IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Volume-5, Issue-6 E-ISSN: 2347-2693

Predicting a Low Latency in Different States of Emergency Events using Web Resources

V.Geetha[1*], S.Rajalakshmi[2]

[1*] Department of Computer Science, S.T.E.T Women's College, Sundarakkottai, India
[2] Department of Computer Science, S.T.E.T Women's College, Sundarakkottai, India

[*]*Corresponding Author: kkmannaig@gmail.com*

Available online at: www.ijcseonline.org

Received: 22/May/2017, Revised: 06/Jun/2017, Accepted: 27/Jun/2017, Published: 30/Jun/2017

***Abstract*—** Low latency is basic for intuitive networked applications. Be that as it may, while we know how proportional frameworks to in-wrinkle limit, decreasing latency particularly the tail of the latency dispersion can be a great deal more troublesome. We contend that the utilization of repetition with regards to the wide-territory Internet is a viable approach to change over a little measure of additional limit into diminished latency. By starting excess operations crosswise over differing assets and utilizing the primary outcome which finishes, repetition enhances a framework's latency even under extraordinary conditions. We show that repetition can fundamentally diminish latency for little yet basic undertakings, and contend that it is a successful broadly useful methodology even on gadgets like phones where data transfer capacity is generally obliged.

***Keywords*—**Performance, Reliability, Latency

I. INTRODUCTION

Low latency is critical for people. Indeed, even somewhat higher page stack times can altogether decrease visits from clients and income, as exhibited by a few destinations. For instance, infusing only 400 milliseconds of counterfeit deferral into Google indexed lists made the postponed clients perform 0:74% less hunts following 4 a month and a half. A 500 millisecond delay in the Bing web search tool diminished income per client by 1:2%, or 4:3% with a 2-second postponement. Human-PC communication ponders also demonstrate that individuals respond to little contrasts in the postponement of operations (see and references in that).

Accomplishing steady low latency is testing: Current applications are exceedingly circulated, and prone to get all the more so as distributed computing isolates clients from their information and calculation. In addition, application-level operations regularly require tens or several assignments to finish because of many items containing a solitary site page, or total of many back-end inquiries to deliver a front-final product. This implies singular errands may have latency spending plans on the request of a couple of milliseconds or many milliseconds, and the tail of the latency dissemination is basic.

Along these lines, latency is a challenge for networked frameworks: How would we make the opposite side of the world feel like it is ideal here, even under outstanding conditions?

One effective method to diminish latency is excess: Initiate an operation numerous circumstances, utilizing as differing assets as could be allowed, and utilize the main outcome which finishes. For instance, a host may inquiry numerous DNS servers in parallel to determine a name. The general latency is the base of the deferrals over each example, in this way conceivably decreasing both the mean and the tail of the latency dispersion. The energy of this procedure is that it diminishes latency definitely under the most difficult conditions: when postponements or disappointments are unusual.

Excess has been utilized in a few past net-worked frameworks: outstandingly, as an approach to manage disappointments in DTNs, and in a multi-homed web intermediary over-lay. In any case, past these particular research ventures, excess is commonly shunned over the Internet. We contend this is a missed open door.

The commitment of this paper is to contend for repetition as a general strategy for the wide-territory Internet. The blend of intelligent applications, high latency, and inconstancy of latency make excess appropriate to this condition. Indeed, even in an all-around provisioned network where singular operations typically work, some measure of vulnerability is unavoidable and the interest for steady low latency exceeds the need to spare band-width which is today nearly shoddy.

To bolster this contention, in §3 we inspect the cost of repetition. Since latency-bound assignments are probably

going to be little, the general overhead is little when workloads are overwhelming followed; we demonstrate that stream estimate conveyance estimations affirm this. We next set a benchmark for when replication is helpful from the point of view of effect on network transmission capacity, demonstrating that replication might be savvy even in to a great degree traditionalist (phone) situations the length of we can spare more than 10 milliseconds for each kilobyte of included movement. In §4, we demonstrate the advantages of replication can be requests of size bigger than this limit in various basic application situations. For instance, questioning numerous DNS servers can diminish the division of reactions later than 500 ms by 6:5x, while the portion later than 1:5 sec is lessened by 50x. We likewise examine applications to multipath steering, TCP association foundation, and nature of administration. In outline, as framework planners we regularly assemble versatile frameworks by maintaining a strategic distance from superfluous work. In any case, we contend that in the wide-zone Internet, additional work is a valuable and exquisite approach to accomplish strength to unforeseen conditions and reliably low latency.

