Computer Networks 109 (2016) 200–210

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

A study of the impact of DNS resolvers on CDN performance using a causal approach

Hadrien Hours[a,b,*], Ernst Biersack[c], Patrick Loiseau[a], Alessandro Finamore[d,e], Marco Mellia[d]

[a] Eurecom, Campus SophiaTech, 450 Route des Chappes, 06410 Biot, France
[b] Ecole Normale Superieure de Lyon - Site monod, 46 Alle d'Italie - 69007 Lyon, France
[c] CAIPY, 06560 Valbonne, Fance
[d] Torre Telefonica Diagonal 00, Plaza de Ernest Lluch i Martin, 5, 08019 - Barcelona, Spain
[e] Politecnico di Torino - Corso Duca degli Abruzzi, 24 - 10129 Torino, Italy

ARTICLE INFO

Article history:
Received 30 September 2015
Revised 9 June 2016
Accepted 17 June 2016
Available online 23 June 2016

Keywords:
Causality
CDN
DNS
Traffic performance
Bayesian networks
Optimization
Monitoring
Knowledge inference
Computer networks

ABSTRACT

Resources such as Web pages or videos that are published in the Internet are referred to by their Uniform Resource Locator (URL). If a user accesses a resource via its URL, the host name part of the URL needs to be translated into a routable IP address. This translation is performed by the Domain Name System service (DNS). DNS also plays an important role when Content Distribution Networks (CDNs) are used to host replicas of popular objects on multiple servers that are located in geographically different areas. A CDN makes use of the DNS service to infer client location and direct the client request to the optimal server. While most Internet Service Providers (ISPs) offer a DNS service to their customers, clients may instead use a public DNS service. The choice of the DNS service can impact the performance of clients when retrieving a resource from a given CDN. In this paper we study the impact on download performance for clients using either the DNS service of their ISP or the public DNS service provided by Google DNS. We adopt a causal approach that exposes the structural dependencies of the different parameters impacted by the DNS service used and we show how to model these dependencies with a Bayesian network. The Bayesian network allows us to explain and quantify the performance benefits seen by clients when using the DNS service of their ISP. We also discuss how the further improve client performance.

1. Introduction

Each time an Internet user wants to access a resource, he uses a human readable name called Uniform Resource Locator (URL), containing the domain name of the administrative entity hosting this resource. However, a domain name is not routable and needs to be translated into the IP address of a server hosting the resource the client wants to access. This is taken care of by the DNS service. At the same time, many popular services such as YouTube, iTunes, Facebook or Twitter, rely on CDNs, where objects are replicated on different servers, and in different geographical locations to optimize the performance experienced by their users. When a client accesses an object hosted by a CDN, its default DNS server contacts the DNS server of the CDN that hosts the resource the client is requesting. Based on the origin of the request, the authoritative CDN DNS redirects the client to the optimal server. Most of the ISPs provide a DNS service, but it is now common to see customers using a public DNS service instead [10]. Clients using the DNS service of their ISP are served by a local DNS server that often provides a more accurate location information to the CDN compared to the information communicated by a public DNS service such as the Google DNS service. Indeed, public DNS servers are usually further away from the clients of a given ISP than the default ISP DNS server. There have been several studies suggesting that public DNS services do not perform as well as local DNS services provided by ISPs, mainly because of the impossibility of public DNS to correctly communicate the location of the clients originating the request [1,7]. This problem is addressed with ECS (edns-client-subnet) [16] but Akamai does not support it currently.

Studying the performance of the users accessing resources in the Internet is a complex task. Many parameters influence the end user experience and the relationships between these parameters is not always observable or intuitive. It is therefore necessary to

* Corresponding author.
E-mail addresses: hadrien.hours@eurecom.fr, hadrien.hours@ens-lyon.fr (H. Hours), erbi@e-biersack.eu (E. Biersack), patrick.loiseau@eurecom.fr (P. Loiseau), alessandro.finamore@telefonica.com, alessandro.finamore@polito.it (A. Finamore), marco.mellia@telefonica.com (M. Mellia).

http://dx.doi.org/10.1016/j.comnet.2016.06.023
1389-1286/© 2016 Elsevier B.V. All rights reserved.

use a simple, yet formal model that allows us to understand the role of a given parameter and its dependencies with other parameters. Bayesian networks offer a simple and concise way to represent complex systems [2]. In this paper, we use a Bayesian network to represent the causal model that captures the impact of the DNS service on the throughput performance experienced by clients accessing resources hosted by the Akamai CDN. Bayesian networks capture the dependencies between the different parameters impacting the throughput of the clients. One very interesting property of causal models is their *stability under intervention*. Causal models can be used to predict how the throughput of CDN users would evolve if we would *intervene* on the different parameters influencing the performance of CDN users. Here an intervention consists in isolating a given parameter of the system being studied, removing all its direct and remote causes and fixing its variations to a pre defined value or distribution. Being able to predict the effect of interventions, we can use causal models to understand the observed performance of a given system and to design strategies to improve its performance. In this work, we infer and use the causal model of CDN performance to understand the impact of choosing one DNS service instead of another. From such a model we are able to explain why clients using the DNS service of their ISP experience better download performance than clients using the Google DNS service. We are also able to indicate how to further improve the performance of the clients using the DNS service of their ISP.

Our work differs from previous studies of DNS services in several important points:

- We use a causal approach that formally models the structural dependencies of the different parameters influencing the throughput obtained.
- Observing that the clients using the DNS service of their ISP (referred to as local DNS) experience higher throughput than the clients using the public DNS service (referred to as Google DNS), we can show that this performance difference is due to the fact that clients using the DNS service of their ISP are redirected to closer servers. We are also able to precisely quantify this performance improvement.
- The causal model of our system also reveals that the parameterization of TCP (initial congestion window) of the servers accessed by the users of the Google DNS plays a key role in their throughput performance. Besides fully explaining the observed performance, this result also indicates how to further improve the performance of the clients using the local DNS.

Overall, the main contribution of our work resides in the methodology adopted and in its use of counterfactuals to understand the causal dependencies of a complex system.

In Section 2, we introduce causal models and their use to predict interventions, summarizing some of the main concepts from [11,15]. We then present, in Section 3, the environment of our study and the description of the parameters constituting our system. Section 4 presents our study of the DNS impact on the throughput. In particular we present the causal model of our system where we can observe the impact of the choice of the DNS service on the throughput. Our approach also allows us to predict the improvement that could be achieved by modifying the parameterization of the servers accessed by the users of the local DNS service. Section 5 compares our approach to the related work and Section 6 summarizes our work and proposes directions to further extend our work.

Several methods mentioned in this paper were designed and validated with parallel studies that are presented in an Appendix. The Appendix is available with the online version of this paper.[1] We give references to these studies in the paper.

2. Causal model: Definitions and usage

To model a complex system such as a communication network and to organize the knowledge obtained from its passive observation is very challenging. Existing work typically looks for the presence of correlation between different events observed simultaneously (see [9] and references therein). However, *correlation is not causation and the detection of correlation between two parameters A and B does not inform us on how they are related. A* can impact *B*, or the other way around, or an unobserved parameter can impact both *A* and *B* simultaneously. The difference between correlation and causation plays an important role if we want to find out how to improve our system by partly modifying its behavior. A causal approach uncovers the structural dependencies between the parameters of the system under study. The ability to predict the effects of a manipulation of the parameters of a system is a major strength of causal models as they are *stable under intervention*. Stability under intervention means that a causal model, inferred from the observations of a system in a given situation, is still valid if we manually change the system mechanisms, redefining the systems laws. The manual modification of the system parameters is called an *intervention*. Interventions consist in modifying the behavior of a component of the system, removing the influence of its direct and remote causes and manually setting its variations. The inference of a causal model and of a causal effect [11,15] is made using passive observations only. The causal theory allows us to predict the behavior of the various parameters of the inferred model after an intervention *without the need of additional observations*.

