	142	114

estimating speed

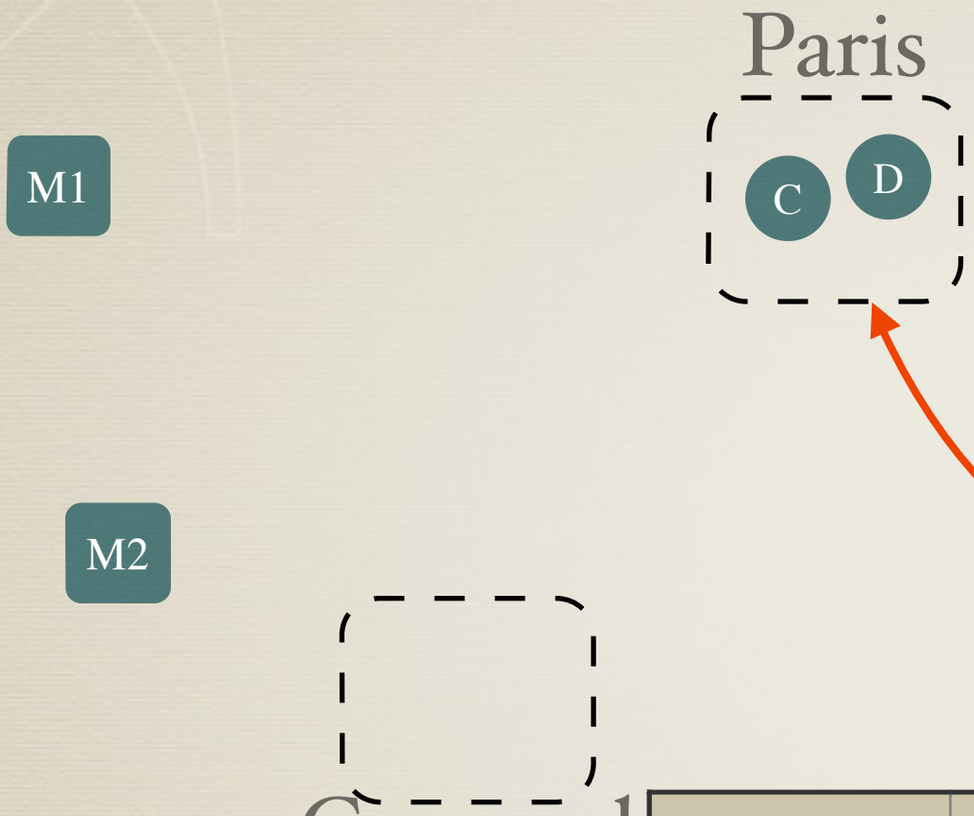


The speed is calculated using the distance between the infer location and the monitors, and the measured RTTs.

$$\text{speed} = \frac{\text{distance}}{\text{RTT}}$$

		ccr + 2 Concord	par + 1 Paris	speed (km/ms)					
				M1		M2		M3	
				ccr+2	par+2	ccr+2	par+2	ccr+2	par+2
C	**ccr**04.**par**01.altas.cogentco.com	**X**	**X**	46	123	26	124	151	120
D	**ccr**02.**par**02.altas.cogentco.com	**X**	**X**	51	129	25	121	142	114

actual location



If the inferred location is the actual location than speeds should be similar across monitors.

		ccr + 2 Concord	par + 1 Paris	speed (km/ms)					
				M1		M2		M3	
				ccr+2	par+2	ccr+2	par+2	ccr+2	par+2
C	ccr04.par01.altas.cogentco.com	X	X	46	123	26	124	151	120
D	ccr02.par02.altas.cogentco.com	X	X	51	129	25	121	142	114

different location

If the inferred location is a different location than speeds should be dissimilar across monitors

		ccr + 2 Concord	par + 1 Paris	speed (km/ms)					
				M1		M2		M3	
				ccr+2	par+2	ccr+2	par+2	ccr+2	par+2
C	**ccr**04.**par**01.altas.cogentco.com	**X**	**X**	46	123	26	124	151	120
D	**ccr**02.**par**02.altas.cogentco.com	**X**	**X**	51	129	25	121	142	114

inferring valid geohint

Paris

C D

M1

M3

M2

Concord

so we infer that **par**+1 is likely a valid geohint

		ccr + 2 Concord	par + 1 Paris	speed (km/ms)					
				M1		M2		M3	
				ccr+2	par+2	ccr+2	par+2	ccr+2	par+2
C	**ccr**04.**par**01.altas.cogentco.com	**X**	**X**	46	123	26	124	151	120
D	**ccr**02.**par**02.altas.cogentco.com	**X**	**X**	51	129	25	121	142	114

DRoP: steps

* Construct geohints from hostnames with shared public suffixes, hints, and hint positions

* Create a 4-dimensional vector of active measurement data for each geohint

* Classify geohint for likely validity

* Derive general geolocation rules

◆DRoP: building the vector

- Create a 4-dimensional vector of active measurement (ITDK) data for each geohint
  - created from RTT and TTL measurements between ark monitors and a geohint's nodes
  - checks if geohint's nodes are in the:
    - same location
    - geohint's inferred location

RTT from Ark monitors to geohint's nodes

3 measurements between anc-us and node 1

RTT* values to nodes matching a *geohint*

monitors	**nodes**	
	node 1	node 2
anc-us	26, 499, 10 ms	17, 23, 15, 199 ms
bed-us	57, 67 ms	70 ms