II. RELATED WORK

Replication is utilized inescapably to enhance unwavering quality, and in numerous frameworks to decrease latency. Dispersed employment execution structures, for instance, have utilized assignment replication to enhance reaction time, both preemptively and to moderate the effect of stragglers.

Inside networking, replication has been investigated to decrease latency in a few particular settings, including imitating DHT questions to various servers and reproducing transmissions (by means of deletion coding) to diminish conveyance time and misfortune likelihood in deferral tolerant net-works. Replication has additionally been recommended as a method for giving QoS prioritization and enhancing latency and misfortune execution in networks equipped for excess end.

Maybe nearest to our work, Andersen et al.'s MONET framework intermediaries web movement through an over-lay network shaped out of multi-homed intermediary servers. While the essential concentrate of is on adjusting rapidly to changes in way execution, they imitate two particular subsets of their activity: association foundation solicitations to various servers are sent in parallel (while the first one to react is utilized), and DNS inquiries are repeated to the neighborhood DNS server on each of the multi-homed intermediary server's interfaces. We demonstrate that replication can be helpful in both these settings even without way differing qualities: a noteworthy execution advantage can be gotten by sending various duplicates of TCP SYNs to a similar server on a similar way, and by reproducing DNS questions to numerous open servers over a similar get to interface. In particular, not at all like the greater part of the above work, our point is that replication is a general strategy that can be connected in an assortment of normal wide-territory Internet applications. We contend this point by concentrate the over-head related with duplicating little streams, the essential advantage for end-clients, and a few utilize cases.

III. MORE IS LESS

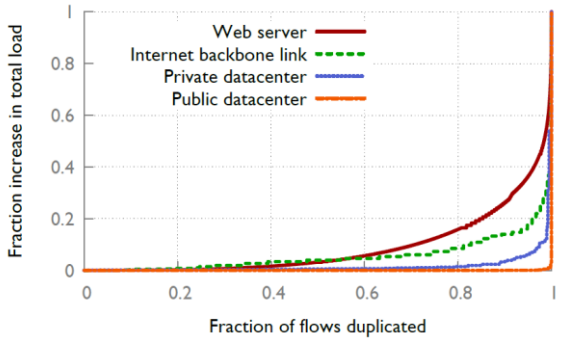
In this segment we talk about the adequacy of the general approach of diminishing latency by means of redundancy.1 Later (§4) we examine particular applications.

A. Avoiding uncertainty

The energy of repetition is to lessen vulnerability without anticipating the reason for that instability. All in all, accept we have numerous choices accessible to get an administration |, for example, various servers, different ways, or even different minutes in time at which to demand benefit. On the off chance that we could foresee which of these choices will perform effectively with most reduced latency, then we would essentially execute that choice. Be that as it may, idealize forecast of excellent conditions, particularly in substantial networked frameworks, is some place be-tween troublesome and outlandish. Quickly high usage on a server, delay because of virtualization, a moderate circle, nonattendance of a protest in a web or DNS reserve, network clog, an assault like a false DNS re-handle, and different variations from the norm all effect the mean and particularly the tail of the latency dispersion. Operations that fall flat can be retried, obviously, yet this includes more noteworthy deferral particularly in the wide-range. Also, outlining frameworks which don't experience such variations from the norm is troublesome and costly.

Assuming, in any case, we execute different choices in parallel and utilize the quickest right outcome, then we can stay away from these uncommon conditions unless they strike all choices at the same time. There are two key challenges. To begin with, we might not have various really autonomous choices, so that excellent conditions are corresponded crosswise over choices. We later give cases of wide-region applications that have adequate assorted qualities crosswise over choices (§4). Second, repetition includes more work and cost.

In §3.2, we contend that the general increment in usage might be little, since latency-delicate errands are regularly a little division of the aggregate network stack. In §3.3, we consider the point of view of an individual client, contending that sparing even a couple of milliseconds for each additional KB of movement is beneficial.