In this section we present the PC algorithm [14] that is used to infer the causal model of our system. We also describe the different properties of a causal model as described in [11,15].

2.1. Causal model: Inference

For our work, we use the PC algorithm [14] to build the Bayesian graph representing the causal model of our system. This algorithm takes as input the observations of the different parameters that characterize our system and infers the corresponding causal model. In our representation of a causal model as a Bayesian network, each node represents one parameter of our system and the presence of an edge from a node X to a node Y ($X \rightarrow Y$) represents the existence of a causal dependence of parameter Y on parameter X.

The PC algorithm starts with a fully connected and unoriented graph, called *skeleton*, where each parameter is represented by a node and connected to every other parameter. The PC algorithm then trims the skeleton by checking for independencies between adjacent nodes:

- First, the unconditional independencies ($X \perp\!\!\!\perp Y$) are tested for all pairs of parameters and the edges between two nodes whose corresponding parameters are found to be independent are removed.
- For the parameters whose nodes are still adjacent, the PC algorithm then checks if there exists a conditioning set of size one that makes two adjacent nodes independent. If this is the case, it removes the edge connecting the corresponding two nodes, otherwise the edge is kept.
- The previous step is repeated, increasing the conditioning set size by one at each step, until the size of the conditioning set reaches the maximum degree of the current skeleton (the

[1] http://dx.doi.org/10.1016/j.comnet.2016.06.023

maximum number of adjacent nodes for any node in the current graph), which means that no more independencies can be found.

The final step of the PC algorithm consists in orienting the edges. First, the PC algorithm orients all the V-structures, i.e. subgraphs $X - Z - Y$ where X and Y are not adjacent, and then orients as many edges as possible without creating new colliders or cycles [11]. A node Z is a *collider* if it is part of an oriented subgraph $X \rightarrow Z \leftarrow Y$ where X and Y are not adjacent. An illustration of the different steps of the PC algorithm is presented in the Appendix A.1.

2.2. Causal model: Properties and theorems

In this section we assume that we have the causal model of our system that is represented by a Bayesian network. We focus on two parameters X and Y, where Y represents the performance of our system and we are interested in the global effect on Y when intervening on X, including the effects mediated by external parameters also impacted by this intervention. We call this causal effect the *total causal effect*. Details of the implementation of the methods presented in this section can be found in the Appendix B.

2.2.1. Atomic interventions

We denote by $do(X = x)$ (or $do(x)$) the intervention that consists in intervening on the parameter X by fixing its value to be x. An intervention that simply assigns to X a fixed value is called an *atomic intervention*. The difficulty of predicting the effect of an intervention comes from the possible presence of spurious associations between the intervention variable and the response variable. A *spurious association* between X and Y is an association between X and Y due to external parameters ($\notin \{X, Y\}$). To obtain an unbiased estimation of the effect of an intervention, we need to remove the effect of spurious associations. As an intervention is equivalent to isolating a given parameter from its direct and remote causes and to assigning it a fixed value, we need to remove the effects of direct and remote causes in our estimations. Such estimation is complex if one needs to consider all the possible inter dependencies between the different parameters influencing the performance of the system being studied. However, the use of a *graphical causal model*, where the different dependencies are present, makes it easy to estimate the outcome of interventions. Different criteria (c.f. [11]) can be used to identify the minimum set of parameters that block the effects of direct and remote causes when estimating the effect of a given intervention.

If G denotes the Bayesian graph that represents the causal relationships between the parameters of our system, we use $G_{\overline{X}}$ to denote the sub-graph of G where all the edges entering X are removed and $G_{\underline{X}}$ the sub-graph of G where all the edges exiting X are removed. We can use the rules of *do-calculus* [11] to estimate the distributions of the parameters of our system after an intervention based on their distributions prior to this intervention. Note that these rules do not make any assumption regarding the distributions or functional dependencies of the parameters.

We briefly recall the *Rules of do calculus* that will be used in Section 4.2 to predict the interventions we are interested in this work. Let P denote a (possibly multivariate) probability distribution specified by the probability mass function or probability density function, depending on the nature of the parameters.

Theorem 1 (3.4.1 from [11]). *(Rules of do calculus) Let G be the directed acyclic graph associated with a causal model [...] and let* $P(\cdot)$ *stand for the probability distribution induced by that model. For any disjoint subsets of variables X, Y and Z we have the following rules.*

Rule 1 *(Insertion/deletion of observation):*

$$P(y|do(x), z, w) = P(y|do(x), w) \text{ if } (Y \perp\!\!\!\perp Z \mid X, W)_{G_{\overline{X}}} \quad (1)$$

Rule 2 *(Action/observation exchange):*

$$P(y|do(x), do(z), w) = P(y|do(x), z, w) \text{ if } (Y \perp\!\!\!\perp Z \mid X, W)_{G_{\overline{X}\underline{Z}}} \quad (2)$$

Rule 3 *(Insertion/deletion of intervention):*

$$P(y|do(x), do(z), w) = P(y|do(x), w) \text{ if } (Y \perp\!\!\!\perp Z \mid X, W)_{G_{\overline{X}\,\overline{Z(W)}}}, \quad (3)$$

where Z(W) is the set of Z-nodes that are not ancestor of any W-nodes in $G_{\overline{X}}$.

2.2.2. Enforcing intervention with a given probability

To study the impact of the DNS service on the performance seen by the clients (c.f. Section 4) we must estimate the effect of interventions on the parameters influenced by the DNS service and on the parameters influencing the throughput. To do so, we cannot use atomic interventions since we intervene on a given parameter by *changing its distribution*. If we want to predict how an intervention on X affects Y, where the intervention on X is enforced with the conditional probability distribution $f^*(X|Z)$, we obtain [11, Section 4.2]:

$$f(y)_{|f^*(x|z)} = \int_{D_X}\int_{D_Z} f_{Y|do(X),Z}(y, x, z) f^*(x|z) f(z) \mathrm{d}x\mathrm{d}z. \quad (4)$$

2.3. D-separation

The d-separation criterion is a graphical criterion to decide, by looking at the graph, if two parameters, represented by their nodes, are independent. D-separation associates the notion of connectedness with dependence. If there exists a directed path between two nodes, the nodes are said to be d-connected and their corresponding parameters are dependent. On the other hand, if we condition on one of the nodes in the path from X to Y, then this node is said to block the path and X and Y are conditionally independent relative to this path. For X and Y to be independent, one must block *all* the paths d-connecting X and Y. When studying d-separation, an important notion is the one of collider. The presence of a collider on a *undirected* path blocks this path. While conditioning on a collider unblocks the path which can be explained by the fact that two independent causes become dependent if one conditions on their common consequence.

2.4. Density estimation

The theory of causality [11] makes no assumption on the distribution of the parameters. We estimate the multidimensional probability density functions via Copulae [8], using the Sklar theorem.

