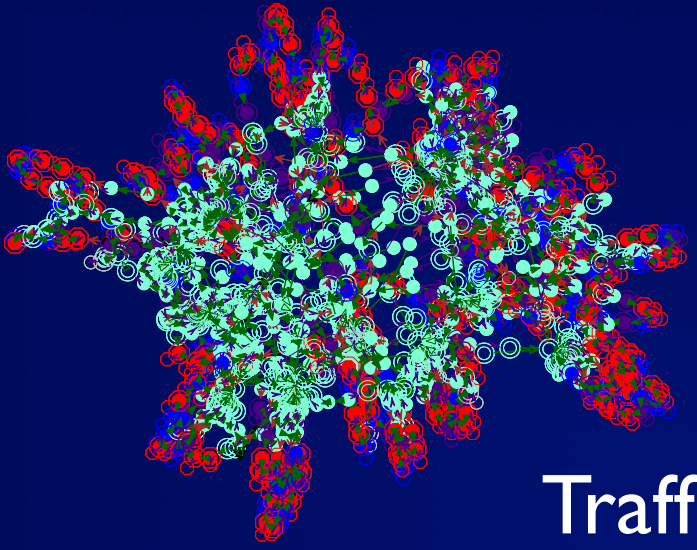


Traffic Causality Graphs: Profiling network applications through temporal and spatial causality of flows

Hirochika Asai <panda@hongo.wide.ad.jp> (*1)
Kensuke Fukuda <kensuke@nii.ac.jp> (*2)
Hiroshi Esaki <hiroshi@wide.ad.jp> (*1)
(*1) Univ. of Tokyo, Japan (*2) NII, Japan
ITC23, San Francisco, USA, Sep. 6, 2011

Background

- Network application profiling
  - Motivations
    - Traffic engineering (traffic forecast, trend analysis)
    - Network forensics (anomaly detection, incident trace)
  - Challenges
    - Various network applications
      - Classes: Web browsers, E-mail clients, P2P file sharing, P2P live streaming etc.
      - Programs: e.g., user agents in Web browsers
    - Multiple protocol utilization by each application
      - Web browsers: HTTP after DNS
      - P2P file sharing: P2P after HTTP

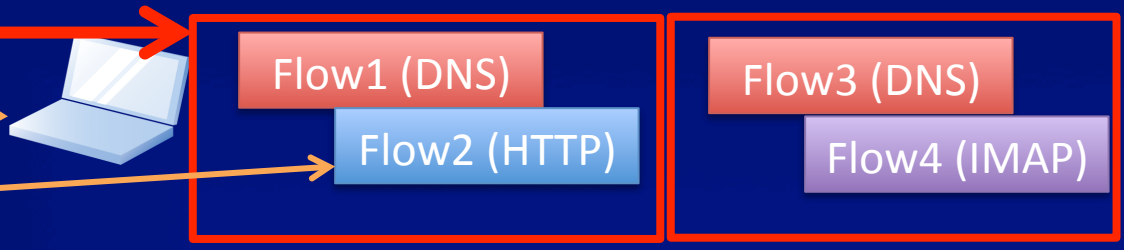
Causality of flows

Target

- Edge network
  - Almost all packets can be captured.

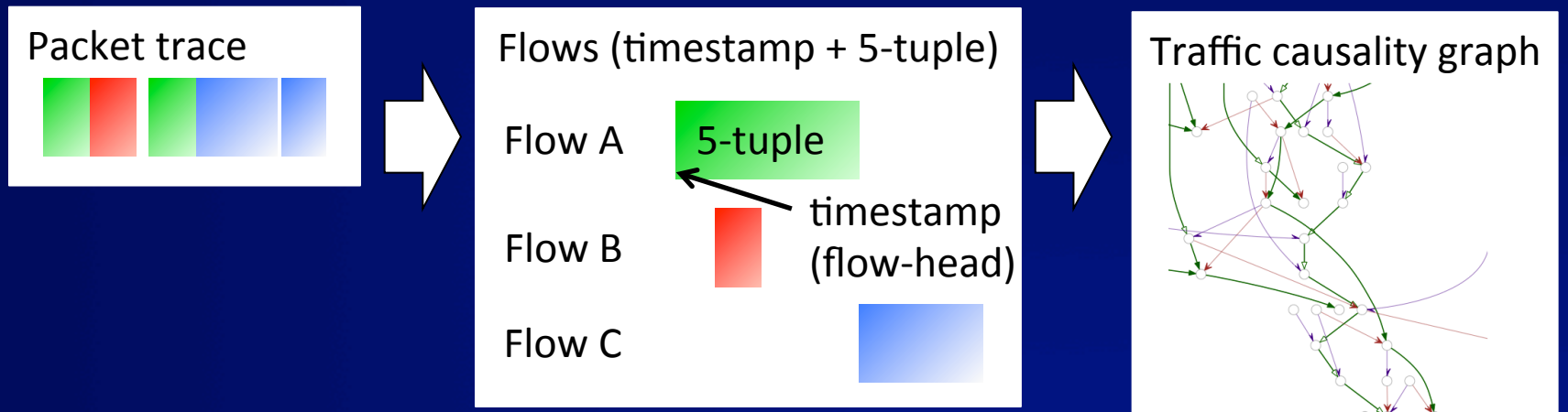
← Different from Iliofotou et al.'s approach

- Flowset profiling
  - Not hosts
  - Not single flows

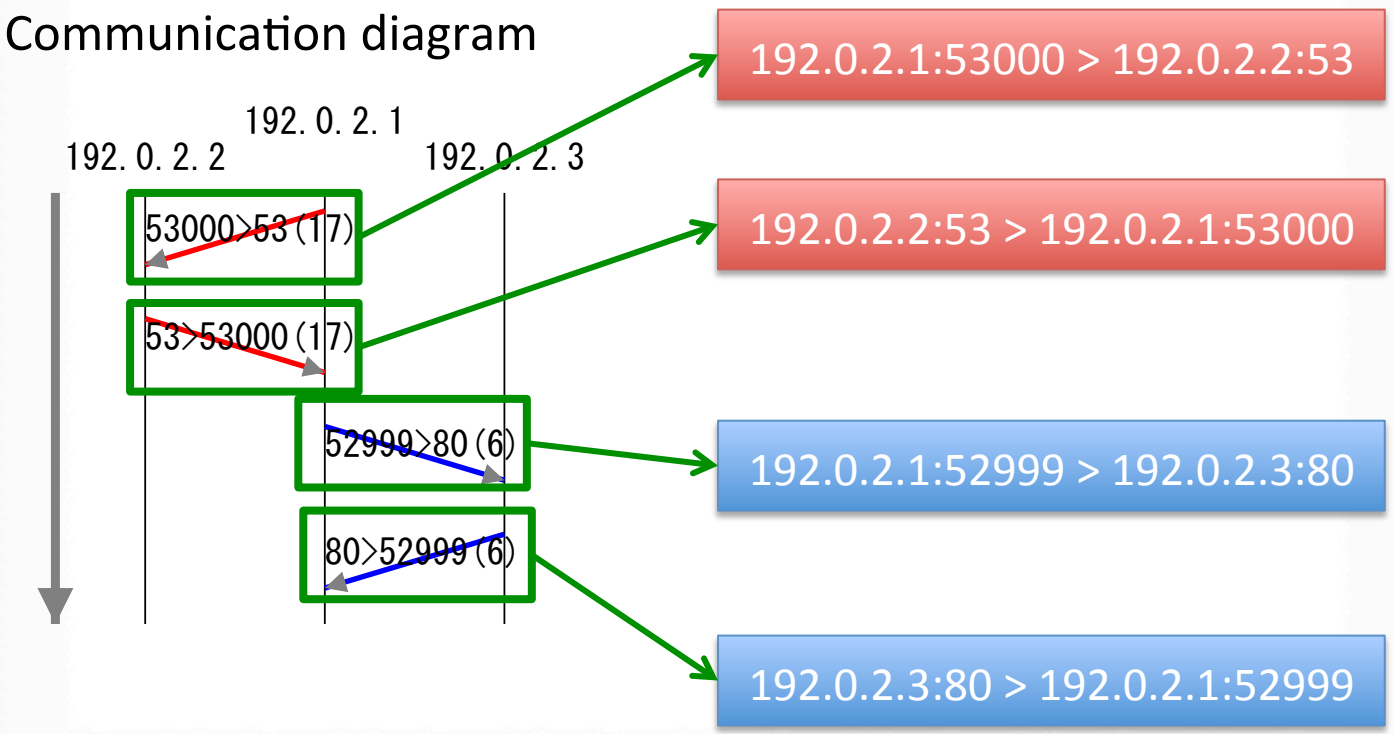


- Flow-base
  - 5-tuple + timestamp
    - N.B., Port numbers can be hints for profiling
      - UDP/53 = Big identity
  - With causality
  - Without deep packet inspection (signature)