4 measurements between anc-us and node 2

*not real numbers

RTT from Ark monitors to geohint's nodes

RTT* values to nodes matching a *geohint*

monitors \ **nodes**	node 1	node 2
anc-us	26, 499, 10 ms	17, 23, 15, 199 ms
bed-us	57, 67 ms	~~70 ms~~

less then two measurements

*not real numbers

RTT_{min} from Ark monitors to geohint's nodes

monitors \ nodes	node 1	node 2
anc-us	10 ms	17 ms
bed-us	57 ms	

minimum (**min**) RTT reduces delay caused by **congestion**
we are interested in **propagation** delay

RTT_{min} from Ark monitors to geohint's nodes

remove nodes with fewer than 2 monitors probed

nodes

monitors	node 1	node 2	node 3	node 5
anc-us	10 ms	15 ms		183 ms
bed-us	57			43 ms
dkr-sn		6 ms	5 ms	13 ms
nrt-jp	63	29 ms		45 ms

not enough monitors for later triangulation

RTT_{std_min} from Ark monitors to geohint's nodes

monitors \ nodes	node 1	node 2	node 3	node 5
anc-us	10 ms	15 ms		183 ms
bed-us	57			43 ms
dkr-sn		6 ms		6 ms
nrt-jp	63	29 ms		45 ms

stand deviation (**std**) is a measure of **similarity**
valid geohints should have **similar** RTTs

RTT_{std_min} from Ark monitors to geohint's nodes

monitors \ nodes	node 1	node 2	node 3	node 5	std. dev.	average
anc-us	10 ms	15 ms		183 ms	80.4	69.3 ms
bed-us	57			43 ms	7	50 ms
dkr-sn		6 ms		6 ms	3.5	9.5
nrt-jp	63	29 ms		45 ms	13.9	45.7ms

nodes — **per monitor**

stand deviation (std) is a measure of similarity
valid geohints should have similar RTTs

$RTT_{avg\_std\_min}$ from Ark monitors to geohint's nodes

monitors	nodes: node 1	node 2	node 3	node 5	per monitor: std. dev.	average
anc-us	10 ms	15 ms		183 ms	80.4	69.3 ms
bed-us	57			43 ms	7	50 ms
dkr-sn		6 ms		6 ms	3.5	9.5
nrt-jp	63	29 ms		RTT	19.9	45.7ms

average (avg) combines measurements of across all monitors

$RTT_{avg\_std\_min}$ from Ark monitors to geohint's nodes

monitors	nodes: node 1	node 2	node 3	node 5	per monitor: std. dev.	average
anc-us	10 ms	15 ms		183 ms	80.4	69.3 ms
bed-us	57			43 ms	7	50 ms
dkr-sn		6 ms		6 ms	3.5	9.5
nrt-jp	63	29 ms		45 ms	13.9	45.7ms
RTT					26.2	

average (**avg**) combines measurements of across **all** monitors

geohint vector

vector			
evidence nodes in same location		**evidence nodes in inferred location**	
RTT avg_std_min	HopCount avg_std	Speed avg	Speed std
26.2			

- **RTT**$_{\textbf{avg\_std\_min}}$ average standard deviation minimum RTT

$speed_{std}/speed_{avg}$ from Ark monitors to geohint's nodes

geohint: [cogentco.com, IATA+2, Concord, CA]

We now use the average minimum (avg_min) RTT, calculated when we calculated $RTT_{avg\_std\_min}$, to calculate the speed.

per monitor

	RTT
anc-us	69.3 ms
bed-us	50 ms
dkr-sn	9.5
nrt-jp	45.7ms

$speed_{std}/speed_{avg}$ from Ark monitors to geohint's nodes

geohint: [cogentco.com, IATA+2, Concord, CA]

$$speed = \frac{distance}{RTT_{avg\_min}}$$

per monitor **to geohint**

	RTT	distance	speed
anc-us	69.3 ms	4872 km	70.3 km/ms
bed-us	50 ms	690 km	13.8 km/ms
dkr-sn	9.5	6660 km	701 km/
nrt-jp	45.7ms	10,311 km	225.6 km/ms

Distance is between the monitor and the inferred location.

$speed_{std}/speed_{avg}$ from Ark monitors to geohint's nodes

geohint: [cogentco.com, IATA+2, Concord, CA]

$$speed = \frac{distance}{RTT_{avg\_min}}$$

per monitor **to geohint**

	RTT	distance	speed
anc-us	69.3 ms	4872 km	70.3 km/ms
bed-us	50 ms	690 km	13.8 km/ms
dkr-sn	9.5	6660 km	701 km/
nrt-jp	45.7ms	10,311 km	225.6 km/ms

To check the inferred location, we check the stand deviation (**std**) and average (**avg**) speed.