The Sklar theorem stipulates that, if F a is multivariate cumulative distribution function with marginals $(F_1, \ldots, F_i, \ldots, F_n)$, there exists a copula C such that

$$F(x_1, \ldots, x_i, \ldots, x_n) = C(F_1(x_1), \ldots, F_i(x_i), \ldots, F_n(x_n)). \quad (5)$$

There are different types of copulae, in our work we focus on T-copulae [3] and G-copulae [13]. T-copulae present the advantage that, by tuning the different parameters of the T-copula, one can better capture the tail dependencies between the different components of the multi-variate distribution that is modeled. This is highly useful in our case where the performance (e.g. the throughput of a Web user) can be strongly affected by changes to the characteristics of the network such as packet loss or delay. Unfortunately, T-copulae are complex to parameterize, which implies that more data is needed to fit such model to our problem. In this paper, we are interested in counterfactuals such as "*How would the system behave under the condition* C1 *if one of its parameter was to behave as it has done under the condition* C2, *knowing that* C1 *and* C2 *are exclusive ?*". Counterfactuals correspond to the predictions of complex interventions, each of which requires conditioning on several variables in order to block the different spurious associations.

We decided to use Gaussian copulae [13], which are known to be less sensitive if the amount of data available is limited (see Appendix C).

In the bivariate case, the Gaussian copula is defined as:

$$C_\rho(u,v) = \Phi_\rho(\Phi^{-1}(u), \Phi^{-1}(v)), \tag{6}$$

where ρ represents the correlation matrix and Φ the CDF of the standard normal distribution. The marginals, $F_i(x_i)$, are estimated using normal kernels.

The choice of Gaussian copulae as well as the methods and their implementation to compute the conditional PDFs have been designed and validated based on studies made on artificial data sets that are presented in the Appendix C.

3. Experimental set up

In this section, we present how we do the data collection and how we extract the parameters of interest. We define our system as the set of parameters (see Section 2.1) and observe these parameters in different situations to capture their dependencies and infer the corresponding graphical causal model.

3.1. Experiment design

We collect IP packet traces at a Point of Presence (PoP) of a large European ISP and extract all the traffic directed to or coming from the Akamai CDN. To model the impact of the choice of DNS service on the client throughput, we make three choices: i) We only focus on the traffic carried by the TCP protocol. ii) To eliminate the impact of TCP slowstart, we only consider large TCP connections that carry at least 2MBytes of data. iii) As more than 90% of the observed connections use either Google DNS (*GDNS*) or the DNS of the local ISP (*LDNS*) we consider only these two DNS services.

The probe capturing the traffic is placed between the client and the server. We call internal network, denoted as *isp network*, the part of the network between the client and the probe. We call external network the part of the network between the probe and the server, assimilated to the Internet network and denoted as *inet network*. The traffic was captured on two different days, a Thursday and a Sunday, from 5.30 pm to 9.30 pm.

3.2. Parameters of our model

We use the Tstat software [4] to extract from the packet traces relevant information on a per connection basis. We have about 7000 connections. We use domain knowledge to select a subset of the information obtained from Tstat that represents the parameters known to impact the throughput. In addition to the information obtained from Tstat we collect for each connection additional information such as the DNS service used, the number of hops between the client and the server and the server address.[2] The packet traces used for this study are confidential and cannot be shared publicly.

3.3. Summary of our data

Each connection is described by 19 parameters. In Table 1, we present the average (μ), minimum (*min*), maximum (*max*), standard deviation (σ) and coefficient of variation ($CoV = \frac{\sigma}{\mu}$) of each of the 19 parameters.

Since we are interested in comparing the performance of LDNS users and GDNS users, Table 2 presents the statistics for the connections where the LDNS is being used and for the connections where the GDNS is being used.

Table 1
Summary of the different parameters.

Parameter	μ	*min*	*max*	σ	*CoV*
dstip	N.A.	1300	34000	N.A.	N.A
dns	N.A	1	3	N.A.	N.A.
dow	N.A.	4	7	N.A.	N.A.
tod (s)	7100	52,000	78,000	4400	0.1
isprttavg (ms)	76	0	19,000	460	6.1
isprttstd (ms)	100	0	37,000	960	9.2
ispnbhops	1.8	1	3	0.51	0.3
inetrttavg(ms)	26	0.48	660	27	1.0
inetrttstd (ms)	8.2	0	4700	61	7.5
inetnbhops	9.4	2	21	2.8	0.3
rwin0	0.83	0	360	11	13
rwinmin (kB)	31.3	0.004	65	22.5	0.9
rwinmax (kB)	213	17.5	2625	150	0.7
cwinmax (kB)	150	7.3	1625	103	0.7
cwinmin (kB)	0.9	0.001	1.5	0.6	0.7
retrscore	0.005	0	0.19	0.009	1.9
rto (bool)	0.11	0	1	0.32	2.8
nbbytes (MB)	23.8	2.1	3875	138	5.7
tput (Mbps)	3.2	0.006	35	2.6	0.8

We use the following notations:

- Parameters with the prefix *isp* represent the *isp network* statistics, while the ones with the prefix *inet* represent the *inet network* statistics.
- The suffix *avg* represents the average value of a given parameter for a single connection (for example the average Round Trip Time between the client and the probe is denoted *isprttavg*).
- The suffix *std* represents the standard deviation of a given parameter for a single connection (for example the standard deviation of the Round Trip Time between the probe and a server is denoted *inetrttstd*).
- The *rto* parameter is set to true if there was at least one packet retransmission due to a time out and to false otherwise
- The *retrscore* parameter represents the fraction of retransmitted packets for a single connection ($= \frac{\text{retransmissions}}{\text{total transmissions}}$).
- The parameters *rwin** and *cwin** represent receiver window and congestion window metrics respectively.
- The day of the week and time of the day are captured by the variables *dow* and *tod* respectively.

Destination IP (*dstip*), DNS (*dns*) and days (*dow*) are categorical data for which the average value, standard deviation or coefficient of variation do not exist.

Without discussing in detail the values of the different parameters in Table 1, we would like to draw the attention to the difference in the RTT values observed inside the ISP network and the RTT values observed in the Internet: the average RTT value *isprttavg* is almost three times as high as the average RTT value *inetrttavg*. The use of an ADSL on the access link and the large buffers used in ADSL networks not only increase the RTT but also result in high variations of the RTT values observed that correspond to a standard deviation of the *isprttstd* being more than ten times bigger than the *inetrttstd*.

4. Causal study of the impact of the DNS service used on throughput

4.1. Modeling causal relationships

We use the PC algorithm [14] and the kernel based independence test from [19] to obtain the Bayesian network showing the

[2] Since the addresses were anonymized we represented the server address by the Autonomous System (AS) number of the AS the server is located in.

Table 2
Summary of the different metrics for the two DNS: Local DNS (LD) and Google DNS (GD) (*dow* and *tod* are similar and provide no insight, so they were removed).