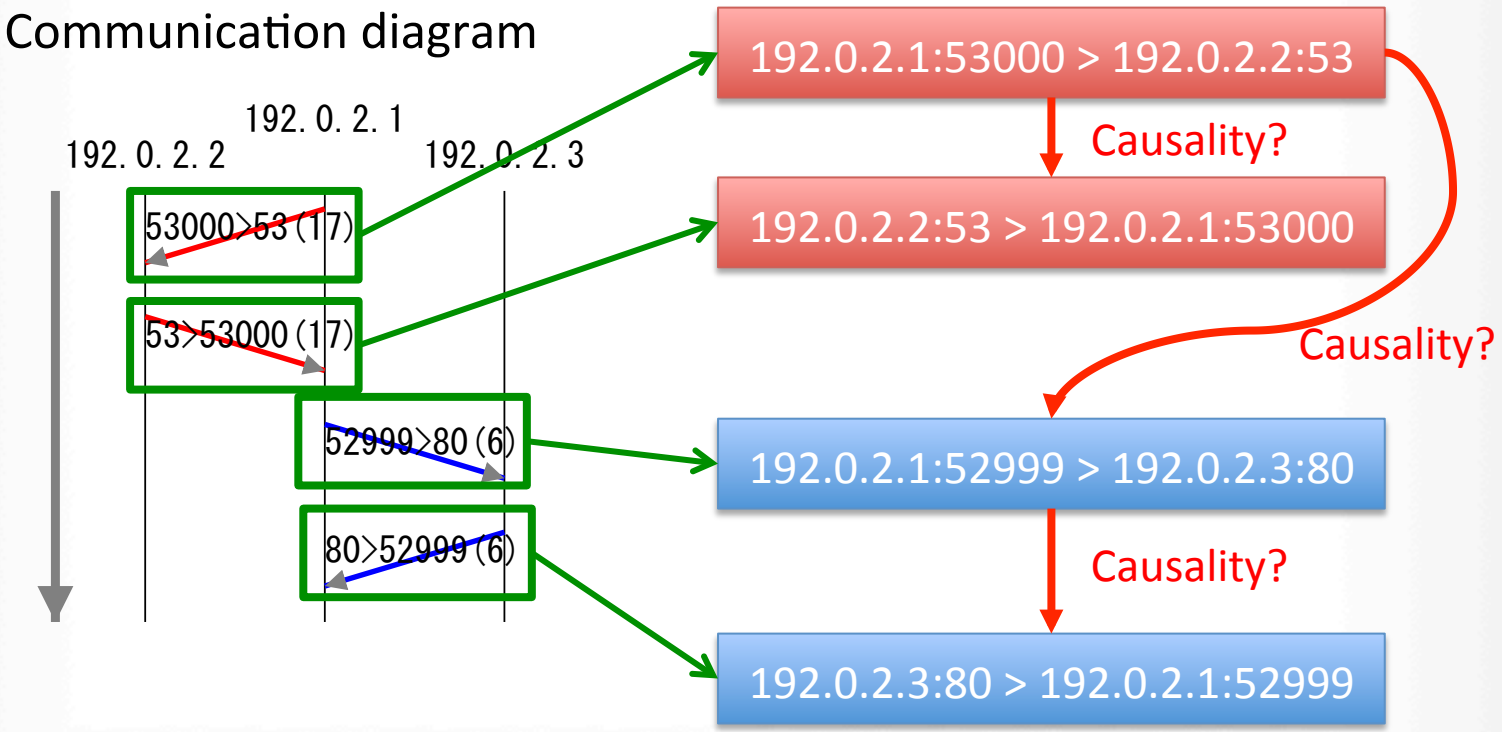
Overview of profiling procedure with traffic causality graph (TCG)



Flow causality
To construct TCG

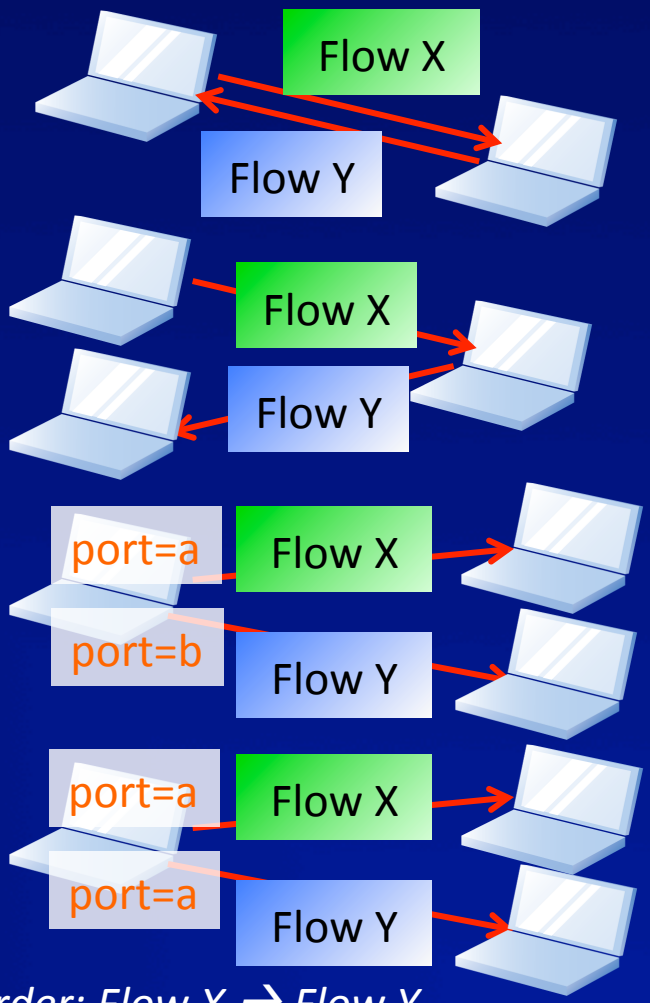


Flow causality
To construct TCG



Four types of causality

1. Communication relationship (CR)
   - Bidirectional communication (e.g., TCP)
   - One-to-one relationship
2. Propagation relationship (PR)
   - Propagate data flow X to flow Y
   - Cause flow Y by flow X
   - Many-to-many relationship
3. Dynamic-port host relationship (DHR)
   - Flows from same host with different port #
   - Many-to-many relationship
4. Static-port host relationship (SHR)
   - Flows from same host with identical port #
   - Many-to-many relationship



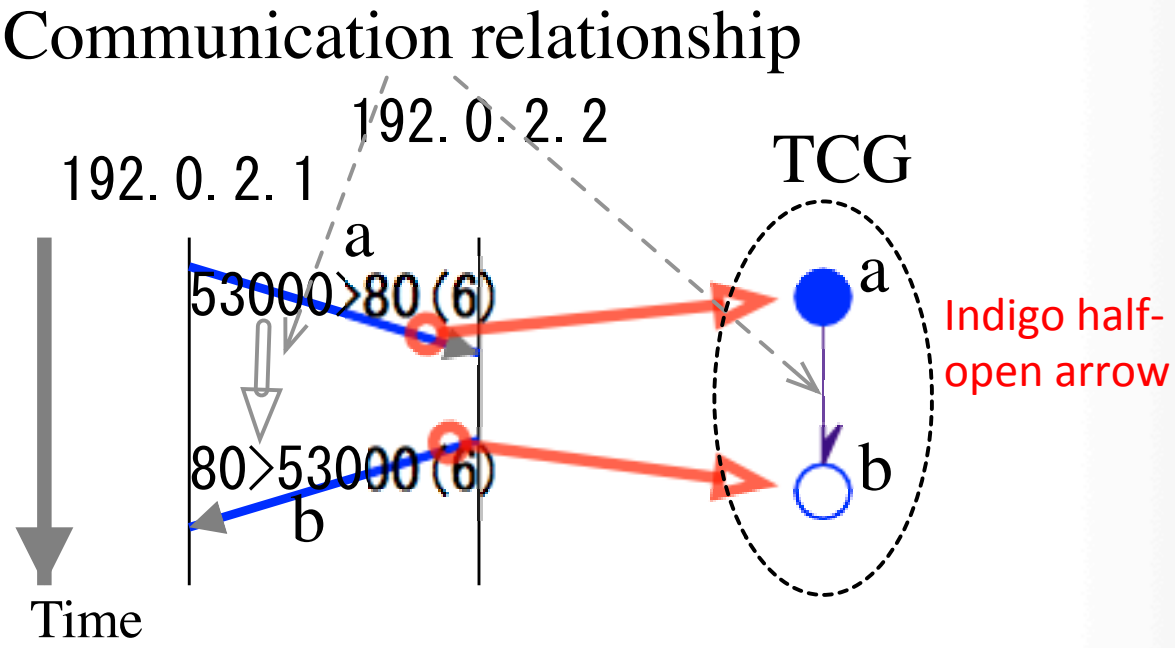
Temporal order: Flow X → Flow Y

Note: Representation of flows (vertices) in TCG

- ICMP
  - Outbound: Triangle
  - Inbound: Inverted triangle
- UDP
  - Outbound: Double bordered octagons
  - Inbound: Octagon
  - Color
    - DNS; Port 53: Red
    - DHCP; Port 67, 68: Orange
    - Well-known: Yellow
    - Others: Grey
- TCP
  - Outbound: Double circle
  - Inbound: Single circle
  - Color
    - SSH; Port 22: Pink
    - HTTP; Port 80: Blue
    - HTTPS; Port 443: Indigo
    - Proxy; Port 8080: Aqua
    - Well-known: Green
    - Others: Grey