B. Overhead can be low

By and large, operations that require steady low latency are probably going to be little in the work they consume per operation. Consider, for instance, a stream over the wide-zone Internet: if the stream is altogether bigger than the network's postponement transfer speed item, the aggregate time it needs to finish will be ruled by its throughput. While downloading a gigabyte motion picture which may have minutes of cushion, a couple of hundred milliseconds postponement or loss of a couple of parcels will probably go unnoticed. Alizadeh et al. have additionally noticed that in various server farm applications, latency-basic employments are little. In such frameworks, the streams that are the well on the way to profit by replication are additionally those that are the minimum costly to repeat.

Besides, if latency-basic operations are little, then they will include a little portion of the network's aggregate work if the dissemination of sizes is substantial followed, which is an inescapable property in the Internet. Figure 1 demonstrates the rate increment in all out network stack that would come about if the littlest x% of all streams were copied in four unique settings: streams in an open and a private server farm, web benefit demands crossing a college wide-zone uplink, and an Internet spine connect. In all cases it is conceivable to du-plicate at any rate the 33% littlest ows while expanding the aggregate load by just 2%.

A few admonitions may constrain or grow the situations in which replication is helpful. (1) as opposed to the general control over, some latency-delicate assignments are substantial, for example, continuous video streams. Replication may in any case be valuable if these streams have low rate (§4.3) or are basic. (2) If overhead is high, a duplicated demand could be set apart as lower need, so it can be dropped in the event that it meddles with other work. Then again, one could reproduce just the most essential operations; for instance, a cloud provider could charge tenants a higher rate for reliable low latency. (3) Even if overhead is high, replication might be adequate when the framework is under-used, as the wide-range Internet normally seems to be. Along these lines, replication can recover an incentive from generally unused assets.

C. End-user cost-benefit analysis

We can gauge when replication is helpful for an end-client through the estimation of time and the cost of expanded usage. The accompanying computations break down just the cost as far as network transfer speed to clients and suppliers, and the advantage to a client.

For replication to sound good to an end-client, we require $\ell\upsilon \geq b$ where l is the normal latency reserve funds in milliseconds for every KB of included activity, υ is the dollar estimation of one millisecond of latency decrease, and b is the cost of included movement per KB. The estimation of time v is the most hard to ascertain. It might be very application-particular, and may rely on upon mean or tail latency in ways best quantized by a human client investigation of nature of experience. In any case, to get a first estimate, we accept the estimation of time is essentially the US normal profit of $23.50 every hour in June 2012, which infers $\upsilon \approx 6{:}53.10^{-6}$ $=ms.

To gauge the cost per kilobyte b, we swing to promoted rates for end-client and cloud benefit data transfer capacity. For cell arranges, we accept a client who has paid for essential network as of now, and figure the cost of band-width from overage charges. For instance, AT&T's littlest remote information arrange brings about $20 per 300 MB. We can now explain $\ell \geq b/\upsilon$ to acquire the equal the initial investment point, as far as the essential latency funds per kilo-byte of extra track.

Connectivity plan	Cost b ($/KB)	Break-even benefit ℓ (ms/KB)
AT&T, small cell	$651.04 \cdot 10^{-7}$	9.970
AT&T, large cell	$95.37 \cdot 10^{-7}$	1.460
T-Mobile Austria	$9.67 \cdot 10^{-7}$	0.148
AT&T DSL	$1.91 \cdot 10^{-7}$	0.029
EC2 and Azure clouds	$1.20 \cdot 10^{-7}$	0.018
MaxCDN	$0.40 \cdot 10^{-7}$	0.006

Take note of that in an association between an end-client and an administration in the cloud, the last's transmission capacity expenses are relatively little. The table shows replication might be practical even in the most moderate availability arrange the length of we can spare more than 10 milliseconds (in the mean or tail, contingent upon the objective) for every kilobyte of included movement. We will see that the advantage can be no less than a request of greatness more prominent than this amount.