270	252 km/ms
Speed	Speed

geohint vector

vector			
evidence nodes in same location		**evidence nodes in inferred location**	
RTT avg_std_min	HopCount avg_std	Speed avg	Speed std
26.2		**270**	**252**

- $RTT_{avg\_std\_min}$ average standard deviation minimum RTT

- **$Speed_{avg}$** average propagation speed
- **$Speed_{std}$** standard deviation of the speed

Hopcount: inferring

We want hop count, but ITDK only has observed TTLs. So we must infer the hop count from the observed TTLs

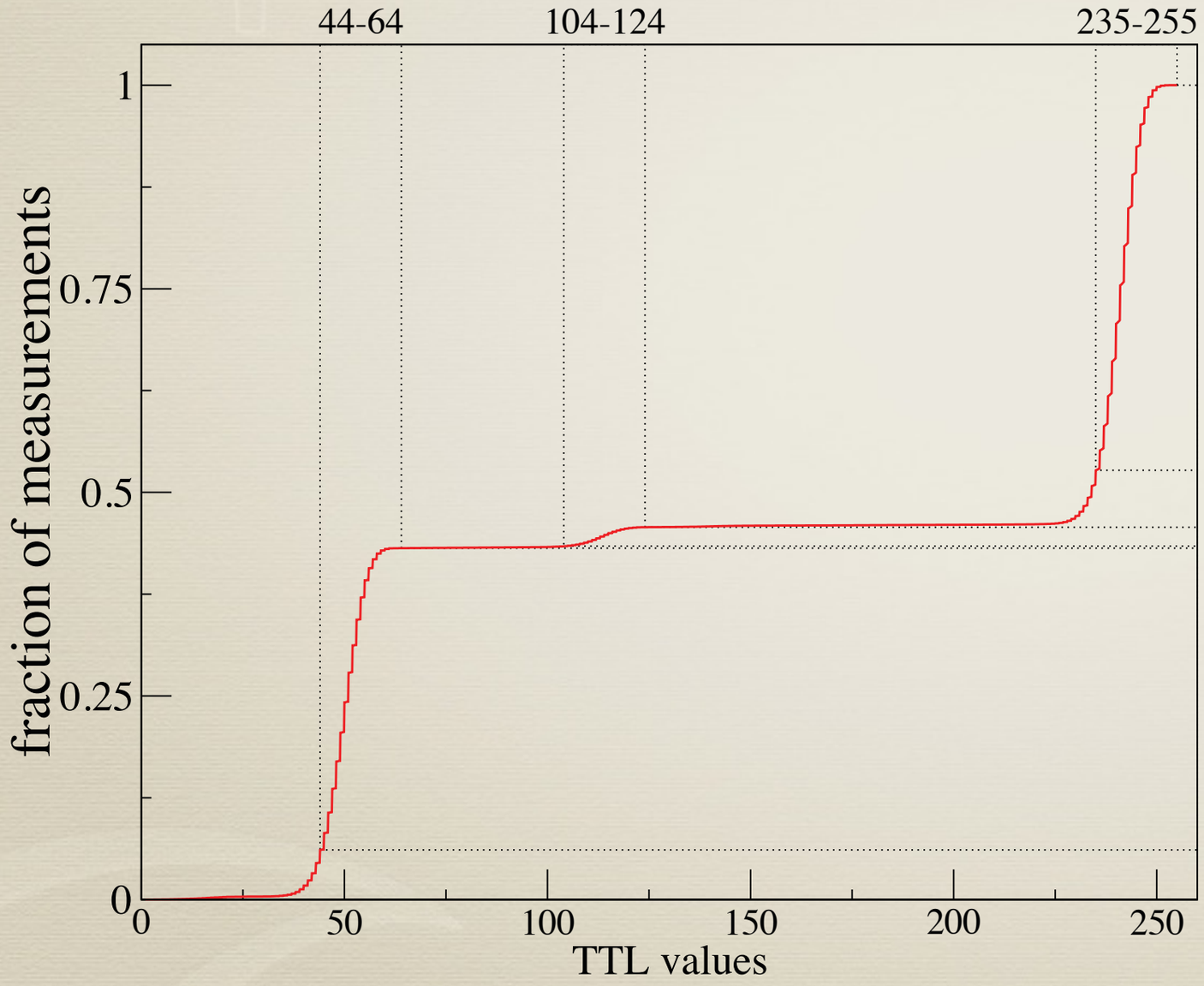
TTL from Ark monitors to geohint's nodes

monitors \ nodes	node 1	node 2
anc-us	51, 49, 49	240, 241,240,239
bed-us	56, 55	246,244

* Observed TTL is function of hop count and initial (initial) TTL.
* Different routers use different $TTL_{initial}$
* To calculate hop count we must first infer each node's $TTL_{initial}$

inferring $TTL_{initial}$

TTL values observed in ITDK July 2013



$TTL_{initial}$

255

124

64

Most likely $TTL_{initial}$ for each observed TTL

inferring HopCount

$$HopCount = TTL - TTL_{initial}$$

monitors

TTL **nodes**

	node 1	node 2
anc-us	51, 49, 49	240, 241,240,239
bed-	56, 55	246, 244

$TTL_{initial}$

64	255

HopCount

anc-us	13, 15, 15	15, 14, 15, 16
bed-	8, 9	9,11

$TTL_{initial}$ is infered for each node.

HopCount is thn calculated from average TTL.

$\text{HopCount}_{\text{avg\_std}}$

$\text{HopCount}_{\text{avg}}$ **nodes**

monitors	node 1	node 2	HopCount
anc-us	14.3	15	0.35
bed-	8.5	10	0.75

HopCount	0.55

HopCount

	node 1	node 2
anc-us	13, 15, 15	15, 14, 15, 16
bed-	8, 9	9,11

$\text{HopCount}_{\text{avg\_std}}$ is calculated in the same was as $\text{RTT}_{\text{avg\_std\_min}}$.

geohint vector

vector			
evidence nodes in same location		**evidence nodes in inferred location**	
RTT avg_std_min	HopCount avg_std	Speed avg	Speed std
26.2	**0.55**	270	252

- $RTT_{avg\_std\_min}$ average standard deviation minimum RTT
- **$HopCount_{avg\_std}$** average standard deviation of inferred hop count