Four types of flow causality relationships and examples

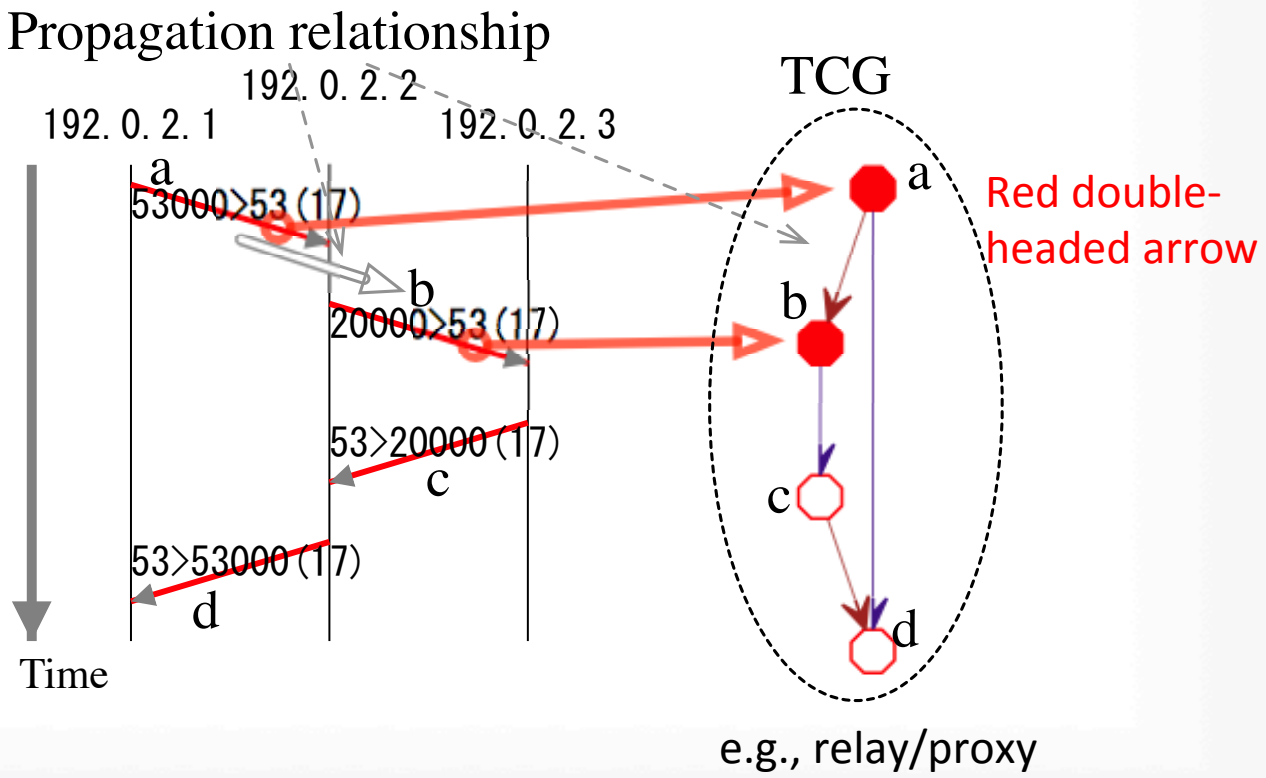
1. Communication relationship



e.g., one TCP session

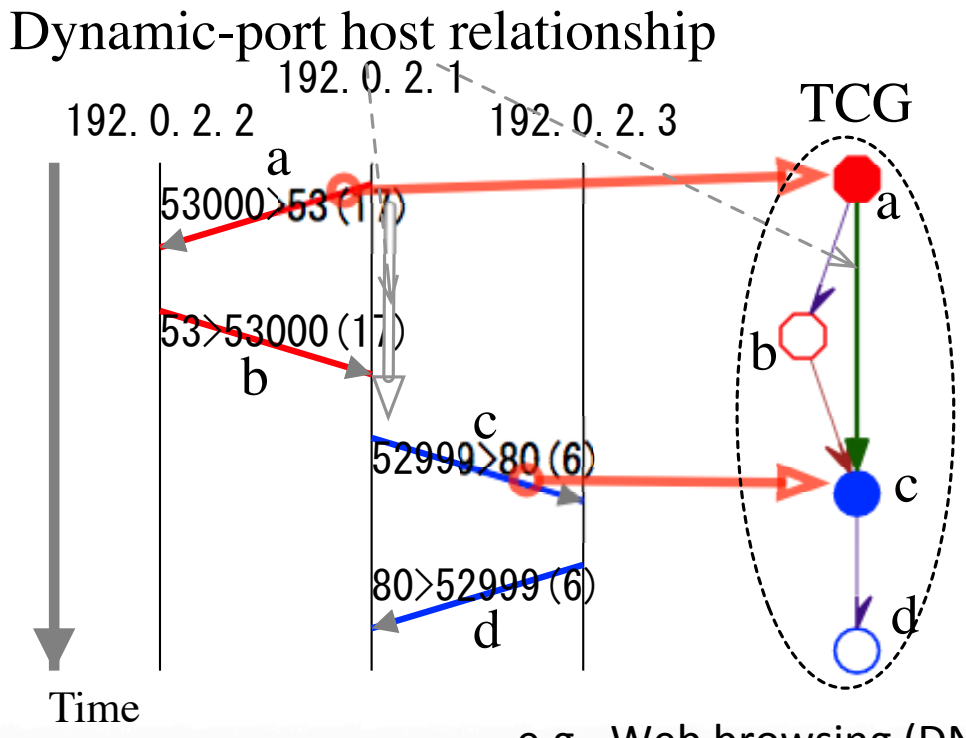
Four types of flow causality relationships and examples

2. Propagation relationship



Four types of flow causality relationships and examples

3. Dynamic-port host relationship

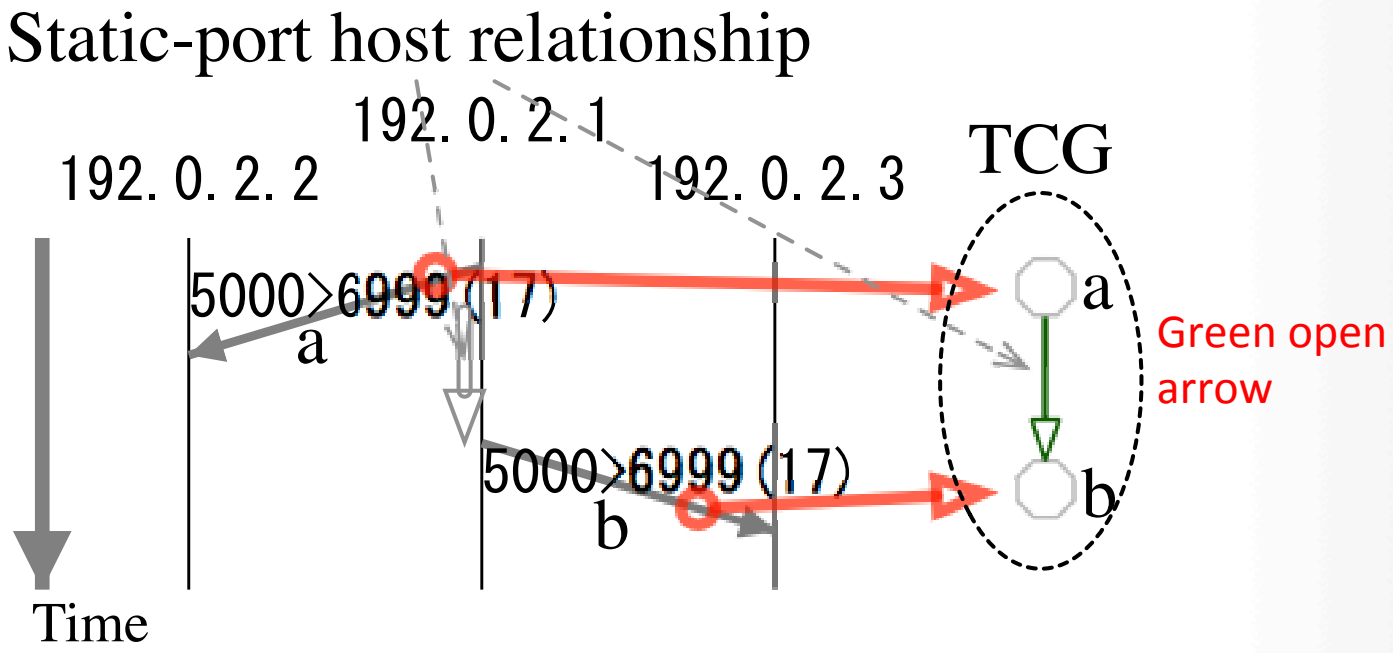


Green filled arrow

e.g., Web browsing (DNS → HTTP)