Par	μ		*min*		*max*		σ		*CoV*	
	LD	GD	LD	GD	LD	GD	LD	GD	LD	GD
isprttavg (ms)	80	61	0	0	19,000	15,000	470	440	5.9	7.2
isprttstd (ms)	1100	76	0	0	32,000	37,000	920	1100	8.3	14.0
ispnbhops	1.8	1.9	1	1	3	3	0.53	0.4	0.3	0.2
inetrttavg (ms)	20	48	0.48	11	510	660	20	38	1.0	0.8
inetrttstd (ms)	8.6	6.5	0	0	4700	1400	65	44	7.6	6.8
inetnbhops	8.7	12	2	5	17	21	2.4	2.7	0.3	0.22
rwin0	0.97	0.29	0	0	330	360	12	9.2	12.0	32.0
rwinmin (kB)	35	12	0.004	0.03	65	65	28	14	0.8	1.1
rwinmax (kB)	213	213	18	18	2625	2000	150	138	0.3	0.7
cwinmax (kB)	163	118	7.3	7.8	1625	738	108	72	0.7	0.6
cwinmin (kB)	0.9	1.2	0.001	0.001	1.5	1.5	0.6	0.5	0.7	0.4
retrscore	0.005	0.004	0	0	0.19	0.06	0.01	0.01	1.9	1.8
rto (bool)	0.11	0.11	0	0	1	1	0.32	0.31	2.8	2.9
nbbytes (MB)	29	7	2.1	2.1	3875	1375	150	44	5.3	6.5
tput (Mbps)	3.2	3	0.006	0.007	35	29	2.7	2	0.9	0.7

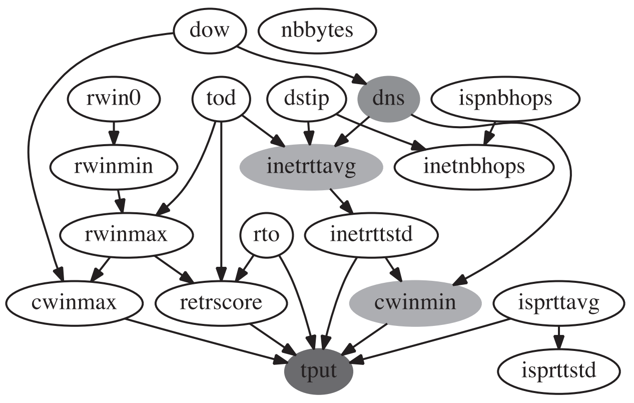

Fig. 1. Bayesian network representing the causal model of Web performance using two different DNS: the public Google DNS and the DNS of the local ISP with the following parameters: Day of the Week (*dow*), Number of bytes exchanged during the connection (*Nbbytes*), first advertised receiver window (*rwin0*), minimum advertised receiver window (*rwinmin*), maximum advertised receiver window (*rwinmax*), minimum server congestion window (*cwinmin*), maximum server congestion window (*cwinmax*), time of the day (*tod*), retransmission score (*retrscore*), presence of time outs (*rto*), server IP address (*dstip*), number of hops between client and probe (*ispnbhops*), number of hops between probe and server (*inetnbhops*), average external delay (*inetrttavg*), standard deviation external delay (*inetrttstd*), average internal delay (*isprttavg*) and standard deviation internal delay (*isprttstd*).

causal model of our system (c.f. Fig. 1). We briefly discuss some of the most interesting dependencies exhibited by this model.

The day of the week (*dow*) and the time of the day (*tod*) are two nodes that have no parents, which is not surprising. The time of the day (*tod*) influences the RTT between the probe and the server (*inetrttavg*), which captures the *peak hour effect* in the Internet.

In Table 1 we saw that that the variance of the internal RTT (*isprttstd*) was much higher than the one of the Inet RTT (*inetrttstd*) . This may lead one to expect that *isprttstd* has a stronger impact on the throughput than the *inetrttstd*. However, the causal model shows something different: we have a direct dependence between (*inetrttstd*) and the throughput (*tput*) but not between the standard deviation of the internal RTT (*isprttstd*) and the throughput. This example illustrates the ability of causal model to exhibit non intuitive dependencies.

We observe that the day of the week (*dow*) influences the DNS service used by the clients (*dns*). As our observations are made on two days (a Thursday and a Sunday), our conclusions are a bit limited. However, it appears that on Thursday 72% of the connections use the LDNS service against 28% using the GDNS service, while on Sunday 93% of the connections use the LDNS service against 7% using the GDNS service. It would be interesting to identify the clients using one DNS service and compare their locations with the ones of the clients using the other DNS service to better understand this dependence. The day of the week may capture the difference in the Internet usage and the devices used at home and at work. However, for privacy reasons, the IP addresses of the clients are obfuscated, which prevents us from investigating this hypothesis.

One of the most interesting dependencies, which motivated this work, is the one between the DNS service (*dns*) and the external RTT (*inetrttavg*). Our data show that most of the time, clients using the DNS of their ISP are redirected to an Akamai server located in the same AS. On the other hand, the clients using the Google DNS service are often redirected to servers located outside the client AS and even, in some cases, to a server outside of Europe.

It has been previously shown [7] that clients using the local DNS service benefit from a redirection to servers closer than the ones of the clients using a public DNS service. Our data (see Table 2) corroborate this observation since the average external RTT for the LDNS service users is of 20 ms, while the users of the GDNS service experience an average external RTT of 48 ms.

We can also see that congestion window metrics (*cwinmin, cwinmax*) have a direct impact on the throughput (*tput*). Additionally, the minimum congestion window (*cwinmin*) has the DNS (*dns*) as direct parent. Its average value for clients using GDNS is 1.2kB against 0.9kB for users served by the LDNS, see Table 2.

A parameter present in a causal model represents also the mechanisms captured by such parameter. This is the case of the *cwinmin* that also captures the tuning of the TCP parameters at the server side. Clients using the LDNS often access their objects from servers that are located *inside* the ISP network. These servers could have a configuration different from the servers accessed by the users of the GDNS. This hypothesis could also explain the fact that both DNS services result in a similar throughput performance despite the difference in the RTTs observed. Other reasons could be the impact of losses on the congestion window or the load of the servers being accessed by the clients. To capture the server load, we estimate the server processing time defined from the time at which a server sends the acknowledgment of the client HTTP/GET message and the time at which it sends the first data packet. However, the server processing time shows an expected value of 43 ms for the LDNS users against 64 ms for the GDNS users. A higher processing time for the servers accessed by the GDNS users suggests that they are more loaded. On the other hand, the congestion

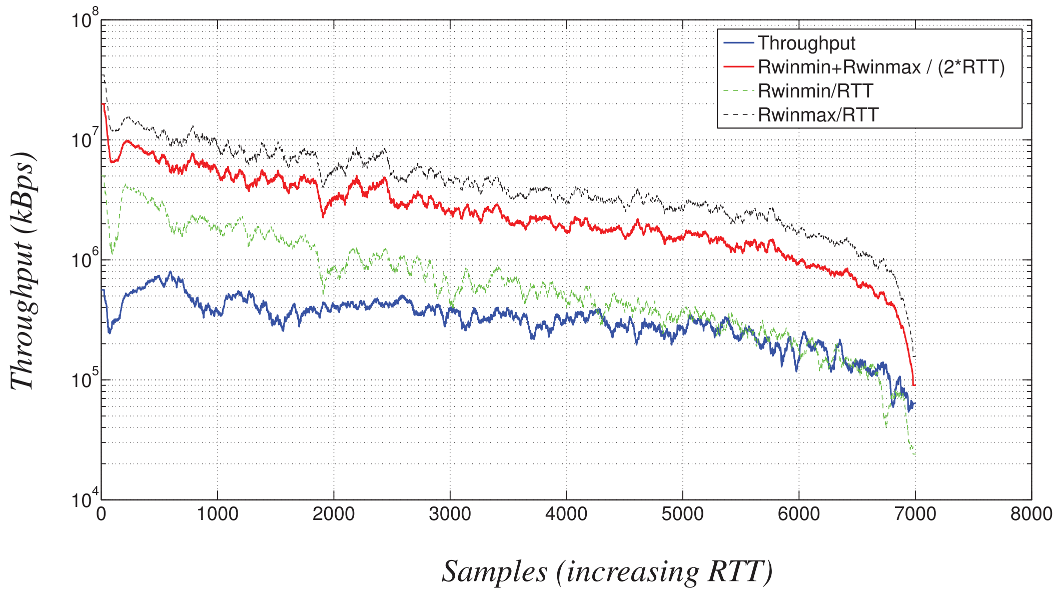

Fig. 2. Comparison of the throughput with the quantity of data a client can handle (rwin*).

window is impacted by the loss. However in our data set, very few losses actually happen and no dependence is found between the loss (*retrscore*) and the DNS service (*dns*).

