DISSERTATION

MANAGEMENT OF INTERNET-BASED SERVICE QUALITY

Submitted by

He Yan

Department of Computer Science

In partial fulfillment of the requirements

for the Degree of Doctor of Philosophy

Colorado State University

Fort Collins, Colorado

Spring 2012

Doctoral Committee:

Advisor: Daniel Massey

Christos Papadopoulos
Shrideep Pallickara
Dan Turk
Zihui Ge
Jennifer Yates

ABSTRACT

MANAGEMENT OF INTERNET-BASED SERVICE QUALITY

An increasingly diverse set of services, content distribution network (CDN), Internet games, streaming videos, online-banking, IPTV, VPN, cloud computing and VoIP, are built on top of Internet. For most of these Internet-based services, best effort delivery is no longer an acceptable mode of operation as ultra-high reliability and performance is demanded to meet the stringent service-level requirements.

In this dissertation, we focus on the research problem: how to manage the Internet-based service quality in a efficient and proactive manner from a service provider's point of view. Managing Internet-based service quality is extremely challenging due to its massive scale, complicated topology, high protocol complexity, ever-changing software or hardware environment and multiple administrative domains. We propose to look into this problem from two views (user view and network view) and design a novel infrastructure that consists of three systems (Argus, G-RCA and TowerScan) to enable managing Internet-based service quality from both views. We deployed our infrastructure in a tier-1 ISP that provides various Internet-based service and it has proven to be a highly effective way to manage the quality of Internet-based services.

ACKNOWLEDGEMENTS

First of all, I would like to express my deepest gratitude to my supervisor, Dan Massey, for his indispensable guidance and invaluable advice throughout my graduate studies at Colorado State University. I am also grateful to Dr. Massey for providing financial assistance to complete this research.

Secondly, I would like to thank my colleagues at AT&T Labs - Research. I am so fortunate to have Zihui Ge as my fantastic mentor at AT&T Labs - Research when I was an intern. He is extremely versatile in many domains ranging from statistical/Mathematical method to system building. He helped me tremendously in laying down the big picture, understanding the formal methods and designing systems. I am extremely grateful to my manager Jennifer Yates. She is very passionate, energetic and enthusiastic about the research in service quality managment. I learned a lot from her regarding operational networks and real-world research problems. She also protects me from being spread too thin and keeps reminding me the high priority of this dissertation. I would also like to thank my other mentors and colleagues at AT&T Labs - Research, Dan Pei, Lee Breslau, Jia Wang, Jeff Erman, Jeff Pang, Alex Gerber, Carten Lund and Oliver Spatscheck. It has been a true pleasure working with them.

Thirdly, I would like to sincerely thank my other committee members Christos Papadopoulos, Shrideep Pallickara and Dan Turk. They provided very good comments in the research exam and preliminary exam and helped me improve the quality of my dissertation.

Finally, I am grateful to my parents and my brother for their unconditional love and support during my graduate studies. Also I need to thank my wife Jia Li for her continual encouragements and unwavering love to me. She also helped me a lot to proofread this thesis.

This research is dedicated to all of my family, friends, mentors and professors who supported me in my academic journey.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

An increasingly diverse set for services have come to use Internet as the underlying infrastructure, including online gaming, stock-trading, IPTV, content distribution network (CDN), VoIP, cloud computing and mobility. In these Internet-based services, service providers are demanded to manage the service quality in an effective and proactive manner in order to meet the ultra-high reliability and performance expectations from users. For example, in an IPTV service, a degraded networking line card starting to drop data packets can cause an unacceptable level of video glitches on customers' home TVs in the region downstream of the line card. In a mobile phone service, users may experience slow download throughput or get connection time-outs due to an overloaded service gateway. In a VoIP service, a routing change may induce a significant transmission delay affecting users' conversation experience. In all cases, it is crucial for the service provider to promptly *detect these service degradations* and *localize their root causes* so that the service impacting conditions can be identified and addressed before user dissatisfaction escalates.

End-to-end service quality typically depends on hundreds of interacting hardware and software elements, each of which may have its own interfaces and protocols and may fail at any point. The Internet was originally designed for exchanging email and files among a few trusted hosts and doesn't facilitate the management of end-to-end

service quality. Any network glitch (e.g.: packet loss or link congestion) could have negative impact on the end-to-end service quality. Even worse, the service provider might only have the control on a subset of the hardware and software elements inside its network. Regardless whether the root cause of service quality issue is actually inside the service provider's own network, the service provider is responsible for tracking the problem and working with the software and hardware vendors or other third parties to fix the issue.

The change of service quality expectations has tremendously transformed the way that service providers manages their networks.

First, network operators have traditionally managed networks on the basis of individual network elements, for example, by detecting and replacing faulty network line cards. Today, managing issues related to end-to-end service quality has become an increasingly significant part of operators day-to-day work. This work typically involves such tasks as monitoring the loss and delay among different sites of a customer's VPN, and the download throughput between clients and CDN servers.

Secondly, network operators primarily focus faults and hard failures on individual network elements previously. Nowadays, the problems that draw their attention are increasingly transient end-to-end service quality issues (e.g: Download throughput drop lasted for a few hours between users and CDN servers). Although transient problems repair themselves later, it is critical to detect troubleshoot, and fix them in a proactive manner for two reasons.

- Even transient problems are short, many of them over time (e.g. chronic issues) would harm customer satisfaction.

- Detecting them potentially a large number of them collectively, classifying their root causes and trending them over time can provide operators with more critical insights This information may help in driving the corresponding failure modes out

of the network and eventually lead to service improvements.

In this dissertation, we focus on managing user-perceived quality of Internet-based services in a effective and proactive manner at the service provide side. We propose to look into this problem from two views (user view and network view) and design a novel infrastructure that consists of three systems (Argus, G-RCA and TowerScan) to enable managing Internet-based service quality from both views. We deployed our infrastructure in a tier-1 ISP that manages the user-perceived quality of various Internet-based service and it has proven to be a highly effective way to manage the quality of Internet-based services.

The organization of the rest of the dissertation is as follows. Chapter 2 describes the problem statement, illustrates the research challenges, discusses the potential data sources and introduces the two different views (user view and network view) of the same problem. Chapter 3 discusses the related works to this dissertation. In Chapter 4, we discuss two systems (Argus and G-RCA) that enable managing Internet-based service quality from user view to network view in a top-down manner. In Chapter 5, we introduces another system call TowerScan that tackles the exact same problem but from network view to user view in a bottom-up manner. Chapter 6 describes the future work and fnally Chapter 7 concludes this dissertation.

Chapter 2

Two Views of Internet-based Service Quality Management : User View and Network View

In this chapter, we first describe the problem statement and the research challenges, then discusses the potential data sources and finally illustrate how to solve the problem from two different views (user view and network view).

2.1 Problem Statement

In the dissertation, we focus on three specific research problems that are the fundamental building-blocks of managing Internet-based service quality.

- *P1- detect the service quality issues from users' point of view:* We aim to detect service quality issues (e.g.: outage or performance degradation) from users' point of view as soon as they occur. As the critical step of service quality management, detection needs to be done in a fast, accurate and scalable way in order to repair the end-to-end service quality issues before users complain. For example, we try to detect slow download throughput in a CDN service and choppy conversations In a VoIP service as soon as they arise.

- *P2 - identify the impact of service quality issues on users:* We aim to understand

the overall impact of service quality issues on users. Specifically, we are try to answer the following questions. Which users are impacted? What do the impacted users have in common? How severe are they impacted? How long are they impacted? The answers to these questions provide key insights to narrow down the search for root causes and prioritize multiple service quality issues. For example, we would like to identify that all CDN users from a particular city only received 50% normal throughput during a two hour period. With the deep understanding of the impact on users, it becomes easier for service provider to find the root cause and prioritize different service quality issues.

- *P3 - analyze the root causes of service quality issues:* We aim to pinpoint the root causes of service quality issues. The root cause analysis is critical in order to eventually fix the service quality issues. For example, a degradation in download throughput in the CDN service could be caused by interface flap, link congestion, router CPU overload, maintenance activities (e.g.: OS, firmware upgrades on routers) to intra-domain/inter-domain routing changes. Note we may not alway find the root cause in the service provider's own network. For example, the slow download throughput may be caused by issues in the neighboring networks, on the peering links between the service provider's own network and the neighboring networks or even in remote networks on the end-to-end path.

2.2 Measurements

The measurements available to service providers to manage their services can be largely divided into two categories: network-view measurements and user-view measurement.

Network-view measurements include the data sources that can be used to understand the status of individual network elements (e.g.: interface, router, switch and server) and calculate the routing path. For example, Syslog and SNMP messages indicate interface

flap, router CPU utilization, link utilization and link bit error while routing protocol (e.g.: OSPF and BGP) messages can be collected to understand the routing path at any given time.

Compared with network-view measurements, user-view measurements are less straightforward. The goal of user-view measurements is to best capture service quality perceived by users. Generally there are two ways to acquire the user-view measurements regarding service quality: intrusive or non-intrusive. In the intrusive approaches (e.g.: Keynote [8], Gomez [5]), agents periodically probe the service (e.g.: download a large file in a CDN service) from at different network locations to acquire the approximated service quality perceived by users. However, this approach has several limitations. First, without probes from a vast number of network locations throughout the Internet, the monitoring coverage is limited and some end-user perceived issues may not be detected. Secondly, probe packets also place additional overhead on the network and may be treated differently than normal packets.

Therefore, the non-intrusive way to obtain user-view measurements regarding the service quality is more preferred by service providers. In this dissertation, we focus on the user-view measurements collected in a non-intrusive way. On one hand, user-view measurements can be obtained through the feedbacks from users directly. For example, set up boxes in a IPTV service send the video quality report to service provide periodically, users complain service quality degradations in a CDN service via social media tools such Twitter and Facebook, Skype users rate the service quality via a survey after ending a call and 3G mobility users provide feedback on the service quality by using a cellphone application provided by the service provider. Figure 2.1 shows where individual users reported the service quality issues via a cellphone application provided by a 3G mobility service provider. This type of user-view measurement are mostly negative feedbacks, which directly indicate user-perceived service quality issues.

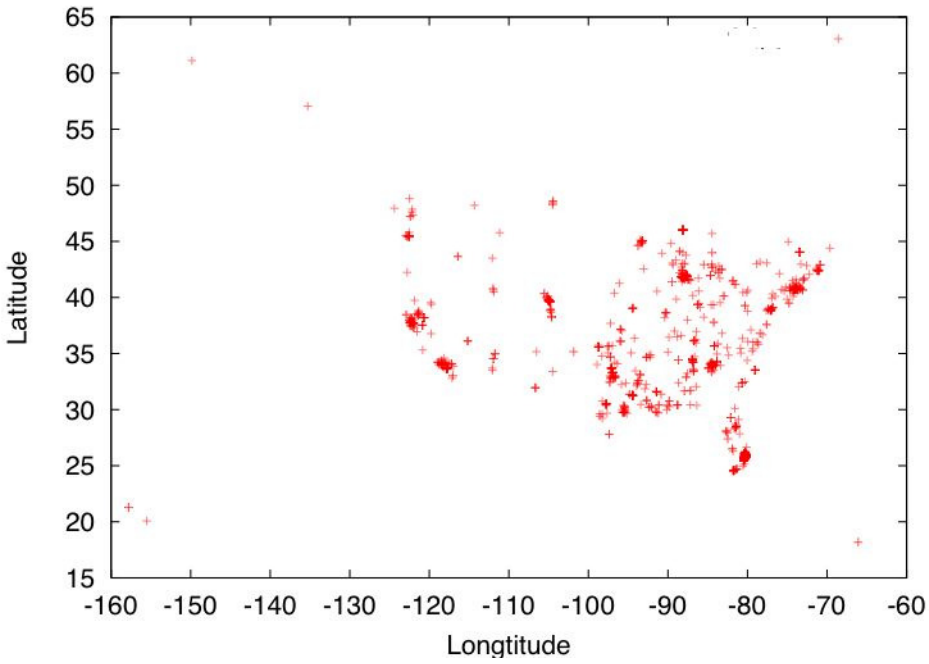
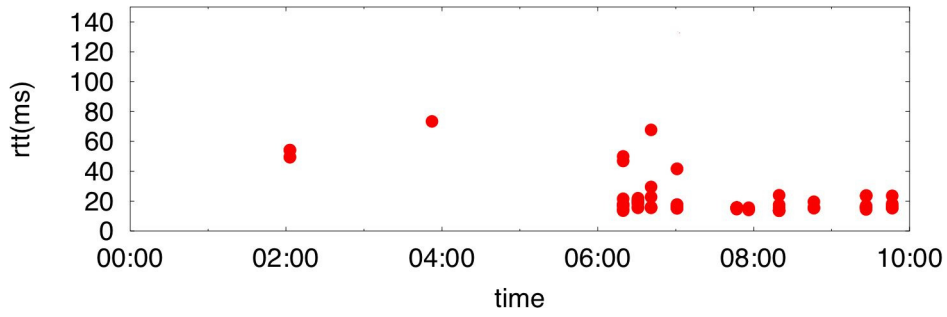
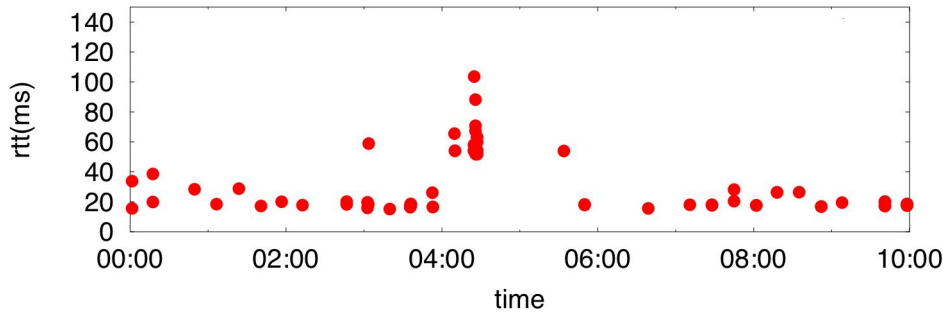


Figure 2.1: An example of user-view measurements: service quality issues reported by individual users in a 3G mobility service during one hour. Each dot is a user report from where he or she has a service quality issue.

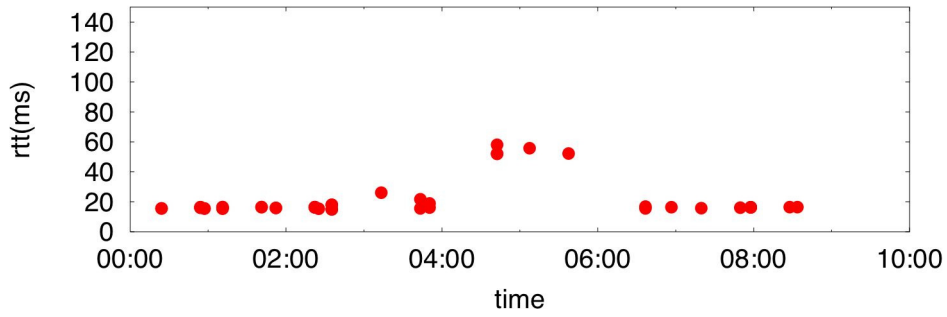
On the other hand, the user-view measurements can also be extracted from service traffic passively and indirectly. For example, the round trip delay between set up boxes and video servers in a IPTV service can be estimated by monitoring the traffic and the download throughput between cell phones and content servers can be also measured based on the traffic. Different from the user-view measurements directly obtained from user feedbacks, these measurement are typically performance numbers and don't directly tell if there is service quality issue or not. Figure 2.2 shows one common user-view measurement in a content delivery network (CDN) service: the round trip delay between server and users estimated by monitoring the traffic at service provide side. Specifically, it shows that the round trip delay associated with three individual users from the same BGP prefix for 10 hours in one day.



(a) User A



(b) User B



(c) User C

Figure 2.2: Another example of user-view measurements: the round trip delay associated with three individual users

2.3 Proposed Directions

In this dissertation, we propose to solve the three problems (Chapter 2.1) in two directions: top-down and bottom-up.

On one hand, the top-down approach starts with user-view measurements (e.g.: com-

plains from Twitter, customer calls and download throughput) to first detect the service quality degradations, then understand the user impact (e.g.: who are impacted? by how much? for how long?), and finally find the root causes (e.g.: link congestion and router CPU overload) that can be extracted from network-view measurements.

On the other hand, the bottom-up approach starts with network-view measurements to detect network events (e.g.: link congestion and router CPU overload) that may have been the root causes of some service quality issues. Then the bottom-up approach needs to determine if a network event has actually caused any service quality issues or not given the service/network redundancy (e.g.: When the primary router fails, the secondary router may work just fine without affecting user perceived service quality.). Finally if the network event has caused service quality issues, similar to the top-down approach, the bottom-up approach needs to understand the user impact (e.g.: who are impacted? by how much? for how long?).

2.4 Challenges

Several significant research challenges must be addressed in order to solve the three problems discussed in Chapter 2.1.

- *Challenge 1 – Complex Cross-Layer Dependency:* A wide range of network devices (e.g.: web servers, communication servers, media gateways, firewalls, multicast servers, name servers, routers, switches and fibers) at different layers (e.g.: physical layer, link layer, transport layer and application layer) need to work seamlessly to assure the end-to-end service quality (e.g.: user equipments download videos from cnn.com smoothly). For example, the service quality of a data session between an user's smartphone and cnn.com depends on the status (congestion level, bit error rate, CPU utilization etc.) of all the routers, links, cell towers, RNCs, SGSNs and GGSNs along the network path carrying the traffic and the

path is dynamically determined by application level redirection and layer-3 routing. Moreover, the layer-3 connectivity of each pair of adjacent routers on the path further depends on the layer-2 line protocol status between them, which in turn depends on the condition of the layer-1 (e.g., a SONET ring)network in between.

- *Challenge 2 – Build-in Service Redundancy:* The service redundancy designed at different layers makes the task of managing service quality even more complicated. At the radio access layer, the user equipments used to be served by a failed cell tower may migrate to some nearby tower depending on the location and load of nearby towers. At the network layer, failed routers and links may be avoided on the end-to-end service path via dynamic routing protocols. At the application layer, the users may be redirect to less busy content server via DNS redirection.

- *Challenge 3 – Incomplete Visibility:* Different segments of the end-to-end service path belong to different administrative domains (e.g.: other autonomous systems). For example, metro area networks may be managed by other companies and other autonomous systems in the Internet could also be involved in the end-to-end service delivery. The incomplete visibility poses a challenge for managing the end-to-end service quality issues. Note even without full visibility into others' networks, it is still helpful to "localize" an end-to-end service quality issue to those networks as the service provider knows where to complain or how to avoid the problematic networks. For example, if an end-to-end service quality issue may be localized to the neighboring network, the service provider may adjust the intra or inter domain routing to avoid that network.

- *Challenge 4 – Large and diverse user population:* Most of these internet-based services already have millions of users from different parts of the Internet today,

and the number is rapidly growing. First, collecting the user-view measurements and detecting the end-to-end service quality issues for each of them pose a scalability challenge first. Secondly, as user-view measurements are only collected when an user accesses the services, the user-view measurements for individual users could be quite sparse. In other words, we may not have dense enough user-view measurements to effectively monitor the service quality perceived by individual users. Thirdly, the user-view measurements from individual users could be quite noisy (as shown in Figure 2.2) due to the variability in users' local conditions such as processing delay or lo- cal access queuing delays. All of these makes it quite challenging to perform any statistical analysis for individual users due to insufficient and varying samples.

Chapter 3

Related Work

In this chapter, we will describe literature related to the three fundamental problems discussed in Chapter 2.1.

3.1 Network Fault Detection and Localization

A class of previous work has focused on detecting and localizing hard failures from network's point of view. In SCORE [37]. Shared Risk Link Group (SRLG) was proposed to model the cross-layer dependency between IP links and underlying optical devices. Their main goal to localize the faults observed on the IP layer to the underlying optical layer in operational networks. Specifically, a set of IP links that use the same underlying optical device such as a fiber span or a signal amplifier form a shared risk link group. In other words, if the shared optical device fails, all of the IP links in the corresponding shared risk link group would fail. SCORE formulates the localization problem using a bipartite graph. Each IP link is represented as a node in the top level of the bipartite graph while each shared risk link group is represented as a node in bottom level of the bipartite graph. An edge exists between a IP link node on the top level and a shared risk link group node on the bottom level if that IP link is a member of the shared risk link group. Then SCORE aims to identify the smallest possible set of shared risk link groups to cover the observed IP link failures using a greedy heuristic of MinSetCover problem.

SHRINK [35] is similar to SCORE [37] as the same bipartite graph is used. However, instead of using the greedy heuristic, SHRINK employes Bayesian network to infer the most likely assignment of the states (0 means alive and 1 means failed) for the shared risk link groups given observations of IP links' status. In contrast to the SCORE, SHRINK takes into account different probabilities of failure of different SRLGs. It uses these probabilities to address the under-determinedness of the problem; from all possible SRLG sets whose failure could have caused the observed IP links to fail, it returns the SRLG set that is most likely. SHRINK also deals with potential inaccuracies in the SRLG database and the possibility of dropped SNMP link status reports.

[38] uses active measurement between edge routers to raise alarms whenever edge-to-edge connectivity is disrupted in a backbone network. These alarms then are fed into localization module that employs spatial correlation techniques to localize the root-cause of edge-to-edge connectivity issues . The localization algorithm used here is similar to the greedy approach for fault localization used in SCORE [37] .

[47] and [27] focuses on the characteristics of Internet end-to-end connectivity among hosts by using pings, traceroutes and routing protocol data. They are trying to understand how stable Internet end-to-end connectivity is, how long failures last and how failures in Internet end-to-end connectivity correlate with routing instability. These approaches are related to our problems P1 and P2. However, this dissertation focuses on service quality issues from user's point of view that mostly are soft and transient failures. These related works instead focus on hard failures from network point of view.

3.2 Service Event Detection and Localization

A large body of recent work focuses on end-to-end service quality. PlanetSeer [61] focuses on detecting end-to-end service quality issues (such as high loss rate and latency) in the CoDeeN [57] (A CDN on PlanetLab [49]) by passively monitoring the traffic and

localizing the scope of detected service quality issues by using traceroute. Compared with PlanetSeer, the dissertation focuses on localizing the end-to-end service quality issues without active probing. Moreover, this dissertation tries to pinpoint the root causes (e.g.: link congestion or router) of the localized service quality issues.

[40] first identifies the inflated prefixes by passively measuring the RTT between clients and Google's CDN servers and then uses traceroutes to Identifying causes (e.g.: lack of peering link) of latency inflation to improve the overall service quality in a long term. Different from [40] that focuses on long term service quality optimization, this dissertation focuses on the day-to-day service quality issues that are typically transient.

[44] is a network tomography approach that infers the internal network characteristics by passively monitoring the end-to-end service quality (e.g.: packet loss rate) between a web server (www.microsoft.com) and its clients. Network tomography is related to our problem P2 but we try to localize the root cause of the service quality issues reported by many users without a clear defined topology. For example, if the download throughput in a CDN service for most of users from a city is low, we may localize the root cause to that particular city.

Commercial systems like Keynote [8] and Gomez [5] are also available to detect the service quality issues from the end-user's perspective by active probing the service from agents at different network locations. Differer from the active probing based systems, this dissertation proposes to measure the service quality passively at the service provider side. Compare to active probing based systems, our approach measures the real user-perceived service quality instead of agent-perceived service quality without introducing overhead to the network.

A more recent work [13] proposed to push service quality management task to the end-hosts themselves and implemented such a prototype system for BitTorrent. Specifically, first each BitTorrent peer detects its local service quality events individually based

on some passively measured performance metric and then local events from different peers are correlated spatially and temporally to determine the scope of the problem. The effectiveness in [13] actually depends the sparsity of passive measurements for individual end systems. An end system with very few measurements would be able to detect event effectively. The deployment is another issue as there is no strong incentive for end systems to corporate. Unlike [13], our approach is not limited by the sparsity of measurements and by the deployment issue as we collect the service quality measurements from individual users passively at the service provide side and aggregate these service quality measurements to avoid the sparsity issue.

3.3 Service dependency graph inference

There has been extensive prior work on inferring the service dependency graph. For example, successfully fetching a SMB file depends on DNS resolution, Kerberos Authentication and accessing SMB server. Furthermore, a success DNS resolution depends on the all the routers and links on the path to the DNS server.

PINPOINT [20] keeps track of users' requests to discover the component dependencies and localize the faulty components in a distributed system. PINPOINT first aims to detect whether these users' requests are successfully served or not by looking for two types of failures: internal and external failure. Internal failures are typically masked before becoming visible to users while external failure are visible to the user. Interval failures include memory leakages while external failures include network outages or machine crashes. PINPOINT then records the components used by each failed and success request and finally a data clustering algorithm is used to discover sets of components that are highly correlated with the failed requests. The data clustering algorithm is an unweighted pair-group method using arithmetic averages.

SHERLOCK [14] passively infers the dependencies among the network, server and

applications from logs. First, SHERLOCK automatically constructs an Inference Graph that captures the dependencies between all components by combining individual views obtained from individual software agents by analyzing the packets. SHERLOCK uses information provided by one agent to fill in any gaps in the information provided by another. The inferred service dependency graph includes primitives that capture the behavior of load-balancers and failover mechanisms. Then SHERLOCK employes a localization algorithm that efficiently pinpoint the root cause using the inferred service dependency graph and measurements of service response times made by the agents.

EXPOSE [34] learns dependency rules in edge networks using spectral graph partitioning to extract significant communication patterns in a packet trace without explicitly being told what to look for. EXPOSE learns the underlying rules that govern communication over a network. From packet timing information, EXPOSE learns rules for network communication that may be spread across multiple hosts, protocols or applications. Orion [21] identifies the dependencies using packet headers and timing information in network traffic based on a novel insight of delay spike based analysis. While none of them solve all the three problems that this dissertation is targeting on, all the techniques are related to our problem P3 as they can complement the root cause analysis by inferring the dependency graph.

3.4 Commercial network management systems

Most of the commercial network management systems, such as EMC SMARTS [4], HP OpenView [6], IBM Tivoli [7] have focused on the performance of individual network elements such as routers, line-cards and interfaces. In practice, these systems have proven inadequate for service quality issues as they treat individual network elements independently without a complete view of service model.

First of all, the performance individual network elements are poor predictors of the

end-to-end service quality that users ultimately care about. These commercial systems are not designed to manage the user-perceived service quality and thus don't have the capabilities to understand the impact of individual network element on the user-perceived service quality. For example, its not clear how high CPU load on a router impacts user-perceived service quality. Therefore, it is hard to set a threshold that alerts only when users are impacted. Moreover, due to the large number of network events, investigating all the potentially user-impacting network events is simply impractical, especially when many of them have no effect on users. This dissertation instead focuses on detecting and localizing the problems that affect users.

Secondly, the commercial systems typically employ simple threshold-based anomaly detection with little embedded intelligence. For example, they are set to generate alerts based on certain threshold violations such as "alarm if router CPU utilization is more than 80%". However, it could be challenging to figure out the right thresholds in practice. In the contrast, we use more intelligent time series forecasting techniques and incorporates important enhancements tailored for service measurement data. In other words, no thresholds need to be set in our approach and anomalies are detected if the actual values differ too much from the forecast values.

Thirdly, these commercial systems typically require too much manual effort in configuration while our approach requires only minimal configuration. For example, as most of the commercial systems are threshold-based, a larger number of thresholds need to be configured for hundreds of devices involved in a service. In contrast, this dissertation focuses on managing user-perceived service quality from both user view and network view by using various statistical analytics that requires minimal human intervention.

3.5 Model based service quality prediction:

There are several model based system to predict service quality based on network measurements. WISE [55] presents a what-if analysis tool to predict the effects of network configuration changes on service response times based on historical data. WISE first identifies features in the dataset that affect response time in a CDN service. Careful choice of features improves the utility of the dataset for evaluating what-if scenarios. Secondly, it constrains the inputs to valid scenarios based on existing dependencies. Then WISE presents the network designers with an easy-to-use interface in the form of a scenario specification language called WISE-Scenario Language. Finally WISE estimates the response time distribution function in a piece-wise manner, and also structures the processing so that it is amenable to parallel processing.