IV. EXAMPLES

A. Connection establishment

We begin with a straightforward persuading illustration, exhibiting why replication ought to be valuable notwithstanding when the accessible decisions are restricted: we consider what happens when numerous duplicates of a bundle are sent on a similar way. Clearly this ought to help if all bundle misfortunes on the way are autonomous. For this situation, sending two consecutive duplicates of a parcel would lessen the likelihood of it being lost from p to p^2. Practically speaking, obviously, consecutive bundle transmissions are probably going to watch an associated misfortune design. Be that as it may, Chan et al. measured a critical lessening in misfortune likelihood regardless of this relationship. Sending consecutive bundle combines between PlanetLab has, they found that the normal likelihood of individual parcel misfortune was ≈0:0048, and the likelihood of both bundles in a consecutive match being dropped was just ≈0:0007 {considerably bigger than the $\sim 10^{-6}$ that would be normal if the misfortunes were autonomous, yet at the same time 7x lower than the individual bundle misfortune rate.3

As a solid illustration, we evaluate the change that this misfortune rate lessening would impact on the time required to finish a TCP handshake. The three bundles in the handshake are, by the measurements talked about in §3, perfect contender for replication: they make up an inconsequential portion of the aggregate movement in the network, and there is a high punishment related with their being lost (Linux and Windows utilize a 3 second introductory timeout for SYN parcels; OS X utilizes 1 second). We utilize the misfortune likelihood measurements examined above to gauge the normal latency investment funds on every handshake.

We consider a glorified network demonstrate. At whatever point a bundle is sent on the network, we expect it is delivered effectively after (*RTT/2*) seconds with likelihood 1-*p*, and lost with likelihood *p*. Bundle de-uniforms are thought to be free of each other. *p* is 0:0048 when sending one duplicate of every parcel, and 0:0007 when sending two duplicates of every bundle. We additionally accept TCP conduct as in the Linux portion: an underlying timeout of 3 seconds for SYN and SYN-ACK parcels and of *3xRTT* for ACK bundles, and exponential back-off on bundle misfortune.

With this model, it can be demonstrated that copying every one of the three bundles in the handshake would diminish its normal finishing time by around (3 + 3 + 3xRTT)x(4:8-0:7)ms, which is no less than 25ms. On the off chance that we expect every bundle is 50 bytes in length, this suggests a funds of around 170 ms/KB, which is more than a request of greatness bigger than the equal the initial investment latency reserve funds distinguished for the most costly arrangement in x3.3. The benefit increments with *RTT*, and is much higher in the tail. Duplication would enhance the 99:9th percentile hand-shake fulfillment time, for example, by no less than 880ms, for a latency funds of around 6000 ms/KB.

B. DNS

An ideal contestant for repetition is a provision that contains small processes and which is simulated at numerous areas, in this manner giving differences crosswise over net-work ways and servers, so that duplicated operations are very free.

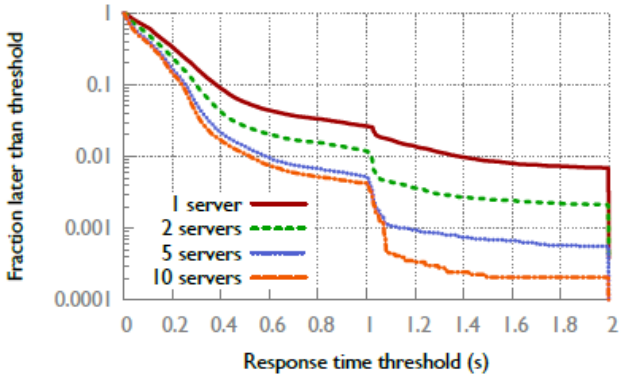


Figure 2: DNS response time distribution.

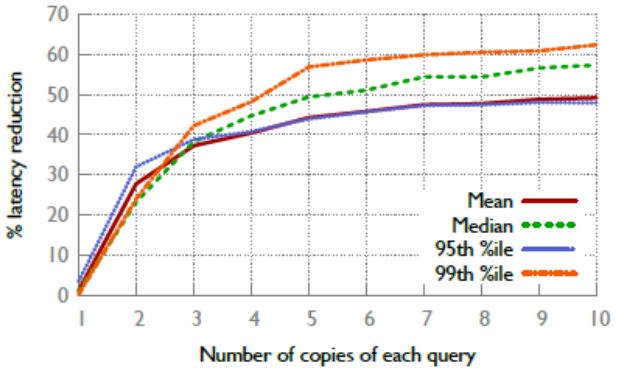


Figure 3: Reduction in DNS response time, averaged across 15 PlanetLab servers.