It is to be expected that the internal RTT (*isprttavg*) is a parent of the throughput. Also, the absence of a dependence between the time of the day (*tod*) and the internal RTT can be explained by the fact that all the observed users are using the same "internal" path (the path from the users to the probe).

We see that the maximum receiver window advertised by the client (*rwinmax*) has the time of the day as one of its parents (*tod*). This could be due to the TCP buffer auto tuning mechanism [5] that adjusts the receiver window according to the quantity and frequency of data received by the client, which is influenced by the time of the day.

There is no edge between the DNS (*dns*) and the destination IP address (*dstip*) and the object size (*nbbbytes*) is not connected at all. This may be explained by the fact that the number of users of the LDNS service (80%) is much higher than the number of users of the GDNS service (20%). The same percentages are observed for the number of servers accessed by the users of the LDNS service (80%) and the number of servers accessed by the users of the GDNS service. The difference between these percentages weakens the dependence between *dns* and *dstip*. A solution to detect weaker dependencies is to increase the acceptance rate in the independence tests. However, increasing the acceptance rate implies a higher risk of failing to reject weak independencies and should be used with caution. The independence of the object size from other parameters influencing the throughput is not necessarily surprising as we consider long connections.

The two loss parameters (*retrscore* and *rto*) and the two RTT parameters (*inetrttstd* and *isprttavg*) are four of the six direct parents of the throughput, which is in line with our domain knowledge of TCP. The additional parents are the congestion window parameters of the server (*cwinmin* and *cwinmax*).

The fact that none of the receiver window metrics (*rwin**) is a direct parent of the throughput (*tput*) is not surprising. By comparing the throughput of a given connection with the minimum and the maximum quantity of information that the client can handle (see Fig. 2), it appears that the receiver window advertised by the client is never limiting the throughput.

4.2. Asking what-if questions

We have seen that the Bayesian network reveals a rich set of causal relationships that indicate how the different parameters impact the throughput. We will now use this model to answer *what-if questions* using only the already collected data, i.e. without the need to collect more data or perform additional experiments.

This reasoning used to answer what-if questions is referred to as *counterfactual thinking*. By asking "*What would be the performance of a user of the*LDNS*service if one of her parameter was to behave as it does when the*GDNS*is used, knowing that the use of the*LDNS*and the*GDNS*are exclusive ?*", we can estimate the impact of the choice of a DNS service on user performance. Such an approach allows to estimate the impact of choosing one DNS service instead of another and, even more interesting, allows us to estimate the impact of this choice on a given parameter that, in turn, impacts the user performance. In our work, we focus on the impact of the choice of a DNS service on the user throughput via the impact of the DNS service on the CDN server location (c.f. Section 4.2.1), and via the impact of the DNS service on the CDN server configuration (c.f. Section 4.2.2).

Since we deal with probabilities, we will compare the expected values of the throughput[3] instead of its average values[4] as we did in the previous section.

4.2.1. Distance and delay

To investigate the impact of the RTT on download performance we investigate the question: "*What would have been the performance of a user served by the local DNS if it would have been redirected to a server whose* inetrtt *corresponds to the one the Google DNS service would have redirected him to ?*".

To answer this question is equivalent to predicting the effect of an intervention where the external delay (RTT) experienced by clients served by the LDNS is modeled by the distribution of the delay experienced by clients served by the GDNS; the distribution of the rest of the parameters being kept identical for the LDNS service users.

[3] $\mathbb{E}[TPUT] = \int_{D_{TPUT}} f_{TPUT}(tput) \cdot tput \cdot dtput$, with D_{TPUT} the throughput domain.

[4] $\mu_{TPUT} = \frac{1}{N}\sum_{i=1}^{N} throughput_i$, with N the total number of observations.

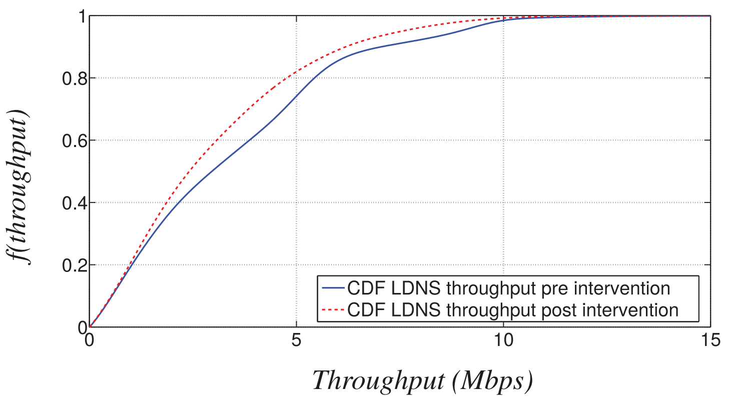

Fig. 3. Evolution of the throughput distribution before and after intervening on the external delay experienced by Local DNS (LDNS) clients.

More formally, if *RTT* denotes the *inetrttavg* parameter, *LD* the local DNS and *GD* the Google DNS, we need to estimate the following distribution:

$$f\big(TPUT = tput|DNS = LD, do(RTT \sim f_{RTT|do(DNS)}(\cdot, GD))\big) \tag{7}$$

The causal graph in Fig. 1 (cf the explanation of d-separation in Section 2.3) tells us that $(RTT \perp\!\!\!\perp DNS)_{G_{\underline{DNS}}}$, which implies (Rule 2 from Theorem 1):

$$f_{RTT|do(DNS)}(rtt, GD) = f_{RTT|DNS}(rtt, GD). \tag{8}$$

To predict how an intervention on *X* affects *Y*, where the intervention on *X* is enforced with the conditional probability distribution $f^*(X|Z)$ we use Eq. (4). The causal graph in Fig. 1 (cf the explanation of d-separation in Section 2.3) tells us that $(RTT \perp\!\!\!\perp TPUT|DNS, TOD)_{G_{\underline{RTT}}}$. It follows, from Rule 2 of Theorem 1 that

$$f_{TPUT|do(RTT),TOD,DNS}(tput, rtt, tod, dns) = f_{TPUT|RTT,TOD,DNS}(tput, rtt, tod, dns). \tag{9}$$

As a consequence, we can rewrite Eq. (7) as:

$$f(tput|LD)_{f(rtt|do(GD))} = \int_{D_{RTT}}\int_{D_{TOD}} f(tput|do(rtt), LD, tod) f(tod) f(rtt|do(GD)) P(GD) = \int_{D_{RTT}}\int_{D_{TOD}} f(tput|rtt, LD, tod) f(tod) f(rtt|GD) P(GD) \tag{10}$$

using Eqs. (8) and (9).