Q-Score [54] accurately learns a small set of network measurements most relevant to user-perceived service quality based on historical data, and proactively predict service quality in a single score. Q-score system takes input from network (including servers, transport and in-home devices) performance indicators, which we refer to as features, the user control activities, and the user feedback in the form of customer call service records. The output is a series of Q-scores, one for each user of the service, quantifying the received service quality. At a high level, Q-Score system is composed of two components: an offline learning component and an online continuous monitoring component. Q-Score carefully designs the appropriate temporal and spatial aggregations to remedy the inherent loss, noise and delay with user feedback. Furthermore, by applying statistical regression over a large quantity of historical data between various network KPIs and the user feedback, a set of regression coefficients which quantitatively capture their relationship is obtained. These regression coefficients are fed into the online monitoring component to predict service issues.

They are related to our problem P1. These approaches can predict service quality is-

sues in a proactive way but the accuracy is still a concern (e.g.: Q-Score claims they can predict 60% of the service quality issues with 0.1% false positives). In the contrast, this dissertation focuses on detecting service quality degradations in a reactive and accurate manner.

Chapter 4

Top-Down Direction

In this chapter, we explore the top-down direction in managing quality of Internet-based services. As we discussed in Chapter 2, there are three main steps in the top-down direction:

1. Detecting service anomaly events based on user-view measurements (e.g.: complains from Twitter, customer calls and download throughput).

2. Estimating the user impact of detected service anomaly events, e.g.: who are impacted? by how much? for how long?

3. Identifying the root causes of service anomaly events by correlating with network-view measurements (e.g.: link congestion and router CPU overload).

We implement the first two steps in a system called Argus while the last step regarding root cause analysis is implemented in another system called "G-RCA". Both Argus and G-RCA are designed in a generic manner for various Internet-based services.

4.1 Argus: Detecting Service Anomaly Events and Estimating their User Impacts

In this section, we first describe our design of Argus, then discuss how we apply Argus in a CDN (content delivery network [46, 25, 52, 56]) service and finally present the

evaluation results and operational experience.

4.1.1 The Design of Argus

Argus is a generic system to detect service anomaly events and estimate their user impacts base on the user-view service quality measurements collected from individual users. Specifically, it turns the service quality measurements from individual users into localized and prioritized service anomaly events in streaming fashion. In order to address the challenges discussed in Chapter 2, we adopt a five-stage approach for Argus as shown in Figure 4.1.

(i) Spatial aggregation: aggregate the user-view measurements associated with individual users into different user-groups. An user-group consists of the set of users that share some common attributes.

(ii) Temporal aggregation: aggregate the user-view measurements in each user-group into time-bins and pick a summary statistic from each time-bin to form a **summary time series** for each user-group.

(iii) Event detection: apply a one-pass time series forecasting technique to extract service anomaly events from the **summary time series** of all user-groups.

(iv) Event localization: adopt a novel hierarchical event localization algorithm to identify one or multiple user-groups that can explain the detected service events on other user-groups.

(v) Event prioritization: prioritize localized service events based on their impacts on users such as scope, severity and lasting duration.

All these stages operate in streaming fashion, which means service anomaly events are detected with some initial user-view device quality measurements and updated in terms of impact scope, duration, status and as more and more measurements arrive. Argus notifies the service operators as soon as service anomaly events are detected and keeps them updated with the latest information regarding these events so that operators can

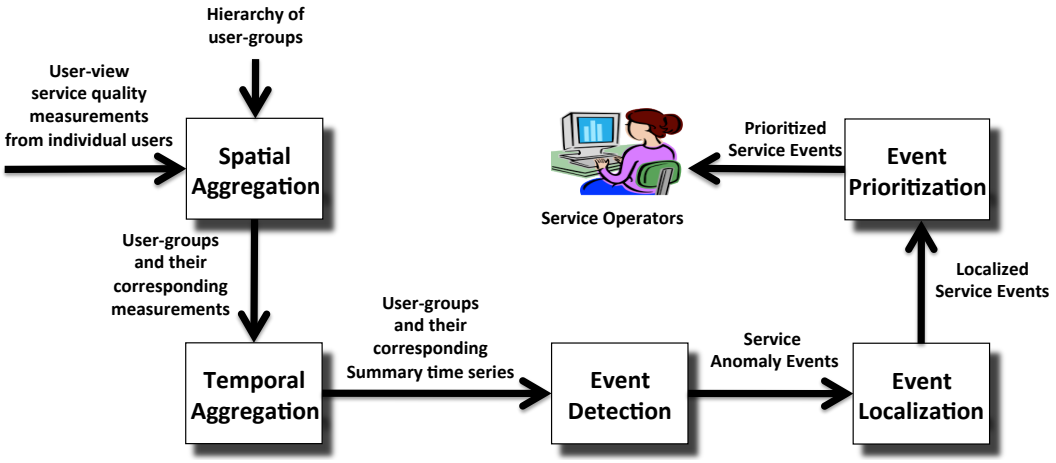


Figure 4.1: The architecture of Argus

keep track of them. We now describe each stage in Argus in detail.

4.1.1.1 Spatial Aggreagtion

While user-view service quality measurements associated with individual users (as shown in Figure 2.1 and 2.2) provide valuable information to manage service quality proactively, it is quite challenging to use them to detect and localize service anomaly events in practice. The critical research question here is how to be oblivious to local dynamics and sparsity in the user-view measurements from a large number of users but remain sensitive to service anomaly problems continuously.

First, service quality measurements associated with individual users could be quite varying due to their local dynamics. In a 3G mobility service, individual users may experience bad performance when they move into some buildings with strong radiant barriers. The bad service quality reported by individual users due to local dynamics are not the primary interests from the point view of service providers as they have very limited control on these issues. Instead, service providers are more interested in the case that the service quality become bad for a larger set of users simultaneously due to some common network issues or server issues. For example, the service provide

typically pays more attention to the significant spatial concentration of red dots instead of individual red dots in Figure 2.1. Similarly, in a CDN service, individual users may suffer from slow download speed due to large queues on the users ADSL links or high CPU utilization on the user's laptops. As one can see from Figure 2.2, the service quality in the form of performance numbers from individual users could be too varying due to local dynamics. Specifically, the round trip delay associated with "User B" and "User C" vary significantly from smaller than 10 ms to larger than 100 ms.

Secondly, service quality measurements are typically collected only when users request the service, which means the service quality samples for individual users could be too sparse to detect service quality issues continuously in practice especially for those that don't request the service frequently. As shown in Figure 2.2, "User C" apparently has too few service quality measurements to keep track of the service quality during the 10 hours. Figure 4.2 shows that the CDF of the number of service quality measurements for all 1053 individual users from the same BGP prefix for 10 hours in one day, where 90% for them has less than10 service quality measurements for the 10 hours. Similarly, Figure 4.3 shows that the CDF of the number of service quality measurements for all 4547 users in a 3G mobility service for 1 hour in one day, where 98% for them has less than 10.

Finally, a Internet-based service typically has a large number of users. For example, there are several millions of users in a CDN service or a 3G mobility service.

In order to avoid keeping track of the service quality perceived by millions of individual users and address the sparsity and local dynamics issue in the user-view service quality measurements for individual users, Argus first spatially aggregates service quality measurements from individual users into user-groups. Each user-group is set of users that share some common attributes. For example, all users from a "BGP prefix" can form a "BGP prefix" user-group and the users are from the same city can form a city

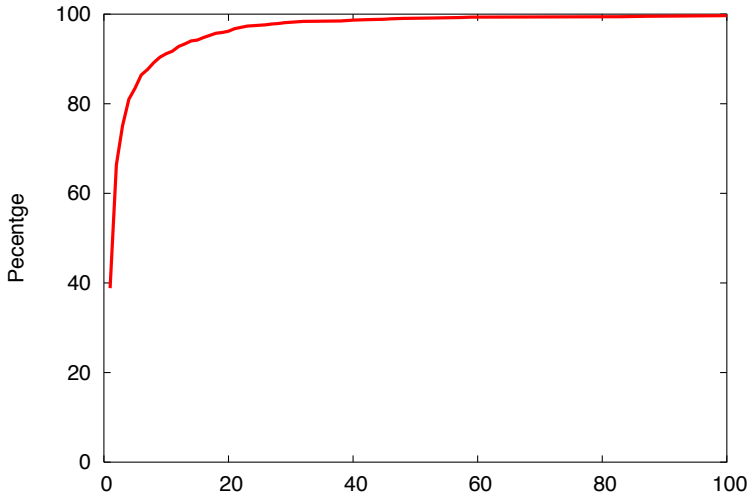


Figure 4.2: CDF of number of service quality measurements per user from the same BGP prefix in a CDN service for 10 hours

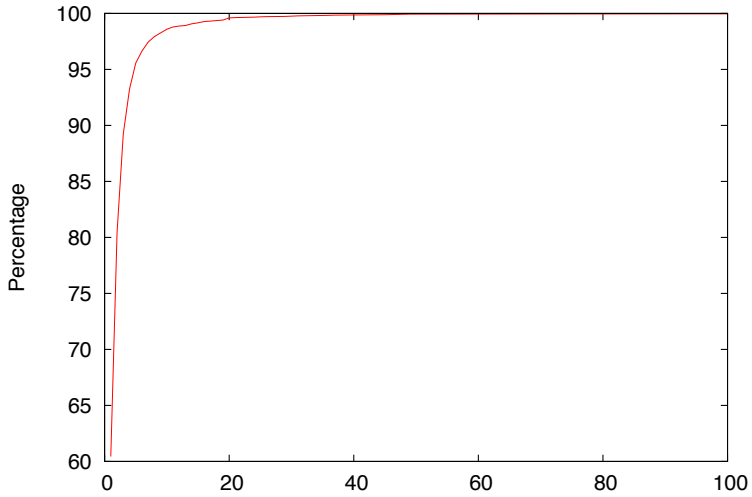


Figure 4.3: CDF of number of service quality measurements per user in a 3G mobility service for 1 hour

user-group. The attributes on individual end-users can be obtained from different data sources such as network topology, routing information and geographic locations. One user might belong to different user-groups.

As shown in Figure 4.1, the input to this module are a stream of service quality measurements from individual users and a hierarchy of defined user-groups. While the

service quality measurement stream could be collected from different services (e.g.: CDN, IPTV) in different ways, it is transparent to Argus. Argus only needs to know the attributes of end-users and how to aggregate them into user-groups using these attributes.

Figure 4.4 shows an example of user-group hierarchy. Each service quality measurement from individual end-users is aggregated into multiple user-groups in the hierarchy. For example, a throughput measurement for end-user "10.1.1.1" is aggregated into 4 user groups (Subnet "10.1.1/24", BGP prefix "10.1/16", AS path "123-789" and Origin AS "789") in the hierarchy. Here "AS path" is the AS-level path that the service provider traverses to reach end-users.

In practice, this hierarchy reflects where operators want to detect and localize service anomaly events. It is worthwhile noting that although the hierarchy shown in this example is a tree, in most cases the hierarchy of user-groups could be a general graph. In other words, one user-group in the hierarchy can have more than one parent user-groups. For example, in the example show in Figure 4.4, AS path"123-789" can have two parents: Origin AS "789" and Next-hop AS "123". Whether the hierarchy is a tree or a general graph determines the computation complexity of event localization later.

4.1.1.2 Temporal Aggregation

After the step of spatial aggregation, each user-group has a set of service quality measurements from individual users associated with it. The next important question to answer is that how to detect service anomaly events for each user-group. The challenge here is that the service quality measurements belonging to each user-group could be quite noisy as they are collected from different end-users. Our solution is to focus on the summary statistics (e.g.: 50th percentile, 95th percentile, min, max or sum) of the distribution instead of based on individual service quality measurements. For example, regarding the numeric service quality measurements such as round trip delay and throughput, average or median is used to be the summary statistics. If the service quality

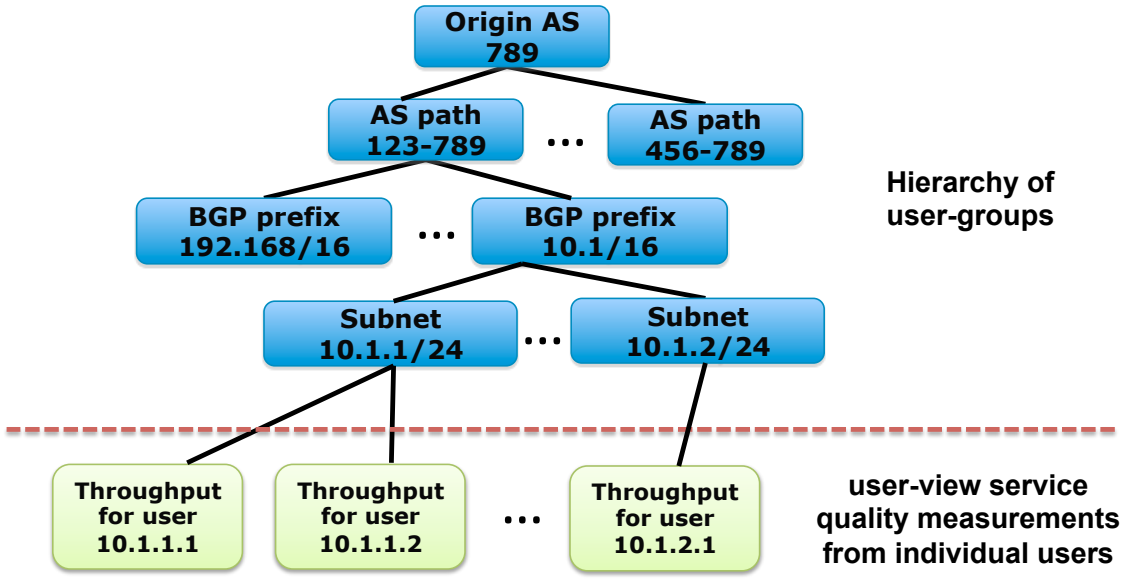


Figure 4.4: an Example of Spatial Aggregation in Argus

measurements are the binary user feedbacks such as customer calls and tweets, summation is typically used as the summary statistic. Obviously, we lost some details regarding individual end-users by focusing on the summary statistics. But it is acceptable as the goal is to detect service events that impact the user-groups. The service events detected on user-groups are more likely to be actionable from service provider's perspective compare to the service events detected on individual users.

In this step, for each user-group (obtained in the pervious step), we temporally aggregate its service quality measurements into time-bins. Binning is a classic data processing technique for data smoothing, which is much needed here for noisy service quality measurements in each user-group. We use two types of binning methods – fixed-size binning and fixed-length binning. In fixed-size binning method, the service quality measurements in each user-group are divided into equal size (e.g., 100) groups. In fixed-length binning method, the service quality measurements in each user-group are divided into equal length (e.g., 10 minutes) groups. Comparing the two approaches, fixed-length binning is more intuitive, however it is more sensitive to data sparsity – smoothing over

one or a few data samples is ineffective. Fixed-size binning on the other hand is more sensitive to variability due to changes in the composition of individual users in each user-group. In our current implementation, Argus runs in either fixed-length binning mode or fixed-size bin mode, while it remains as our future work to evaluate how much benefit we can achieve to combine the two.

Once time-bins are formed, a summary statistic is selected from all the service quality measurements in each time-bin to form a **summary time series**. Several summary statistics can be used here – minimum, maximum, average, median, other percentile values and summation. Different statistics may have advantage for tracking certain type of issues. For example, if the service quality measurement is round trip time (RTT), the minimum may well capture baseline RTT due to network propagation delay while being oblivious to varying queuing delay that may be due to network congestion while average can well capture the service event due to network congestion. If the goal is to detect service events that impact a relatively large collection of users in each user-group, median is a good option as the summary statistic for each time-bin by default. We find median quite effective in tracking service side or network side issues while being robust to variability in performances of individual users due to their local processing or local queuing delays.

4.1.1.3 Event Detection

Once we transform the user-view service quality measurements into **summary time series** of each user-group, we can apply time series analysis techniques to extract service anomaly events from them. There are a wide range of time series anomaly detection algorithm in the literature, ranging from Box-Jenkins [45] linear time-series forecasting techniques, to frequency domain Fourier analysis [15] or Wavelet analysis [48] based approaches, to structural analysis such as principal component analysis [26]. Due to the scale of our application, it is desirable to have online anomaly detection with minimal

runtime complexity and memory requirement. We base our approach on the classic additive Holt-Winters (HW) algorithm [17], a widely used one-pass online time series forecasting method. One of the key strengths of HW is that it involves very light-weight computation and has very few states to maintain.

At a high level, HW runs three exponential smoothing processes on three components of the **summary time series**: the baseline, the linear trend, and the seasonal effect to dynamically formulate forecast values based on historical values. Specifically, the forecast value $\hat{y}_t$ at time t is formulated as follows:

$$\hat{y}_t = a_{t-1} + b_{t-1} + c_{t-n}$$

where a_t is the baseline at time $t-1$, b_t is the linear trend at time $t-1$ and c_{t-n} is the seasonal effect at time $t-n$. Note n is the number of cycles in one season (e.g., 24 cycles in one day to model hourly seasonality) and $t-n$ means the same cycle in the previous season (e.g., the seasonal effect for 1pm in the previous day is needed to formulate the forecast value for 1pm today). Given a new value y_t, a_t, b_t and c_t are updated exponentially with parametersα, β, γ respectively:

$$a_t = \alpha \times (y_t - c_{t-n}) + (1-\alpha) \times (a_{t-1} + b_{t-1})$$

$$b_t = \beta \times (a_t - a_{t-1}) + (1-\beta) \times b_{t-1}$$

$$c_t = \gamma \times (y_t - a_t) + (1-\gamma) \times c_{t-n}$$

In order to determine if a new value y_t in the summary time series is abnormal or not, we compare the current residual error (e.g., absolute difference between the actual value y_t and the forecast value $\hat{y}_t$) with the historical residual errors from the same cycle in previous seasons. In this way, seasonal variability in residual errors can be captured. The update formula for d_t is similar to that of c_t and uses the same parameter γ:

$$d_t = \gamma \times |y_t - \hat{y}_t| + (1-\gamma) \times d_{t-n}$$

Specifically, for each y_t, we calculate its descretized abnormal level A as follows: $\mathcal{A} = 0, 1, 2, 3, 4, 5$ when $|y_t - \hat{y}_t|$ is in $[0, 0.5 \times d_{t-n})$, $[0.5 \times d_{t-n}, 1 \times d_{t-n})$, $[1 \times d_{t-n}, 1.5 \times d_{t-n})$, $[1.5 \times d_{t-n}, 2 \times d_{t-n})$, $[2 \times d_{t-n}, 2.5 \times d_{t-n})$ and $[2.5 \times d_{t-n}, \infty)$ respectively. Although it is configurable, we typically consider $\mathcal{A}$ of 4 or above as anomalous as suggested in [18]. This is a relatively aggressive setting (i.e., more anomalies). However, it is an appropriate setting as our event localization and prioritization (next two stages) is robust to false positives. We further combine consecutive anomalous values in the summary time series into a single anomaly event and keep track of all on-going anomaly events, with the begin time of the event being the begin time of the first anomalous value.

Although the classic HW has been proven effective in many different application settings, when dealing with the real service performance measurements, we have uncovered several serious deficiencies with HW and proposed corresponding enhancements.

- **Robust Forecast against Dirty Data and Service Disruptions**

  "Dirty data" is unfortunately unavoidable in real service performance measurements – problems in various software/hardware components of data collectors and ingestion servers can occur, rendering the performance data nonsensical. Furthermore, there are many situations in which network problems can cause service disruptions, driving the performance data out of the norm. For example, in Figure 4.5, the huge dip (the average RTT is decreased significantly due to missing measurements) is caused by a bug in the data collector and the spike is caused by a link congestion. It is highly desirable that forecast and anomaly detection can be robust against the "dirty data" and service disruptions in our context. In our approach, a_t, b_t and c_t are not updated if the abnormal level A of the current value y_t is above a configurable threshold A_u, which depends on the nature of the summary time series. For a stable time series, A_u should be configured tightly so that most anomalies are excluded in formulating the forecast values. For a noisy

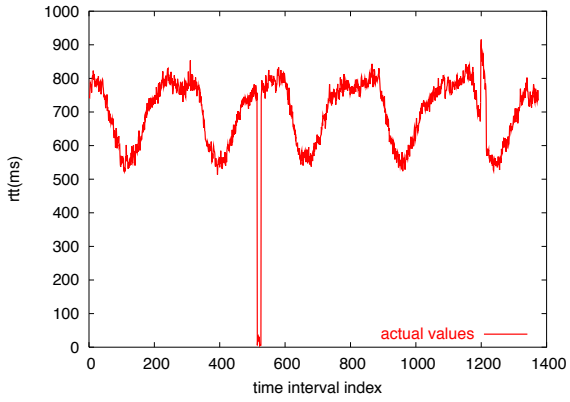


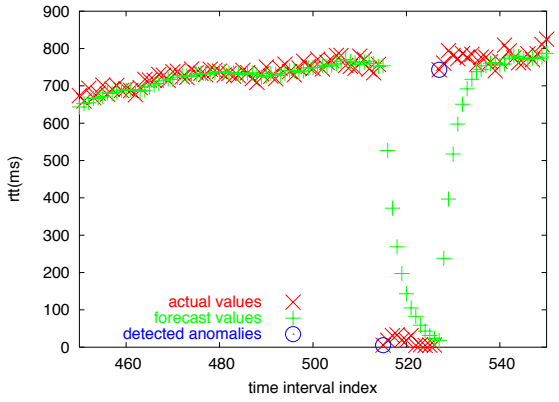
Figure 4.5: Anomalies in a summary time series that consists of the average RTTs, one for each 5 minutes

time series, less tight A_u is desirable to avoid excessive number of anomalies.

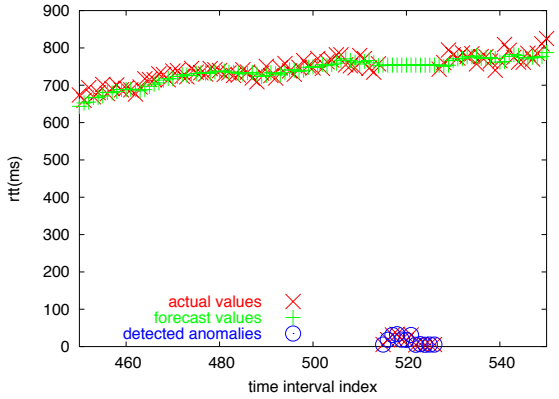
As a head-to-head comparison, we take a close look at the dip in Figure 4.5 and plot the different behaviors for classic HW and our approach in Figure 4.6. One deficiency in classic HW is that the forecast values can be easily contaminated by nonsensical performance data as shown here. On the contrast, our approach has a more stable forecast line and thus more robust in detecting anomalies.

- **Fast Adapt to Permanent Service Performance Changes**

  Network and service upgrades can often introduce a permanent level-shift on the end-to-end service performance. Once Argus is confident that a permanent change has taken place, it is ideal to "forget" about the extended long historical data and adapt the model to capture the recent performance only. To achieve so, a shadow set of baseline a_t^s, linear trend b_t^s, seasonal effect c_t^s and residual error d_t^s is updated in parallel using its own α_s, β_s and γ_s. Compared with the working set of baseline a_t, linear trend b_t, seasonal effect c_t and residual error d_t, the shadow set is updated differently in two aspects. First, the shadow set gives more weights to the recent observations via using relative large α_s, β_s and γ_s while the working set gives more weight to the history compared to the recent observations by using relative



(a) Classic HW



(b) Our Approach

Figure 4.6: Forecast values and detected anomalies for classic HW and our approach using the same set of parameters. A_u is set to 4 in our approach.

small α, β and γ. Secondly, anomalies are used to update the shadow set while they are ignored when updating the working set. In this way, the shadow set can quickly adapt to the permanent level-shift.

A moving window of size L is used to keep track the recent values. Once the percent of abnormal values in the moving window exceeds a threshold P (suggesting that a permanent level-shift has occured), the working set is replaced with the shadow set as the shadow set should have adapted to the permanent level-shift. The configuration of L and P determines the trade-off between the timeliness in adapting to permanent changes and the risk of wrongly adapting to temporary

abnormal changes.

- **Leverage Temporal Continuity in Residual Errors**

  In the classic HW, the exponential smoothing is applied on the residual errors d_t on a per cycle basis. For example, if there are 24 hourly cycles in one season, the exponential smoothing only applies to the residual errors of the same hour in different days. In order to leverage temporal continuity in residual errors perform in neighboring cycles (e.g., the neighboring hours in a day), weighted moving average is used for smoothing the residual errors d_t among neighboring cycles. Specifically, given a new value y_t, the residual errors for current cycle and its neighboring cycles $t-w, \cdots, t, \cdots, t+w$ are updated as follows:

$$d_{t+i} = \gamma_w \times (1 - \frac{|i|}{w+1}) \times |y_t - \hat{y_t}| + (1 - \gamma_w \times (1 - \frac{|i|}{w+1})) \times d_{t+i-n}$$

  where w is the smoothing window size, i is offset compared with the current cycle ranging from $-w$ to w and γ_w is derived from γ given the window size w. Similar to the working set, the residual errors in the shadow set are also smoothed among neighboring cycles in this way.

  Leveraging temporal continuity is critical in particular when the the number of cycles in one season is large (e.g., 288 5-min intervals in a day). Smoothing the residual errors over neighboring cycles restores the continuity and makes anomaly detection more robust.

- **Fine-Grained Detection with Coarse-Grained Model**

  In the classic HW, the detection interval has to be aligned with the cycles in a season. For example, if the classic HW models a time series with 24 hourly cycles in one season, the anomaly detection interval has to be hourly. In order to trigger

anomaly detection more frequently (e.g., once per 5 minutes), the classic HW has to model the time series with more cycles (e.g., 288 5-minutes cycles in one season). The memory consumption in HW is proportional to the number of cycles as one c_t and one d_t need to be maintained per cycle. Thus increase from 24 hourly cycles to 288 5-minutes cycles in one reason can considerably increase memory consumption by 12 times. Considering the millions of time-series need to be monitored in a service, fine-grained model might not be affordable.

In our approach, we enable the fine-grained detection with a coarse-grained model by linear interpolation of the parameters from two consecutive cycles in the model. If the anomaly detection interval is T (e.g., 20 minutes), one hourly bin (e.g., 11:20 - 12:20) may span over two consecutive hourly cycles (e.g., 11:00 -12:00 and 12:00 - 13:00) in the model. The c'_t for the hourly bin (11:20 - 12:20) are derived by the sum of two third of c_t for cycle (11:00 -12:00) and one third of c_t for cycle (12:00 -13:00). d'_t is calculated similarly.

4.1.1.4 Event Localization

As each service quality measurement contributes to the **summary time series** of multiple user-groups (as shown in Figure 4.4), a single underlying network failure such as a link failure may manifest itself at multiple user-groups in the spatial aggregation . For example, if an underlying network event has caused an increase of RTT for most of end-users associated with the same "BGP prefix A", Argus by design should detect the RTT service anomaly event for the user-group "BGP prefix A". Due to the nature of BGP routing, these end-users should share the same origin AS and AS path, and if these end-users from "BGP prefix A" dominate other end-users associated the same origin AS or AS path, Argus would also detect RTT service anomaly events for the corresponding user-groups of origin AS and the AS path. In this case, it is desirable for Argus to localize the issue to the "BGP prefix A" and report a single service anomaly event. For

another example, if a router failure has caused a throughput service anomaly event for all the end-users that are reached by the service provider via "AS path A", all the children user-groups of "AS path A" at the lower level in the hierarchy (as shown in Figure 4.4) such as the BGP prefix user-groups would experience the throughput service anomaly event as well. In this case, it is desirable for Argus to localize the anomaly to the AS path.