We started with a rundown of 10 DNS servers4 and Alexa.com's rundown of the main 1 million site names. At each of 15 PlanetLab hubs over the mainland US, we ran a two-arrange explore: (1) Rank every one of the 10 DNS servers as far as mean reaction time, by more than once questioning an irregular name at an arbitrary server. Take note of that this positioning is particular to each PlanetLab server. (2) Repeatedly pick an arbitrary name and play out an irregular one of 20 conceivable trials-either questioning one

of the ten individual DNS servers, or questioning somewhere in the range of 1 to 10 of the best servers in parallel (e.g. on the off chance that sending 3 duplicates of the question, we send them to the main 3 DNS servers in the positioned list). In each of the two phases, we performed one trial at regular intervals. We ran each phase for about seven days at each of the 15 hubs. Any question which took over 2 seconds was dealt with as lost, and considered 2 sec when ascertaining mean reaction time.

Figure 2 demonstrates the dissemination of inquiry reaction times over all the PlanetLab hubs. The change is considerable, particularly in the tail: Querying 10 DNS servers, the portion of inquiries later than 500 ms is lessened by 6:5x, and the division later than 1:5.sec is decreased by 50x. Averaging over all PlanetLab hubs, Figure 3 demonstrates the normal percent lessening accordingly times contrasted with the best settled DNS server recognized in stage 1. We acquire a significant decrease with only 2 DNS servers in all measurements, enhancing to 50-62% lessening with 10 servers. At long last, we contrasted execution with the best single server all things considered, i.e., the server with least mean reaction time for the questions to individual servers in Stage 2 of the investigation, since the best server may change after some time. Indeed, even contrasted and this stringent pattern, we found an outcome like Fig. 3, with a decrease of 44-57% in the measurements while questioning 10 DNS servers. Is this change beneficial? Sending 10 DNS questions rather than 1 costs < 4500 additional bytes for a latency funds of 0:1 sec in the mean or 0:9 sec in the 99th percentile. This is around 20_ superior to anything the most negative make back the initial investment advantage in x3.3, which means the data transfer capacity cost is irrelevant. We likewise take note of that questioning numerous servers increments storing, a side-advantage which would enthusiasm to measure.

We take note of that prefetching that is, preemptively starting DNS queries for all connections on the present site page makes a comparative tradeoff of expanding burden to decrease latency, and its utilization is boundless.

C. Multipath routing

We now consider replication of bundles in a vast scale multipath directing setting. This setting might be accessible in a future Internet condition, additionally imitates decently nearly (in topology and information rate) the Skype overlay network, where predictable low latency is useful. We led probes three diverse overlay topologies, each comprising of a source and a goal hub associated by means of 8 transitional hubs, yielding 8 unmistakable end-to-end ways. We utilized two topologies crossing the US and one with the source and a large portion of the middle hubs in Europe and the goal and the staying moderate hubs in the US. The vast majority of these hubs were on PlanetLab, yet in one topology we utilized only held ProtoGENI hubs for the source and goal in light of the fact that the PlanetLab hubs we utilized at first with were too vigorously stacked (see encourage dialog beneath). P2P applications, for example, Skype utilize comparative overlay topologies, transferring activity through middle of the road hubs to bypass NAT and firewall issues.

Throughout about 48 hours, we sent UDP information bundles at rates of r = 32kbit/s and r = 56kbit/s over the m∈{1,2,3}best ways by then. The blends of r and m were haphazardly picked each moment. We utilized a basic moving-normal (SMA) with a window-size of 5 to rank the ways.

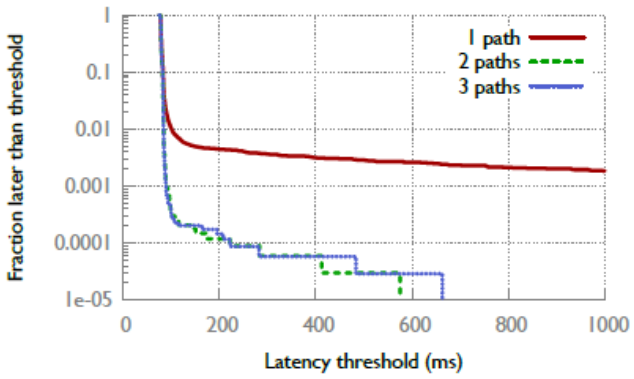


Figure 4: Multipath directing: Latency circulation in agent intra-US topology at r = 56kbit/s.