- $Speed_{avg}$ average propagation speed
- $Speed_{std}$ standard deviation of the speed

DRoP: steps

* Construct geohints from hostnames with shared public suffixes, hints, and hint positions
* Create a 4-dimensional vector of active measurement data for each geohint.
* Classify geohint for likely validity
* Derive general geolocation rules

◆DRoP: validate geohints

- ✳ Classify geohint for likely validity
  - ❋ train classifier with ground truth (operator-provided naming conventions)

    akamai.com, belwue.de, cogentco.com, digitalwest.net, ntt.net, peak10.net

    <iata>[0-9]+.atlas.cogentco.com
    par01.altas.cogentco.com

  - ❋ classify geohints as likely valid or invalid

ground truth

assign **labels** based on ground truth

actual hint

				public suffix
hostname	ccr21.	par01.	atlas.	**cogentco.com**
position	2	1	0	
hint	ccr	par	atlas	
location	Concord, CA	Paris, FR	Salas Atlas, ES	

geohint	vector				label
	RTT avg_std_min	Speed std	Speed avg	HopCount avg_std	
[cogentco.com, ccr+2, Concord, CA]	9.2	270	252	0.55	FALSE
[cogentco.com, par+1, Paris, FR]	13.2	50	12	0.35	TRUE
[cogentco.com, atlas+0, Salas Alta, ES]	30.4	72	33	0.90	FALSE

ground truth

assign **labels** based on ground truth

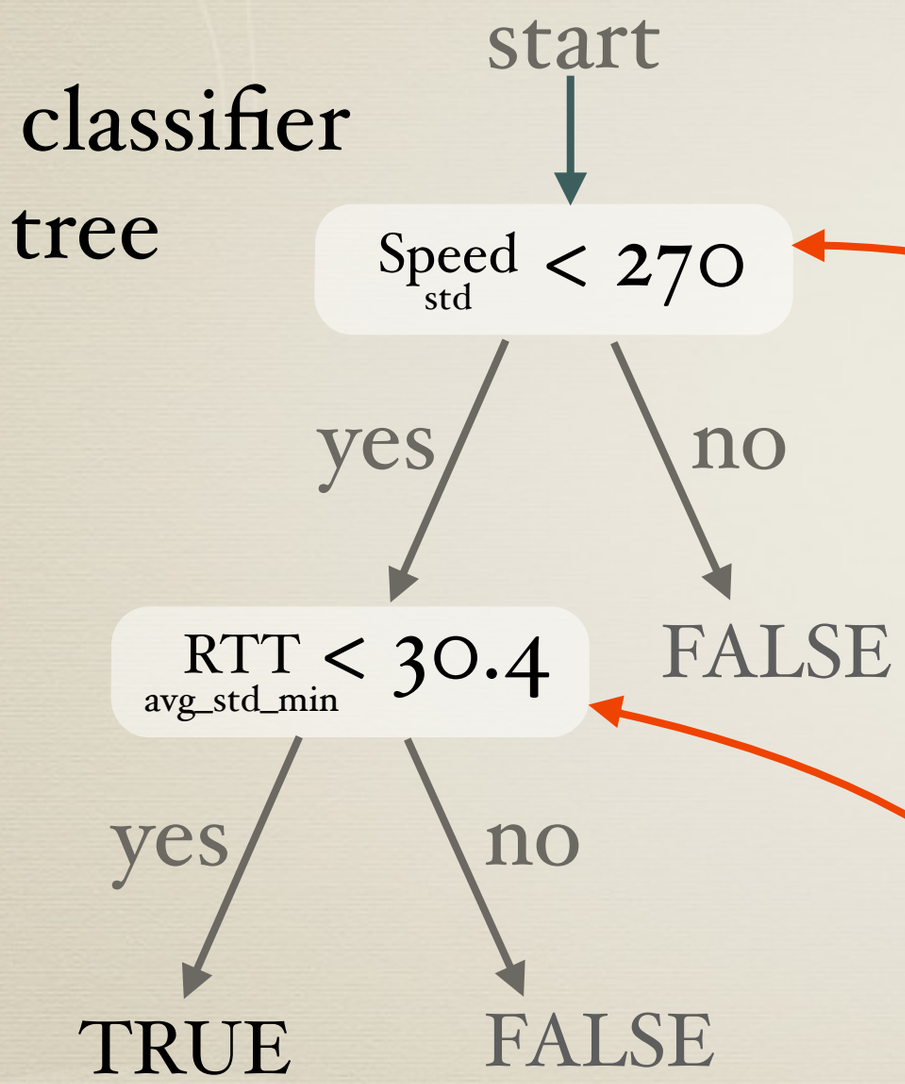
actual hint

				public suffix
hostname	ccr21.	par01.	atlas.	**cogentco.com**
position	2	1	0	
hint	ccr	par	atlas	
location	Concord, CA	Paris, FR	Salas Atlas, ES	

geohint	vector				label
	RTT avg_std_min	Speed std	Speed avg	HopCount avg_std	
[cogentco.com, ccr+2, Concord, CA]	9.2	270	252	0.55	FALSE
[cogentco.com, par+1, Paris, FR]	13.2	50	12	0.35	TRUE
[cogentco.com, atlas+0, Salas Alta, ES]	30.4	72	33	0.90	FALSE

train classifier

use labels to **train** the **classifier**



classifier

train

vector				label
RTT avg_std_min	Speed std	Speed avg	HopCount avg_std	
9.2	270	252	0.55	FALSE
13.2	50	12	0.35	TRUE
30.4	72	33	0.90	FALSE

valid geohints

using classifier

geohint	vector			
	RTT avg_std_min	Speed std	Speed avg	HopCount avg_std
[ntt.net, parsfr+4, Paris, FR]	9.2	45	22	0.15
[ntt.net, paris+6, Paris, FR]	98	170	12	0.55
[ntt.net, sttlwa+4, Seattle, US]	13.2	34	24	0.09