Four types of flow causality relationships and examples

4. Static-port host relationship



e.g., scan using identical src port

Algorithm based on 5-tuple

Algorithm 1 Get the type of relationship between the flows f_1 and f_2

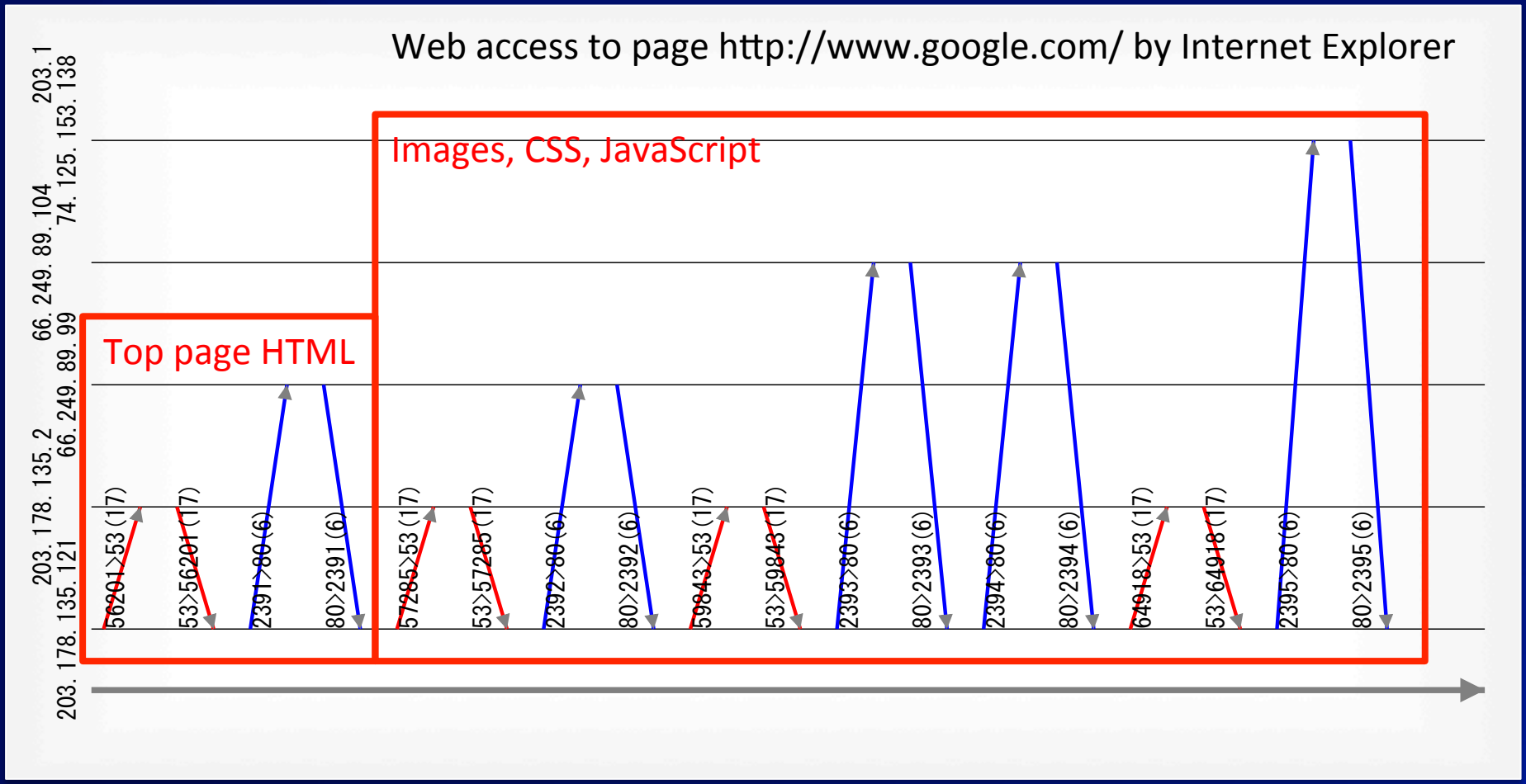
procedure getRelationship(f_1, f_2, τ):

```
1:  if timestamp(f2) − timestamp(f1) > τ then
2:      return Nil
3:  end if
4:  if proto(f1) = proto(f2) and srcIP(f1) = dstIP(f2) and srcPort(f1) = dstPort(f2)
        and dstIP(f1) = srcIP(f2) and dstPort(f1) = srcPort(f2) then
5:      return COMMUNICATION_RELATIONSHIP
6:  else if dstIP(f1) = srcIP(f2) then
7:      return PROPAGATION_RELATIONSHIP
8:  else if srcIP(f1) = srcIP(f2) and srcPort(f1) ≠ srcPort(f2) then
9:      return DYNAMIC_PORT_HOST_RELATIONSHIP
10: else if srcIP(f1) = srcIP(f2) and srcPort(f1) = srcPort(f2) then
11:     return STATIC_PORT_HOST_RELATIONSHIP
12: else
13:     return Nil
14: end if
```

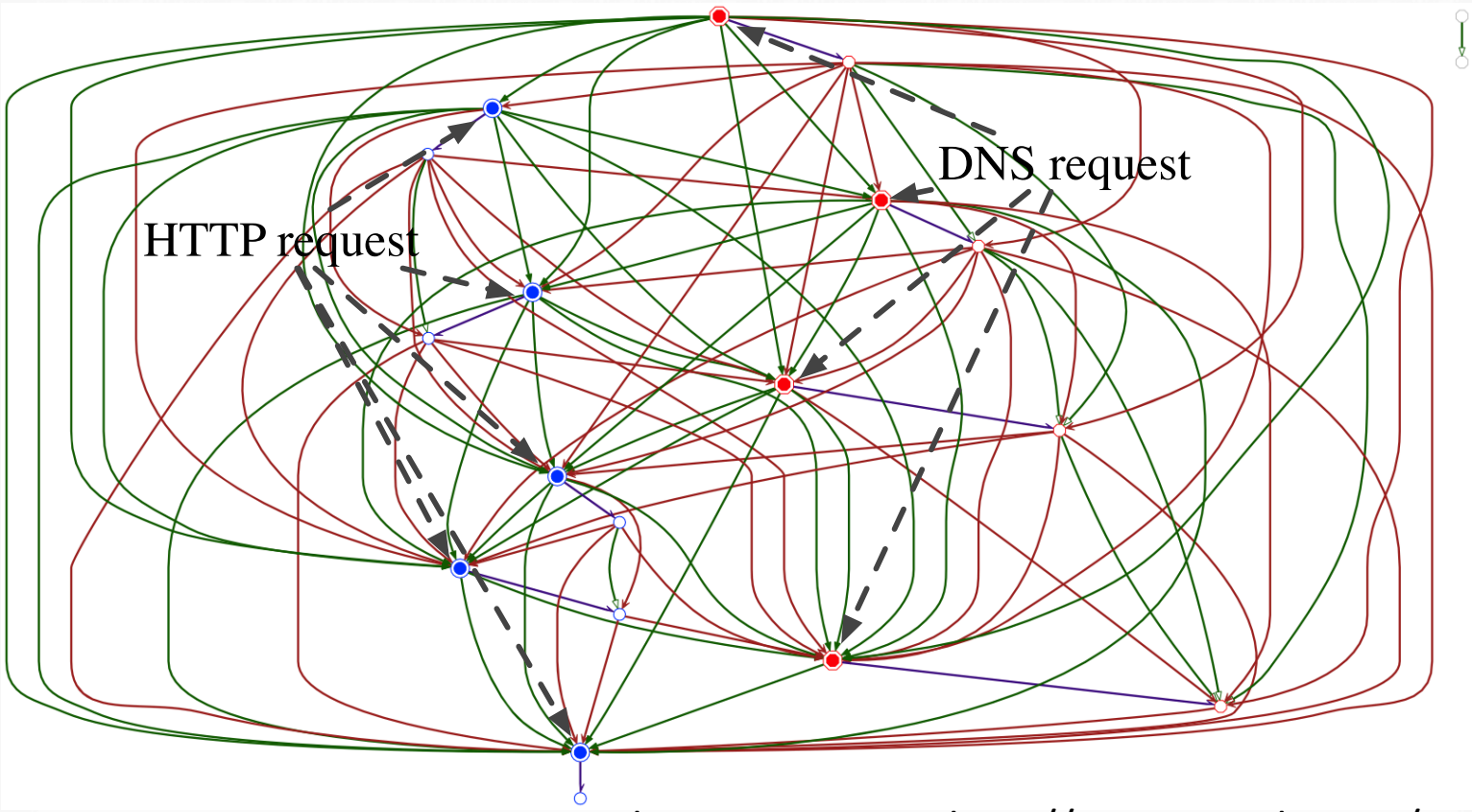
end procedure

Checking temporal order and threshold, here

An example trace (flow diagram)



Traffic causality graph (TCG)