We watched a noteworthy execution change when utilizing replication, in spite of the variety in elements, for example, the topology, the information rate, and the state of the latency conveyance on every way. Notwithstanding, we take note of that in one earlier estimation (comes about not announced here) replication altogether corrupted framework execution. We trust this was because of bizarrely substantial load on the sending PlanetLab hub: when we changed it out for a delicately stacked ProtoGENI hub we again watched an indistinguishable level of execution from in alternate tests. The last outcomes are accounted for here, however this underscores the way that a bigger scale study is important to portray the conditions under which replication makes a difference.

Fig. 5 and 4 demonstrate that most or the majority of the change we watch originates from utilizing two ways rather than one. While the change in the mean latency is moderately little, replication essentially enhances the latency tail and decreases the parcel misfortune likelihood. Contrasted with a solitary way we see better latencies beginning at the 95th percentile, and with only one extra way the 99th percentile latency falls by around 60%.

	mean latency			99% latency			99.9% latency		
	1 path	2 paths	3 paths	1 path	2 paths	3 paths	1 path	2 paths	3 paths
Intra-U.S. 1	103.5ms	7.28%	6.30%	292.1ms	61.61%	59.62%	loss	1846.8ms	1997.0ms
Intra-U.S. 2	86.0ms	6.63%	6.41%	101.9ms	17.19%	17.19%	1888.5ms	95.04%	95.22%
Trans-Atlantic	142.1ms	4.02%	4.19%	233.7ms	37.26%	37.27%	594.9ms	69.52%	71.81%

Figure 5: Multipath routing: Measured RTTs in 56kbit/s experiment for 1 path and percent reduction when replicating over 2 or 3 paths (packets later than 4.5 seconds were considered lost).

The two information sending rates we tried, which are sensible for constant sound activity, yielded pretty much indistinguishable conduct, however we expect that testing a bigger scope of rates will uncover that replication stops to help when the information rate is adequately high. We leave an examination of this question to future work. We take note of that the examinations we have directed so far are too little scale to be illustrative {a a great deal more exhaustive assessment would be important to legitimately evaluate the execution benefits that replication can yield. Nonetheless, our outcomes propose that there are sensible situations in which replication can offer assistance.

D. Quality of service

Low latency is a vital part of nature of administration (QoS) on the Internet. Conventional ways to deal with giving QoS (e.g. IntServ) include moderately complex flagging and parcel planning instruments, do not have a demonstrated valuing model, and have not seen boundless organization. Regardless of the possibility that they were sent, these components manage just a single aim of high latency (clog) to the rejection of other capricious occasions (switch disappointment, carriage QoS usage, physical layer defilement, and so on.). Replication might be a successful intends to acquire a portion of the advantages of QoS. On the off chance that multipath steering is available| with various network suppliers, passages, or more propelled instruments in future Internet architectures| then latency-delicate activity can be repeated along these ways. Contrasted and conventional QoS, this plan (1) facilitates organization, by requiring no information plane QoS bolster and reusing existing evaluating models in light of data transfer capacity volume; (2) can be adaptably connected to a sub-set of activity so just a little extra load is presented where it is best; (3) manages any erratic occasions, for example, switch disappointments, the length of they are not related over the freeways.

V. CONCLUSION and Future Scope

Using repetition as a typical system in the Internet raises a few fascinating exploration headings. Computerizing excess. Our examination proposes that excess is beneficial even with generally little changes in postponement, and in this way, it might be advantageous to robotize. One straightforward way would be, for instance, to imitate the main k parcels of every TCP stream, for some little k. Past this, would we be able to robotize repetition to decrease latency in wide-zone application-level administrations?

Way determination: Rather than picking ways (or servers) in light of their mean execution, it might be advantageous to pick alternatives that are as free as could reasonably be expected. Picking an arrangement of k ways whose latency-tails are as \maximally autonomous" as conceivable is a fascinating frameworks and algorithmic issue.

Childishness: Assume clients pick the amount to reproduce while childishly limiting their latency. We have developed cases demonstrating that the Nash balance of such a procedure in a lining framework can be more terrible than without replication. In any case, can execution genuinely debase? In functional frameworks, are end-have costs (e.g., vitality or uplink requirements) adequate to avoid poor results?