The result of the intervention is presented in Fig. 3. The CDF of the throughput for the LDNS before intervention is plotted as blue solid line and the CDF of the throughput for the LDNS service users after an intervention setting their external delays distribution to the delay distribution seen by the GDNS users is plotted as red dotted line. The throughput after invention is degraded due to the higher RTTs experienced by the clients': The expected throughput for clients using the local DNS service prior to intervention is 3.5 Mbps and 3.0 Mbps after intervention (14% decrease). This result quantifies the gain in performance that the redirection to closer CDN servers, provided by the use of the local DNS service, represents.

This result also illustrates the use of *counterfactual thinking*. We can deduce the gain in performance for a user who chose the LDNS service by estimating the change in performance if the GDNS would have been chosen instead.

The results obtained cannot be validated in practice as this would require the modification of the behavior of the local DNS servers. In fact, this difficulty nicely illustrates the benefit of the causal approach: it offers the possibility to predict the effect of interventions that are impossible to perform experimentally. Our approach allows us to estimate what would have been the effect on a user performance if she would have chosen the GDNS service, knowing that in reality the LDNS was used.

Fig. 4 shows the distribution of the external RTT for GDNS users and LDNS users. Both conditional probability distributions present a long tail and very few values are actually observed for a RTT > 200 ms. It is important to mention that RTT values are observed for the LDNS users for the range [0.5ms,200ms] and for GDNS users for the range [10ms,200ms]. This condition is necessary to perform the prediction preformed in this section, which is a limitation of the method used: The prediction formulated in Eq. (7) is only possible since the range of the external RTT values observed for GDNS users represents a subset of the range of values observed for the LDNS users.

If one wants to study the opposite intervention, where the users of the GDNS service would be given access to servers placed at the locations of the servers the LDNS service users are redirected to, the prediction would be more complex. We do not have samples to estimate $f(tput|rtt, GD, tod)$ for some of the smallest RTT values (RTT < 10 ms) for which we have $f(RTT|LDNS) > 0$. However, this limitation should not surprise us, since it is common to many machine learning problems where the amount of available information determines the predictions we can make. The reason why we cannot predict the opposite intervention is due to the use of kernels to estimate distributions, which requires the presence of samples in a given region to estimate the value of the distribution in this region. One possible way to overcome this problem would be to develop a parametric model that allows to extrapolate the different PDFs beyond the value range where the variables of our system are observed.

It is important to note that our model considers the impact of the change in the delay distribution but also the impact of the servers themselves, captured by the minimum congestion window and parameters such as the loss (*retrscore*) that are different between the two DNS services. In fact, the influence of these parameters may explain that the throughput experienced, in the original dataset, by the users of the GDNS service is only 7% smaller than for the users of the local DNS service. To evaluate the impact of the servers on download performance we focus on the impact of the minimum congestion window since *cwinmin* is a direct parent of the throughput (*tput*) and is influenced by the DNS service choice (*dns*). Also, other parameters such as the loss parameters (*retrscore* and *rto*), the delay parameters (*isprttavg* and *isprttstd*) or the maximum congestion window (*cwinmax*) are not influenced by the choice of the DNS service (*dns*) (c.f. Fig. 1).

4.2.2. Minimum congestion window

The minimum congestion window (*cwinmin*) is a direct parent of the throughput (*tput*), see Fig. 1. Its average value is higher for the clients using the GDNS service than for the clients using the

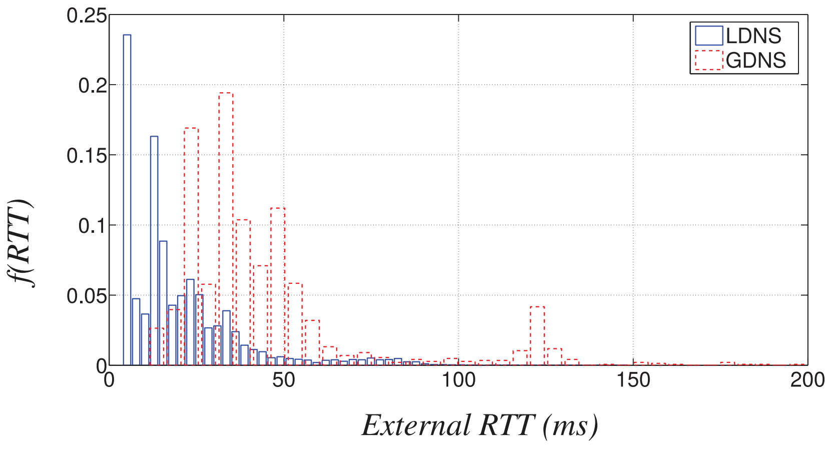

Fig. 4. Histogram of the external RTT for the local DNS (LDNS) and Google DNS (GDNS).

LDNS service (1.2kB and 0.9kB respectively). The difference in the expected value of the throughput of LDNS users (3.5 Mbps) and GDNS users (3.3 Mbps) is 6%, smaller than the gain for the LDNS users being redirected to closer server, that is estimated to be 14%. Our hypothesis is that the minimum congestion window represents a difference in the configuration of the servers accessed by the LDNS users and the configuration of the servers accessed by the GDNS users. To evaluate this hypothesis we estimate the causal effect of the minimum congestion window on the throughput, mediated by the choice of the DNS service. This is equivalent to asking the question: "*What would be the throughput for the clients using the local DNS if the servers they are redirected to would present the same minimum congestion window as the ones Google DNS users are redirected to ?*".

We observe from the causal graph of Fig. 1 (cf the explanation of d-separation in Section 2.3):

- $(CWINMIN \perp\!\!\!\perp DNS)_{G_{\underline{DNS}}}$
- $(CWINMIN \perp\!\!\!\perp TPUT|DNS, INETRTTSTD)_{G_{\underline{CWINMIN}}}$

For space reasons, and because the approach is the same as in Section 4.2.1 for the external delay (*inetrttavg*), we only present the final equation.

Let denote *cmin* the minimum congestion window (called *cwinmin* in our model) and σ_{rtt} the standard deviation of the external rtt (called *inetrttstd* in our model). As before *LD* refers to the local DNS and *GD* to the Google DNS. We obtain the following equation:

$$\begin{aligned} f(tput \mid LD)_{f(cmin|do(GD))} &= \int_{D_{CMIN}} \int_{D_{\sigma_{RTT}}} f(tput|do(cmin), LD, ts) f(\sigma_{rtt}) f(cmin|do(GD)) \\ &\quad \times P(GD) \\ &= \int_{D_{CMIN}} \int_{D_{\sigma_{RTT}}} f(tput|cmin, LD, \sigma_{rtt}) f(\sigma_{rtt}) f(cmin|GD) P(GD) \end{aligned} \tag{11}$$

Eq. (11) allows the prediction of the distribution of the throughput for the LDNS users after an intervention when we use for the minimum congestion window the distribution seen by GDNS users. The CDFs of the pre-intervention throughput (solid line) and post-intervention throughput (dotted line) are presented in Fig. 5. We can see the gain in throughput due to the intervention on the minimum congestion windows of the LDNS servers. The expected throughput for LDNS service users after the intervention is 4.6 Mbps (compared to 3.5 Mbps prior to intervention), which represents an increase of more than 30%. This increase is due to the fact that the servers GDNS service users are redirected to use higher values for their minimum congestion window.

The study of the opposite intervention, where GDNS service users are redirected to servers with a minimum congestion window following the distribution of the minimum congestion window seen by the LDNS service users, in the original dataset, is not possible. The reason is the same as the one mentioned in Section 4.2.1. If we compare the distribution of the minimum congestion windows for LDNS service users and GDNS service users, Fig. 6, we can notice the absence of *cmin* values for GDNS users to estimate *f*(*tput*|*cmin, GD,* σ_{rtt}) for values of *cmin* where *f*(*cmin*|*LD*) > 0.