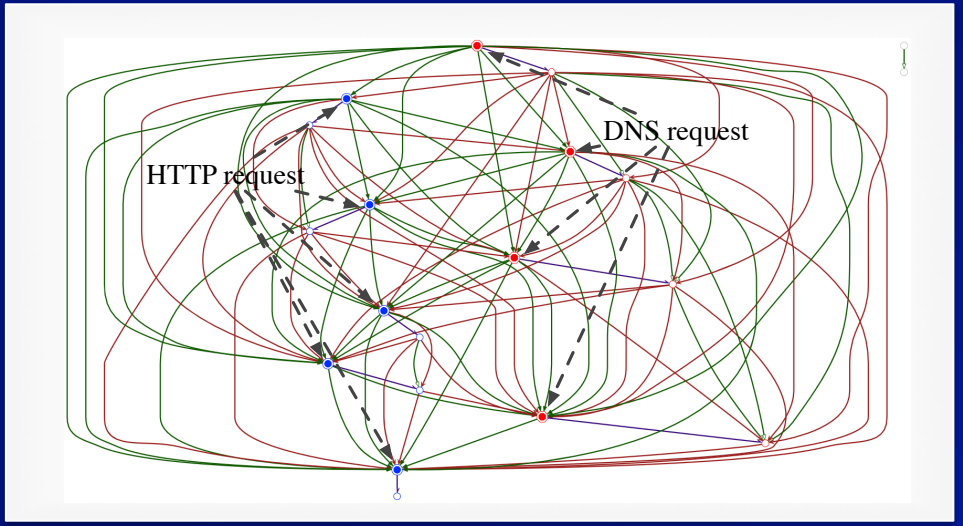
Two problems of this TCG

- Too many edges for PR, DHR, and SHR
  - Some are not direct causality

Not good to generate patterns or features for profiling

- Irrelative edges
  - Due to the simple algorithm

Edge reduction

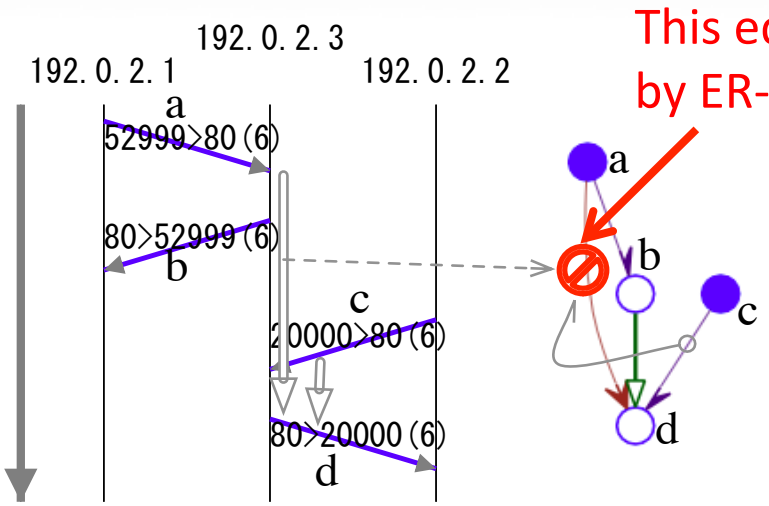


Edge reduction

- ER-Rule 1: Removing tenuous edges
  - Remove all PR, DHR, and SHR edges except for temporally closest ones, i.e., max-degree = 1
- ER-Rule 2: Removing irrelative edges
  - Modify the simple algorithm by looking at neighbors
- ER-Rule 3: Removing insignificant/ uninteresting edges (optional)
  - Remove some edges by looking at neighbors

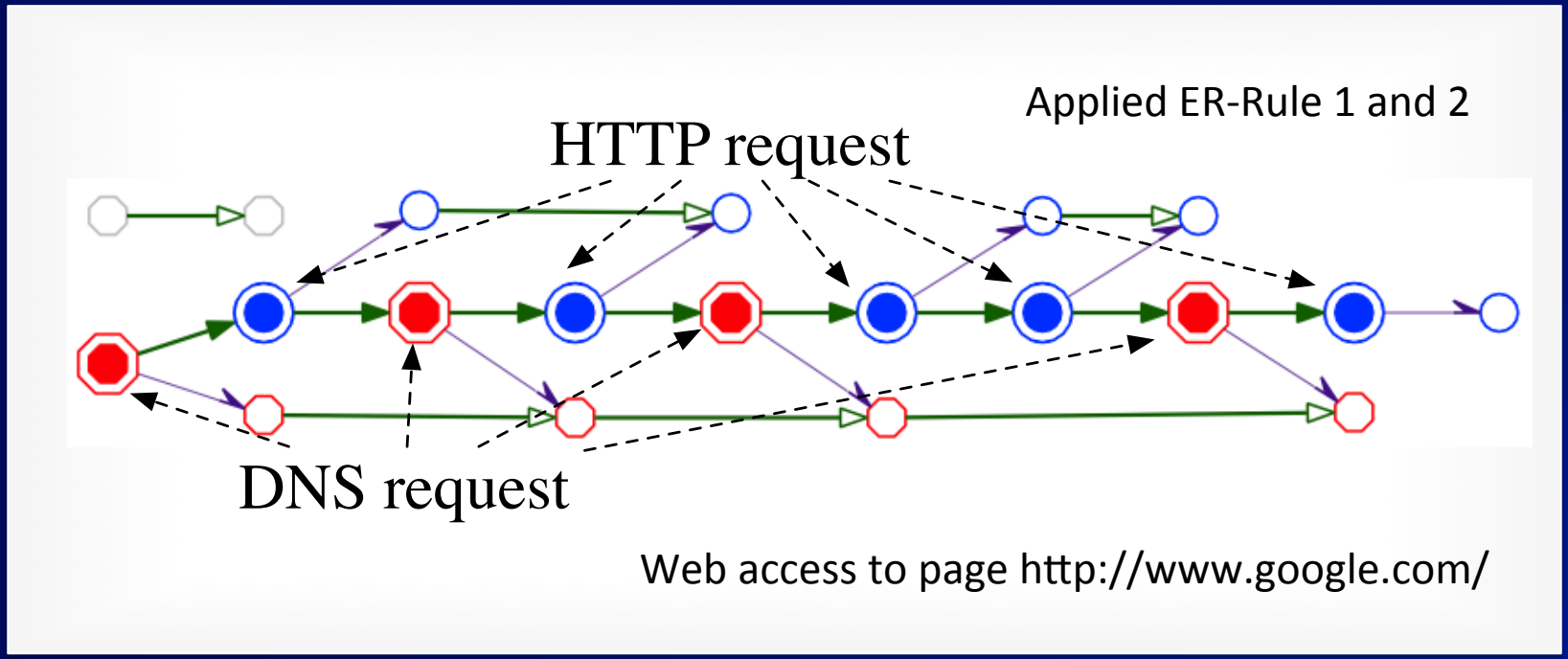
Details in paper

An example of edge reduction



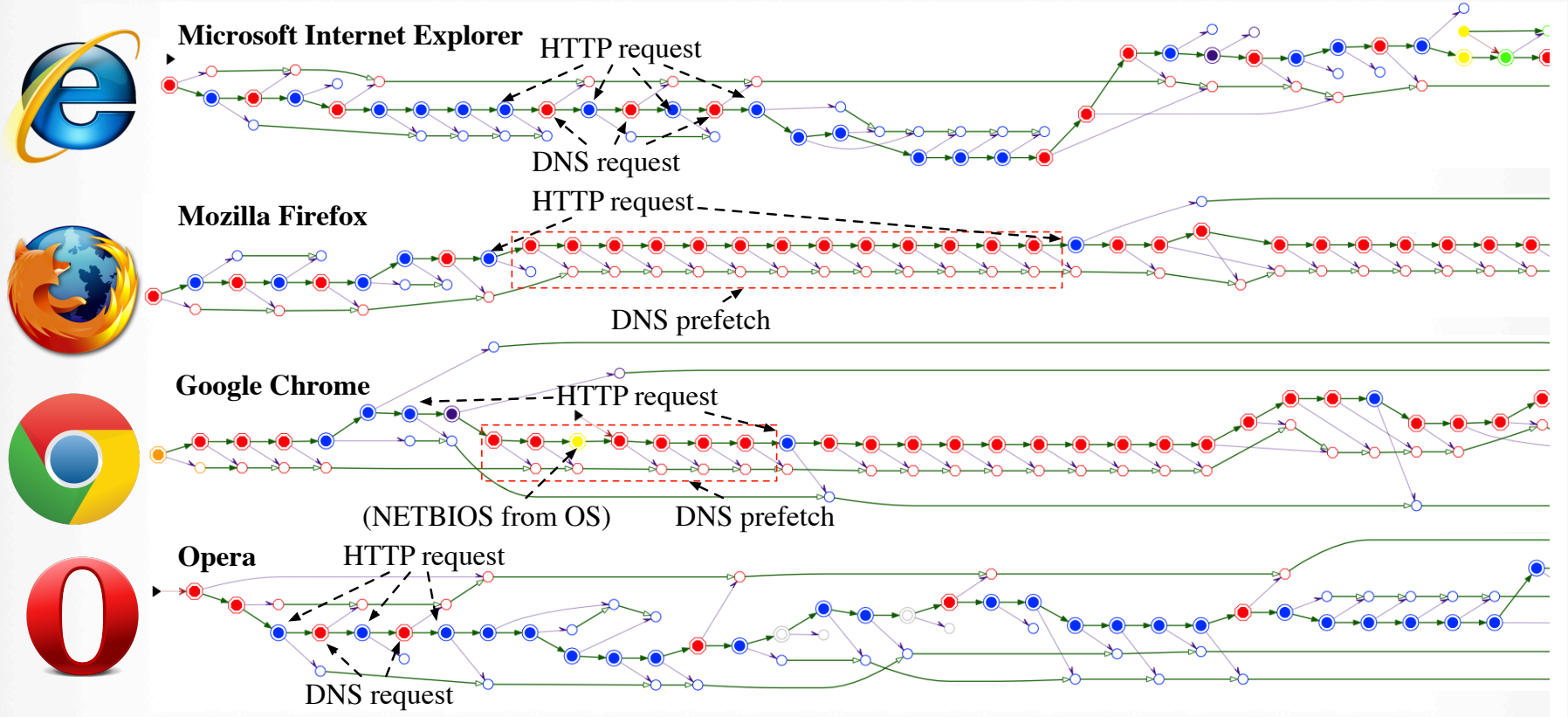
(b) PR edges to CR-response and DHR/SHR edges from CR-request to CR-response