In general, the event localization problem (**ELP**) can be modeled as follows. Given a graph of user-groups, each user-group is marked abnormal, normal or undetermined due to insufficient samples as shown in Figure 4.7. The goal of ELP, in accordance with the

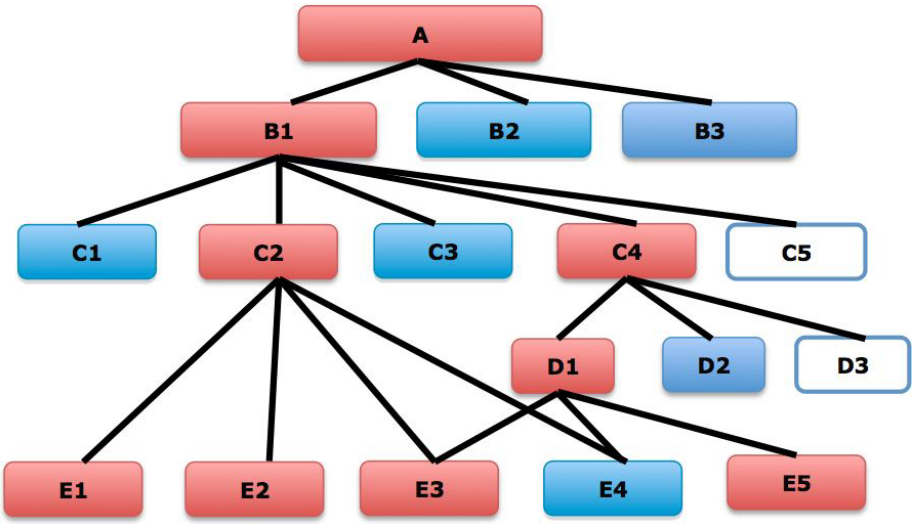


Figure 4.7: An example of the event localization problem

Occam's razor principle [16], is to identify the **smallest set** "A" of root abnormal user-groups that can explain all the detected abnormal user-groups and satisfy the following 2 constraints. First each abnormal user-group is either in "A" or a descendent of a user-group in "A" or a ancestor of a user-group in A. Secondly, an user-group is selected into "A" only when most of its direct descendants are abnormal. Next we present the formulation of event localization problem (**ELP**), its complexity analysis and a heuristic to solve the problem.

- **Problem Formulation**

  First we introduce the notations used in the problem formulation. In ELP, the user-group hierarchy (e.g. Figure 4.4) is a directed acyclic graph (DAG). Let N represent the set of user-groups in the hierarchy. $\forall n \in N, D(n)$ denotes the set of n's descendants (i.e. other user-groups can be reached from n by traversing edges). $\forall n \in N, A(n)$ denotes the set of n's ancestors (i.e. other user-groups can reach n by traversing edges). $\forall n \in N, d(n)$ denotes the set of n's direct descendants (i.e. other user-groups can be reached from n by traversing only one edge). $\forall n \in N, a(n)$ denotes the set of n's direct ancestors (i.e. other user-groups can reach n by traversing only one edge). The user-groups hierarchy has the following two properties.

  (**P1**) Each user-group is in one of the three status: abnormal, normal or insufficient measurements.

$$\forall n \in N : f(n) = \begin{cases} 1 & \text{if n is abnormal} \\ 0 & \text{if n is normal} \\ -1 & \text{if insufficient measurements for n} \end{cases}$$

  (**P2**) Each abnormal user-group has at least one abnormal or "insufficient measurements" descendant.

$$\forall n \in N : f(n) = 1 \Rightarrow \exists x \in D(n) : f(x) = 1 \vee f(x) = -1$$

  Objective Function:

$$\arg\min_{A \subseteq N} |A|$$

  The goal of ELP is to find a smallest user-group subset A that subjects to the following three constraints (C1-C3).

  Constraints:

(**C1**) Each user-group in A must be abnormal.

$$\forall a \in A : f(a) = 1$$

(**C2**) Each abnormal user-group in N is either in A or is a descendent of a user-group in A or a ancestor of a user-group in A. In other words, all abnormal user-groups are covered by the subset A.

$$\forall n \in N : f(n) = 1 \Rightarrow \exists a \in A : n = a \vee n \in D(a) \vee n \in A(a)$$

(**C3**) For any user-group in A, the number of its direct abnormal and "insufficient measurements" descendants is larger than the number of it direct normal descendants.

$$\forall a \in A : |\{x \in d(a) | f(x) = 1 \wedge f(x) = -1\}| > |\{x \in d(a) | f(x) = 0\}|$$

- **Complexity Analysis**

Theorem 4.1 The event localization problem *is NP-hard.*

Proof It is easy to show that ELP $\in NP$. Given a user-group subset $A \subseteq N$, validating that A satisfies the two above constraints can be done in polynomial time.

To show that ELP is NP-hard, we prove that set-overing problem [31] is polynomially reducible to ELP. i.e. set-covering $\leq_p$ ELP. Given an instance of set-overing problem with the universe $\{u_1, u_2, ..., un\}$ and a family of subsets $\{s_1, s_2, ..., sm\}$ of the universe, we can construct an instance of ELP as follows:

- Create a two-level user-group hierarchy with m abnormal user-groups at the bottom level (one for each subset) and n abnormal user-groups at the top level (one for each element in the universe).

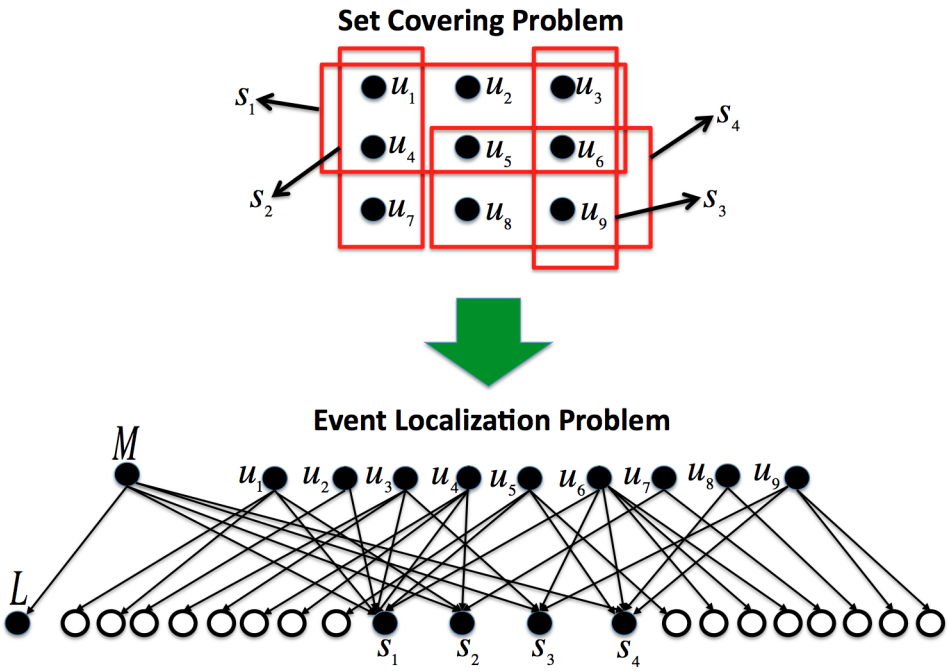


Figure 4.8: Polynomial Reduction Example

- Add a directed edge from one abnormal user-group u_i at the top level to one abnormal user-group s_j at the bottom level if u_i is a member of s_j.

- For each abnormal user-group u_i at the top level, add the same number of normal user-groups as the number of sets that u_i belongs to at bottom level.

- At the top level, add a common ancestor to abnormal user-group M that connects to all abnormal user-groups $\{s_1, s_2, ..., sm\}$ at bottom level.

- At the bottom level, add one special abnormal user-group L whose ancestor is M.

Figure 4.8 shows how the polynomial reduction is done from a set-overing instance to an ELP instance.

We claim that the constructed ELP instance has a feasible solution A with size at most $k + 1$ if and only if the original set-covering instance is satisfiable with at most k subsets. Indeed, if the set-cover is satisfiable with k subsets, then all abnor-

mal user-groups (i.e., the universe $\{u_1, u_2, ..., un\}$ in the set-covering) at the top level can be covered by k abnormal user-groups (i.e., k subsets in set-covering) at the bottom level. The remaining abnormal user-groups at the bottom, including a special abnormal user-group L, can be covered by the common ancestor abnormal user-group M at the top level. Conversely, suppose our ELP instance has a feasible solution A with size $k + 1$. Then, A must include i) the common ancestor user-group M to cover all the abnormal user-groups at the bottom level, and ii) k abnormal user-groups at the bottom to cover all the top level abnormal user-groups (note: those abnormal user-groups cannot be included in A, as each of them has normal children user-group). By selecting the subsets each of which corresponds an abnormal user-group at the bottom in our ELP solution A, the set cover instance is satisfiable with a collection of k subsets. Since this reduction is in polynomial time and the set-overing problem is NP-hard, the event localization problem is NP-hard[1]. This completes the proof.

- **A Greedy Solution**

  The event localization problem above can be mapped to the set-covering problem [31] . Using the set-covering terminology, all the abnormal user-groups in the hierarchy form the **universe**. By picking an abnormal user-group x that satisfies the constrains **C1** and **C3**, a **subnet** S_x ($S_x = x \cup D(x) \cup A(x)$) of the universe is formed. For each event localization problem instance, there is a family of n subsets ($S_1, S_2, ..., S_n$) that are corresponding to n abnormal user-groups that satisfy the constrains **C1** and **C3**. The goal is to find the smallest subfamily from whose

[1]If the user-group hierarchy is a tree, then the event localization problem is a P problem. In this special case, one can simple do a breadth-first search. Once a user-group that satisfy the constrains is found, one can immediately select it into output subset A and stop searching its descendants.

union is the universe. As set-covering problem is NP-hard [31], the event localization problem is NP-hard too. [31] presents a simple greedy heuristic for the set-covering problem and proves it is a factor-$\lceil \ln n \rceil$ approximation algorithm. Here we discuss a similar greedy heuristic to ELP, which keeps choosing the abnormal user-groups (subsets in set-covering problem) that covers most uncovered abnormal user-groups (elements in universe in set-covering problem) until all abnormal user-groups (the whole universe in set-covering problem) are covered. Here we discuss a similar greedy algorithm to the event localization problem, which keeps choosing the abnormal nodes (subsets in set-covering problem) that covers most uncovered abnormal nodes (elements in universe in set-covering problem) until all abnormal nodes (the whole universe in set-covering problem) are covered.

4.1.1.5 Event Prioritization

After event localization stage, Argus employes a ranking function to prioritize the localized service anomaly events. Since each anomaly event may contain multiple anomalous time-bins, we first estimate the severity score of each time-bin and then use the aggregate score of all time-bins as the severity score of the service event. The ranking function used to estimate the severity score of each anomalous time-bin incorporates two factors – the significance of the relative size of the anomaly and the breadth of its impact scope. The significance of the anomaly can be measured by the deviation score $|d|$ from the EHW algorithm. The impact scope can be measured by the the number of distinct end-users observed in the time-bin, which we denote as c. We choose distinct end-users since it is robust against anomalies dominated by a few outlier end-users.

Specifically, for anomaly event e, its baseline ranking score r_e is defined as:

$$r_e = \sum_{b \in \text{ bins of } e} A_b \times C_b$$

where A_b and C_b is the abnormal level and the number of distinct end-user for bin b. In

Algorithm 1: : Greedy Algorithm for Event Localization Problem

Let A denote the output subset
Initialize $A = \emptyset, UNCOV = \{x \in N | f(x) = 1\}$
for each $u \in UNCOV$ **do**
  $SET_u = u$
  for each $v \in D(u)$ **do**
    if $f(v) = 1$ **then**
      $SET_u = SET_u \cup v$
    end if
  end for
  for each $w \in A(u)$ **do**
    if $f(w) = 1$ **then**
      $SET_u = SET_u \cup w$
    end if
  end for
end for
while $UNCOV \neq \emptyset$ **do**
  Choose $u \in UNCOV$ such that $|SET_u|$ is maximized
  $A = A \cup u$
  $UNCOV = UNCOV - SET_u$
  for each $i \in UNCOV$ **do**
    $SET_i = SET_i - SET_u$
  end for
end while

this way, long lasting events are likely given higher priority than short events.

4.1.2 Apply Argus in a CDN service

Our objective here is to illustrate how Argus can be applied to monitor the service quality in a CDN service. In particular, we monitor the end-to-end TCP round trip time (RTT) between users and CDN servers as the key indicator of service quality. Many applications are extremely sensitive to network RTT (e.g., gaming). In the context of CDN service, the TCP throughput of large objects, which are more likely hosted by CDN, are expected to be inversely proportional to RTT [43], making it an important factor for CDN service providers.

A simple and common way to measure end-to-end RTT is to compare the timestamps of IP packets during the TCP handshake. In our case, one traffic monitor is installed for each CDN node (data center). The monitor observes the access links that connect the CDN node to the ISP backbone and it is configured to capture TCP handshake packets. When a request is observed, the traffic monitor calculates the time difference between the first SYN (from user to CDN server) and the corresponding ACK that completes the handshake (also from user to CDN server). This becomes the estimated RTT between the CDN node and the users. This RTT includes network propagation delay, any queuing delay (e.g., due to congestion inside network), and server side as well as user side processing delay.

We analyze the RTT data from three CDN nodes over a 10-day period (April 1st to April 10th, 2010). These three CDN nodes are located in northeast, southeast and northwest regions of USA respectively. The three datasets are hence named Northeast, Southeast and Northwest. Table 4.1.2 summaries the details of the three datasets. Subnet means the /24 prefix of the user's IP. BGP prefix means the longest matching prefix in the BGP table that covers the user's IP. AS path means the AS path from the CDN node to the user's IP. Egress router means the router at which data traffic from the CDN node

to the user's IP exits the ISP network. To protect proprietary information, we cannot list actual numbers of connections, user IPs and egress routers here. For example, for the dataset collected from the Northeast node includes tens of millions of connections were observed from several millions of user IP, which spanned 202,252 subnets, 23,869 BGP prefixes, 5,116 different AS paths and several hundred different egress routers. Note that the differences in coverage among three datasets are not caused by CDN assignment strategy. Instead, they are due to the incomplete deployment of traffic monitoring devices (for Southeast and Northwest) at the time of this study.

Table 4.1: Network Coverage of Three Datasets

Dataset	# Connections	# user IPs)	# Subnets	# Prefixes	# AS paths	# Egress Routers
Northeast	tens of millions	several millions	202,252	23,869	5,116	several hundreds
Southeast	tens of millions	several millions	41,784	3,613	649	several tens
Northwest	tens of millions	several millions	66,464	14,269	2,583	several hundreds

4.1.2.1 Variability in user RTT Series

Each TCP connection (e.g. successful handshake) made by a user results in a single RTT measurement. We simply refer to a series of passively measured RTTs associated with **a single user IP** as the **user RTT series**. These RTT series can be an important indicator quantifying the service quality perceived by the CDN users over time.

- **Variability across user RTT Series** Figure 4.9 shows the CDF of RTTs from all user RTT series in three different datasets. We normalize each RTT by the maximum RTT of all three datasets to protect proprietary information. We observe: (a) There is a large disparity in RTT's distribution for each dataset. All three datasets show significant variation (4 orders of magnitude) in per-connection RTT. (b) To a large extend the three datasets show a similar RTT distribution. In particular, for every dataset, a large faction of all RTTs has small or medium values while a small fraction of RTTs have large values.

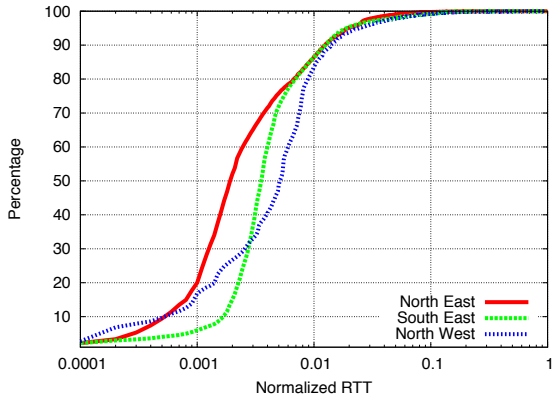


Figure 4.9: Distribution of normalized RTT across user RTT series

These observations suggest that the variability of RTTs across all user RTT series is large. On one hand, the large variability may be due to path diversity – connections from different users traverse different paths and each path may have different typical RTT. For example, a user in South America assigned to the Southeast node would more likely experience a greater RTT than a user in Florida simply due to the longer distance. On the other hand, the large variability may be due to time dynamics – the RTTs from the same user varies over time. For example, different connections from the same user may have different RTTs because of routing change or queuing fluctuation during a day. In our context of anomaly detection, we are more interested in the latter case, where different connections from the same user have largely varying RTTs. These variations may indicate some potential service performance anomalies. But it is not clear from Figure 4.9 how largely RTTs of different connections from the same user vary. To better understand this issue, we next examine the variability in RTTs within individual user RTT series.

- **Variability within Individual user RTT Series** We use the coefficient of variation (CV) metric to quantify the variability in RTTs within individual user RTT series. In other words, we are interested in the variability in RTTs measured from different requests of the same user during the 10-day period. Figure 4.10 shows

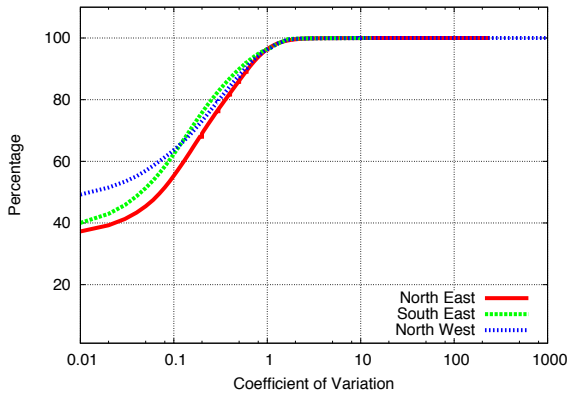


Figure 4.10: Distribution of coefficient of variation for individial user RTT series

the cumulative distribution of CV for all user RTT series in each of the three datasets. Note that x axis is in log scale. We can see from Figure 4.10 that all three datasets have similar patterns. Specifically, around 60% of user RTT series have very small CV (less than 0.1), which implies good predictability when using average historical RTTs to forecast future RTTs. Almost 35% of user RTT series show medium CV (ranging from 0.1 to 1), which indicates reasonable predictability. However, we also noticed that 5% of user RTT series exhibit large CV (ranging from 1 to 240). The large variability observed in these user RTT series suggest that RTT anomaly detection for the corresponding users may be challenging.

- **Self-inflicted RTT Increase**

  In examining the 5% user RTT series with huge variability, we discovered an interesting phenomena that contributes to the large variability – consecutive requests within a very short time period (second or sub second level) have almost monotonic increasing RTT value. For example, in one case, we observe 32 requests from the same user having subsequent RTT value increased from 25.84 ms to 202.04 ms within one second.

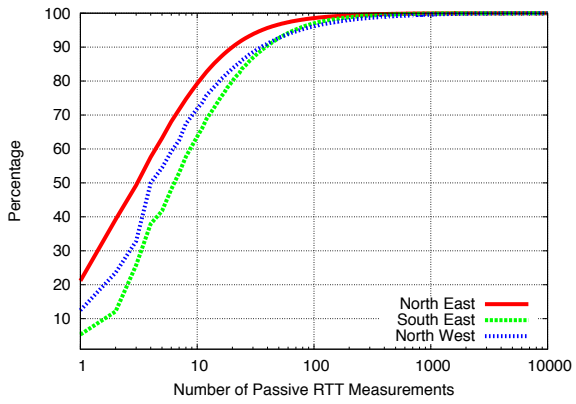


Figure 4.11: Distribution of number of measurements in user RTT series

Our conjecture to this behavior is that the RTT increase is *self-inflicted*. As recommended by HTTP 1.1 standard [28], modern browsers such as IE7, Firefox, Safari and Opera use multiple TCP connections in parallel to fetch different objects on the same page. Although HTTP 1.1 recommend 2 parallel TCP sessions, most of the latest releases of these browsers use much more concurrent connections: Firefox 3.5.9 and IE8 use 6 and Safari 4.0.5 uses 4 TCP sessions. Thus TCP SYN-ACKs from the CDN server are likely queued one after another at the user side access link or in processor buffer. Furthermore, data packets from different web servers may also get into the queue – for example, advertisement, javascripts, stylesheet file on the same webpage may not be hosted on the CDN server. Since each 100-byte packet queued over a 64 Kbps access link would increase the RTT of subsequent TCP sessions by 12.5 ms, it can quickly create a significant increase over several packets. Such self-inflicted RTT increases do not reflect any real performance problem for the CDN service, hence need to be carefully handled when we use user RTT series for performance impairment detection.

4.1.2.2 Sparsity of user RTT Series

RTT measurements are only collected when a user contacts a CDN node. In order to have timely measurements to detect service quality issues between users and CDN nodes, users need to communicate with CDN nodes often enough. In order words, if a user doesn't contact a CDN node very often, its user RTT series may be too sparse to reflect any service quality issues. In order to understand how often a user contacts a CDN node, we first plot the CDF of number of connections for individual users using the three datasets.

Figure 4.11 shows that most of users have very few connections over a 10-day period. This is true of all three data sets. More specifically, in Northeast dataset, 80% users have less than 10 connections; in Southeast dataset, 70% users have less than 10 RTT measurements; in Northwest dataset, 65% users have less than 10 connections. In other words, 10 passive measurements from 10 connections are too few to reflect service quality over a period of 10 days. The number of RTT measurements alone may not be sufficient to determine the measurement sparsity. For example, even though a user contacts a CDN node many times within the same second (doesn't contact the CDN node at other times), its user RTT series is still considered sparse as all these measurements only reflect the service quality at that single second.

In order to better understand the sparsity of user RTT series, we further define a RTT measurement to be "valuable" only if it is at least 600 seconds later than the previous RTT measurement. As in general path RTT appears steady at least for 600 seconds[63], passive measurements are within a period of 600 seconds should be considered as a single sample of end-to-end RTT. Ideally, we want to have one passive measurement every 600 seconds in order to better monitor the path RTT. Figure 4.12 shows most of users have even fewer "valuable" RTT measurements over a 10-day period compared to Figure 4.11. For all three datasets, 90% users have less than 10 "valuable" RTT

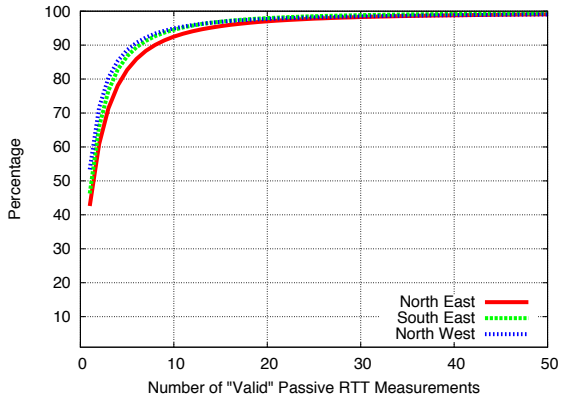


Figure 4.12: Distribution of number of “valuable” measurements in user RTT series

measurements.

As our approach purely depends on passive monitoring, we cannot change how often users contact CDN nodes to solve the sparsity problem. All of these suggest user RTT series are too sparse to detect service quality issues.

A naive approach of detecting service quality issues would be applying anomaly detection algorithms directly on the user RTT series. In other words, for each user IP, keep track of its user RTT series and detect abnormal RTTs deviated from its normal behavior that is built based on the history. But the above analysis suggests there are several limitations in this naive approach. **(i) Scalarbility:** It won’t scale with respect to the number of users. For example, in Northeast dataset, there are several millions of user IPs during a 10-day period. It is not trivial to keep track of several millions of user IPs. **(ii) Sparsity:** user RTT series usually are too sparse to conduct a statistical anomaly detection. **(iii) Variability:** The larger RTT variability within some user RTT series makes anomaly detection challenging.

4.1.2.3 Topological Aggregation of RTTs

As anomaly detection based on user RTT series is not practical, we adopt a different approach by aggregating user RTT series into higher level user-groups according to the

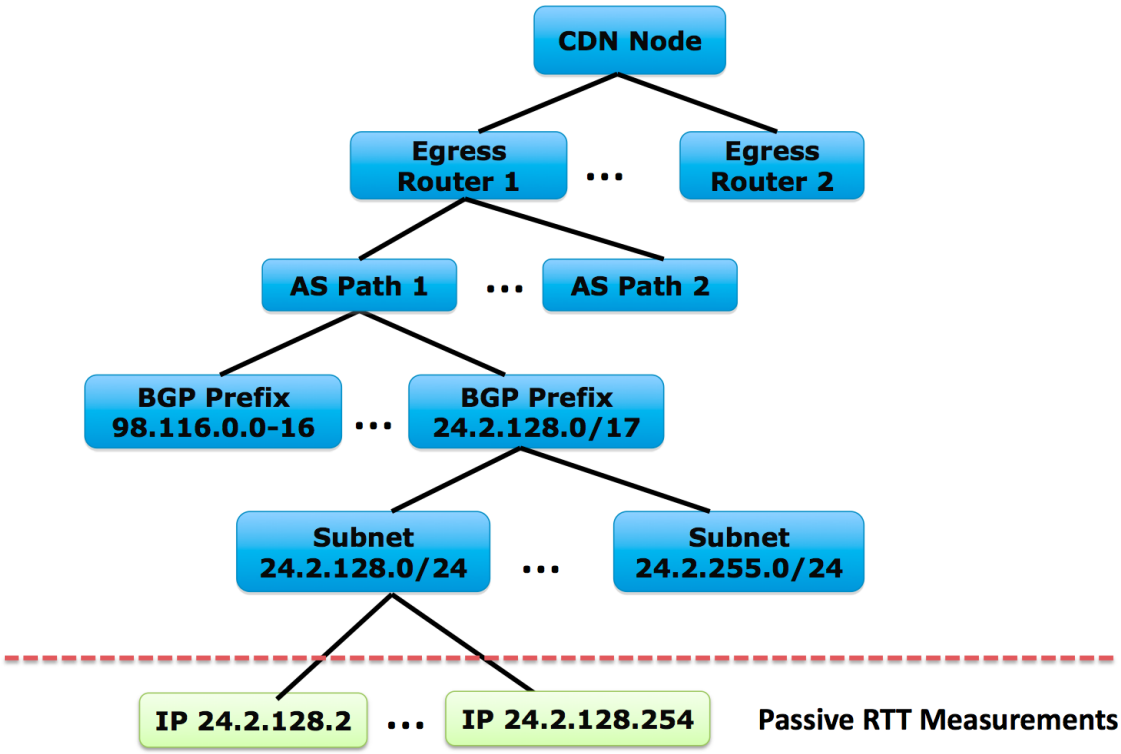


Figure 4.13: Topological hierarchy

topological hierarchy. In order to illustrate the idea of aggregation, we use the hierarchy shown in Figure 4.13 as an example, where the user RTT series are aggregated into subnet user-groups, BGP prefix user-groups, AS path user-groups and egress router user-groups. In other words, for each user-group in the hierarchy, a **aggregate RTT series** is formed by aggregating the user RTT series from all the users that are its children in the hierarchy. As a result, anomaly detection approach can be applied on the newly formed aggregate RTT series instead of the user RTT series.

This approach immediately solves the first problem of anomaly detection for individual users. More specifically, scalability is not a big issue here as there are much fewer user-groups we need to keep track of compared to the number of individual users. Let's take Northeast dataset as an example. After aggregating, instead of monitoring several millions of users, now we only keep track of 202,252 subnets, 23,869 BGP prefixes,

5,116 AS paths and several hundred egress routers.

Moreover, individual user level anomalies are not meaningful for localizing service quality issues as operators are more interested in the service quality issues that affect the RTTs of a large number of user IPs. For example, if most of users that traverse the same AS path experienced abnormal RTTs during a time period, it is more meaningful to report a single AS path anomaly to operators compared with reporting many anomalies for individual users. Due to aggregation, the anomalies are naturally reported for subnets, BGP prefixes, AS paths and egress routers. They are more useful to localize service quality issues compared to individual user anomalies.