- classifier labels remaining geohints as likely or unlikely
- drops unlikely geohints

valid geohints

using classifier

classifier

geohint	vector				likely
	RTT avg_std_min	Speed std	Speed avg	HopCount avg_std	
[ntt.net, parsfr+4, Paris, FR]	9.2	45	22	0.15	TRUE
[ntt.net, paris+6, Paris, FR]	98	170	12	0.55	FALSE
[ntt.net, sttlwa+4, Seattle, US]	13.2	34	24	0.09	TRUE

- classifier **labels** remaining geohints as likely or unlikely
- **drops** unlikely geohints

DRoP: steps

* Construct geohints from hostnames with shared public suffixes, hints, and hint positions
* Create a 4-dimensional vector of active measurement data for each geohint.
* Classify geohint for likely validity
* Derive general geolocation rules

decode rules

rule generation

geohint	vector				likely
	RTT avg_std_min	Speed std	Speed avg	HopCount avg_std	
CLLI **[ntt.net, parsfr+4, Paris, FR]**	9.2	45	22	0.15	TRUE
CLLI **[ntt.net, sttlwa+4, Seattle, US]**	13.2	34	24	0.09	TRUE

<<clli>>([^a-z]+[a-z]+[0-9]*){3}.ntt.net

xe-1.telefonica-data.asbnva02.us.bb.gin.ntt.net
ae-0.francetelecom.parsfr01.fr.bb.gin.ntt.net
xe-0.dt.amstnl02.nl.bb.gin.ntt.net

◆ DRoP: build rules

- ✳ Derive general geolocation rules
  - ❋ combine classified geohints into a single rule
    - 60% classified as likely valid
    - share same geographic hint type / position
  - ❋ conflicting rules are resolved by rejecting the rule matching fewest hostnames (6% of rules)
  - ❋ resulting rules
    - 1,711 general rules
    - 1,398 domains

validation of rules

domain	type	positive		negative		number of hostnames
		true	false	true	false	
akamai.com		0%		100%		170
		0%	0%	**100**%	0%	
belwue.de	city name	52%		48%		161
		86%	14%	**99**%	1%	
cogentco.com	IATA	90%		10%		13,129
		99%	1%	**100**%	0%	
digitalwest.net	IATA	49%		51%		111
		100%	0%	**98**%	2%	
ntt.net	CLLI	96%		4%		2,584
		100%	0%	**100**%	0%	
peak10.net	IATA	100%		0%		115
		100%	0%	**0**%	0%	
–total–		90%		10%		16,270
		99%	1%	**100**%	0%	

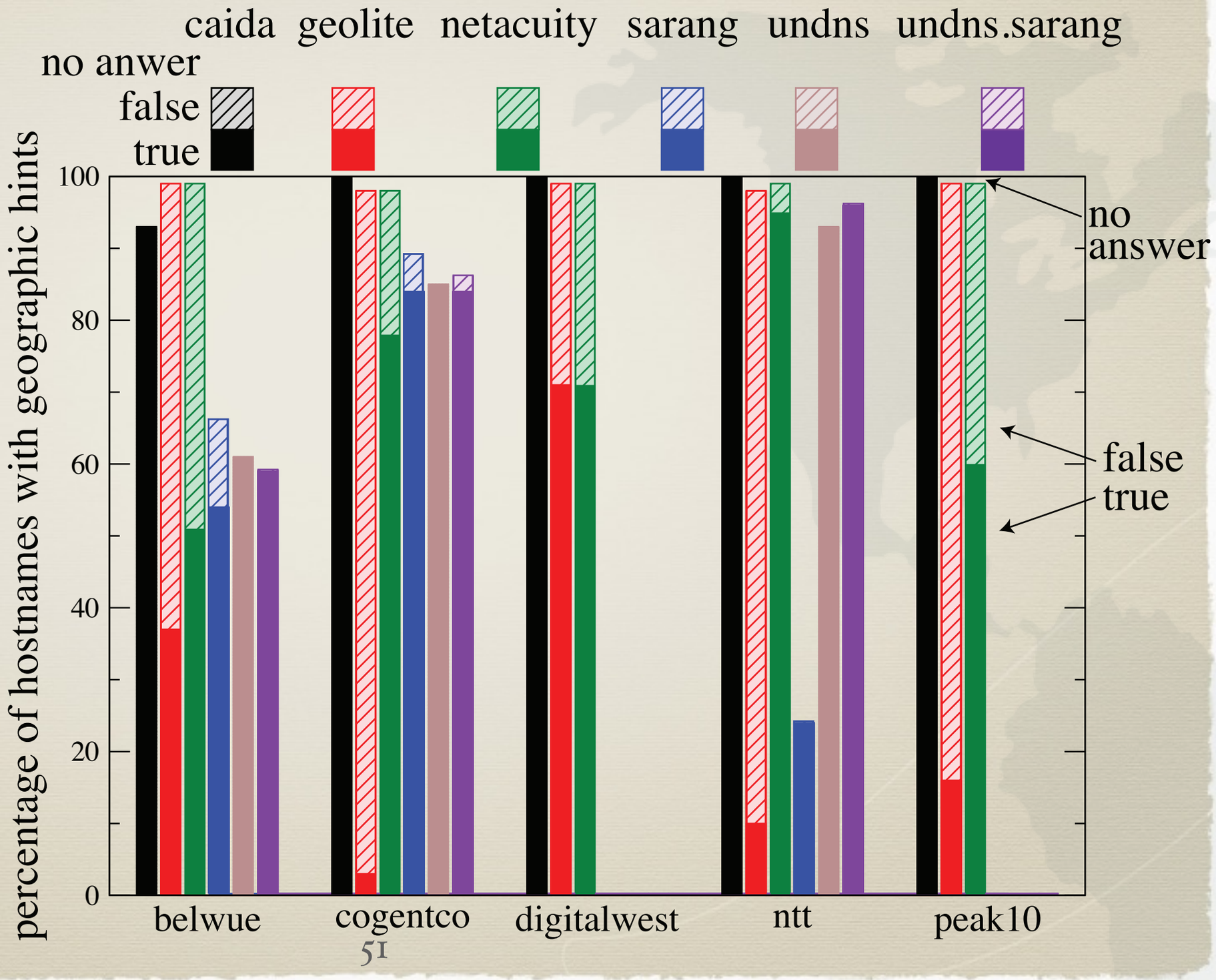
- **"true positive"** correctly placed this router within 10 km of its actual location
- **"false positive"** mapped the hint to a wrong location.
- **"true negative"** operators told us these hostnames have no geographic hint
- **"false negative"** failed to recognize a geographic hint.