Modified traffic causality graph (TCG) w/ edge reduction



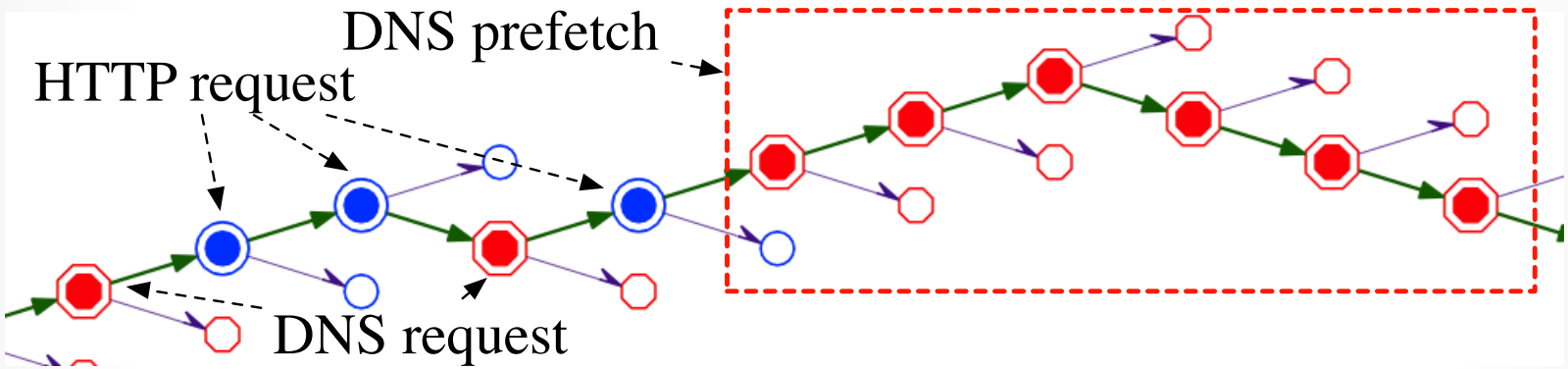
Case Studies

Well-known Web browsers (ground truth packet trace)