- **Spatial Locality among user RTT Series**

  Aggregating users into user-groups based on topological hierarchy only makes sense if users that are topologically close to each other have similar user RTT series. Towards this end, we collect user RTT series at different user-groups and examine whether user RTT series in the same user-group are similar. Specifically, for each user RTT series, one key statistical indicators such as median and minimum is extracted. Then the similarity test among user RTT series is done by using this key statistical indicators.

  Here we consider the four different user-group, namely subnet user-group, BGP prefix user-group, AS path user-group and egress router user-group. We also form random user-group for comparison. First user RTT series are aggregated into user-groups according to different user-group definitions. We only consider user RTT series that have at least 100 measurements to keep the computation meaningful. Then for each user-group, we calculate the median(or minimum) RTT for each user RTT series in it and the CV of these median(or minimum) RTTs. In other words, the smaller the CV is, the stronger spatial locality is.

  Figure 4.15 and 4.14 plot CDF of CV for user-groups using median and minimum

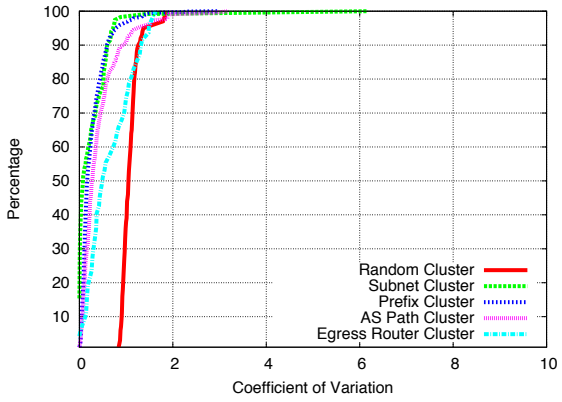


Figure 4.14: Distribution of coefficient of variation for user-groups (Minimum is used as the key statistical indicator for each user RTT series.)

as the statistical indicator respectively. These plots are generated based on Northeast dataset. Overall we can find from both plots that subnet user-group, BGP prefix user-group, AS path user-group and egress router user-group all exhibit significant stronger spatial locality than random user-group. Both plots also suggest that spatial locality is strongest in subnet user-group; BGP prefix user-group and AS path user-group show similar degree of spatial locality; egress router user-group exhibits less significant degree of spatial locality than others. Even though all user-groups in the topological hierarchy exhibit significant degree of spatial locality, we do notice the percentage of user-groups that show no spatial locality or very limited degree of spatial locality increases as the user-group moves up in the hierarchy. For example, in Figure 4.15, the max coefficient of variation for random user-group is 2.93 while there are almost 0.4% of BGP prefix user-groups has coefficient of variation larger than 0.4%. The number for AS path and egress router user-groups are 0.4% and 2%. We also conduct the same experiments using Northwest and Southeast dataset and they show the similar results.

- **Sparsity of Aggregate RTT Series**

  In addition to reducing the number of entities we need to track, aggregation may

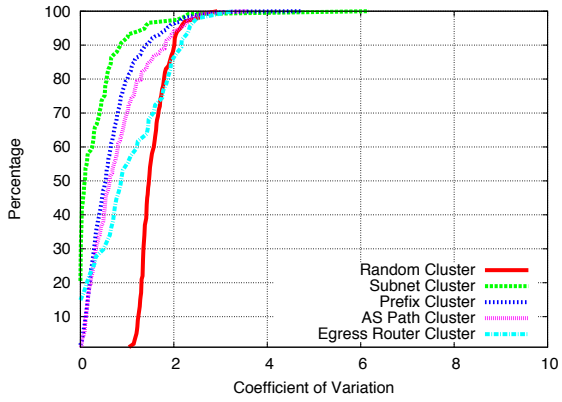


Figure 4.15: Distribution of coefficient of variation for user-groups (Median is used as the key statistical indicator for each user RTT series.)

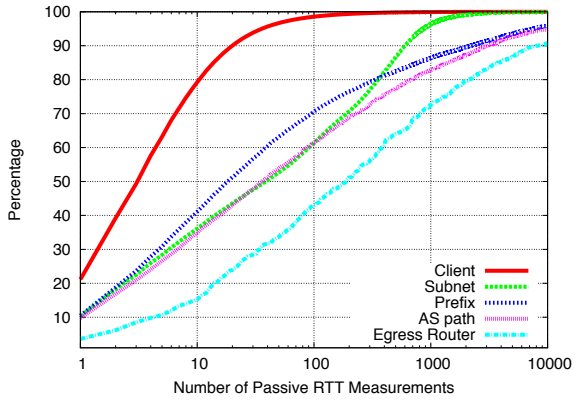


Figure 4.16: Distribution of number of RTT measurements for different user-groups

also overcome some of the sparsity problems seen when tracking individual user RTT series. As one may expect, the number of RTT measurements at aggregated user-group levels increases significantly compared to individual users. Figure 4.16 shows that for Northeast dataset: only 20% users have more than 10 measurements while 65% subnets, 60% prefixes, 65% AS paths and 85% egress routers have more than 10 measurements. The reason why more BGP prefixes have less than 10 measurements compared to subnets is that there are many BGP prefixes have length longer than 24 (subnet) in our BGP data.

Similar to section 4.1.2.2, we further define a RTT measurement to be “valuable”

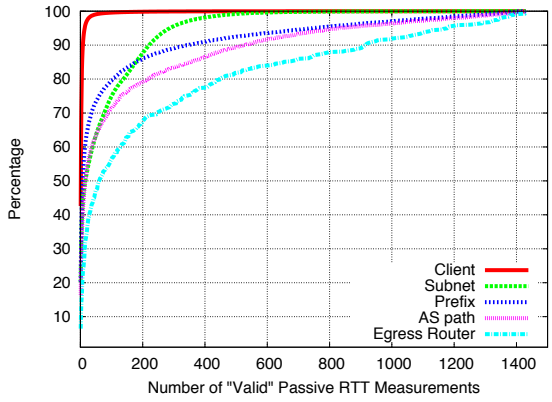


Figure 4.17: Distribution of number of ”valuable” RTT measurements for different user-groups

only if it is at least 600 seconds later than the previous RTT measurement. Figure 4.17 shows that for Northeast dataset: most of user-groups at all different levels have much more“valuable” RTT measurements compare to individual user IPs.

The above analysis suggests that sparsity is significantly improved at aggregated user-group levels. We also conduct the same experiments using Northwest and Southeast dataset and they show the similar results.

- **Summary**

  Aggregating user RTT series along the topological hierarchy addresses the scalability issue and measurement sparsity issue. It also naturally provides the ability of isolating sevice anomalies due to the topological significance in the hierarchy.

4.1.3 Evaluation Results

Evaluating the accuracy of an anomaly detection system is typically achieved through comparison with a set of “ground truth” events. However, there simply is no defined set of ground truth events in our context here. We instead evaluate Argus through comparison against two independent data sources:

- anomaly events detected by using RTTs actively measured from Keynote[8] agents

- authoritative DNS server change events provided by the CDN service team

In both situations, Argus would be expected to identify a broader set of events. However, Argus should also be able to identify the set of events detected via the other mechanisms. Figure 4.18 shows a concrete example of an event (on AS path "3356 6079") reported by Argus. Note each plot has a blank area around 16:00 April 8th 2010 as there was not RTT measurement for a hour. We focus on identifying how many of the events from these two data sources are reported by Argus, how many of them are missed by Argus and investigate the reasons for the differences.

In the remainder of this section, for each data source, we first describe the data source and then outline the evaluation results. All timestamps in the section are in GMT. Argus runs in fixed-length time bin mode with bin size as 10 minutes based on the RTT measurements passively collected at the North-East CDN node.

4.1.3.1 Comparison With Keynote Anomaly Events

Keynote [8] monitors service quality through active injection of probe packets. Each Keynote [8] agent machine has 3 unique IPs in the same /24 subnet, and each IP periodically sends probe packets to CDN servers in order to measure end-to-end performance. For the North-East CDN node, Keynote has six agents from different geo-locations and connected to different ISPs as shown in Table 4.1.3.1. While the keynote agent tracks a broad set of metrics, we focus here only on the initial connection time as this provides a direct comparison with the passively measured RTTs in Argus. The Keynote "initial connection time metric measures the RTT between the keynote agent and the CDN server by calculating the time difference between the first SYN (from the Keynote agent to the CDN server) and the SYN+ACK (from the CDN server back to the keynote agent) during the TCP handshake.

Note that Keynote does not directly provide the anomaly data. Therefore, for each of the six Keynote agents (each with 3 unique IP addresses in the same /24 subnet), we

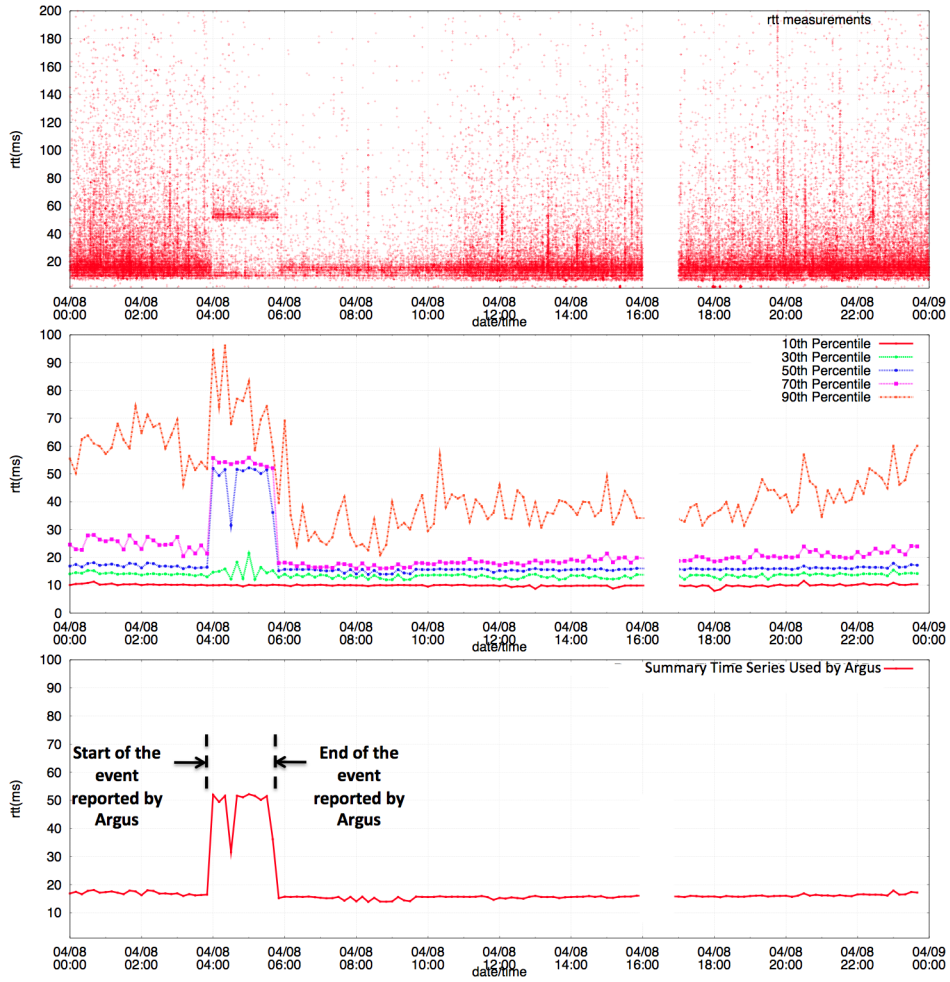


Figure 4.18: An event reported on AS path "3356 6079" around 4:00 April 8th 2010 by Argus. The top scatter plot shows the raw RTT measurements on that AS path. The plot in the middle shows the 10th, 30th, 50th, 70th and 90th percentile time series. The bottom one shows the event detected by Argus based on the summary (50th percentile) time series.

Table 4.2: Locations of six Keynote agents

Agent ID	Contry	State	City	ISP
1	US	MA	Boston	Sprint
2	US	MA	Boston	Verizon
3	US	CT	Hartford	Same ISP
4	CA	QC	Montreal	Peer1
5	US	NY	New York	Cogent
6	US	NY	New York	Sprint

first collect the initial connection time over a two-month period (from 1st July 2010 to 31st August 2010), and group them into 10-minute bins. Then we run the Shadow Holt-Winters algorithm to detect anomaly events (called **Keynote anomaly events** in the rest of the section) for comparison with the anomalies detected by Argus for the same time period and with the same bin mode and size.

Results:

There are totally 310 anomaly events from all six Keynote agents during the two-month period. We compare Keynote anomaly events with the results regarding their corresponding /24 subnet reported by Argus, and classify the results as follows.

- **(i) Match:** Argus reported an event on the same subnet that is temporally overlapped with the Keynote anomaly event.
- **(ii) Miss:** Argus didn't report anything for the same subnet around the period of the Keynote anomaly event.
- **(iii) More:** Argus reports anomalies about a Keynote agent's subnet, but there are no temporally overlapped Keynote anomalies event.

As shown in Table 4.1.3.1, Argus in total successfully detected 91% (281 events) out of 310 Keynote anomaly events, missed 29 events (9%), and detected 51 more anomalies. As also shown in the table, Argus observes RTT for more IPs in the subnet covering each Keynote agent subnet (each has 3 IPs). For example, in Keynote agent 2's subnet, except the 3 IPs used by Keynote for probing, another 10 IPs also visited the North-East CDN node during the two-month period. As a result, even though an anomaly event is detected using keynote measurements from the 3 IPs, it may not show up in Argus as a subnet anomaly by looking at the RTT measurements from all 7 IPs. This is the expected difference between Argus and active measurement-based anomalies. Coverage of more IPs can detect anomalies not directly experienced by the probing IPs, and

also anomalies experienced only by the probing IPs might not contribute enough to the subnet to have subnet level anomalies.

Table 4.3: Comparison with Keynote anomaly events

Agent	# IPs from its subnet seen by Argus	# Keynote Anomaly Events	# Match	# Miss	# More
1	6	44	38	6	0
2	13	93	85	8	20
3	5	32	29	3	1
4	7	49	45	4	3
5	6	72	67	5	4
6	15	20	17	3	23

4.1.3.2 Comparison With Authoritative DNS Server Change Events

We first describe how CDN node assignment works in the tier-1 ISP and then how the authoritative DNS server change helps with our evaluation.

The tier-1 ISP CDN's authoritative DNS server, when receiving a DNS query from the client IP's local DNS server, responds with a CDN server address closet to the client's local DNS server, with the hope that this server address is also closet to the client IPs. As with most of DNS-based CDNs [1, 9], this approach assumes that the network location of a client IP can be approximated by that of its local DNS server since the authoritative server can see the local DNS server address, but not the client IP's address.

The tier-1 ISP CDN uses anycast IP addresses for their authoritative DNS servers (i.e.,they have the same IP addresses) located at many PoPs in the ISP network. Therefore, a local DNS server's DNS query is naturally routed to the closest authoritative DNS server. The CDN service team monitors the local DNS server addresses that queried each authoritative DNS server in their daily operations. Since the authoritative DNS servers are located near the edge of the ISP network, a change in the authoritative DNS server 'hit' by a local DNS server is indicative of the routing from the local DNS server to the authoritative DNS server anycast address has changed its ingress point to the tier-1

ISP. Furthermore because the tier-1 ISP announces the same BGP path attributes for the anycast prefix and the CDN node prefix, the routing path from local DNS server (and also the client IPs) to the assigned CDN node might also (but not always) change its ingress point to the tier-1 ISP.

For example, if a local DNS server that was 'hitting' an authoritative DNS server in Boston PoP starts hitting an authoritative DNS server in Chicago PoP, it is likely the client IPs that local DNS server serves now enter the ISP network via Chicago PoP instead of Boston PoP, to reach the North-East CDN node and result in a longer RTT. Note that the returning path from the North-East node to the client IP might or might not change since sometimes the routing is asymmetric, but when the routing of this direction also changes, e.g., going through Chicago POP as well, the RTT increase will be even larger. In other words,a local DNS hit change at the authoritative servers is a good (although not perfect) indication of the RTT increase experienced by the client IPs, and we obtained a list of such change events from 1st April 2010 to 15th April 2010 for all the local DNS IPs that should be assigned to the North-East CDN node according to the global assignment table. Note that this list was manually compiled and may not be comprehensive.

Results:

Using these labelled DNS events as an independent data source, we evaluate the accuracy of Argus by running it over the same period. We first extract the AS path that the ISP took reach the local DNS server from each labeled event in the list and then classify each labeled event into three categories based the output of Argus.

- **(i) Match:** Argus reported an event on the same AS path that is temporally overlapped with labelled event, as shown in the top plot of Figure 4.19. There is a labelled event from 3:45am to 6:00am on 8th April 2010 to for a local DNS server that uses AS path "3356 6079" to reach the ISP. The local DNS server was routed

to an authoritative DNS server close to the North-East CDN node before 3:45am on 8th April 2010 and after that it was routed to another authoritative DNS server several hundred miles away from the North-East CDN node till 6:00am on 8th April 2010. It is a "match" as Argus reported an event around the same period of the labeled event.

- **(ii) No Data:** As there were no passive RTT measurements on the AS path around the period of the labelled event, Argus didn't report anything. The middle plot in Figure 4.19 shows an example of "No Data" labelled event as there is not measurement data around its period. That means during the period of labeled event, no traffic is seen on this AS path. We are still working with CDN service team to figure out the reason of this while it is not the focus of this paper. We don't worry about this case as Argus did't report anything simply due to lack of data.

- **(iii) Miss:** There were passive RTT measurements on the AS path around the period of the labelled event but Argus didn't report anything. The bottom plot in Figure 4.19 shows an example of "Miss" labelled event. This case worries us more as Argus was silent when the clients on this AS path entered the ISP from an new ingress city that much farther to the North-East CDN node compared to the old ingress. One plausible explanation is that only the DNS server experienced a routing change and the end-users were unaffected.

The duration of the 68 labeled events in the list varies from 15 minutes to 5 hours. We show the evaluation results by breaking down the 68 events into 3 classes according their durations. As you can see from table 4.1.3.2, out of 68 labeled events, Argus successfully detected 72% (49 events) and missed 13%. The rest 15 % is hard to decide due to lack of data. In addition, Argus tends to miss more short ($15m \leq D \leq 1h$) labeled events.

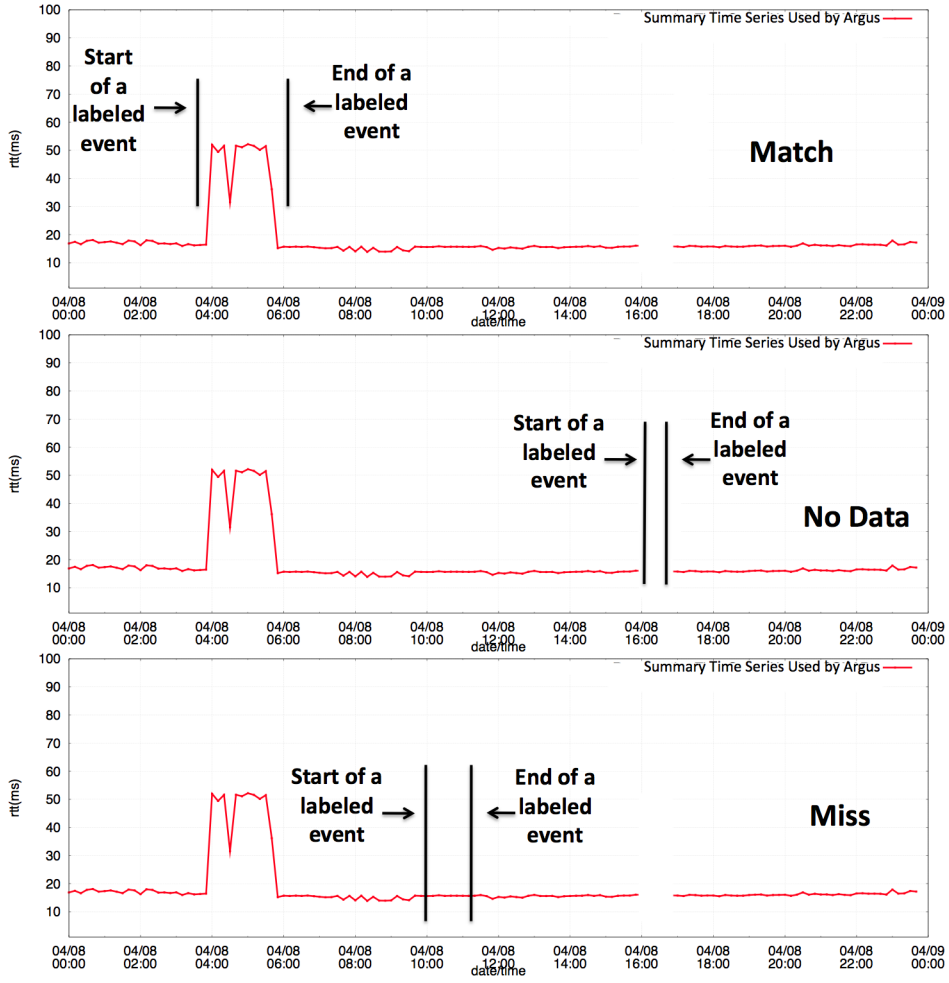


Figure 4.19: Examples of three categories of labeled DNS events

Table 4.4: Evaluation results using authoritative DNS server change events

Duration (D)	# Labelled Events	# Match	# No Data	# Miss
$15m \leq D \leq 1h$	24	16	2	6
$1h < D \leq 2h$	32	25	5	2
$D > 2h$	12	8	3	1

4.1.4 Operational Experience

In this section, we present two representative case examples from our operational experience by running the Argus over several months with the CDN service. This ISP is referred to as "local ISP" in the remainder of this section. All timestamps in the section are in GMT. The following examples demonstrate the effectiveness of Argus for detecting and localizing service events in a CDN service. Both case examples are based on the RTT measurements passively collected at the North-East CDN node by using the fixed-length bin mode with a bin size of 10 minutes.

- **Permanent RTT Level Shift Caused by BGP Next-hop Changes**

  Argus reported an 12-hour event on AS path"3561 7922 7015" [2] from 11pm on 7th May 2010 to 11am on 8th May 2010. We were interested in looking into this event as it ranked first among all events from 1st May 2010 to 10th May 2010. AS path"3561 7922 7015" has 7 children (BGP prefixes) in total. Each child is a BGP prefix that can be reached via this AS path from the perspective of the local ISP. The problem is localized to the AS path level because all of these 7 BGP prefixes were showing the same behavior as the AS path around the same time. The highest ranking score of this event comes from three aspects. This event lasted for a long period with a high "confidence" (several hundred unique end-users) and a significant "severity"(2 to 4 standard deviations from predicted value).

  The top left plot in Figure 4.20 shows the percentile time series plot for AS path and the start and end (highlighted using two vertical lines) of the event detected by Argus. We can see clearly that there was a RTT level shift started around 3:30am

[2]This is the reverse AS path that CDN nodes use to reach end-users.

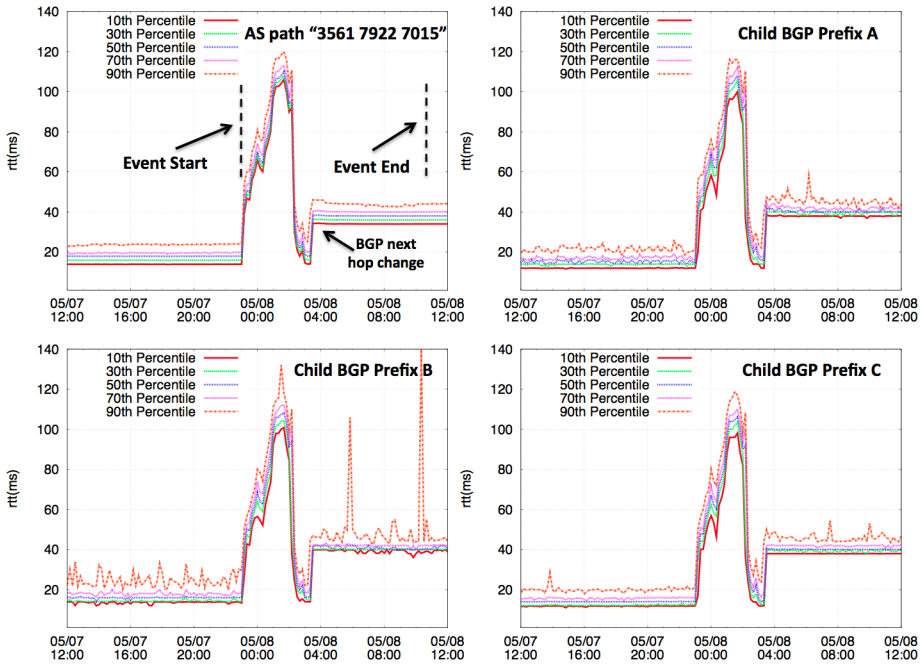


Figure 4.20: Percentile time series for AS path "3561 7922 7015" and its three child BGP prefixes. The reported 12-hour event on the AS path is marked using two vertical lines.

on 8th May 2010. It is a permanent RTT level shift as it didn't change back to the previous baseline. In Argus, the maximum length of an event is 12 hours and Argus will adapt to the new baseline if an event lasts longer than 12 hours. The other three plots of Figure 4.20 show three child BGP prefixes under AS path "3561 7922 7015" in the hierarchy. Due to space limitations, we omit the other 4 children BGP prefixes that exhibited exact the same behavior as the three shown in Figure 4.20.

According to the BGP updates from a route reflector that is co-located with the North-East CDN node, we determined that the BGP next-hop to reach all of the 7 BGP prefixes changed at 3:28am on 8th May 2010 from the same city as the North-East CDN node to another city 1,000 miles away from it. Thus, BGP next-hop change is very likely to be the reason for the RTT level shift illustrated in Figure 4.20. The instability before the level shift may be indicative of network

instabilities that caused the BGP next-hop change.

- **Temporary RTT Level Shift Related to Remote ASes**

  Among all events in March 2010 reported by Argus, the top ranked event is on AS path "7922 7015" from 4:10am to 5:50pm on 21st March 2010. This event is so significant as it effected all 65 BGP prefixes on the AS path "7922 7015" for more than 10 hours. The top plot in Figure shows the percentile time series for AS path "7922 7015" and the event reported by Argus is marked by the two vertical lines.

  More interestingly, Argus didn't report anything for the two other AS paths "7922 33287" and "7922 33657" that share the same *next-hop AS* and *egress router* with the problematic AS path "7922 7015" from the viewpoint of the North-East CDN node. The two plots at the bottom in Figure depict that the corresponding percentile time series for AS paths "7922 33287" and "7922 33657". Even though these two AS paths didn't have as many RTT measurements as the problematic AS path, we can still infer that the RTT level shift might be caused by something outside the local ISP (in remote ASes 7922 or 7015). The limitation of the above inference is that the current hierarchy (Figure 4.13) doesnt incorporate forward AS path that end-users use to reach CDN nodes. For example, suppose AS 7015 is multi-homed to AS 7922 and another AS X. If AS X is used for end-users in AS 7015 to reach the North-East CDN node. Then the problem may be in AS X. No information in Argus is able to identify this possible cause. Incorporating forward path information into Argus forms part of our future work.