validation (other services)

To make databases co-locate cities with the same name, we relaxed **threshold from 10 km to 40 km**.

For these hostnames, our algorithm provided the right answer in 99% of cases.
Excluding belwue.de, we got 100%.

Netacuity was the most successful existing solution with 81% accuracy.

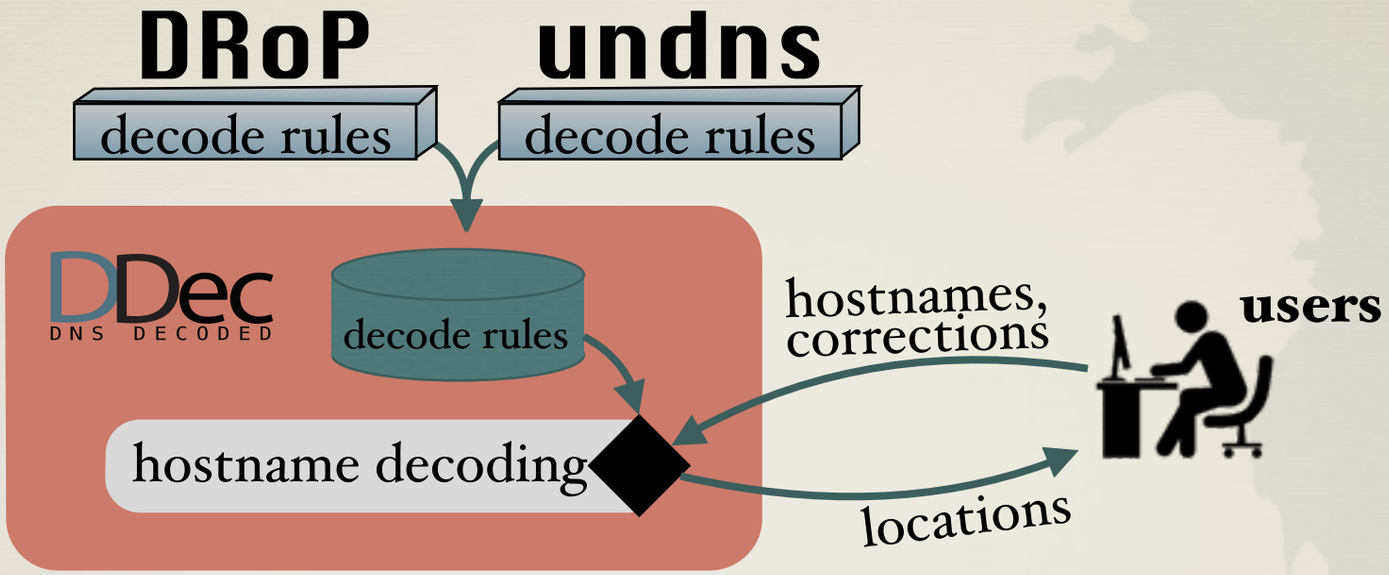


DRoP: summary

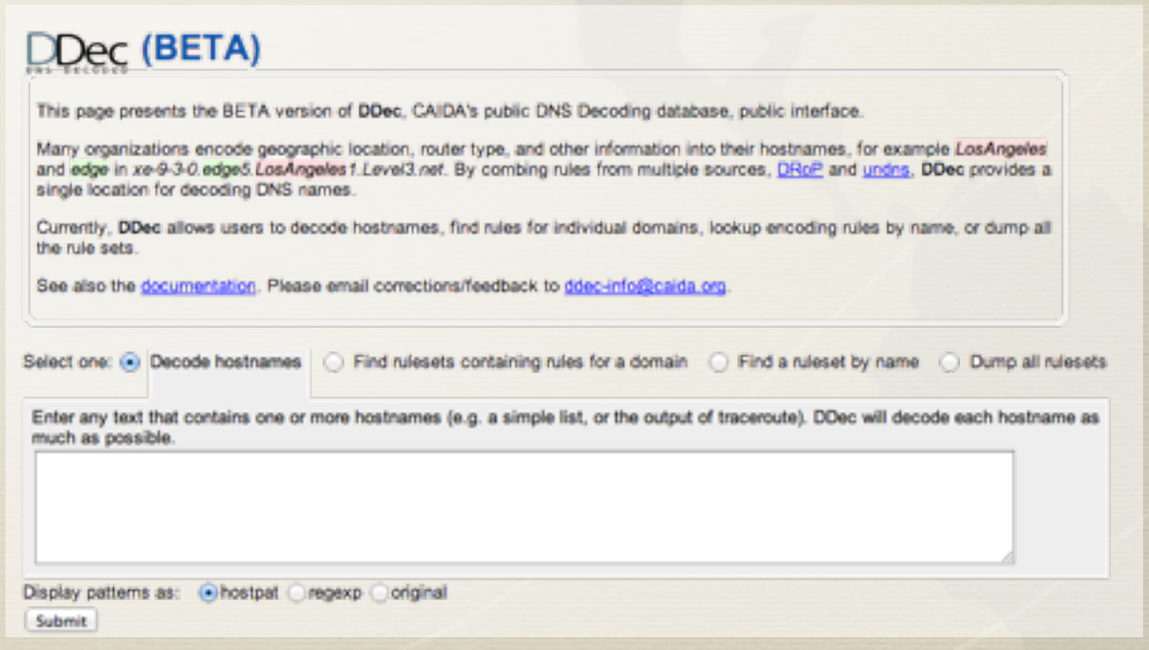
- DRoP provides a automated method for inferring geographic hints in hostnames.
- only works for hints in our library
- For available ground truth, our algorithm provided the right answer in 99% of cases.
- DRoP inferred
  - 1,711 general rules
  - 1,398 domains

DDec: public interface

ddec.caida.org



DDec provides a public interface for corrections and lookup



DDec: public interface

DNS DECODED

ddec.caida.org

- DNS Decode (DDec)
  - decode geolocation, bandwidth, etc
- public interface to inferred DNS encodings
  - CAIDA's DRoP
  - University of Washington's undns
  - RIPE NCC's OpenIPMap[1]
- interface for operators to provide corrections to either rules or locations
- hostpat custom pattern syntax, replacement for regular expression

[1]beta, not yet public/supported

hostpat: pattern syntax

ddec.caida.org/help.pl

- **custom pattern syntax** designed for hostnames

```
regex:\b([a-z]+)([^a-z]+[a-z]+\d*){2}\.2iij\.net$
        loc=$1 {#include pop}
hostpat:B<pop>([^L]+L+D*){2}.2iij.net
```