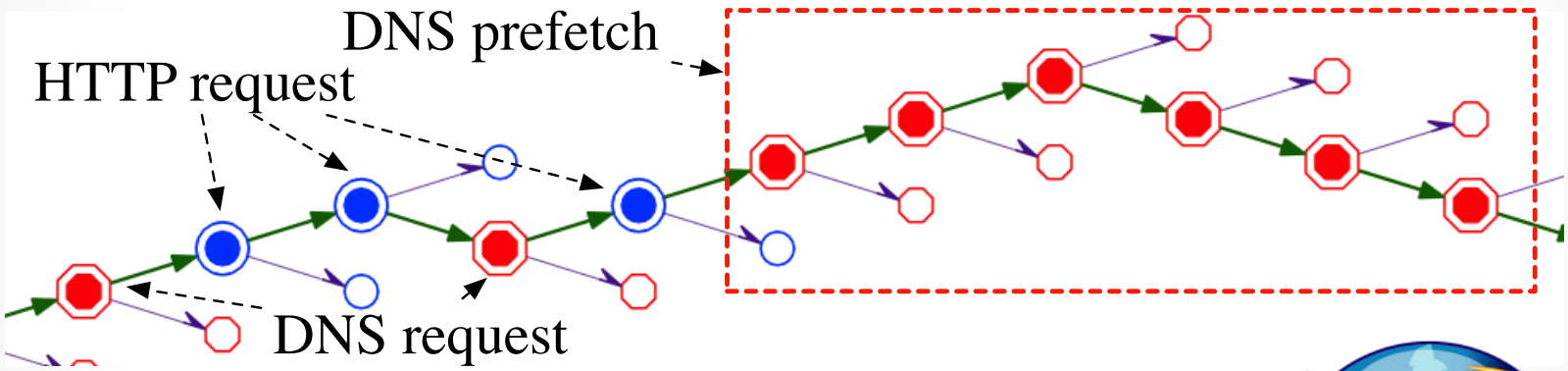


Browsed same pages in the same manner

Web browsing TCG from a measured traffic trace

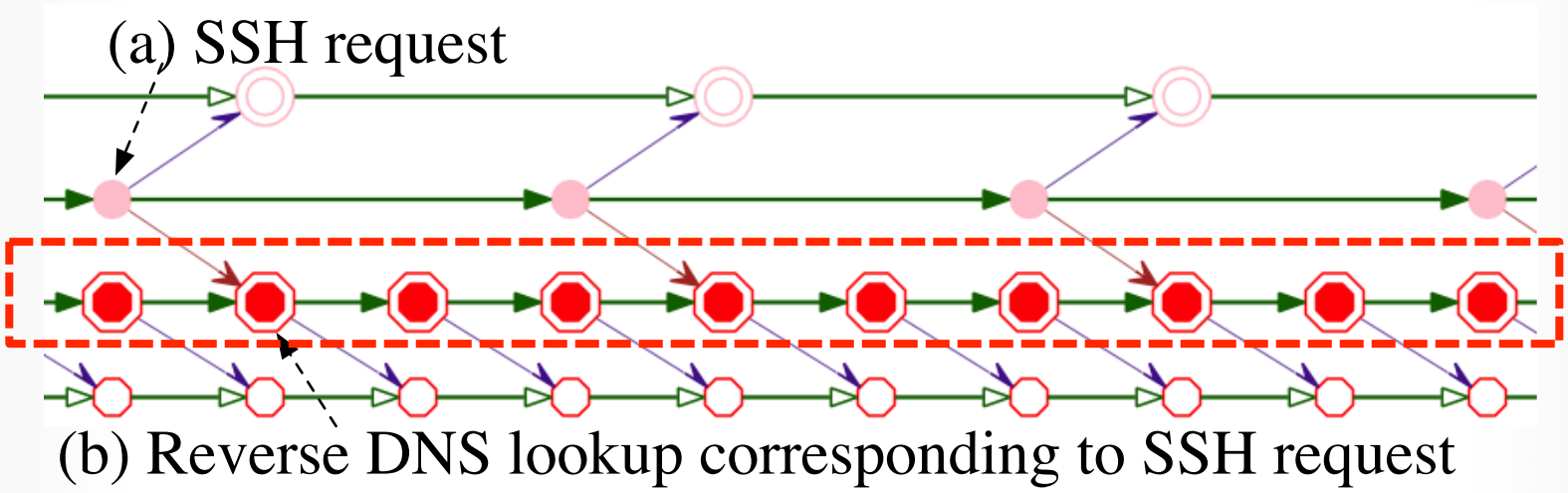


Web browsing TCG from a measured traffic trace



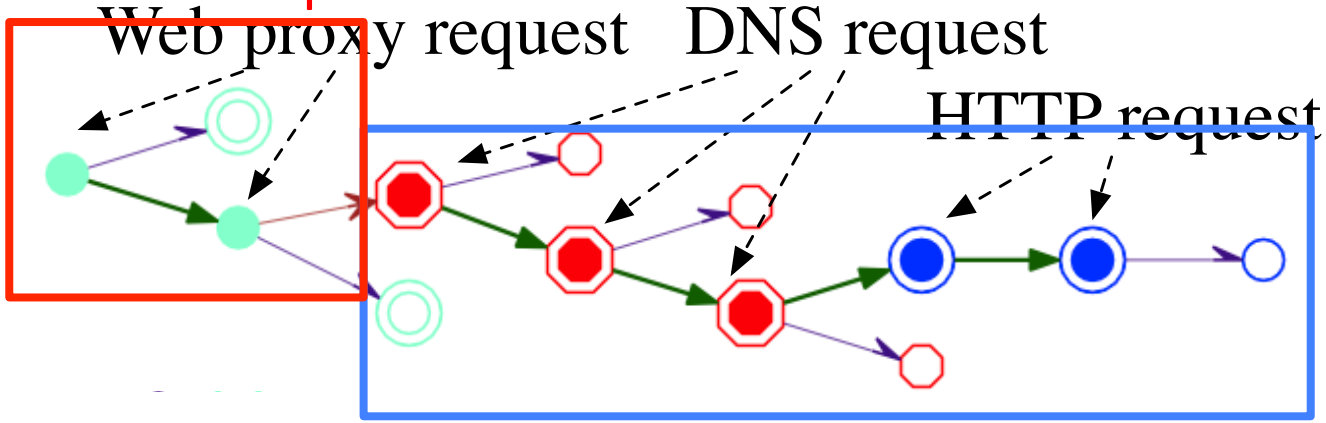
Firefox by manual inspection

SSH brute force attack



Web proxy

Important for operators



Similar to Web browsing behavior

Towards Automated Profiling

Preliminary evaluation

- Simple features (not patterns now)
  - PR-CR
    - #PR/#CR in a TCG
  - DNS-DHR
    - # of a DHR edges from dst/UDP/53 to dst/UDP/53
      - i.e., consecutive DNS requests

I know these are far too much simple…

Results

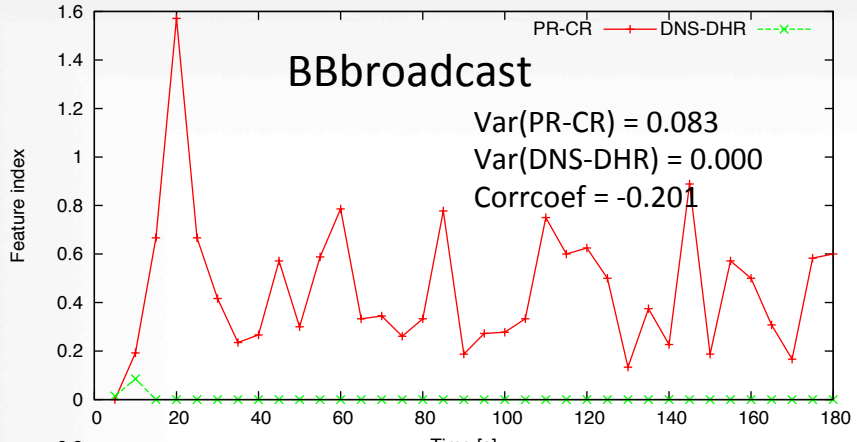
TABLE II
RESULTS: FEATURES OF TCGS ($\tau = 1$[S]; ER-RULES: 1, 2, AND 3(A))

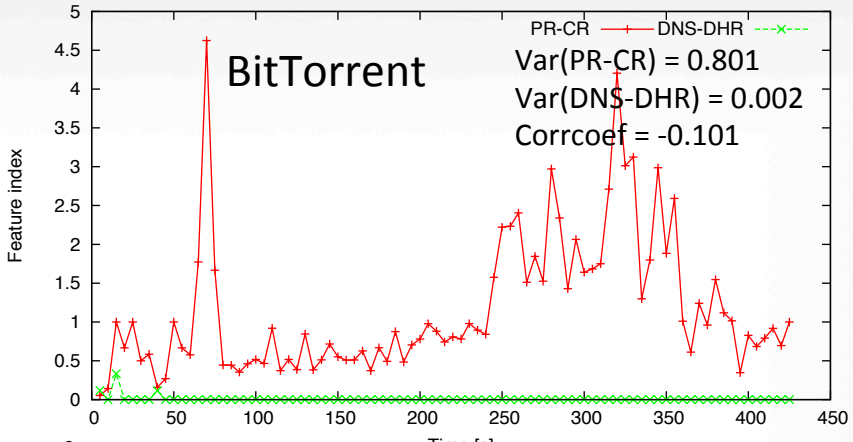
Application program	#Edges	PR-CR	DNS-DHR
Microsoft Internet Explorer	270	0.029	0.077
Mozilla Firefox	739	0.008	0.600
Google Chrome	1161	0.025	0.580
Opera	516	0.020	0.034
BitTorrent	3444	0.595	0.025
LimeWire	4803	0.320	0.058
Perfect Dark	905	0.345	0.000
BBbroadcast	373	0.246	0.006
SSH brute force attacks (60 s)	619	0.259	0.586
Web proxy(5 s)	3668	0.614	0.100

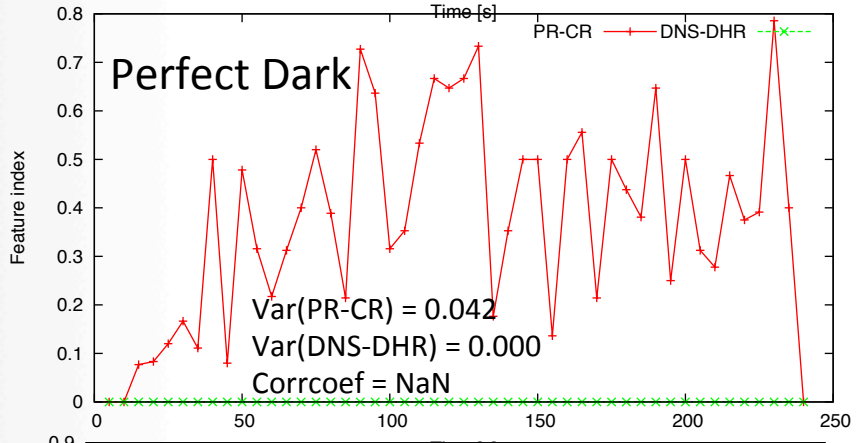
Evidence of prefetch-enabled

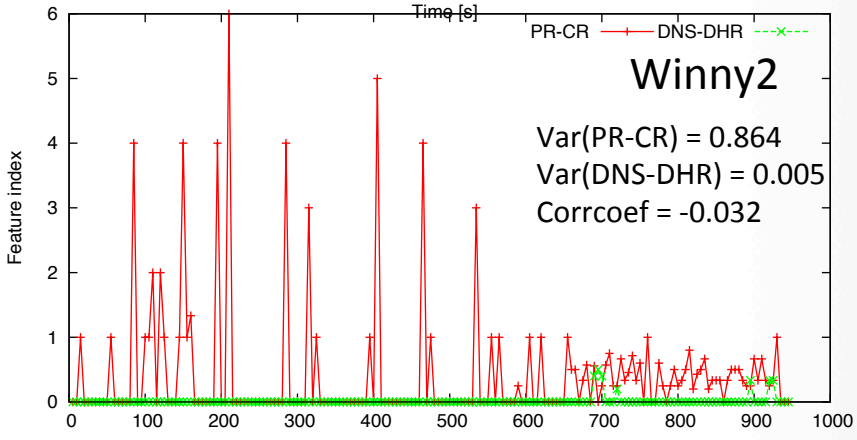
Evidence of data-relay

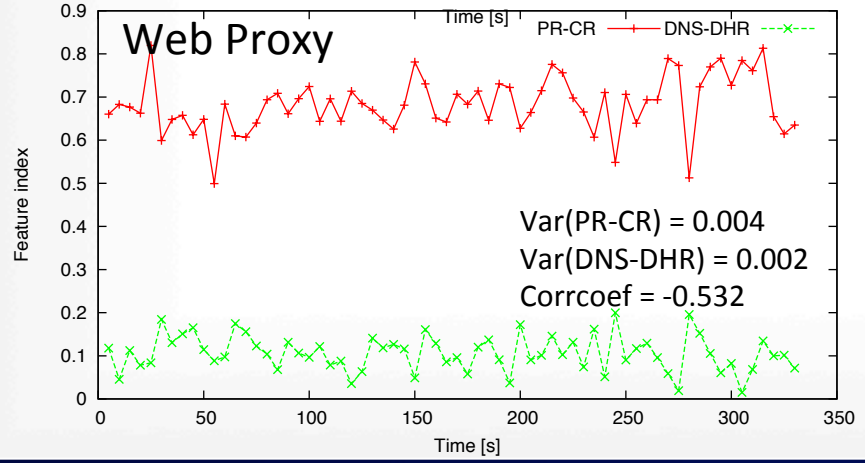
"Simple" features of TCGs are discriminative.