4.2 G-RCA: Identifying the Root Causes of Service Anomaly Events

Given these new challenges in managing the quality of Internet-based service, traditional fault diagnoses and root cause analysis (RCA) systems [24, 32, 58, 59, 6, 7, 4] that

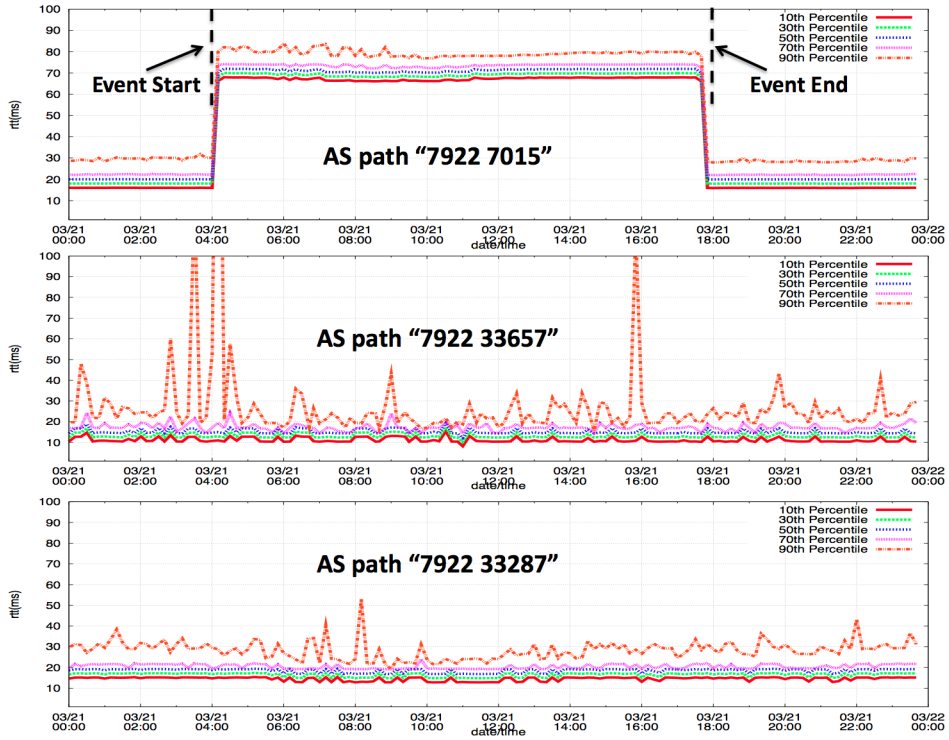


Figure 4.21: Percentile time series for AS path “7922 7015” and other 2 AS paths that share the same *next-hop AS* and *egress router*. The period of this plot is the entire day of 21st March 2010.

network operators have relied on are reaching their limits for the following five reasons.

First, the narrow view provided by the per network element perspective of the traditional systems tends to miss rather complicated **service dependency relationships**. For example, the quality of a VOIP call across the ISP network depends on the status (congestion level, bit error rate, etc.) of the routers and links along the network path carrying the traffic, which is dynamically determined based on the link weights at the time. In another example, the health of a BGP session connecting to a peer router depends on the route processor resource on both routers and the layer-2 line protocol status between them (with complicated timer/protocol interactions), which in turn depends on the condition of the layer-1 (e.g., a SONET ring) network in between. Capturing such service dependency relationships is vital for identifying root causes of service quality issues.

Second, traditional information gathering processes (such as running *traceroute* or invoking *show* command on routers) that are effective at diagnosing problems for large on-going service anomaly events are unable to cope with minor and transient service disruptions. RCA for transient failures should only rely on proactively collected data.

Third, achieving ultra-high service quality requires going beyond break-and-fix operation and single-event troubleshooting. It is critical to understand the root causes of a large number of service anomaly events and extract actionable information from the identified root causes in the aggregate. For example, when analyzing sporadic high packet loss events detected by Argus between users and CDN servers, one should examine these high packet loss events over an extended period (e.g., a month) and diagnose their root causes. Should link congestion be determined to be the primary root cause, capacity augmentation is needed along the corresponding network path. Alternatively, if packet losses are found to be largely due to intra-domain routing re-convergence, deploying technologies such as MPLS fast reroute becomes a priority.

Fourthly, various services (e.g.: CDN, VOIP, IPTV, Mobility, BGP and MVPN)

provided by the same service provider typically share the same underlying network. Operators that manage different services often need to perform the similar (if not the same) steps for root cause analysis. Therefore, it is highly desirable to have a generic root cause analysis infrastructure for various services instead of one root cause analysis system per service.

Finally, as new technologies (e.g., MPLS TE) and new devices (e.g., OC768 line cards) are introduced into the underlying networks at a fast rate, service providers often have to learn through experience about service-impacting issues. Should unexpected failure modes or performance impairments occur, operators need to act quickly to understand the problem, diagnose the root cause(s), and eliminate or mitigate the failure mode to improve service quality. In such an ever-changing network and service environment, domain knowledge and operator's experience may become insufficient. RCA systems that solely rely on expert input can fail to capture unexpected service dependencies, which unfortunately are not unusual in practice due to the variety of hardware/software errors and configuration mistakes that can occur. An root cause analysis system should allow rapid instantiation of new RCA tasks based on existing expert knowledge as well as flexible data exploration and data mining capability to improve operators' domain knowledge and understanding over time.

In this chapter, we introduce our Generic Root Cause Analysis platform (G-RCA) that is designed to address the above limits. A key feature of G-RCA is a comprehensive service dependency model that includes network topological and cross-layer relationships, protocol interactions, and routing and control plane dependencies. Thus, network operators can look for undesirable network conditions that are potentially **related** to service anomaly events without specifying the details of the topology and cross-layer relationships, the protocol interactions, or routing dependencies.

We also ensure that the service dependency relationships in G-RCA can be deter-

mined using only data that is proactively collected. For example, network paths can be computed from BGP and OSPF route-monitoring data, as opposed to requiring multiple *traceroutes*.

We implemented G-RCA for a service provider that manages many different services. Our design decomposes the RCA process into data collection for *symptom event* (user-view service degradataions) and *diagnostic event* (network-view measurements), temporal and spatial event correlation, and reasoning and inference logic. Here, symptom events are the service problems to be analyzed, and diagnostic events refer to the evidence of a potential root cause. We define a simple yet flexible rule specification language that allows operators to quickly customize G-RCA into different RCA tools as new problems need to be investigated and understood. We integrate into G-RCA data trending, manual data exploration, and statistical correlation mining capabilities that are tailored for service quality management. G-RCA has proven to be a highly effective root cause analysis platform in several different applications. In particular, using the G-RCA platform network operators are able to quickly investigate new service problems, uncover unexpected service impacts, and quantify the scale and trend of different factors contributing to service performance issues.

4.2.1 G-RCA Architecture

This section presents our design of the Generic Root Cause Analysis (G-RCA) platform for various services built on top of the same network. We focus on the following aspects of G-RCA: (1) data collection and management (2) the service dependency model (3) spatial-temporal correlation (4) reasoning logic and (5) domain knowledge building. The architecture of G-RCA is shown in Figure 4.22.

G- RCA obtains the symptom events of interest and the set of diagnostic events (network-view) via the data collector. For a given symptom event at service level, G-RCA uses service specific diagnosis graph to identify the relevant diagnostic events.

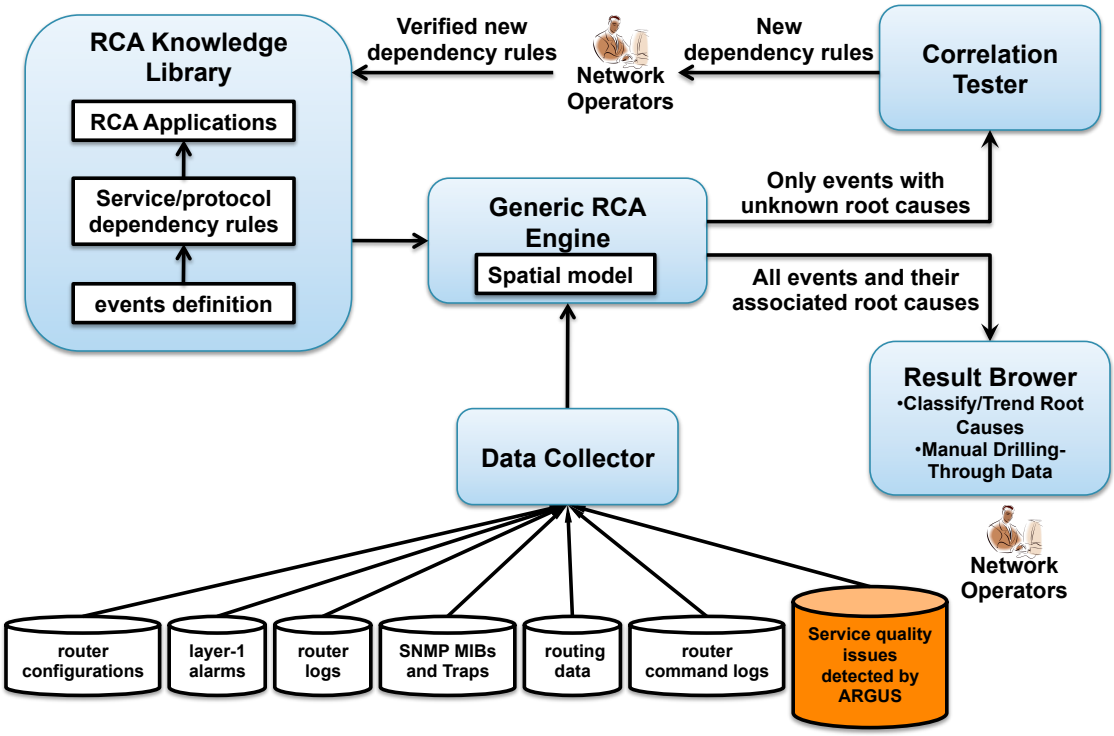


Figure 4.22: G-RCA Architecture

Specifically, G-RCA determines where and when to look for diagnostic events based on the location and time of the symptom event. Once these diagnostic events are identified, G-RCA then applies reasoning logics to examine all of the different diagnostic events observed for the given symptom to identify the most likely explanation(s) of the symptom event. Operators ususally start with an inaccurate and incomplete diagnosis graph and G-RCA allows them to gradually acquire new knowledge or learn unexpected network behaviors to improve the diagnosis graph.

Overall, there are two types of scenarios in which G-RCA is frequently applied:
(i) troubleshooting individual on-going service quality issues: These issues may currently be impacting customers, in which case service operators are under great pressure to quickly go through a large number of alarms, logs, and measurement data and identify the root cause.

(ii) investigation past behaviors in order to improve future service quality: Besides critical outages and major degradations, there are many cases of non-critical undesirable service quality. Some are very short in duration, such as a low throughput issue that clears itself before a human operator can get to it. These "small" service impairments can add up becoming a chronic issue, causing customer dissatisfaction. It is critical for operators to keep track of a potentially overwhelming number of "small" service quality issues and analyze their root causes so operators can prioritize efforts to improve the overall service quality. For example, if link congestion is determined as the primary root cause for low throughtput in a CDN service, capacity should be added to the corresponding network path. Or if long round-trip time is found to be largely due to intra-domain routing re-convergence, perhaps priority should be put on deploying technologies such as MPLS fast reroute.

4.2.1.1 Data Collection and Management

Understanding service quality issues often requires an integrated view of different parts of the network. As mentioned earlier, G-RCA relies on a wide range of *proactively* collected information containing alarms, logs and performance measurement data from various network management systems. As simple as it sounds, there are tremendous instrumentation challenges for data management. Moreover, these data come from many devices and network management systems provided by different vendors, all reporting different statistics, from different time zones, and at varying intervals. The same device may be referenced in different ways by different systems or at different network layers (by a circuit identifier, an IP address, or an interface name). The timestamps can be a mixture of local-time (depending on the time-zone of the device), network-time as defined by the service provider, and GMT. To facilitate trouble-shooting service quality issues, one has to look across data sources efficiently. Hence, in G-RCA, the first optimization is on data integration – G-RCA's Data Collector pulls all the data together, normalizes them so that they can be readily correlated, and stores them in database tables in real-time. The normalization across naming conventions, time zones and identifiers takes place as data is ingested into the Data Collector. This hides the data processing complexity from the remaining G-RCA components and eliminates the need for the operators to be painfully aware of the original data source details when correlating data. The data sources in our implementation of G-RCA include layer-1 alarms, router logs, SNMP MIBs and traps, routing data, router command logs and router configurations. Currently the Data Collector is collecting around 600 data sources in total and the daily data volume is about 7 TB. Using two data sources (syslog and SNMP) as examples, the daily numbers of new records for them are tens of millions and hundreds of millions respectively.

Expectedly, raw data are typically difficult to work with. In G-RCA, we introduce

Table 4.5: Common Event Definitions for the Service Provider's Network

Event Name	Event Description	Location Type	Data Source
Router reboot	router was rebooted	router	syslog
CPU high (average)	$\geq$ 80% average utilization in 5-minute intervals	router	SNMP
CPU high (spike)	$\geq$ 90% average utilization over the past 5 seconds	router	syslog
Interface down	LINK-3-UPDOWN msg	interface	syslog
Interface up	LINK-3-UPDOWN msg	interface	syslog
Interface flap	LINK-3-UPDOWN msg	interface	syslog
Line protocol down	LINEPROTO-5-UPDOWN msg	interface	syslog
Line protocol up	LINEPROTO-5-UPDOWN msg	interface	syslog
Line protocol flap	LINEPROTO-5-UPDOWN msg	interface	syslog
Regular optical mesh network restoration	regular restoration events in layer-1 optical mesh network	layer-1 device	layer-1 device log
Fast optical mesh network restoration	fast restoration events in layer-1 optical mesh network	layer-1 device	layer-1 device log
SONET restoration	restoration events in the layer-1 SONET network	layer-1 device	layer-1 device log
Link congestion alarm	$\geq$ 80% link utilization in 5-minute intervals	interface	SNMP
Link loss alarm	$\geq$ 100 corrupted packets in 5-minute intervals	interface	SNMP
OSPF re-convergence event	link weight update in OSPF	interface	OSPF monitor
Router Cost In/Out	Router cost in/out inferred from link weight changes	router	OSPF monitor
Link Cost Out/Down	Link cost out or link down inferred from link weight changes	interface	OSPF monitor
Link Cost In/Up	Link cost in or link up inferred from link weight changes	interface	OSPF monitor
Command to Cost In Links	Command typed by operators to cost in links	interface	TACACS
Command to Cost Out Links	Command typed by operators to cost out links	interface	TACACS
BGP egress change	BGP next hop to some external prefix changed	ingress:destination	BGP monitor
In-network delay increase	delay increase between two PoPs	ingress:egress	performance monitor
In-network loss increase	loss increase between two PoPs	ingress:egress	performance monitor
In-network throughput drop	throughput drop between two PoPs	ingress:egress	performance monitor

the notion of **event** – an event is a *signature* that captures a particular type of network condition. We associate a *location type* with each event as it provides a key piece of information required for modeling service dependency (in 4.2.1.2). Figure 4.23 shows the location types that can be associated with a single event. A type of event can be extracted from raw input data through a parsing script, a database query, or some more sophisticated processing such as Argus. Specifically, an **event definition** in G-RCA is a tuple consisting of (*event-name, location type, retrieval process, additional descriptive information*), in which the retrieval process points to the actual scripts/queries needed to obtain the matching event instances.

Each **event instance** consists of an (*event-name, event start-time, event end-time, event location, additional info*). For example, the event definition (*link-congestion, interface,*

myscript) indicates that the G-RCA Engine will use *myscript* to query SNMP traffic counter data to identify links that are nearly 100% utilized, and output event in-

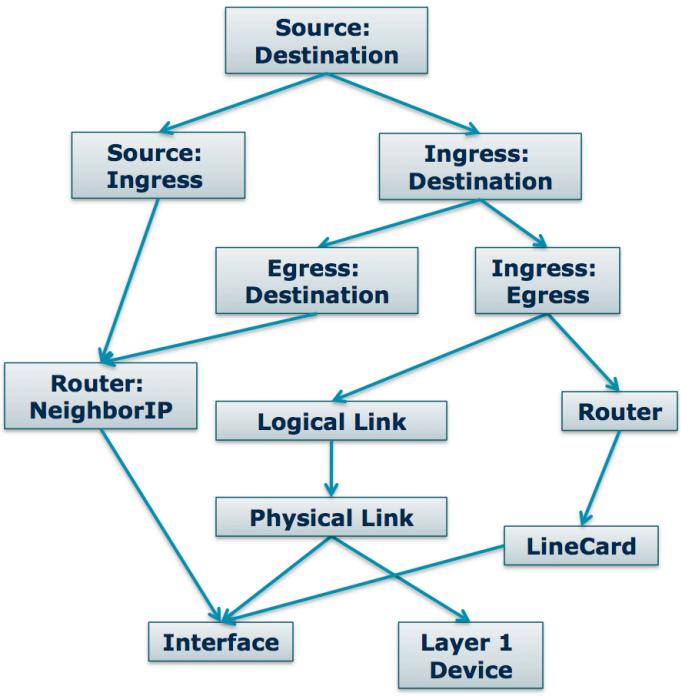


Figure 4.23: Spatial Model: Location Types and Mapping

stances with location type *interface*. A corresponding event instance example is (*link-congestion*, 2010-01-01 12:30:00, 2010-01-01 12:35:00, *newyork-router1:serial-interface0*).

In order for service operators to quickly analyze new service problems, G-RCA defines and implements a wide range of commonly used event signatures. These are included in the RCA Knowledge Library. For example, various services running on the same network may all be interested in identifying link congestion events. Furthermore, there can be multiple signatures defined for the same network conditions. For example, in G-RCA Knowledge Library, a link congestion event is defined as either a near 100% link utilization in the SNMP traffic counter or a high number of overflow packets in the SNMP interface MIB. The number of overflow packets is a more reliable metric to reflect packet loss as the impact of link utilization on packet loss depends the network type. In the service provider's network, as traffic is highly aggregated, there is rarely any packet loss (overflow) even for links with 5-minute average utilizations around 90%

level. However, it can be expected that in the access network where traffic is more bursty, packet loss can occur with significant lower link utilization measure, hence impacting TCP performance. Service operators can pick the event definitions that are best suited for the service quality issues under investigation.

Table 4.2.1.1 lists some common events in G-RCA for the service provider's network. Note that any event defined in the Knowledge Library can be redefined by an application. For example, the event "link congestion alarm" in CDN low-throughput-analysis can be easily redefined as "$\geq$ 90% link utilization in the SNMP traffic counter" when needed. At the time of writing this dissertation, there are more than 200 common events that are defined in the G-RCA Knowledge Library.

4.2.1.2 Service Dependency Model

The key to identifying root causes of service quality issues is to understand the service dependency relationship between a user-view service problem and the underlying network devices and protocols supporting the service. G-RCA uses the model in Figure 4.23 to capture such dependencies. Though it appears simple, this model actually incorporates topological information (e.g., physical link connecting two different routers), cross-layer dependency (e.g., layer-1 devices supporting layer-3 links), logical and physical device association (which requires router configuration), and dynamic routing (e.g., BGP and OSPF routing in determining the path between source and destination).

The service dependency abstraction is the most powerful component of G-RCA. By specifying the type of service problem (e.g, Ingress:Destination [3]), G-RCA can automatically expand the service dependency to include all network elements that are associated with the service. However, realizing this model in practice is quite challenging. One

[3]In the paper, the notation "A:B" denotes all locations between points A and B.

crucial aspect to the dependency model is that the relationship is time varying – egress points to a destination network can change upon BGP updates; network paths can change as operators modify link weights; logical to physical mappings can change with configuration changes; even physical connectivity can change over time. Associating the right network elements with a service event at a given time in history requires reconstructing the "network condition" at the time. G-RCA tackles this by implementing a range of sophisticated conversion utilities as follows:

(1) A source and destination pair where both are outside the service provider's network is first mapped to "Source:Ingress router" and "Ingress router:Destination". This mapping is typically done by looking at the traffic sampling data (e.g.: NetFlow) to figure the ingress router and sometimes needs external mapping information. For example, if the traffic enters the service provider's network from a data center that is also under the service provider's control, the mapping can be easily obtained by looking at the configuration (e.g. the list of ingress routers that connect to the data center). Then in order to map from "Ingress router:Destination" to "Ingress router:Egress router" and "Egress router:Destination", G-RCA looks up historical data of BGP tables to find out the longest prefix match and the network egress point for the destination. Note that BGP routing changes are typically not available at all ingress routers and only those changes at the BGP route-reflectors are available. In such cases, approximation is needed. The reflectors which feed the ingress router with BGP updates are extracted from the daily archive of router configurations; the BGP decision process at the ingress router is emulated based on the BGP route changes from its reflectors as well as the OSPF distance to available egress routers, and one best egress router is picked based on BGP best path selection.

(2) Both "Source:Ingress" and "Egress:Destination" can be mapped to a pair of access router and neighbor's IP according to router configuration. The mapping from

"Router: NeighborIP" to "Interface" can also be acquired by looking at the router configuration. This is particularly useful for diagnosing some protocol (e.g. BGP) events with a neighbor IP that typically belongs to a router outside the service provider's network.

(3) Given the ingress router to egress router pair, the logical link or router level path between them can be computed via an OSPF [42] routing simulation based on network-wide link weights from route-monitoring tools such as OSPFMon [53] (which listens to the flooded messages in OSPF). In the case of ECMP (Equal Cost Multipath), all network elements along all paths will be considered.

(4) A point-to-point logical link can be associated with its attached routers by matching the IP addresses of the logical interface to a $/30$ network.

(5) A logical link may be mapped to more than one physical link for redundancy and capacity purposes by using techniques such as SONET APS (Automatic Protection Switching)[12] and Multilink PPP bundle [10]. This mapping can be obtained from the router configuration.

(6) G-RCA parses daily router configuration snapshots to infer that a router consists of a set of line-cards, which comprises multiple interfaces.

(7) An external database that keeps track of layer 1 inventory provides G-RCA with the mapping from physical links to all the layer-1 devices in between.

These conversion utilities are specific to the service provider's network that we work with. However, we believe similar capability can be established when applying G-RCA to other networks.

4.2.1.3 Spatial-temporal Correlation

The most commonly asked question when service operators perform root cause analysis tasks is *what happened in the network at the time that can be related to the service problem?* Breaking this question into more rigid and programable logic, G-RCA defines

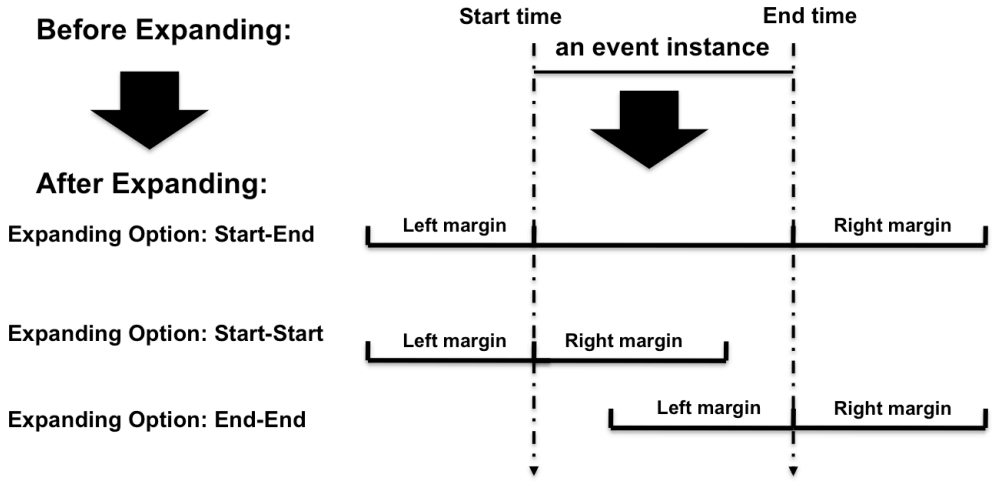


Figure 4.24: Three Different Expanding Options

a temporal and spatial join rule as follows.

The simple concept "at the same time" can be quite entangled with each application. First, there are typically various delay timers or expiration timers in each network protocol. Cause and effect rarely follows one another instantly. Secondly, there is always inaccuracy and uncertainty in the timing of network measurements. For example, a router CPU measurement in a five-minute interval (via SNMP) may indicate a CPU overload condition within that interval, but not any more precisely. G-RCA captures the above by defining a time-window to allow symptom event and diagnostic event to be joined (or "at the same time").

Specifically, each temporal joining rule consists of six parameters: the left expansion margin X, right margin Y and an expanding option (Start/End, Start/Start, or End/End) for each of the symptom event and diagnostic event. The margin values can be positive or negative in seconds, indicating forward-shift or backward-shift in time. The expanding option (Figure 4.24) specifies how the time window of an event is expanded. G-RCA determines a joint event pair when their expanded time windows overlap. Note typically operators decide the parameters based on their domain knowledge.

For example, consider a diagnosis event "Interface flap" (Start/End, X=5, Y=5) to

be correlated with a symptom event "eBGP flap" (Start/Start, X=180, Y=5). Here 180 is used to model the cause-effect delay between "eBGP flap" and "Interface flap". 180-seconds is the default setting for the eBGP hold timer. In other words, "eBGP flap" is likely to occur 180s after an "Interface flap" event takes place. 5-seconds is used to model the inaccurate time-stamps in syslog messages. For an "eBGP flap" starting at time 1000 and ending at time 2000, its expanded time interval is [820, 1005]. For an "Interface flap" starting at time 900 and ending at time 901, its expanded time interval is [895, 906]. The two event instances are considered temporally joined since the two time intervals overlap.

For a diagnostic event to be correlated with a symptom event spatially, G-RCA defines the spatial joining rule that consists of three parts: 1) symptom event location type, 2) diagnostic event location type, and 3) *joining level*. The first two follow directly from the event definitions and must be one of the location types specified in Figure 4.23. The *joining level* is used to link symptom event locations with diagnostic event locations. G-RCA automatically converts the locations of symptom and diagnostic events into the same "join level" location so that they can be directly compare. For example, the symptom is an end-to-end packet loss event that has a location type as "Source: Destination". The diagnostic event is a CPU overload event that has a location type as "Router". The joining level can be "Backbone Router-level Path", which means only CPU overload event on the router along the backbone path (not all the routers on the backbone) will be joining with this end-to-end packet loss event. As another example, consider the symptom event of "packet loss on the uplink of an access router" [4] with location type "Interface". Consider the diagnostic event of "packet loss on an service provider's access router customer-facing interface" also with location type "Interface". If joining

[4] An uplink is the link that connect an access router to a core network router.

level is "Router", two event instances are spatially joined only if they take place on the same router. The Generic RCA Engine evaluates the built-in spatial model that ensures the symptom and diagnostic events are related according to the spatial joining rule specified. With this capability, when building a new application from G-RCA, operators are alleviated from the details of routing information, network topologies, router configurations, and cross layer dependency.

The above defines the temporal and spatial relationship between a pair of symptom and diagnostic events. For any RCA application, typically many diagnostic signatures are investigated as potential root causes. We model this using a **diagnosis graph** — an example of diagnosis graph is shown in Figure 4.27. We refer to each edge in the diagnosis graph (the pair of symptom and diagnosis events and their temporal and spatial joining rules) as **"diagnosis rule"**. Given a diagnosis graph (for a specific SQM application), G-RCA evaluates the time and location conditions and collected data according to the data *retrieval process* in the event definition to determine the presence or the absence of diagnostic signature events.

Similar to the event definition library for frequently used event signatures, G-RCA also includes a library of diagnosis rules. Table 4.2.1.3 lists some of the commonly referenced rules in the service provider's network and some others in the case studies in the rest of this section. These are maintained in G-RCA Knowledge Library in Figure 4.22. Some statistical correlation tests such as [41] can help operators find more diagnosis rules automatically. Note even with these statistical correlation tests, domain knowledge is still indispensable to check if the identified rules are actually meaningful and to decide the right parameters for the rules . At the time of writing this dissertation, there are more than 300 common diagnosis rules that are defined in the G-RCA Knowledge Library.