(undns format)

- **rulesets** collections of related rules

```
name: examplecorp
note: ExampleCorp, Inc.
rules:
- hostpat: <iata>D+.example.com
- hostpat: %.<clli>.example.net
```

- increased **simplicity lowers barrier** to provide feedback

DDec: sample output

DNS DECODED

ddec.caida.org

hostnames parsed from input

rulesets name undns.* DRoP.*

☑ Show rules ☐ Show decoding steps

	Hostname	Decoded values		Ruleset	feedback
		loc	type		
1	r06.snjsca04.us.bb.gin.ntt.net	San Jose, CA		undns.AS2914	correction
	r06.snjsca04.us.bb.gin.ntt.net	SAN JOSE, CA, US		DRoP.ntt.net	correction
2	ppp4-225.lvsb.vsnl.net.in	Mumbai, India	customer	undns.AS4755	correction

location

type

DDec: sample output

DNS DECODED

ddec.caida.org

hostnames parsed from input

rulesets name undns.* DRoP.*

☑ Show rules ☐ Show decoding steps

	Hostname	Decoded values		Ruleset	feedback
		loc	type		
1	r06.snjsca04.us.bb.gin.ntt.net	San Jose, CA		undns.AS2914	correction
	r06.snjsca04.us.bb.gin.ntt.net	SAN JOSE, CA, US		DRoP.ntt.net	correction
2	ppp4-225.lvsb.vsnl.net.in	Mumbai, India	customer	undns.AS4755	correction

location

type

feedback

DDec: feedback

DNS DECODED

ddec.caida.org

feedback: none Provide feedback on this rule sets. Edit the rule sets.

Enter text descripting your feedback. Submit Message

```
# mx0-ae7--thor-ae0.sdsc.edu | loc=LaJolla, CA
```

feedback: none Provide feedback on this rule sets. Edit the rule sets.