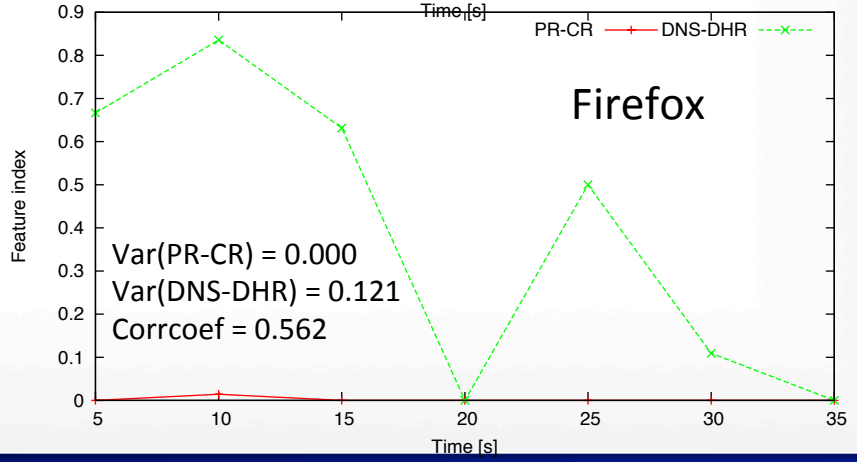

BBbroadcast
PR-CR
DNS-DHR
Var(PR-CR) = 0.083
Var(DNS-DHR) = 0.000
Corrcoef = -0.201
Feature index
Time [s]

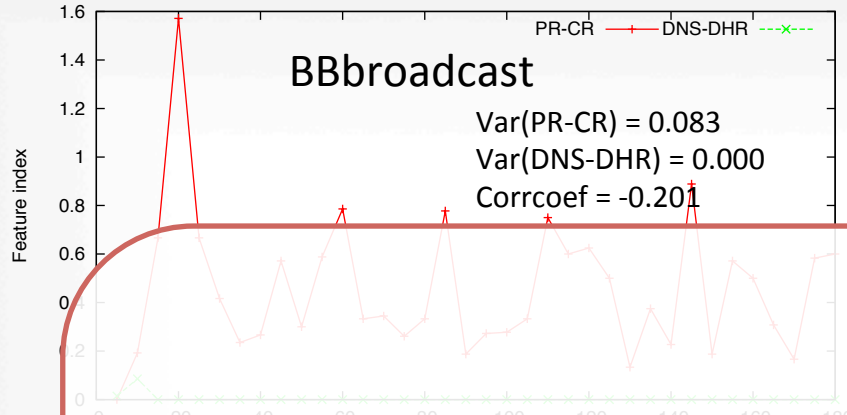

BitTorrent
PR-CR
DNS-DHR
Var(PR-CR) = 0.801
Var(DNS-DHR) = 0.002
Corrcoef = -0.101
Feature index
Time [s]

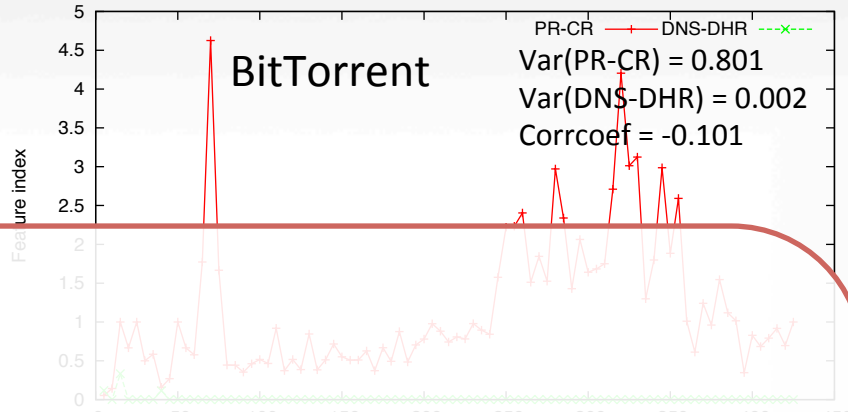

Perfect Dark
PR-CR
DNS-DHR
Var(PR-CR) = 0.042
Var(DNS-DHR) = 0.000
Corrcoef = NaN
Feature index
Time [s]


Winny2
PR-CR
DNS-DHR
Var(PR-CR) = 0.864
Var(DNS-DHR) = 0.005
Corrcoef = -0.032
Feature index
Time [s]


Web Proxy
PR-CR
DNS-DHR
Var(PR-CR) = 0.004
Var(DNS-DHR) = 0.002
Corrcoef = -0.532
Feature index
Time [s]

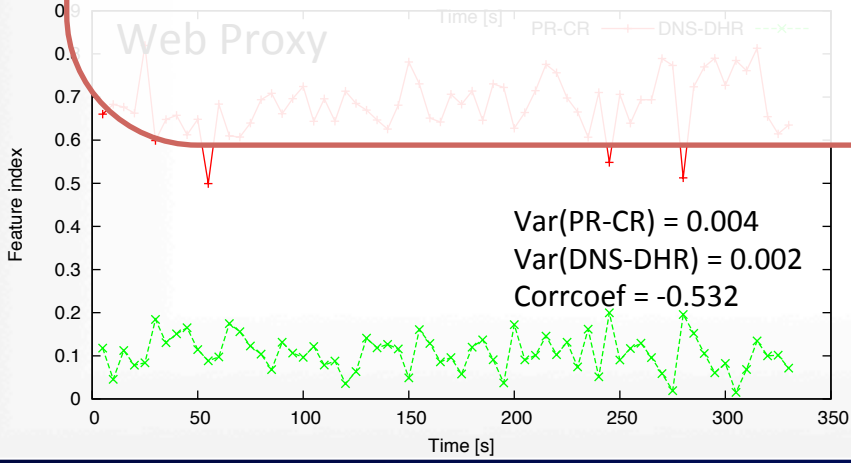

Firefox
PR-CR
DNS-DHR
Var(PR-CR) = 0.000
Var(DNS-DHR) = 0.121
Corrcoef = 0.562
Feature index
Time [s]

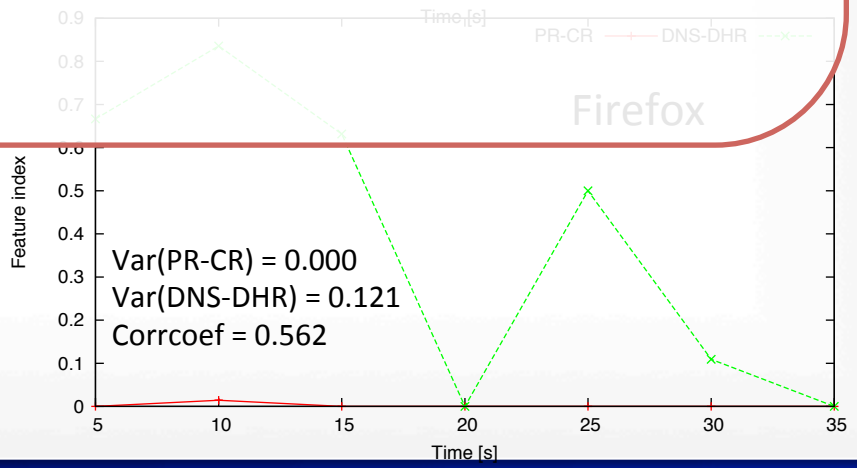




Statistical properties of the features can be a profile of network applications.

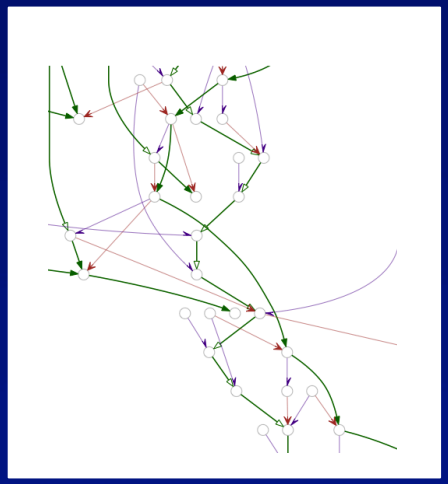
Other features or graph analysis approaches will be examined in future.

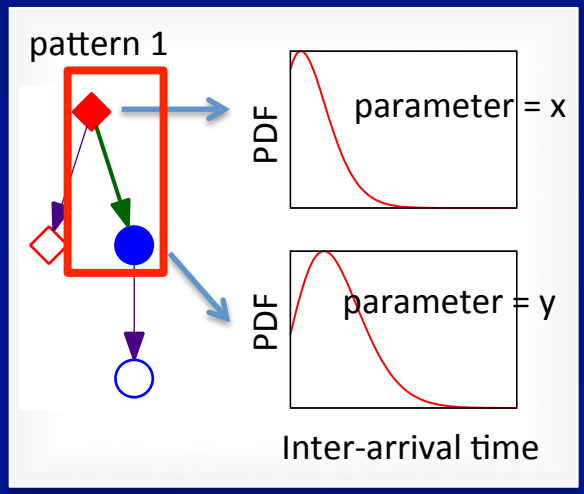




Conclusion

- Traffic causality graphs help operators
  - identify root causes by TCG visualization
  - profile application programs by TCG features
- Future work
  - TCG edge reduction vs. weighted edge w/ heuristics
  - Use flow properties against port randomization
  - Extend the automated profiling to graph mining and pattern matching





Thank you for your attention.
Questions or comments?

Hirochika Asai

THE UNIVERSITY OF TOKYO

Ph.D candidate at Esaki Lab.
Graduate School of Information Science and Technology,
The University of Tokyo, Japan
Email: panda@hongo.wide.ad.jp