Table 4.6: Common Diagnosis Rules for the Service Provider's Network

Symptom Event	Diagnostic Event
Line protocol down/up/flap	Interface down/up/flap
Interface down/up/flap	SONET restoration
Line protocol down/up/flap	SONET restoration
Interface down/up/flap	Regular optical mesh network restoration
Line protocol down/up/flap	Regular optical mesh network restoration
Interface down/up/flap	Fast optical mesh network restoration
Line protocol down/up/flap	Fast optical mesh network restoration
BGP egress change	Interface down/up/flap
BGP egress change	Line protocol down/up/flap
Edge-to-edge delay increase	BGP egress change
Edge-to-edge loss increase	BGP egress change
Edge-to-edge throughput drop	BGP egress change
Edge-to-edge delay increase	Link congestion alarm
Edge-to-edge loss increase	Link congestion alarm
Edge-to-edge throughput drop	Link congestion alarm
Edge-to-edge delay increase	OSPF re-convergence event
Edge-to-edge loss increase	OSPF re-convergence event
Edge-to-edge throughput drop	OSPF re-convergence event
Link loss alarm	Link congestion alarm
Link loss alarm	Line protocol down/up/flap
OSPF re-convergence event	Line protocol down/up/flap
OSPF re-convergence event	Interface down/up/flap
OSPF re-convergence event	Commands to Cost In/Out Links
Link Cost Out/Down	Line protocol down
Link Cost Out/Down	Interface down
Link Cost Out/Down	Command to Cost Out Links
Link Cost In/Up	Line protocol up
Link Cost In/Up	Interface up
Link Cost In/Up	Command to Cost In Links
Link congestion alarm	OSPF re-convergence event

4.2.1.4 Reasoning Logic

Once data are collected regarding the presence or absence of diagnostic signature events, the next step is to determine the root cause of the symptom events based on this "evidence". This reasoning logic can be implemented in many ways. In particular, G-RCA includes two reasoning engines: rule-based decision-tree-like reasoning and Bayesian inference.

- **Rule-based Reasoning Module:**

  In our rule-based reasoning module, we allow operators to associate a priority value for each edge in the diagnosis graph (such as in Figure 4.27). The higher the priority value, the stronger support that the operator believes the diagnostic event to be the real root cause. After temporal spatial correlation, each symptom event instance is at the root of diagnosis graph and diagnostic event instances are located at other nodes of diagnosis graph. The rule-based reasoning engine starts from the root, searches through each node (if there is a diagnostic event instance) and identify the leaf node with the maximum priority as the root cause. In the case of a tie between different leaf nodes, all of them are output as joint root causes.

  Regarding the priorities of root causes, G-RCA relies on the domain knowledge from operators. In general, the priority assignment for the root causes on the same branch is trivial, operators just need to make sure the deeper root cause has a higher priority. For example, event Interface flap and event line protocol flap can both the root cause of symptom event BGP flap. Because line protocol flap is typically caused by Interface flap, the priority for Interface flap is higher. It is more tricky to assign priorities for the unrelated root causes on different branches. For example, the priorities for "Router reboot", CPU overload and Interface flap are purely determined by operators according to their knowledge about which one is more likely to the real root cause of BGP flap. If operators aren't sure about

which root cause is more likely, they simply use the same priority for all of them. G-RCA's Result Browser allows them to exam all potential root causes ordered by the priority.

- **Bayesian Inference:**

  An alternative to the classic priority and rule-based reasoning is the Bayesian inference technique, which has proven successful in many networking applications [22, 62, 23, 36, 14]. While it is considerably more complex in parameter setting (a drawback based on operators feedback), including Bayesian inference in G-RCA provides several key advantages. For example, it naturally models unobservable root cause conditions (i.e., those that do not have strong observable evidence or signatures) and captures the uncertainty of diagnostic evidence. Using Bayesian inference also allows multiple symptom events to be examined together and deduces a common root cause (or causes) for them – this typically achieves better accuracy than when each individual symptom event is diagnosed separately.

  We model the root cause analysis problem using a Naive Bayesian Classifier [11], in which the potential root causes are the *classes*, and the presence or absence of the diagnostic evidence as well as the symptom events themselves are the *features*. The likelihood for a particular root cause r given the features observed $(e_1, ..., e_n)$ is

$$p(r|e_1, ..., e_n) = \frac{p(r)p(e_1, ..., e_n|r)}{\sum_{r \in R} p(r)p(e_1, ..., e_n|r)} \tag{4.1}$$

  where R is the set of potential root causes. Determining the root cause is to identify the one producing the following maximum *likelihood ratio*:

$$\arg\max_{r \in R} \frac{p(r|e_1, ..., e_n)}{p(\bar{r}|e_1, ..., e_n)} = \arg\max_{r \in R} \frac{p(r)}{p(\bar{r})} \times \frac{p(e_1, ..., e_n|r)}{p(e_1, ..., e_n|\bar{r})}, \tag{4.2}$$

  in which $\bar{r}$ denotes when the root cause is not r.

Suppose operators want to assess the likelihood ratio for a BGP session flap due to an overloaded router CPU. In this case, $p(r)$ is the a priori probability of an overloaded router CPU inducing BGP session timeout. And $p(e_1, ..., e_n|r)$ is the probability of the presence of evidence (such as SNMP 5-minute average CPU measurement being high, or a BGP hold-timer expiry notification observed in router syslog) under such a scenario; it is divided by $p(e_1, ..., e_n|\bar{r})$, which is the chance for the same evidence to appear when the BGP flap is due to other root causes. Hence, the first term in Eq.(4.2) quantifies how likely is the root cause without any additional information, and the second term quantifies how much confidence you gain or lose from observing or not observing the set of evidence. When the features are conditionally independent, the second term can be decoupled to $\Pi_i \frac{p(e_i|r)}{p(e_i|\bar{r})}$, in which each term quantifies the support of root cause r given evidence e_i.

The parameters (ratios: $\frac{p(r)}{p(\bar{r})}$ and $\frac{p(e_i|r)}{p(e_i|\bar{r})}$) can be difficult to configure. These can be trained from classified historical data, which we can bootstrap using the rule based reasoning from the previous subsection. Alternatively, we also define a fuzzy type of discrete values. Operators can simply specify the ratios as "Low", "Medium" and "High", which corresponds to values 2, 100, and 20000, respectively. Note that the unscaled values are likely to be fractional numbers less than one as the root causes are rare events. But from the operational point of view, it is undesirable to use fractional numbers. According to Eq.(4.2), multiplying a constant scaling factor doesn't change the final results. Thus, instead we use integers like 2, 100, and 20000. Coarse as they are, the classification results using these are quite reasonable, in that the performance of the Naive Bayesian classifier is often not sensitive to the probability parameters [50].

- **Comparison:**

Interestingly, in our operational practice, we have found that rule-based reasoning logic is often preferred over its more sophisticated counterpart – this is because (1) it is easier to configure, (2) it gives simple and direct association between the diagnosed root cause and the evidence(s) for result interpretation, and (3) it is found to be very effective in most applications that we have explored. However, there are a few cases where Bayesian inference is prefered – for example when the root cause condition is unobservable (e.g., no direct evidence can be collected).

4.2.1.5 Domain Knowledge Building

One of the important challenges in root cause analysis is that operators' domain knowledge and operational experience can be unreliable or incomplete. This implies that the specification of a diagnosis graph for a new service offered by an operator, especially the initial version, can be inaccurate and incomplete.

G-RCA assures the accuracy of diagnosis graph by using Correlation Tester (see Figure 4.22) to check each edge/rule in the graph. Specific, for each diagnosis rule, we run Correlation Tester to test the statistical correlation between symptom event and diagnostic event. Regarding Correlation Tester, G-RCA implements the statistical correlation algorithm proposed in NICE [41]. In comparison to other canonical statistical tests, NICE handles the event autocorrelation structure very well, which is commonly observed in networking event series. The diagnosis rule is only considered to be accurate when it passes the test. The idea is that the number of coincidental correlation should be bounded when examining the instances of symptom and diagnostic events in bulk. Thus, when the diagnosis rule is inaccurate it fails the test due to lack of statistical correlation between symptom event and diagnostic event. We also periodically re-test each diagnosis rule in the diagnosis graph to keep the diagnosis rules up to date.

G-RCA addresses this concern regarding incomplete diagnosis graph through iteratively using the Correlation Tester and Result Browser (see Figure 4.22). G-RCA first

allows operators to filter out the symptom events with known root causes with the root cause classification capability provided in Result Browser Secondly, operators are able to focus on the rest of symptom events by comparing with other suspected diagnostic events (regardless if they are not currently defined in the diagnosis graph) that occur at about the same time and that are spatially related to the service problem under investigation. On one hand, the second step can be done via manual drill-down and data exploration capability in Result Browser. The manual-discovered diagnosis rules need to be tested by CorrelationTest before incorporating into diagnosis graph. On the other hand, operators can also choose to run the Correlation Tester *blindly* between the symptom events without known root causes and each type of suspected diagnostic events. Note that a relation between the symptom events and one type of suspected diagnostic events might be buried in the noise if we don't take out the the symptom event with known root causes. As G-RCA emphasizes usability, the newly uncovered diagnosis rules need to be verified by operators before incorporating into diagnosis graph. For example, a large number of BGP flaps between customer routers and provider edge routers where found to be due to link flaps between the customer and provider edge routers, which typically are caused by customer activities. These BGP flaps could be easily filtered out by Result Browser and operators can concentrate on the rest of BGP flaps without know root causes. This has proven tremendously useful from operational experience. Operators can often spot the signature of overlooked root causes and add them to the diagnosis graph. Similarly, instead of focusing the symptom events with unknown root cause, one can concentrate on the symptom events with a particular type of root cause to dig out deeper root causes. For example, by only looking at the BGP flaps caused by "CPU overload", one may find the deeper root cause that results in "CPU overload".

Through iteratively using the Result Browser and Statistical Correlation Tester, operators can start with inaccurate and incomplete domain knowledge and gradually acquire

new knowledge or learn unexpected network behaviors exhibited in the network data, which can then be incorporated into the diagnosis graph.

4.2.2 G-RCA Applications

The key advantage of G-RCA is its capability to be rapidly customized for various services built on the service provider's network. We have customized G-RCA for diagnosing service quality issues in CDN, DNS, cellular service, multicast VPN and customer BGP service. In this dissertation, we focus on the following three case studies – 1) end-to-end round trip time degradations detected by Argus in a CDN service, 2)customer BGP flaps, and 3) PIM (Protocol-Independent Multicast) flaps in multicast VPN service – to demonstrate the effectiveness of G-RCA.

4.2.2.1 Root Cause Analysis for CDN Service Quality Issues Detected by Argus

- **Overview of Service Anomaly Events Detected by Argus:**

  We focus on the service quality issues in round trip time (RTT) detected by "Argus" from 20th July 2010 to 20th August 2010. Note that these service anomaly events were detected by running "Argus" in fixed-length bin mode with bin size as 3,600 seconds (1 hour) based on the RTT measurements passively collected at the North-East CDN node.

  During this one-month period, "Argus" detected 2,909 anomaly events across all user-groups in the hierarchy (Figure 4.13). Table 4.2.2.1 shows the anomaly event distribution across all user-groups. In general, the lower level user-groups are responsible for more anomaly events as they are much more than the higher level user-groups. In addition, for each type of user-group, only a small fraction (bad user-groups) are responsible for the anomaly events and generally there is no heavy hitter among the bad user-groups.

According to Table 4.2.2.1, the CDN nodes are extremely stable across the month, with no anomalies detected at their user-groups. Anomaly events localized to "Origin AS", "City", "BGP prefix" and "City+BGP prefix" are most likely attributable to events outside the local ISP. In contrast, the anomalous events localized to "Egress Router", "Nexthop AS", "AS path" are more likely to be attributable to events within the local ISP (although they could still have been caused within other ISPs).

Table 4.7: Anomaly events breakdown by User-group Type

User-group Type	# Total User-groups	# Events	# Bad User-groups
CDN Node	1	0	0
Egress Router	185	66	36
Nexthop AS	125	54	25
Origin AS	820	172	97
AS path	1055	213	119
City	1758	593	289
BGP prefix	4646	600	425
City+BGP prefix	8784	1211	851

We now we focus on the time durations of the 2,909 anomaly events. Our results clearly show that the majority of the anomalies are very short in duration, whilst long-lasting events are rare. Specifically, the events with duration 3,600 seconds are the most common and represent 90% of all anomaly events. In addition, this statement holds true for every user-group type in the hierarchy (due to space limitations, we don't discuss the details of per user-group type distribution here).

- **Customize G-RCA to Diagnose Service Anomaly Events:**

  To troubleshoot a sudden RTT degradation in the CDN detected by Argus, operators must determine whether it is caused by network or service element faults, routing changes, congestion, or significant packet error and loss conditions that can impact the delivery of the web objects from the CDN server to the web client.

Furthermore these conditions could be in the service layer (e.g., servers which host the content), somewhere in depths of the service provider's network, or even in the wilds of the Internet. With thousands of routers and servers within the service provider's responsibility, and lack of data regarding events outside the service provider's network, investigating such end-to-end service quality issues is challenging, to say the least.

The primary challenge is to identify the network and service elements involved in servicing the requests at the precise time of the service performance degradation. This is challenging to achieve during a real-time event and practically impossible to manually identify for historical events. However, G-RCA's spatial model and proactive data collection enables such determination, and is the key to providing the ability to automatically troubleshoot these service issues.

To customize G-RCA for diagnosing these service anomaly events , we defined the application specific events (Table 4.2.2.2) and diagnosis rules (dashed lines in Figure 4.25). Note that the majority of the events and rules could again be drawn from the RCA Knowledge Library. The most important application-specific event is the "CDN end-to-end RTT degradation" detected by Argus. This event indicates an increase in round trip time between users and CDN servers and is the input to the G-RCA. Each "CDN end-to-end RTT degradation" event is associated with a start time, a end time and a location defined on "source:destination" as shown in Figure 4.23. Specifically, "source" is defined by a random server in the CDN node and "destination" is defined by a random user in the user-group.

After defining the application specific events, we then need to add a few service specific diagnosis rules, which are not already included in the RCA Knowledge Library (Table 4.2.1.3). As shown in Figure 4.25, a few service specific diagnosis rules and some other rules from the RCA Knowledge Library together form a full

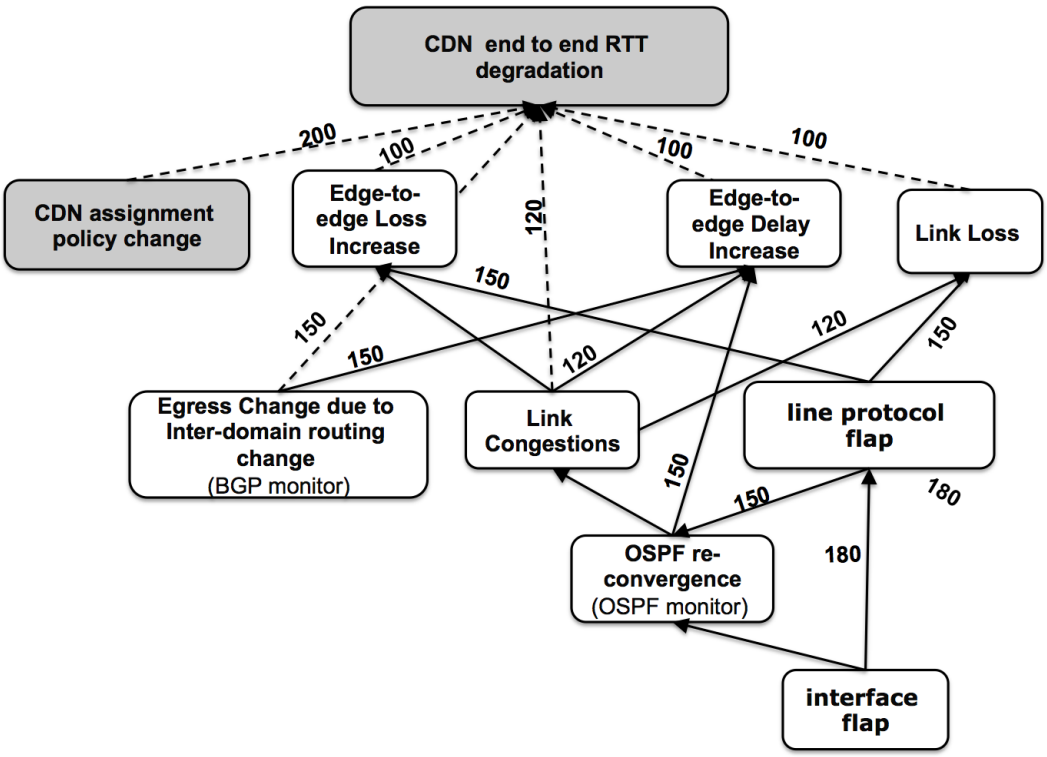


Figure 4.25: Diagnosis Graph for CDN RTT Degradation Root Cause Analysis

diagnosis graph for root cause analysis of service anomaly events in the CDN. Note that in Figure 4.25, service specific events are shown in gray boxes and service specific rules are shown with dashed lines. Priorities (numbers on the edges) for different diagnosis rules are also defined in Figure 4.25.

- **Results:**

  Table 4.2.2.1 shows the root cause breakdown generated by the Result Browser in G-RCA. At a high level, only 25.17% of the RTT degradations are identified as caused by either events that happened within our network (e.g., interface flap, link congestion and CDN assignment policy change) or from events that are visible in our network (such as BGP routing changes announced by other ISPs). For the rest (majority) of them, we did not find any evidence from inside our network, which suggests that those RTT degradations may be caused by other ISPs on the

end-to-end path.

Table 4.8: Root Cause Breakdown of End-to-end RTT Degradations

Root Cause	Percentage (%)
CDN assignment policy change	3.83
Egress Change due to Inter-domain routing change	5.71
Link Congestions	3.50
Link Loss	3.32
Interface flap	4.65
OSPF re-convergence	4.16
Outside of our network	74.83

According to the CDN service operations team, G-RCA is quite useful in finding the root causes and is much more rapid than could be achieved by Operations personnel. As an example event, G-RCA successfully determined that a given RTT degradation was caused by the failure of the peering link between our network and the neighboring networks. This failure resulted in a routing change which in turn resulted in traffic experiencing larger delays and degraded TCP performance. Although network operations team was already aware of the peering link failure and working on it, the CDN service operations team could in parallel repair service even before the network was repaired by updating the DNS tables to route impacted users to "closer" CDN nodes as measured by the new network routing. Thus, the two teams could work in parallel - with CDN performance being repaired even whilst the network issue was still being resolved. Having rapid root cause analysis through G-RCA thus enables faster intervention on customer-impacting issues and fast service improvement. The average diagnosis time per symptom event is less than 3 minutes. Most of the delay is incurred computing inter-domain (BGP) routes and intra-domain (OSPF) routes.

4.2.2.2 Root Cause Analysis for Customer BGP Flaps

- **Overview of eBGP Flaps:**

  In the second case study, we focus on customizing G-RCA to understand the root causes of eBGP [2] session flaps between customer routers (CR) and provider edge routers (PER).

  Customer networks exchange routes with the service provider's network through the eBGP session - the routes learned from the service provider's network inform the customer network how to route to locations across the Internet and other sites of the same customer; routes shared from the customer network ensure that other sites can reach the sites. If a session flaps, all routes are withdrawn and traffic is disrupted. Although relatively short (on the order of a minute), these flaps can disrupt applications. For example, VoIP sessions may be lost and financial transactions may be interrupted. We therefore aim to minimize the number of eBGP session flaps, taking actions to drive avoidable flaps permanently out of the service. The first step to achieving this is to understand the root cause of the flaps - a particularly challenging problem across a trust domain (between customer and provider networks). We achieve this using G-RCA by constructing service specific events and rules.

- **Customize G-RCA to Diagnose eBGP Flaps:**

  We start by constructing our BGP flap-specific events - those events which are not already included in the common event definitions in the RCA Knowledge Library (Table 4.2.1.1). These new service specific events are illustrated in Table 4.2.2.2. Note that there are only 3 of them, in contrast with 7 other events which we reuse from the Knowledge Library.

  After defining the service specific events, we need to add a few service specific

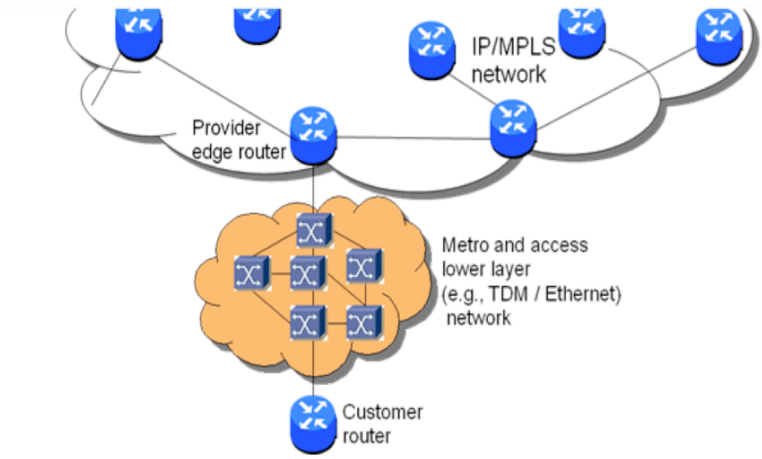


Figure 4.26: Physical connectivity between CR and PER

Table 4.9: Service Specific Events for BGP Flaps Root Cause Analysis

Event Name	Event Description	Data Source
eBGP flap	eBGP session goes down and comes up, BGP-5-ADJCHANGE msg.	syslog
Customer reset session	eBGP session is reset by the customer, BGP-5-NOTIFICATION msg.	syslog
eBGP HTE	eBGP hold timer expired, BGP-5-NOTIFICATION msg.	syslog

Table 4.10: Service Specific Events for Root Cause Analysis of RTT Degradations in CDN

Event Name	Event Description	Data Source
CDN RTT degradation	end-to-end RTT increase between CDN servers and user-groups detected by Argus	Argus
CDN server issue	CDN server load is high	server logs

Table 4.11: Service Specific Events for Root Cause Analysis of PIM Adjacency Change in Multi-cast VPN

Event Name	Event Description	Data Source
PIM Neighbor Adjacency Change	a PE lost a neighbor adjacency with another PE in the MVPN.	syslog
PIM Configuration change	a MVPN is either provisioned or de-provisioned on a router.	router command logs
Uplink PIM adjacency change	a PE lost a neighbor adjacency with its directly connected router on its uplink to the backbone.	syslog

diagnosis rules. The complete diagnosis graph is depicted in Figure 4.27. This combines events and rules taken from the RCA Knowledge Library (Table 4.2.1.3) with BGP service specific events (shown as gray boxes) and service specific rules (dashed lines). Let us now examine the diagnosis graph (Figure 4.27) from bottom

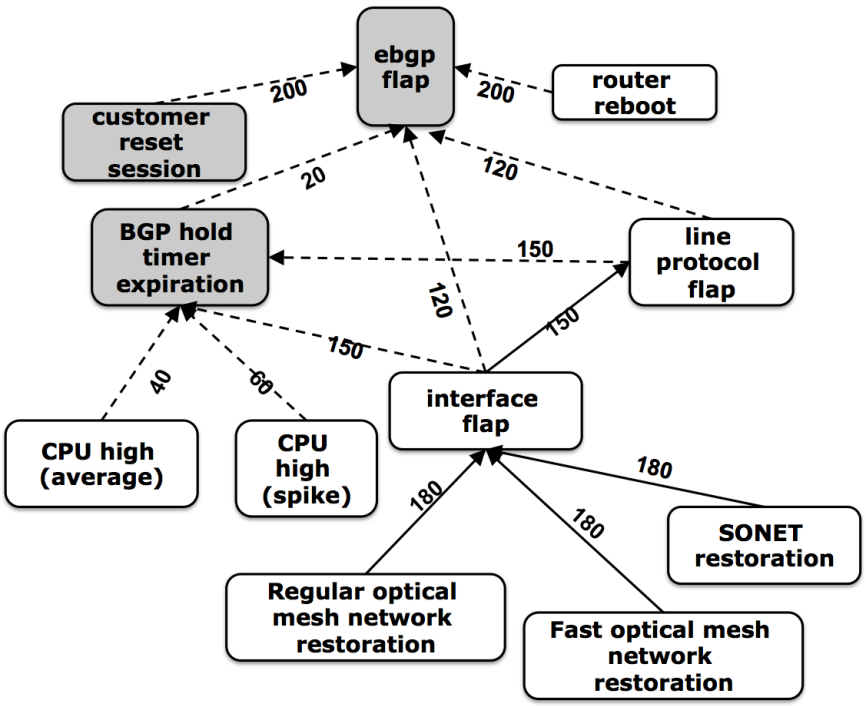


Figure 4.27: Diagnosis Graph for BGP Flaps Root Cause Analysis

to top. Layer-1 events such as fast restoration in optical mesh network and SONET restoration may cause interface flaps on PERs. Further, interface flaps may induce BGP hold timer expirations, line protocol flaps or even eBGP flaps. If BGP fast external fallover [3] is enabled, an interface flap can directly trigger an eBGP flap without requiring the BGP hold timer to expire. All the CPU utilization related events such as "CPU high (average)" and "CPU high (spike)" can only cause eBGP flaps through BGP hold timer expiration. On the top of this figure, we can see the events "router reboot" and "customer reset session" could also cause eBGP flaps.

Finally, we specify priorities for different diagnosis rules for BGP flaps RCA, as depicted via the numbers on the edges in Figure 4.27. The highest priority is used to determine the most likely root cause among multiple root causes by the G-RCA Engine. For example, if a BGP flap joins with both a high CPU event and

a layer-1 flap, the layer-1 flap is identified as the root cause of this BGP flap as it is associated with a higher priority (180) edge.

- **Results**

  In order to demonstrate how effectively G-RCA can identify the root causes of BGP flaps, we ran the BGP flap RCA tool configured above for more than 600 provider edge routers in different locations, each of which has several hundred eBGP sessions established with customer routers. Table 4.2.2.2 shows the root cause breakdown generated by the Result Browser in G-RCA for all the BGP flaps on these provider edge routers during a month.

Table 4.12: Root Cause Breakdown of BGP Flaps

Root Cause	Percentage (%)
Router reboot	0.33
Customer reset session	1.84
CPU high (average)	0.02
CPU high (spike)	6.44
Interface flap	63.94
Line protocol flap	11.15
eBGP HTE (due to unknown reasons)	4.86
Regular optical mesh network restoration	0.04
Fast optical mesh network restoration	0.14
SONET restoration	0.29
Unknown	10.95

One of the most critical insights revealed by applying our BGP flap RCA application to the operational network is in the BGP flaps induced by fast optical mesh network restoration. The service provider's routers are configured so that these fast optical mesh network restoration by design should not cause BGP flaps. Yet Table 4.2.2.2 clearly indicates that they are. When considered in comparison with all BGP flaps, they are a relatively small percentage of events. However, further analysis revealed that for customers who tightly manage their networks,

these layer-1 induced BGP flaps were a far greater percentage of all their BGP flaps, and were causing concerns. As this behavior was inconsistent with design, it represented an opportunity for improving service.

After detailed lab replication and close collaboration with the router vendor, it was revealed that the routers were reacting to fast optical mesh network restoration when they should not – a bug in the router software. The router vendor rapidly repaired the code, which was then deployed across the service provider's network as a software upgrade. Although the new software was extensively tested in the lab before being rolled out, the operational network is far more complex than can be replicated in a lab environment, and it was thus necessary to validate that the software repair did indeed have the desired effect in the operational network. The G-RCA application was the means through which this was achieved. The same set of provider edge routers as used in Table 4.2.2.2 were picked to test the software fix in the field. Specifically we ran the BGP flap RCA tool again with all BGP flaps on these routers for a month after the software fix was applied. G-RCA's proved that the software repair was successful as no BGP flaps were caused by fast optical mesh network restoration after the upgrade.

This RCA application is now an integral part of the customer BGP monitoring. It is used to trend flaps and identify anomalous behavior which requires investigation (e.g., behavioral changes after new software upgrades). It is also used by operations and customer service representatives to provide automatic analysis of specific customer BGP flaps, for rapid responses to customer inquiries about such events. In the BGP RCA application, the average disgnosis time per symptom event is less than 5 seconds.