Edit the rule sets below and submit. Submit Rulesets

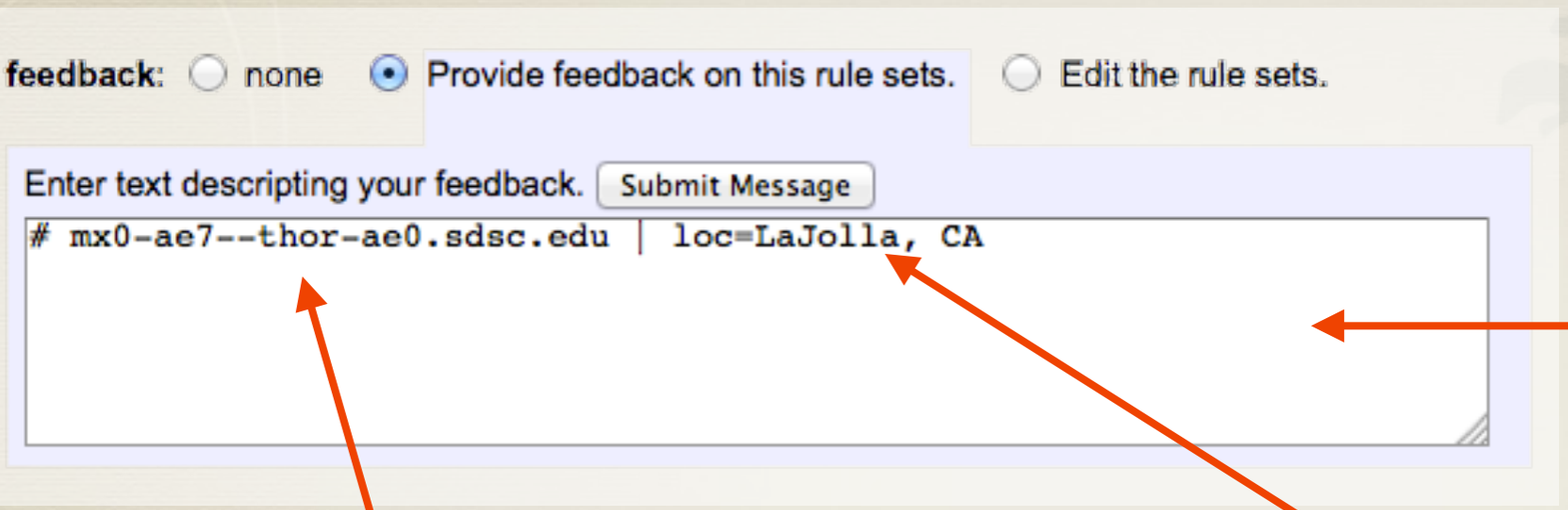
```
# mx0-ae7--thor-ae0.sdsc.edu | loc=LaJolla, CA

---
name: undns.AS195
source: undns/keys/edu-us
rules:
- hostpat: '%.sdsc.edu'
  vars:
  - loc: LaJolla, CA
```

DDec: feedback

DNS DECODED

ddec.caida.org

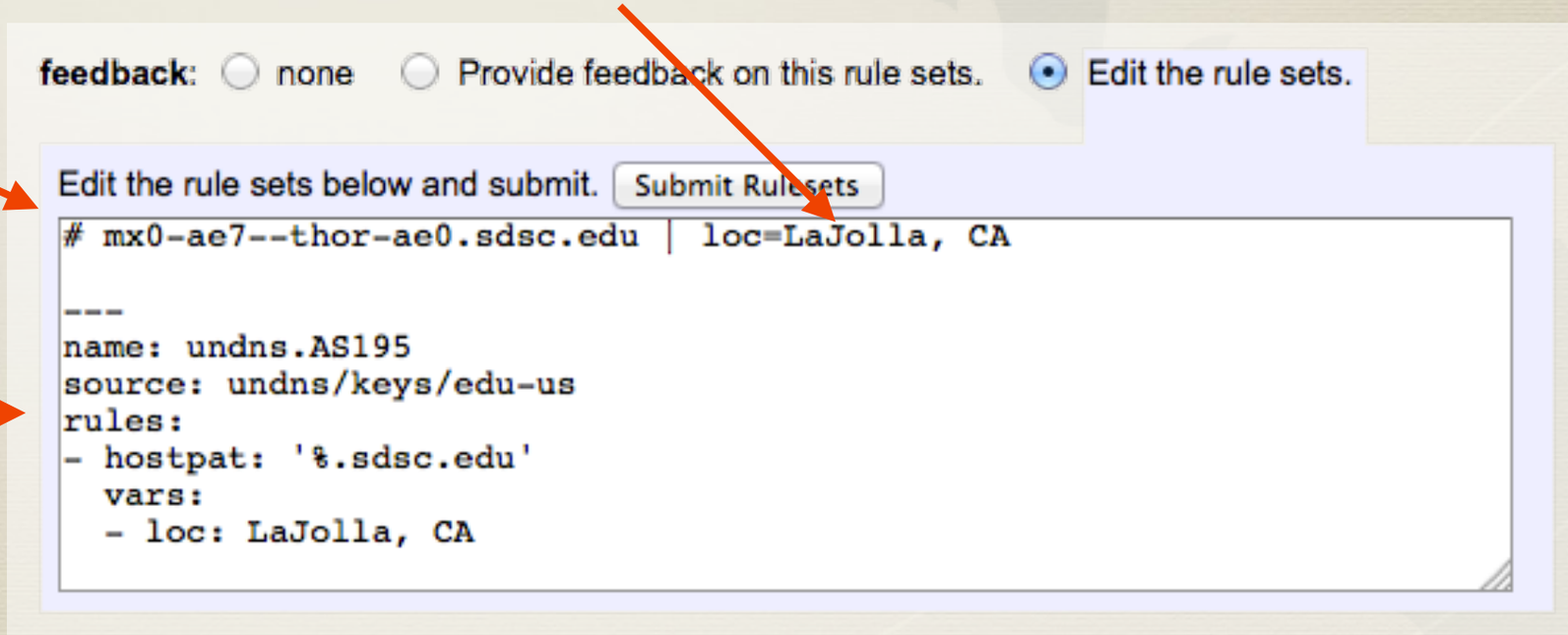


space for message

hostname

location

editable rulesets



future

* use DRoP-geolocated routers to geolocate adjacent routers
* infer less common geographic hints
* improve DDec feedback methods
* increase DRoP and DDec public visibility
    - present to RIPE and NANOG