If we summarize the findings of the last two sections, we can say that by using a causal model and its graphical representation we were able to quantify that it is not only the proximity of the server that has an important impact on the throughput but also the configuration of the server hosting the content a client wants to access.

In a causal model such as the one presented in Fig. 1, a given node *X* also represents the influence that external factors impacting *only this parameter* have on the rest of the system. This means that the difference in behavior of Akamai servers that the Google DNS redirects the clients to compared to the behavior of the servers the LDNS redirects the clients to may not be solely the effect of the minimum congestion window but may also be the effect of other un-observed parameters of TCP such as the *additive increase value* for each acknowledged packet. Unfortunately, we have no means to validate this hypothesis.

5. Related work

The two works closest to ours are WISE [18] and Nano [17].

WISE uses, as does our work, the PC algorithm [14] to infer a graphical causal model from which interventions are then predicted. However, WISE requires a lot of domain knowledge in its feature selection and in the definition of external causes that guide the inference of the causal model. Also, WISE uses the *Z*-Fisher independence, which assumes linear dependencies. We have tested the *Z*-Fisher independence criterion in our work and obtained very poor results as the test fails to detect parameter independencies resulting in incorrect models [6]. In addition, WISE considers much simpler scenarios of intervention and requires a much larger data set. Our approach takes full advantage of the causal theory developed by Pearl [11,15] to predict interventions and counterfactuals. Counterfactuals are very useful to understand the role of the different parameters of a system and, to our knowledge, scenarios such as the ones presented in Section 4.2 have not been treated so far.

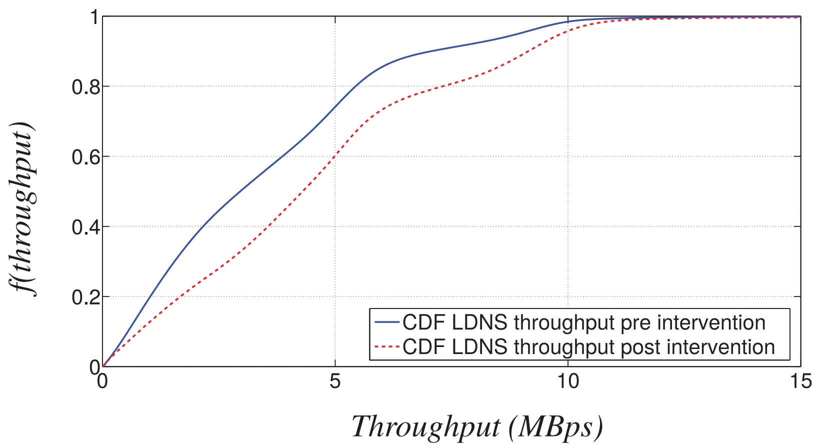

Fig. 5. Evolution of the throughput distribution before and after intervening on the minimum congestion window of servers of the users of the local DNS.

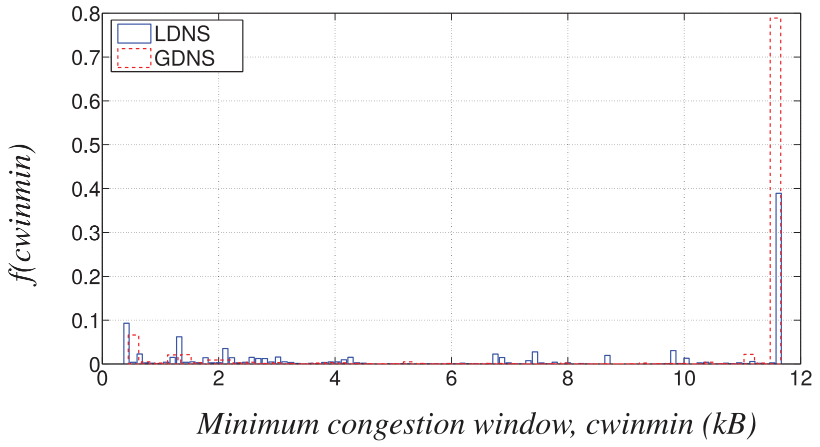

Fig. 6. Histogram of the minimum congestion window for the local DNS (LDNS) and Google DNS (GDNS).

Nano tries to detect network neutrality violation by assessing the direct causal effect between the quality of experience of a user from a given ISP and the type of content being accessed. A performance baseline is defined based on observations made for different ISPs sharing similar configurations and then compared to the one observed for a particular scenario. Again, this approach uses domain knowledge to define the possible confounders and to condition on these variables to remove spurious associations. Since Nano has not derived a formal causal model, its approach has serious limitations since one of the confounders could be a collider in the corresponding causal graphical model. Also, conditioning on a common effect induces a dependence between two independent causes whose influence tries to be canceled, questioning the obtained results.

Several papers study how the choice of the DNS service impacts client performance [1,7,10]. These works rely on active measurements and differ greatly in their approach and objectives from our work.

In our previous work [6] we presented solutions to the problem of causal model inference and to the prediction of atomic interventions for cases where the assumptions of normality and linearity do not hold. We also validated our approach and showed for simple systems and scenarios that it was possible to use a causal approach to study communication network performance.

The work presented in this paper goes much further. First, we study a more complex system with more parameters and diverse categories of data (including categorical data). We use the causal model obtained to explain non intuitive observations (namely a similar throughput for connections experiencing a different RTT). Second, the major contribution of the work presented in this paper is due to the use of *counterfactuals* and *counterfactual thinking*, Section 4.2. The use of *counterfactuals* gives us access to a deeper understanding of the causal mechanisms ruling the performance of the system and it allows us to quantify the impact of each of these mechanisms on the performance of this system.

6. Concluding remarks

The main contribution of our paper resides in the methodology based on the inference and in the usage of a causal model that allows us to estimate the causal effect of the DNS service on user performance. Using a causal approach and inferring the causal model, which is then represented as a Bayesian graph, we are able to study the causal effect of a DNS service on the TCP throughput. We compare the performance of clients using their ISP local DNS service to the performance of clients using the Google DNS service. The causal model allows to unveil dependencies that would be very difficult, if not impossible, to extract otherwise from the data. We showed that the choice of the DNS service has a strong impact on the location of the servers the clients are redirected to, which in turn impacts not only the distance from clients to servers but also the type of configuration of the servers. Distance and configuration are captured by the dependence between the DNS and the RTT and the dependence between the DNS and the server minimum congestion window.

A very interesting property of causal models is their "*stability under intervention*". The model inferred from data following a given distribution is still valid when we predict the effect of modifying this distribution. When comparing the performance of the users of the local DNS and the users of the Google DNS, we can observe that the performance difference cannot simply be explained by the redirection of Google DNS users to more distant Akamai servers. Based on the causal graph obtained, we can formulate the hypothesis that the configurations of the Akamai servers Google DNS users are redirected to allow them to experience a performance close to the one of the local DNS service users. This hypothesis is confirmed by our prediction where we give to Akamai servers serving the local DNS users a minimum congestion window equivalent to the one of the Akamai servers serving Google DNS users. We estimate the gain in throughput corresponding to this intervention to be 32%. By comparison, the gain in terms of throughput corresponding to the redirection of the local DNS users to closer servers is estimated to be only 14%.