4.2.2.3 Root Cause Analysis of PIM Adjacency Change in Multicast VPN

- **Overview of PIM Adjacency Change in Multicast VPN:** In the final case study, we describe the use of G-RCA to identify the root cause of problems in the Multicast VPN (MVPN) service. For each MVPN customer, all PERs at which the customer attaches to the provider network maintain PIM (Protocol-Independent Multicast) Neighbor adjacencies with each other using a Hello protocol. The loss of PIM neighbor adjacencies, which is reported via syslog, are often a good indicator of service related problems. However, not all such changes are indicative of an actual problem (e.g., some are due to customer disconnects). Due to the sheer volume of these messages (thousands per day), manual analysis to determine the root cause of each event to determine those which are indicative of an actual problem is infeasible.

- **Customize G-RCA to Diagnose PIM Adjacency Changes in Multicast VPN:** A G-RCA application to determine the root cause of PIM neighbor adjacency changes within the MVPN service was created. The resulting diagnosis graph is shown in Figure 4.28. The kinds of root cause events that were determined to have caused PIM neighbor adjacency changes include router configuration changes, problems on the provider-customer link, routing changes within the service provider's network, and problems on the PER uplinks to the backbone network. Since the application was able to reuse many of the events and rules in the RCA Knowledge Library, we only needed to add 3 multicast-specific events (Table 4.2.2.2) and 7 multicast-specific diagnosis rules (dashed lines in Figure 4.28). For example, we reused many events and rules regarding the path changes between a pair of PERs and the unstablities on the provider-customer link. Actual development time was no more than 10 hours. Without G-RCA, building a root cause analysis tool for this problem would have required months of development,

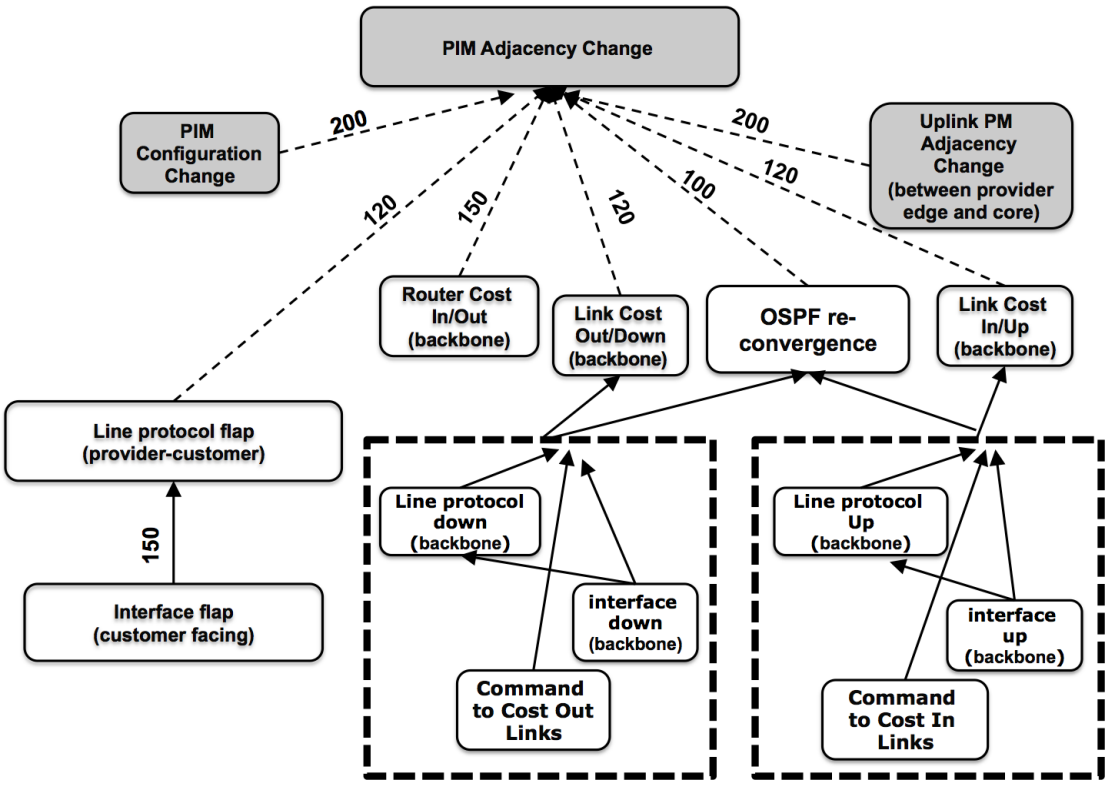


Figure 4.28: Diagnosis Graph for PIM Adjacency Change Root Cause Analysis

and may not have happened in practice.

- **Results**

  The PIM RCA application has proved to be extremely useful in classifying root causes of PIM adjacency losses and in guiding operators and engineers to a better understanding of actual MVPN performance in the network, allowing them to focus their effort on those issues that require their attention. Running the G-RCA PIM application on a day's worth of events required about 1-2 hours. For each day, the application is currently able to identify the root causes for more than 98% of PIM neighbor adjacency changes. We expect that with additional attention to those remaining unclassified events, the G-RCA PIM application will determine root causes for more than 99% of the events.

In order to demonstrate how effectively G-RCA can identify the root causes of PIM adjacency losses in the service provider's network, we ran the PIM RCA application configured above for all the PIM neighbor adjacency changes observed in 2 weeks on more than 600 provider edge routers. Table 4.2.2.3 shows the root cause breakdown generated by the Result Browser in G-RCA. We skip the breakdown of some detailed root causes (shown in the dashed rectangle in Figure 4.28) to ease the discussion by simply using their parent events as the root causes.

Table 4.13: Root Cause Breakdown of PIM Adjacency Losses

Root Cause	Percentage (%)
PIM Configuration Change (to add and remove customers)	4.04
Router Cost In/Out	10.34
Link Cost Out/Down	1.50
Link Cost In/Up	0.84
OSPF re-convergence	10.36
Uplink PIM adjacency loss	1.95
interface (customer facing) flap	69.21
Unknown	1.76

One of the primary benefits of the G-RCA PIM application was its ability to identify events that could be ignored. For example, a large number (69.21%) of PIM adjacency changes between PERs where found to be due to link flaps between the customer and provider routers, which typically are caused by customer activities. These link flaps did not affect multicast service between the PERs, but they did result in spurious PIM adjacency change syslog messages which could be safely ignored. Along similar lines, another root cause of the PIM adjacency changes was configuration change (to add and remove customers) on the PE router. By identifying those events that did not require investigation or further action, the G-RCA PIM application enabled operators to focus on those events that were potentially performance impacting. As an example, G-RCA found cases (10.36%) in which

OSPF routing changes in the ISP's backbone network led to PIM neighbor adjacency changes. This violates the protocol designs and was thus unexpected - PIM adjacency loss should only be induced by outages far longer than typical OSPF convergence events. Through detailed G-RCA analysis and close collaboration with the network operator and relevant router vendors, several different causes of this unexpected protocol interaction have been identified such as various software bugs in devices. Once G-RCA was able to identify the root causes, appropriate solutions to the problems have been identified and put in place to permanently improve customer multicast performance. More importantly, these solutions may also help other ISPs who may not already be aware of these problems.

In the PIM RCA application, the average diagnosis time per symptom event is similar to the BGP RCA application, which is typically less than 5 seconds.

4.2.3 Operational Experience on Improving Domain Knowledge

The main challenge in creating G-RCA applications is identifying the diagnosis rules. Domain knowledge typically provides a solid starting point, but our experience indicated that collating domain knowledge across potentially many domain experts can be surprisingly challenging. Domain knowledge is often distributed across multiple experts - no one expert understands the entire domain. These experts often have trouble thinking of the relevant rules when "put on the spot", or they are so busy fighting issues in the network that it is difficult to obtain their attention for long enough to obtain the information. In other cases, the network operator's domain knowledge may be wrong either because the relationships between events are extremely complex and not well understood, or because the network is not behaving as designed (as in Section 4.2.2.2). We thus found it extremely critical to provide mechanisms integrated in G-RCA to facilitate diagnosis rule learning.

4.2.3.1 Learning Diagnosis Rules via Manual Iterative Analysis

With G-RCA, the individual responsible for creating an RCA application can follow an iterative process to identify new diagnosis rules. For example, in the PIM case, domain experts use data exploratory tools [33] to manually inspect unexplained neighbor adjacency changes and determine root cause(s). Once a new root cause was identified, it was codified in the RCA application, which was then run to identify all those events which could be explained by the augmented set of rules and, more importantly, those which were still unexplained. The domain experts would then further sample remaining unexplained PIM flaps searching for new signatures which could be incorporated. The PIM application developer thus continually whittled down the number of unexplained flaps, by iteratively incorporating new rules and examining those which fell outside the scope of the new rules. By using G-RCA's Result Browser which made individual event analysis easy, the PIM application developer rapidly identified new diagnosis rules for the application and therefore revealed the anomalous behaviors discussed in Section 4.2.2.3.

4.2.3.2 Learning Diagnosis Rules via Statistical Correlation Test

Although the manual iterative analysis was effective in the PIM application, we used a more "intelligent" approach to analyzing BGP flaps. We illustrate this here by discussing our experiences in analyzing BGP flaps which were related to high CPU events.

Table 4.2.2.2 illustrated that a significant portion of BGP flaps occurred at the same time as CPU overload was observed on the router. A naive assumption may be that these BGP flaps were in fact induced by high router CPU load. However, further inspection cast doubt on this assumption.

With the integrated data drilling-through functionality implemented in the Result Browser of G-RCA, it is easy for operators to explore additional information such as

syslog messages and workflow logs that appear on the same router or location as the event being analyzed. Equipped with the powerful GUI, operators revealed via manual drilling-down that not all BGP flaps with a high CPU signature are actually due to CPU overload on PERs. In most cases, the high CPU utilization is likely *caused by* BGP flaps that are triggered from the customer side. Specifically, large amount of routing computation on PER in response to the BGP flaps produces high CPU utilization.

With this cyclic causal relationship – "BGP flap causes CPU overload" and "CPU overload causes BGP session timeout", evidence based diagnosis systems including our RCA tool hit their limit. We needed further refined signatures such as searching for other potential causes of the high CPU events to identify those which were not BGP flap induced and could thus explain BGP flaps.

Rapid manual inspection of events through G-RCA's Result Browser worked well in some situations but our experience demonstrated that it does not work effectively if looking for relatively rare explanations among a sea of events. Instead, we took a different approach (Figure 4.29), using G-RCA's correlation tester module to examine the statistical correlation between CPU-related BGP flaps and other types of events on the same PER. Specifically, we created a time series from all CPU-related BGP flaps as defined by our G-RCA application - those BGP flaps associated with BGP hold timer expiries, but where there was no evidence of link failures that could explain the flap, and which joined with one of the high CPU signatures. We then executed a statistical correlation test [41] between this time series and 831 other time-series created from workflow logs, and 2533 time series from syslog messages.

We fed three months worth of data into the correlation tester to analyze the CPU-related BGP flap. Of the 3361 time series, 80 time series exhibited significant statistical correlation with our CPU-related BGP flaps. A rapid examination of these events by domain experts revealed that many of them were readily explained and/or incorporated

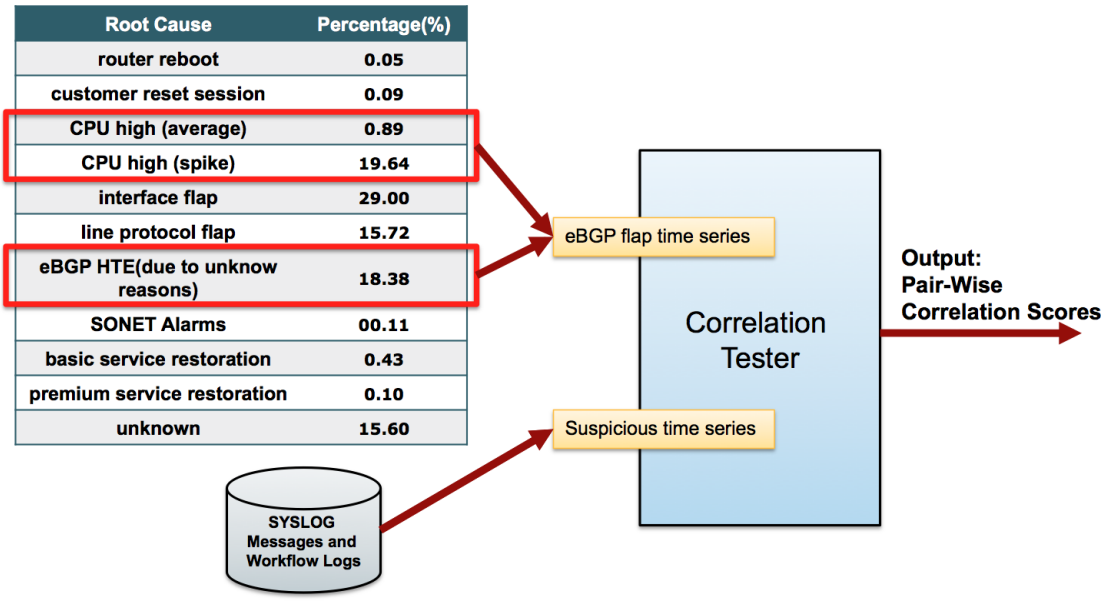


Figure 4.29: Interaction between Generic RCA Engine and Correlation Tester

into our existing application rules. For example, these CPU-related BGP events were strongly correlated with BGP notifications - a generic message logged for any BGP flap. However, the statistical correlation test did reveal some unexpected correlations. For example, the result revealed that certain provisioning activities (as derived from work-flow logs) are strongly correlated with CPU-related BGP flaps. Drilling into individual cases, we identified a small number of incidents where unrelated provisioning activities on some routers appear to have caused customer BGP sessions to flap, an unexpected router software behavior. As a result, 10 such incidents were sent to the router vendor for further investigation; the vendor has since implemented software changes to eliminate this issue.

It is worth noting that the pre-filtering of BGP flaps by their root causes as diagnosed by the Generic RCA Engine made a significant difference here. When we fed all BGP flaps to the correlation tester module, the correlation with provisioning activity was no longer statistically significant. By instead focusing on a small subset of the BGP flaps, the correlation "signal" is amplified, revealing the hidden issue. Thus, the interaction

between G-RCA engine and the Correlation Tester is crucial to revealing subtle issues.

4.2.3.3 Learning Unobservable Root Causes via Bayesian Inference Engine

Thus far all examples have been based on rule-based reasoning. We now demonstrate the power of Bayesian inference engine in the G-RCA. In particular, we show how the inference engine can identify a line-card problem as the root cause, which is not readily detectable using rule-based reasoning: a line-card problem is an unobservable root cause as no logs or alarms for line-card issues were incorporated into the RCA tool at the time of our analysis. The configuration for inference engine is shown in Figure 4.30. Three virtual root cause events are defined as "CPU High Issue', "Interface Issue" and "Line-card Issue".

We ran both the rule-based reasoning engine and the Bayesian inference engine using one month of eBGP flaps on a PER that has several hundred eBGP sessions. While most of the results are consistent with each other, there are 133 eBGP flaps that the rule-based reasoning engine diagnoses as "Interface flap" induced. However, the Bayesian inference engine identifies these same flaps as "Line-card Issue" caused. By manually drilling down into these 133 eBGP flaps (on 125 different eBGP sessions) using G-RCA's Result Browser, we find that all of them are associated with the same line-card and are within three minutes. This is a strong indication for a line-card related problem. This was later confirmed by network operators, who actually pointed us to a line-card crash signature that was not incorporated into G-RCA's Knowledge Library at the time, and we confirmed that the linecard in question indeed crashed. Since Bayesian inference easily allows analysis across multiple symptom event instances, the common root cause of these 133 eBGP flaps was successfully inferred.

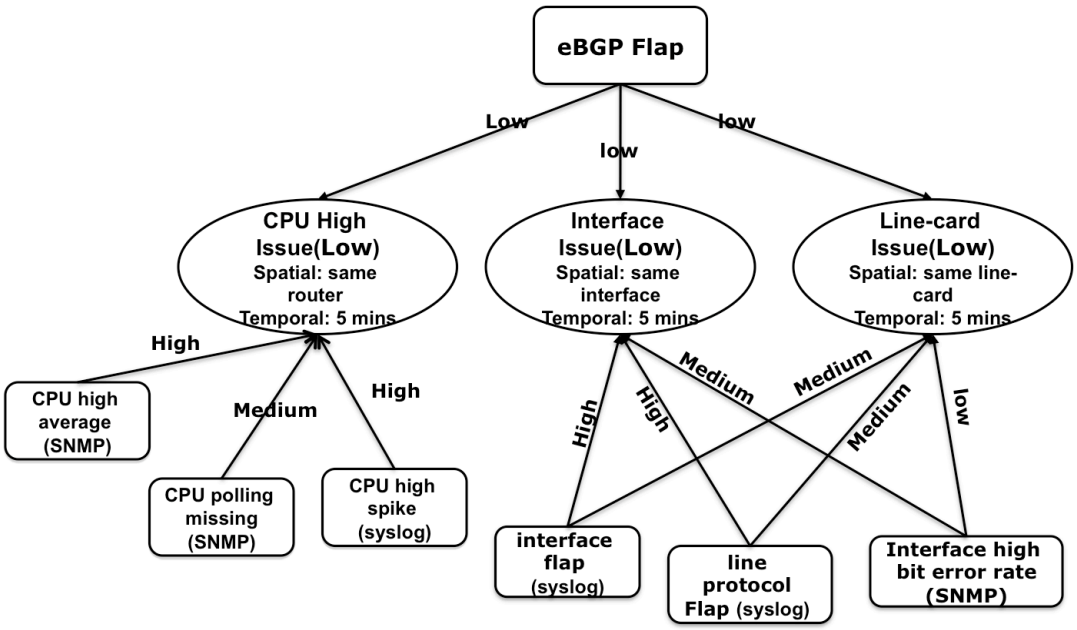


Figure 4.30: Bayesian Inference Configuration for BGP Flaps RCA application

Chapter 5

Bottom-Up Direction

In the previous chapter, we described the top-down approach that first detects the service quality degradations based on user-view measurements (e.g.: complains from Twitter, customer calls and download throughput) and then correlates service quality degradations with network-view measurements (e.g.: link congestion, router CPU overload or cell tower failure) to find the root causes. In this chapter, we discuss the opposite bottom-up direction that first detects network events (e.g.: link congestion, router CPU overload and cell tower failure) based on network-view measurements and then determines if a network event has actually caused any service quality issues or not.

There are two main steps in the bottom-up direction:

1. Detecting network events based on network-view measurements (e.g.: link congestion, router CPU overload or cell tower failure)

2. Estimating the user impact (e.g.: who are impacted? by how much? for how long?) of detected network events based on user-view measurements (e.g.: complains from Twitter, customer calls and download throughput).

As discussed in the previous chapter, Argus and G-RCA are designed and implemented in a generic manner for various services in the top-down direction. In this chapter, we focus on the 3G mobility service and design a system called TowerScan

to detect cell tower failures and determine their user impact. Although our work in the bottom-up direction is not as generic as the top-down direction, TowerScan reveals the common challenges and complexities in the bottom-up direction. Extending TowerScan to a generic infrastructure for various services is left for our future work. In the rest of this chapter, we first give an overview of TowerScan, then describes the design of TowerScan and finally presents evaluation results and operational experiences.

5.1 TowerScan: Understanding the Impact of Network Events on Service Quality

5.1.1 Overview

Figure 5.1 shows the overall architecture for 3G mobility networks, which consists of two parts: UMTS Radio Access Network (UTRAN) and UMTS core network. As our objective is to quantify the user impact of cell tower outages, we focus on the UTRAN in this dissertation. The UTRAN mainly consists of the User Equipments (UE), the NodeBs (or commonly known as the mobility towers), and the Radio Network Controllers (RNC). mobility towers perform wireless link transmission/reception to/from the UEs via the Uu radio interface, and communicate with the RNC via the backhaul link called the Iu-B. In the most common configuration, a single mobility tower serves 6 cells, also called sectors. Each RNC typically manages tens to hundreds of cell towers and serves as a gateway to the UMTS core network.

In order to achieve an end-to-end communication in 3G UMTS network, sophisticated signaling protocols among the UTRAN and the UMTS core network devices are required. First, a UE acquires a physical radio channel through the RRC (Radio Resource Control) signaling by exchanging connection establishment request/response with the RNC. Once the physical radio connectivity is established, a logical Radio Access Bearer (RAB) between the UE and the Core Network needs to be established. This

is achieved through a set of RAB signaling. RAB is designed to provide users with multimedia services (voice, video and data) with assured QoS. The success rate of RRC and RAB establishment is often used to quantify the service availability of 3G UMTS networks.

Once a UE is active (connected to a sector/tower via radio link) on the 3G UMTS network, the affiliation between the UE and the sectors/towers is managed through maintaining a so-called active RLS (Radio Link Set) at both the UE and the RNC end. The RLS contains the identity of the sectors that a UE is simultaneously listening to and transmitting to. These sectors may belong to different towers, which may be further controlled by different RNCs. When transmitting data from UEs, all sectors in the RLS participate in receiving and decoding the data frames, and when transmitting data toward UEs, a primary sector in the RLS is responsible for sending the data frames. UEs continuously monitor the received signal strength from all "visible" sectors and communicate with RNC periodically so that RNC is able to optimize the channel allocation for all UEs in the proximity. The decision of RLS updates (inclusion of new sectors or removal of existing ones) is signaled back to UEs and coordinated with cell towers for future data transmission. This signaling and adaptation of RLS provide a natural mechanism in UTRAN to achieve load-balancing and fault-tolerance to cell/tower outages.

5.1.2 Challenges

With vast numbers of cell towers scattered across often challenging terrain and interconnected with a wide array of complicated devices, cell tower outages are simply a fact of life. Cell **tower outages** may be triggered by external conditions, such as weather events or fiber cuts, by hardware or software failures, or may result from necessary maintenance activities. However, not all cell tower outages actually have significant impact on the users that they service. Neighboring towers may well be able to service users who would normally have been serviced from a failed tower. On the other hand,

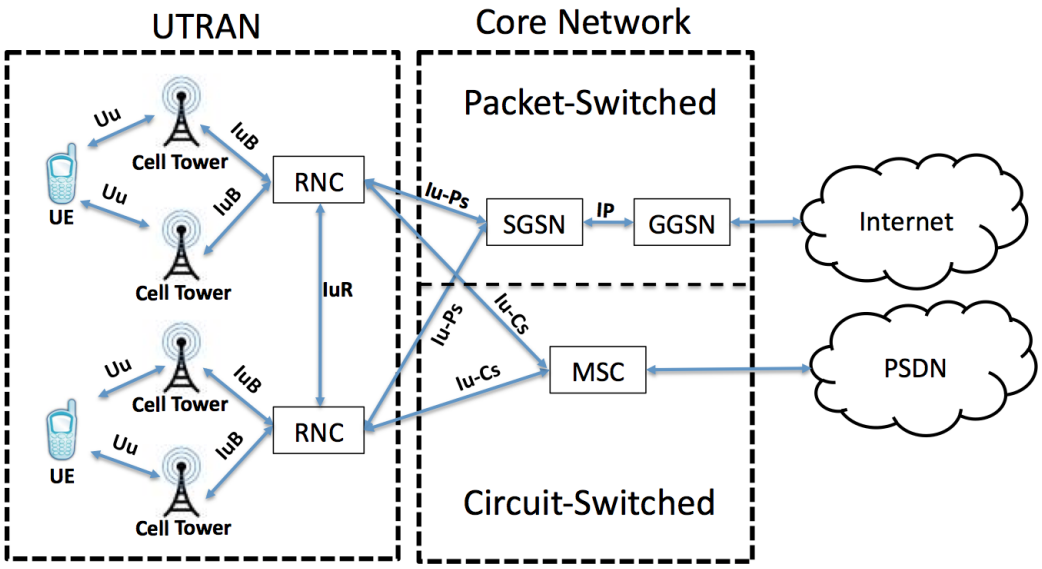


Figure 5.1: 3G UMTS network architecture

depending on signal propagation characteristics and network load, the extra workload due to the failed tower may be redistributed to the towers in a much larger area, causing degraded service quality to users that were far away from the failed tower. [1]

As tower outages are common and inevitable, and yet their service impact on users varies to a great extent, it is critical that an operator be able to rapidly and accurately determine the impact of cell tower outages. The accurate assessment of service impact will guide many facets of network operations and design, including the prioritization of activities should there be conflicting issues necessitating attention, guiding customer care service representatives on how to react to network events (e.g., providing accurate feedback to mobile users about the potential impact of a network issue), reporting network issues and their estimated user impact to the FCC (Federal Communications

[1] We use the term "tower outage" here loosely, as we use it to refer to any scenario where a cell tower is unable to service mobile users. The root cause may not even be within the cell tower – for example, a failure in the Radio Network Controller (RNC) could result in the failure of a subset or all of the cell towers that it supports.

Commission) and guiding network engineering in improving network design. In this paper, we focus on developing a systematic approach to quantifying the service performance impact of tower outages from the user's point of view.

Quantifying the user impact of tower outages is a complex task, primarily due to the inherent redundancy in radio coverage. Under normal conditions, each active UE is typically "connected" to a set of nearby cell towers via multiple radio links for fast data transmission and service redundancy. The active RLS (Radio Link Set) for each active UE is maintained dynamically in response to the physical movement of the mobile user, or even when the UE remains stationary due to fluctuations in radio channel conditions (e.g., signal strength), or due to load-balancing and optimizations triggered by the RNCs. This change in the active RLS is achieved automatically through sophisticated signaling among UEs, cell towers, and RNCs. With such an inherent service redundancy mechanism built in, the impact of a tower outage may very well be "absorbed" by the surrounding towers through a rippling series of signaling, with the result being that users perceive no service impact. On the other hand, it is also possible that users associated with fully functional towers may suddenly suffer performance degradations resulting from the sudden migration of service load from a nearby tower that failed. This makes the impact analysis of a tower failure dependent on its nearby tower locations, antenna configurations, mobile UE distributions, local landscape features (e.g., hills, buildings), and a wealth of other potential factors. Detailed modeling of each of these features can be extremely challenging, if not impossible.

The tower outage scenario becomes even more complicated when multiple tower outages "collide" in time and space. Towers may fail as a result of issues within the local tower - thereby being isolated to only that single tower - or because of issues which may impact a broader region, such as the failure of the RNC to which cell towers are connected. As many towers are attached to a single RNC, a RNC failure could in fact

induce the simultaneous failure of potentially large numbers of cell towers, all located within close geographical proximity to one another. More generally, tower outages are often likely to be correlated due to shared risks in common network elements or weather conditions. Individually evaluating the impact of correlated tower outages in isolation makes little sense as they have compounding impact on each other. It is thus important to jointly consider temporally and spatially convoluted cell tower outages.

Although operators devote tremendous effort to quantifying the user impact of tower outages, the state-of-the-art approach is rather rudimentary. Typically, analyses are conducted manually by an operator based on his/her understanding of the nature of the outage, its root cause and expected impact, and the user or usage estimates. Outages are also often analyzed on a per incident basis with each outage examined in isolation and the fault-tolerance mechanisms ignored, simply due to the complexity of performing such an analysis manually. Consequently, the impact analysis result may be quite inaccurate, for example when the operator misjudges the failure scenario, or under- or over-estimates the user population.

5.1.3 Our Approach

In TowerScan, we seek to develop an objective data-driven approach that does not require operators' insights into the nature of the tower outages. Our impact analysis approach should incorporate key factors such as mobile UE density and usage levels, mobility fault-tolerance and load-balancing mechanisms, and correlated tower outages. It is also imperative that our analysis not require a priori knowledge of the root cause diagnosis of the cell tower outage, as such knowledge is often not available until a thorough investigation can be carried out. An estimate of the service impact of an outage must typically be known *before* such a root cause analysis is completed.