Security: Past latency, there are more extensive compositional explanations behind assortment. Similarly as excess makes it harder for nature to bring about an issue, so it is harder for aggressors to bring about an issue. By recreating DNS questions, for instance, one may identify vindictive or erroneous reactions. Could we plan an Internet with inescapable excess, enhancing both latency and security?

REFERENCES

[1] J.Makkonen, *"Investigation on event evolution in TDT"* In 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language, PP.43-48, 2003.

[2] C.Yang, X.Shi, and C.Wei, *"Discovering Event Evolution Graphs from News Corpora"* IEEE Trans. on Systems, Man and Cybernetics—Part A: 39(4):850-863, 2009.

[3] J.Abonyi, B.Feil, S.Nemeth, and P.Arva, *"Modified Gath– Geva clustering for fuzzing segmentation of multivariate time- series"* Fuzzy Sets and Systems, Data Mining, Special Issue 149:39-56, 2005.

[4] C. C. Foster and E. M. Riseman, *"Percolation of code to enhance parallel dispatching and execution"* IEEE Trans. Comput., 21(12):1411{1415, Dec. 1972.

[5] W. Gray and D. Boehm-Davis, "Milliseconds matter: An introduction to microstrategies and to their use in describing and predicting

interactive behavior", Journal of Experimental Psychology: Applied, 6(4):322, 2000.

[6] A. Greenberg, J. R. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, D. A. Maltz, P. Patel, and S. Sengupta, *"VL2: a scalable and flexible data center network"*, In ACM SIGCOMM, pages 51{62, New York, NY, USA, 2009. ACM.

[7] D. Han, A. Anand, A. Akella, and S. Seshan. *"RPT: re-architecting loss protection for content-aware networks"* In 9th USENIX conference on Networked Systems Design and Implementation, NSDI'12, pages 6{6, Berkeley, CA, USA, 2012. USENIX Association.

[8] S. Jain, M. Demmer, R. Patra, and K. Fall, "Using redundancy to cope with failures in a delay tolerant network", In ACM SIGCOMM, 2005.

[9] J. Li, J. Stribling, R. Morris, and M. Kaashoek, *"Bandwidth-efficient management of DHT routing tables"* In USENIX NSDI, 2005.

[10] U. D. of Labor. *Economy at a glance*. http://www.bls.gov/eag/ eag. us.htm.

[11] F. Qian, A. Gerber, Z. M. Mao, S. Sen, O. Spatscheck, and W. Willinger, *"TCP revisited: a fresh look at TCP in the wild."* In IMC, pages 76{89, New York, NY, USA, 2009. ACM.

[12] S. Ramachandran, "*Web metrics: Size and number of resources*" May 2010. https://developers.google.com/speed/articles/web-metrics.

[13] E. Soljanin, *"Reducing delay with coding in (mobile) multi-agent information transfer"* In Communication, Control, and Computing (Allerton), 2010 48th Annual Allerton Conference on, pages 1428{1433. IEEE, 2010.

[14] S. Souders, *"Velocity and the bottom line"* http://radar.oreilly.com /2009/07/velocity-making-your-site-fast.html.

[15] J. Whiteaker, F. Schneider, and R. Teixeira, *"Explaining packet delays under virtualization"* ACM SIGCOMM Computer Communication Review, 41(1):38{44, January 2011.

[16] X.Wu, Y.Lu, Q.Peng, and C.Ngo, "*Mining Event Structures from Web Videos"* IEEE Multimedia, 18(1):38-51, 2011.

[17] Q. He, K. Chang, E. Lim and A. Banerjee. Keep It Simple with Time: A Reexamination of Probabilistic Topic Detection Models. IEEE Transactions on Pattern Analysis and Machine Intelligence, 32(10):1795-1808, 2010.

[18] C. Sung and T. Kim Collaborative Modeling Process for Development of Domain-Specific Discrete Event Simulation Systems. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, 42(4):532-546, 2012.

[12] J. Allan, G. Carbonell, G. Doddington, J. Yamron, and Y. Yang. Topic Detection and Tracking Pilot Study Final Report. In Proceedings of the Broadcast News Transcription and Understanding Workshop, 1998.