We demonstrated the potential of adopting a causal approach using counterfactuals. Counterfactuals are one of the possible way to approach Causality and we use this technique to evaluate the effect of a parameter on the system performance by predicting the effect that changing its parent would have with the rest of the system parameters left unchanged. We manage to answer questions such as "*How would the system behave under the condition* C1 *if one of its parameter was to behave as it has done under the condition* C2*, knowing that* C1 *and* C2 *are exclusive ?*". The ability to make predictions for such scenarios illustrates the power of the inherent mechanisms underlying the development of Causality. Counterfactuals are relatively complex to study, explain and even more so to predict. However, thanks to the Bayesian network as a representation of the causal model of our system, using counterfactuals becomes easier.

Complex interventions, where many parameters are modified simultaneously, require important resources in terms of the amount of data and computational power. The results presented in this paper document a first successful attempt. Based on this work, we are confident that the underlying tools and methods can be improved to reduce the required resources and increase both, the accuracy of such predictions and the range and complexity of the interventions that one can consider.

We see the following directions for future work:

- By fitting a parametric model we could extend the prediction of counterfactuals for cases where the two conditional probabilities have only partial overlap.
- The weight of an edge, $X \rightarrow Y$, corresponds to what is known as the *direct effect* of one parameter, X on another, Y. However, in the absence of linearity, the estimation of the direct effect of X on Y is complex and requires predicting the effects of several interventions [12] for each direct effect, which requires a lot of computational resources.
- Regarding a selection criterion, the absence of any assumption regarding the distribution of the parameters and the nature of their dependencies prevents us from using a classical selection criterion such as maximum likelihood. Two possibilities could be used instead: *(i)* A *Bootstrap* approach, where, by resampling the original data set to create new data sets, we could infer one causal model for each data set and, by comparison, derive a confidence level for our model. This approach is simple to implement. However, the inference of the causal model presented in this paper took up to one week running on a cluster of 30 machines. Therefore, a *bootstrap* approach requires important resources in terms of computation time. On the other hand, when creating sub-data sets, we work with smaller data sets, which has an impact on the accuracy of the results. *(ii)* We could use the *independence test p-values* to obtain a confidence in the presence or absence of any edge in the graph we obtain to give us a confidence in the model. This approach becomes complex due to the number of tests to consider for a given pair of nodes and no general criterion has been designed as this stage of our work.
- We had to design several solutions to build a reliable framework for causal knowledge inference [6] that implied an increase in complexity and resource requirements. While we have used very small data sets to validate our approach and to show its benefits, there are many directions to explore to make Causal reasoning work more efficiently on large quantities of data thanks to the use of distributed computing. The parallelization of the independence testing for causal model inference [6] and the parallelization of the estimation of interventions (see Appendix B.2) fit very well a Big Data approach. Working with a bigger and partitioned data set on which parallel computing could be done, would improve the performance of the Causal knowledge inference framework we presented in this work.

Acknowledgment

We would like to thank Elias Bareinboim from UCLA for his advice and support in developing the methodology to predict counterfactuals. We also would like to thank the reviewers for their constructive remarks. The research leading to these results has received fundings from the European Union under the FP7 Grant Agreement n. 318627 (Integrated Project "mPlane").

Supplementary material

Supplementary material associated with this article can be found, in the online version, at 10.1016/j.comnet.2016.06.023

References

[1] B. Ager, W. Mühlbauer, G. Smaragdakis, S. Uhlig, Comparing DNS resolvers in the wild, in: IMC, ACM, 2010, pp. 15–21.
[2] A. Darwiche, Modeling and Reasoning with Bayesian Networks, first, Cambridge University Press, Cambridge, 2009.
[3] S. Demarta, A. McNeil, The T-copula and related copulas, Int. Stat. Rev. 73 (1) (2005) 111–129.
[4] A. Finamore, et al., Experiences of internet traffic monitoring with tstat, IEEE Netw. Mag. 25 (3) (2011) 8–14.
[5] P. Ford, A. Shelest, N. Srinivas, Method for automatic tuning of TCP receive window, 2002, US Patent App. 09/736,988.
[6] H. Hours, E. Biersack, P. Loiseau, A causal approach to the study of TCP performance, ACM Trans. Intell. Syst. Technol. 7 (2) (2015) 25:1–25:25.
[7] C. Huang, D.A. Maltz, J. Li, A.G. Greenberg, Public DNS system and global traffic management., in: INFOCOM, IEEE, 2011, pp. 2615–2623.
[8] P. Jaworski, F. Durante, W. Härdle, T. Rychlik, Copula theory and its applications, Lecture Notes in Statistics, Springer, Berlin, Heidelberg, 2010.
[9] V. Mayer-Schönberger, Big Data: A Revolution That Will Transform How We Live, Work and Think., John Murray Publishers, UK, 2013.
[10] J.S. Otto, M.A. Sánchez, J.P. Rula, F.E. Bustamante, Content Delivery and the Natural Evolution of DNS: Remote DNS trends, performance issues and alternative solutions, in: IMC, ACM, 2012, pp. 523–536.
[11] J. Pearl, Causality: Models, Reasoning and Inference, Cambridge University Press, New York, NY, USA, 2009.
[12] J. Pearl, Direct and indirect effects, CoRR abs/1301.2300 (2013).
[13] M. Pitt, D. Chan, R. Kohn, Efficient Bayesian inference for gaussian copula regression models, Biometrika 93 (3) (2006) 537–554.
[14] P. Spirtes, C. Glymour, An Algorithm for Fast Recovery of Sparse Causal Graphs, Soc. Sci. Comput. Rev. 9 (1991) 62–72.
[15] P. Spirtes, C. Glymour, R. Scheines, Causation, Prediction, and Search, second, The MIT Press, Cambridge MA 02142-1209, 2001.
[16] F. Streibelt, J. Böttger, N. Chatzis, G. Smaragdakis, A. Feldmann, Exploring edns-client-subnet adopters in your free time, in: IMC, ACM, 2013, pp. 305–312.
[17] M. Tariq, M. Motiwala, N. Feamster, M. Ammar, Detecting network neutrality violations with causal inference., in: CoNEXT, ACM, 2009, pp. 289–300.
[18] M. Tariq, et al., Answering what-if deployment and configuration questions with WISE, in: SIGCOMM, ACM, 2008, pp. 99–110.
[19] K. Zhang, J. Peters, D. Janzing, S. B., Kernel-based conditional independence test and application in causal discovery, CoRR abs/1202.3775 (2012).

Hadrien Hours was born in 1986 in Aix-en-Provence, France. From September 2004 to July 2006, he did Engineering Science and Mathematics and Physics at the Ecole Preparatoire of Lycée Thiers, Marseille, France. He joined Télécom Bretagne in September 2006 and did one Erasmus semester in the Facultad de Informatica de Madrid (Spain). From 2008 to 2009, he worked for Bouygues Télécom (Aix en Provence, France) as an intern with Performances and Optimizations missions, along with Dimensioning and Architecture. In March 2010 he attended the Networking Track in EURECOM. He got his engineering diploma from Télécom Bretagne in 2011 for Engineer in Telecommunication in Networking Track. He made his master thesis in Technicolor Paris Research Lab on House Automation: Energy Monitoring under the supervision of the Professor Ernst Biersack from EURECOM and Laurent Massoulié from Technicolor. He started his PhD under the supervision of Professor Ernst Biersack and Professor Patrick Loiseau on the subject of A Causal Approach to the Study of Communication network performance in November 2011. He obtained his Ph.D. in September 2015. Since November 2016 he is a post-doc in the Dante team at ENS Lyon, France.