There are many ways that a tower outage may impact user-perceived service. An outage could cause calls to be dropped or blocked, or may result in degraded data service

quality (e.g., slower web responses). We thus need to quantify the impact of tower outages using a comprehensive set of critical metrics that capture the users' service quality. Such metrics range from accessibility (measuring voice/data connection success rate) to retainability (measuring dropped voice/data connections, e.g., dropped calls) through to data performance (measuring service quality such as data download throughput). The impact analysis must accurately identify the impact of the outage on this range of different metrics, and enable accurate and meaningful comparisons across different tower outages.

Ideally, we would like to track detailed measurements from all UEs to achieve the most comprehensive and accurate analysis. However, for a mobility service provider with tens of millions of UEs, tracking detailed UE level data for determining the service impact can be prohibitively expensive in measurement overhead, not to mention the controversial issue of subscriber privacy. Given the unavailability of such fine grained UE level measurements, we choose to instead base our approach on the aggregated service monitoring data readily available in the form of performance counters produced by the cell towers (conforming to the 3GPP standards). These counters are periodically polled from the towers and the values are stored in a centralized data repository.

In order to determine the impact scope of given tower outages, we develop a method that borrows techniques from statistics (time series analysis) and computer vision (diffusion and region extraction). Our approach effectively identifies the affected service region without tracking detailed signaling exchanges. Furthermore, we collate multiple data sources to generate numerical quantifications of the service impact pertaining to the affected service region. We applied our method in a 3G UMTS mobility network operated by a large mobile service provider in North America, and worked through numerous cases studies with the network operators. The operator feedback provided us with positive confirmation – our method effectively automates the "art" of impact anal-

ysis on tower outages and our quantification result is consistent with the intention of service impact analysis and is often more accurate than the manual analyses performed today. Note our approach can be applied to other mobility networks (e.g., 4G/LTE) although so far we have only applied it in a 3G UMTS mobility network.

5.1.4 Data Sources

TowerScan uses the following two types of data sources. The device syslogs are the messages that network devices generate to record the hardware and software conditions observed by them, such as link and protocol-related state changes (e.g., down or up), alarming environmental measurements (e.g., high voltage or temperature), and warning and debugging messages (e.g., a configuration change becoming effective). Logging is enabled at different devices in UTRAN (cell towers, RNCs, and routers and switches in between) with varying logging level settings (e.g., critical or warning) and message exporting setup. The syslog messages, either in its raw form or filtered and processed by an operations management platform, provide information including the timestamp of the reported event, the identity of the reporting device, a message body describing the nature and some details of the event. The device syslogs are considered the most valuable data source for monitoring network faults, or with respect to this study, the diverse types of cell tower outages.

A wide range of UMTS performance measurement counters are maintained at cell towers and RNCs. Similar to the SNMP MIBs in routers, the UMTS counters track the frequency of network/service events (e.g., number of successful/unsuccessful RRC requests), the packets and bytes flowing through each network interface, and other service and device self-monitoring metrics (e.g., queue length and CPU utilization). The counters and combination of them can be turned into performance metrics that are generally divided into four broad categories: accessibility (e.g., voice/data call setup success rate and delay), retainability (e.g., call drops rate), mobility (e.g., hand-off frequency and

delay), and performance (e.g., voice call quality and data throughput). In the 3G UMTS provider network that we study, there are over 300 different types of UMTS counters tracked and collected at 15-minute granularity and associated with varying levels (sector/cell tower/RNC), enabling quantification of service impact of tower outages.

One of the most important metrics for our analysis is the "number-of-UEs" associated with each tower, which is the total of "number-of-UEs" associated with the sectors within the tower. The metric is based on counters maintained at RNC that provide summary statistics of the active RLS of all UEs under its control. Particularly, for each sector, there is a counter that reports the time averaged value of the number of UEs affiliated with the sector when the size of the UE's active RLS is k. To avoid double counting of UEs, the number-of-UEs metric is generated by taking a weighted sum of those summary counters with the weight being $1/k$. For example, an UE with 4 sectors in its active RLS would contributes 0.25 to the value of number-of-UEs metric for each sector in its RLS. The number-of-UEs metric reflects the workload redistribution under cell tower outages, which is critical for our approach in the next section.

5.2 TowerScan Design

As described in the previous section, mobility service operators need to accurately determine the impact of cell tower outage(s). One simple and seemingly logical approach is to examine the workload on the failed tower right before the outage and use it to quantify the impact. Alternatively, one may apply some time series analytic techniques to estimate the expected normal-condition workload on the failed tower during the outage for impact analysis. However, these approaches are too crude to reflect the true impact, as the effective impact is also largely determined by the distribution and the workload on the surviving towers in the area. For example, outages on two "neighboring" towers are likely to cause higher damage than two individual outages on towers that are far apart,

even when their combined workload before their outages is the same in the two scenarios. This is because in the former case users from the failed towers are less likely to be able to switch to another functioning tower to continue the service than in the latter case. Experienced operators would consider factors such as the tower density in their impact assessment. However, without a systematic approach, their impact analysis result often varies person by person and case by case.

In TowerScan, we provide a way to systematically quantify the impact of tower outage(s) on the mobile users. Ideally, this can be done by tracking all mobile users at all time throughout the network. However, collecting and processing this user-level data can be overwhelmingly costly. Furthermore, we cannot derive the per-user level data from the modeling of mobility communication – accurately predicting the mobile user-tower association and determine the service capacity under all terrain, weather, and device conditions is a far more challenging research problem. Instead, TowerScan relies on aggregate performance counters that are readily available from towers' OSS (Operations Support System). Based on the aggregate counters, TowerScan first computes the impact region for the tower outages under investigation and then quantifies the user impact by various metrics (using a variety of data sources) pertaining to the identified impact region.

5.2.1 Determining the Status of Cell Towers

The first task in quantifying the service impact of cell tower outages is to detect that a cell tower is out of service. This is achieved by parsing and mining the device syslogs — in the 3G UMTS network studied in this paper, these device logs are collected in near real-time to a central repository. We compiled a list of "signatures" of tower failures contained in the device syslog messages from both the cell tower side and the RNC side. These signatures are defined based on domain knowledge input from experienced operators. Some of them are closely tied to certain types of failure root causes (e.g.,

lower layer communication failure), while others are generated via service and signaling monitoring between RNC and cell towers. The redundancy in failure signatures and source devices is critical for TowerScan, as collection of all device syslogs may not be reliable or timely under outage conditions.

One implication of using multiple sources and message types for tracking cell tower outages is that there may be inconsistency in the message timestamps regarding the same outage. Hence, in the design of TowerScan for impact analysis, we do not rely on precise timing information about tower outages. Furthermore, as flapping (devices repetitively switching in and out of service) is a common phenomenon in operational networks, TowerScan agglomerates the detected cell tower outages that are close in time (e.g. within 600 seconds) into a single long-lasting outage for the purpose of impact analysis.

In case that the above automation system fails to infer any out-of-service cell towers, TowerScan also allows manual input from operators to set the aliveness status of the cell towers and to initiate the impact analysis as described in the next two subsections.

5.2.2 Identifying impact regions

Given the time and location of a tower outage, TowerScan first identifies *the region on which the tower outage is likely to have impact*, which we will refer to as the **impact region**. Once impact regions are determined, TowerScan can then quantify the degradation in accessibility, retainability, voice and data quality and performance within these regions (Section 5.2.3).

The complexity in scoping the impact region lies in the sophisticated fault-tolerance and load-balancing mechanism in mobility networks, which is achieved through extensive signaling among UEs, towers, and RNCs (radio controller coordinating multiple towers) so as to adapt to normal radio resource dynamics as well as anomalous outages. The impact due to the loss of resource is expected to be gradually "absorbed" in a rippling fashion by the surrounding towers. However, because of the diversity in tower

density and UE distribution, the complexity of landscape features (e.g., hills, buildings, etc.), and the mobile nature of UEs, it is extremely challenging, if not impossible, to model this impact absorption process, hence precisely calculating the impact regions. Furthermore, the outage of one tower can compound onto other pre-existing or concurrent tower outages. One has to examine the the spatially/temporally correlated outages jointly to determine the impact regions. In fact, as we will see in Section 5.3, the outages in operational 3G networks are more likely to be clustered in space and time due to common shared-risk components [39, 36, 38]or common influence of adverse weather condition, causing compounding impact to each other.

Before we present our solution to impact region scoping, let us first consider a couple of straw-man approaches.

- Use the (maximum) coverage scope of the tower under outage as the impact region – for example, using a 5-mile radius circle centered at the tower under outage. This however can be both too broad and too narrow. On one hand, during a non-peak time in a densely deployed 3G area, the impact can very well be localized to its immediate surrounding towers, making 5-mile circle excessively large. On the other hand, in a tightly-resourced area under peak demand, the cascading effect can be farther reaching.

- Track per UE level performance and determine who as opposed to where has been impacted. The problem with this approach is the complexity and the daunting cost of per UE level measurement. Even if per UE measurement could be practically managed, the variability of service performance on individual UE can be too high to reliably indicate whether one has been affected.

Our approach focuses on the aggregate performance metrics (e.g., IuB link utilization, call drop rate) at the tower level. It seems straightforward to define one impact

region based on each aggregate performance metric. Simply put, if on some *surrounding* tower, one aggregate performance metric *deviates significantly* from its expected value, the tower is considered impacted from that metric's perspective. We will bind the loose terms of "surrounding" and "significant deviation" with more rigorous mathematics processes in later sections. However, there are two drawbacks with this approach. First, many metrics are not quite meaningful when examined at individual tower in isolation. For example, "IuB link utilization" at a tower may be associated with two different sets of UEs due to different UE associations before and after an outage. Comparing the "IuB link utilization" loses the physical meaning. Second, using each metric separately for impact region identification would not only increase the computational overhead, but also produce inconsistent (e.g., due to varying sensitivity of different metrics) and hence confusing impacted regions, which is very undesirable to the network operators.

To address these limitations, we choose the single key metric number-of-UEs (associated with individual towers) to identify the impact region. The number-of-UEs is a suitable metric in scoping the impacted region because it ties closely to the UE and tower association, which largely determines the impact propagation or absorption among surrounding towers.

5.2.2.1 Anomaly Detection in number-of-UEs

We rely on time series analysis techniques to detect significant deviations from expectation on the tower-level number-of-UEs metric. An example of such time series is shown in Figure 5.2 (a). In particular, we adopt the time series decomposition approach, which is a light-weight offline time series analysis algorithm and has been found very effective in modeling economic data and recently in network traffic as well [51]. At a high level, time series decomposition technique deconstructs a given time series into the secular trend component ($\{T_t\}$), the seasonal component ($\{S_t\}$), and the noise component ($\{N_t\}$) [19]. In the additive model, the original time series ($\{V_t\}$) is the summation of

these three components.

Figure 5.2 (b), (c), (d) show the corresponding components for the time series in Figure 5.2 (a). The general idea of time series decomposition is very simple – with a specified seasonality window W, secular trend can be obtained through smoothing over long term (multiples of W), i.e., by *centered moving average*:

$$T_t = \sum_{i=-W}^{W-1} V_{t+i}/2W$$

The seasonal trend can be obtained by averaging the phase value (after removing secular trend) across seasons, i.e., by *seasonal moving average*:

$$S_t = \sum_{i=0}^{K} V_{t-iW} - T_{t-iW}$$

where K is the number of seasonal windows contained in the historical data. And the remainder becomes the noise component:

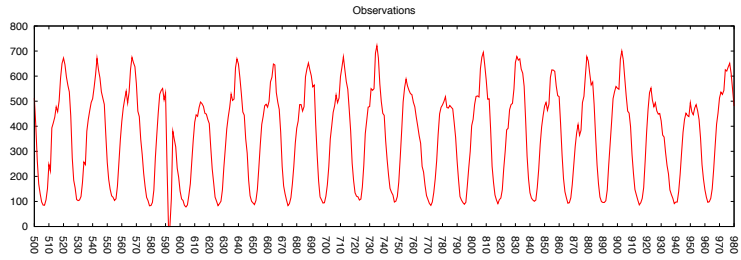
$$N_t = V_t - T_t - S_t$$

Note that time series decomposition as the above can be applied for analyzing both long range of historical data and in a moving window fashion for the recent data (as new data append to the time series).

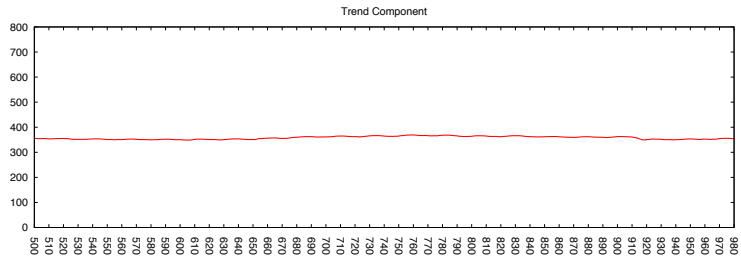
In our approach, we further model the noise components, N_t, at different phase as zero-mean Gaussian variables with different variance, $\sigma^2_{t|W}$, where the phase $t|W$ represents $t \mod W$. We tag the corresponding time series value, V_t, as anomalous (critical value 1.96 at 95% confidence interval) if

$$|N_t/\sigma_{t|W}| > 1.96 \tag{5.1}$$

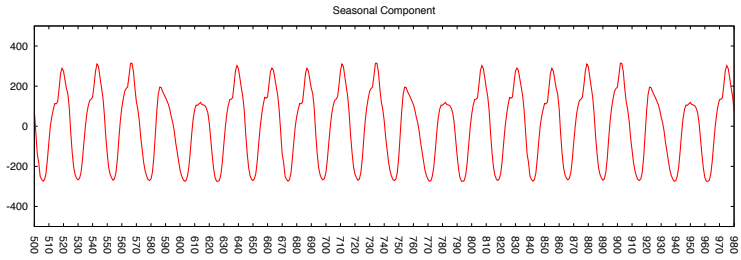
This is consistent with the classic anomaly detection techniques. We also apply an iterative process such that we remove the anomalous points in the previous iteration



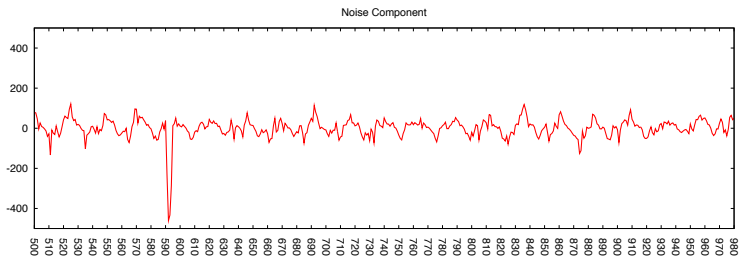
(a) Observations



(b) Trend Component



(c) Seasonal Component



(d) Noise Component

Figure 5.2: Time series decomposition example on number-of-UEs

from the trends and noise variance computation, which makes our approach robust to bad data or significantly bad anomalies.

Reverting back to our application, for a given tower outage, we apply the decompo-

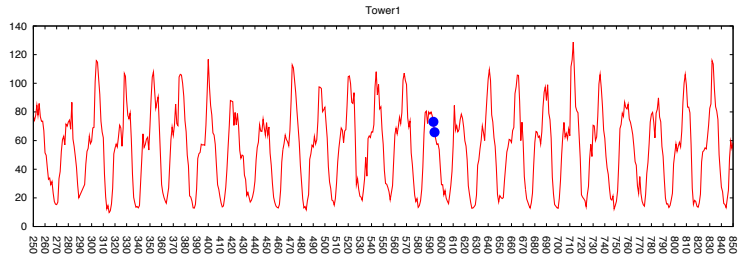
sition technique to detect significant deviation from expectation on the number-of-UEs metric measured at the surrounding towers during the outage time. To ensure that we include all impacted towers, we use a fairly large radius (e.g., 10 miles) from the tower under outage, and we recursively include additional towers for consideration if there exists other towers under outage within the included region and having overlapping outage time. In this way, TowerScan can track the compounded service impact due to multiple tower outages colliding in time and space. We set the seasonality window, W, to one week as our data demonstrate strong time-of-day and day-of-week patterns.

Figure 5.3 shows three examples regarding the number-of-UEs on some surrounding towers to where a two-hour outage took place. Each data point corresponds to a one-hour aggregate. The outage was during hour 592 and 593 (marked with big dots in the graphs). Figure 5.3 (a) is an example tower where no anomaly is found and Figure 5.3 (b) shows a significant dip and Figure 5.3 (c) shows a significant spike anomaly. We should note that considering the day-of-week pattern is critical here. In Figure 5.3 (c), the spike is not very distinguishable compared to the day before. However, it is very significant when compared to a week before (or after).

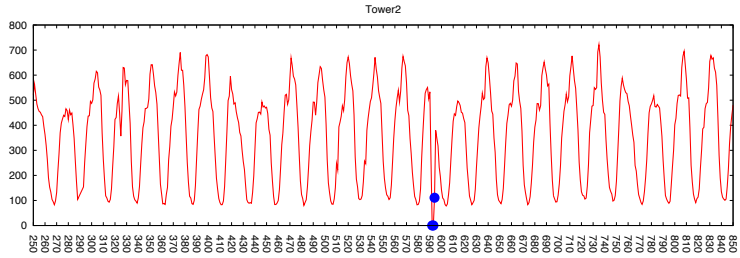
5.2.2.2 Scoping the Cascading Impact

With the technique in subsection 5.2.2.1, we can now annotate all surrounding towers with their status measured in number-of-UEs. Figure 5.4 shows an example of towers under outage and the surrounding towers (within 10 miles from outage). The rectangles are the towers under outage. The crosses are towers without significant change in number-of-UEs. The triangles are towers with an increase (spike anomaly), and the circles are towers with a decrease (dip anomaly) in number-of-UEs. Some markers are overlapping to each other as there are co-located towers at the same spot. Our goal is to scope the impact of outage in an automated fashion.

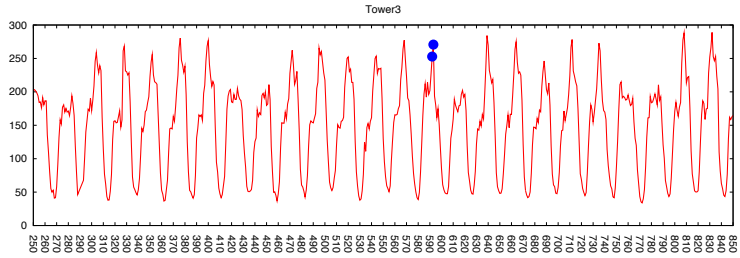
We first make two important observations from the above example.



(a) No Anomaly



(b) Dip Anomaly



(c) Spike Anomaly

Figure 5.3: Three possibilities regarding the status of number-of-UEs on surrounding towers

- First, besides towers under outage, there exist neighboring towers that have significantly decreased number-of-UEs (circles). One would expect the "workload" of the tower under outage would be redistributed to other towers, hence neighboring towers should have increased number-of-UEs. We have investigated into those cases and found two reasons why sometimes some neighboring towers have significantly decreased number-of-UEs. On common reason is that the source that we extract the signature of tower outage is not 100% reliable. Another reason is that some towers only suffer from partial loss of capacity (e.g., losing three out

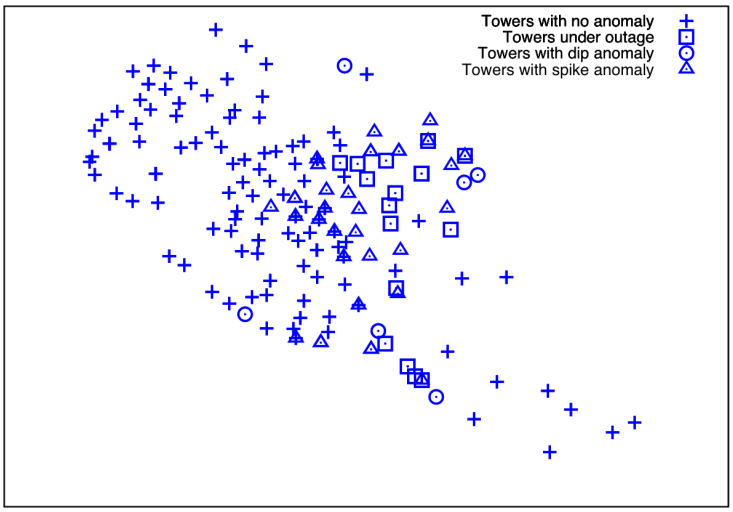


Figure 5.4: Annotated graph of the surrounding towers to a cluster of tower outages. X-axis and Y-axis are longitude and latitude respectively.

of six sectors (or roughly an antenna)). We have confirmed with network operators that some circles in the example are due to missing tower outage signature in the logs and others can be explained by "partial outages" (The number-of-UEs decreases significantly but not to zero). Fortunately, TowerScan is quite robust against these corner cases as any neighboring tower with significant changes in number-of-UEs will be included in the following analysis no matter whether it is identified as a outage or not.

- Secondly, the towers that have significantly increased number-of-UEs are NOT always the closest to the towers under outage. There can be crosses (no-change in number-of-UEs) between the rectangles (towers under outage) and triangles (towers with increased number-of-UEs). We have drilled down into several cases and found that the closest tower to the outage were already near its capacity, hence the extra workload were likely passed through to towers farther away (through rebalancing of UEs in between).

Based on the second observation, we conclude that we cannot simply use the union

of towers with significant deviation in the number-of-UEs as the impacted scope. The towers that remain unchanged in the total number-of-UEs (e.g., due to capacity limit) may associate to a sub-optimal set of UEs during the outage, and passing through the extra workload to its neighbors. The annotation of "no significant change" in the total number-of-UEs hence does not equate to "no service impact". Furthermore, data quality problem in the performance counters or the false negatives in the time series decomposition algorithm can also produce wrong tower annotation. As a result, we seek a robust method to scoping the impact region.

We draw some similarity between this impact region identification problem to the blob extraction problem [29] in the domain of computer vision, which involves identifying a continuous area with largely similar color. A commonly used technique in computer vision is to first apply some low-pass filter on the image of interest (to denoise) and then expand through adjacent pixels to track the border of the desired blob. We borrow this idea in our application and design a three-step approach.

- **Step1: Diffusion Low-pass Filter:** We adopt the diffusion analysis technique to denoise the tower annotation. At a high level, diffusion describes the process of mass/energy spreading from regions of higher concentration to regions of lower concentration, which draws a parallel of UEs settling to find available resources when an outage takes place.

  We set the diffusion process as follows. We define the heat source vector S_i for each tower i in the area: $S_i = 1$ iff tower i is under outage or its number-of-UEs significantly deviates from expectation. We adopt the 2D Gaussian kernel function with the width parameter $\sigma = \frac{5}{1.96}$ so that two towers 5 miles apart have negligible impact on each other. We then define an energy conserving diffusion transition matrix (with a cut-off of 5 miles):

$$A[i,j] = \frac{f(d(i,j))}{\sum_{j \text{ within 5 miles from } i} f(d(i,j))}$$

where $d(i, j)$ is the Euclidean distance between towers i and j, and $f(\cdot)$ is the diffusion kernel function. The detailed diffusion process is described in Algorithm 2.

Algorithm 2: : Diffusion Algorithm

Let H_0 denote the source heat vector.
Let A denote the transition matrix.
Let ϵ denote the convergence threshold.
Let α denote the reinforcement parameter.
$H = H_0$
$H_1 = H_0 + \epsilon$
while sum$(|H_1 - H|) \geq \epsilon$ **do**
  $H_1 = H$
  $H = H \times A$
  $H = \alpha(H \cdot H)$sum$(H_0)/$sum$(H \cdot H) + (1 - \alpha)H_0$
  $H($find$(H > 1)) = 1$
end while

Note that the above diffusion process captures both the distance information between towers and the tower density information within the area. The parameters in Algorithm 2 is set as the following: convergence threshold $\epsilon = 10^{-3}$ and reinforcement parameter $\alpha = 0.5$.

- **Step2: Separating Foreground and Background:** We then determine the towers under impact based on their denoised change-significance values (in number-of-UEs). This is equivalent to distinguishing the foreground color and the background color in computer vision. We adopt the one-dimensional k-mean thresholding approach [64], which is a commonly used method in turning a gray-scale image into a binary image.

  Once a threshold is determined, we can then re-annotate the towers according to H – this is simply marking the towers with H value higher than the threshold as under service impact.

- **Step 3: Region Extraction:** To extract the impact region from the annotated tower map is equivalent to extracting blob from image in computer vision. However, here we do not have the concept of pixels, as towers are most likely non-evenly distributed in space. To compensate for this, we define a connection graph of the towers as follows.

  On each tower, we divide its surrounding space by six 60° sectors. Within each sector, we identify the closest tower in term of the Euclidean distance (with an upper bound of 5-miles) and add an edge between the two towers. In this way, we obtain a tower graph with annotated nodes. We can then simply apply Connected Component Labeling algorithm [30] to extract the impact region.

  Figure 5.5 shows the source heat vector generated from Figure 5.4. The gray dots are the failed towers (rectangles) and the surrounding towers with significant anomaly in number-of-UEs (triangles and circles). The black dots are the surrounding towers without significant changes (crosses). After applying diffusion and thresholding on Figure 5.5, all towers deemed under impact are shown as gray dots in Figure 5.6.

  Figure 5.7 shows the results of region extraction. Two regions are extracted: one includes most of the gray dots in Figure 5.6 and another only has one gray dot.

5.2.3 Quantifying Service Impact

Once TowerScan determines the impact region, it becomes relatively straightforward to track a set of service and performance metrics defined within the region to numerically quantify the service impact. As mentioned in Section 5.1.4, these metrics can be broadly divided into a few categories – service *availability/accessibility*, *retainability*, *mobility*, and *performance*. We list below a few of the most frequently referenced metrics for user impact quantification.

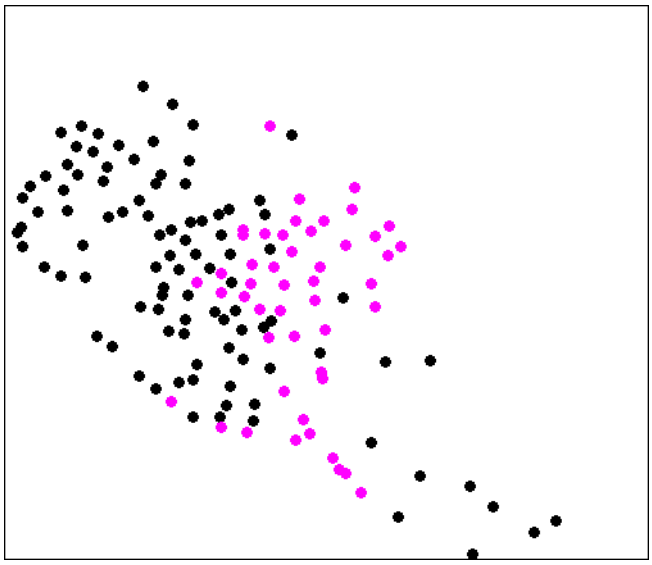


Figure 5.5: The source heat vector displayed on a 2-dimension map. X-axis and Y-axis are longitude and latitude respectively.

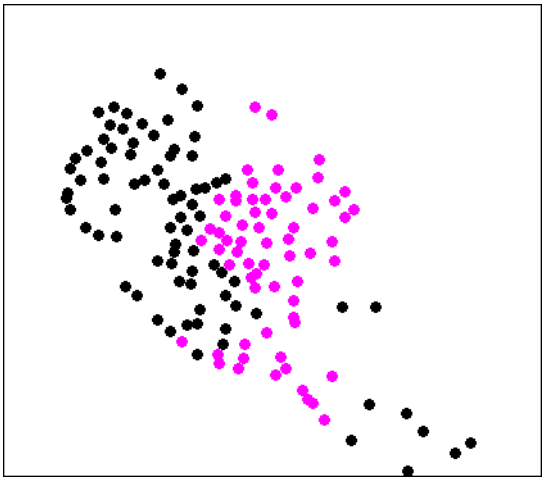


Figure 5.6: The result heat vector after diffusion and thresholding displayed on a 2-dimension map. X-axis and Y-axis are longitude and latitude respectively.

- **affected population** — This metric is derived from population density per zip code based on publicly available census data, and the area estimate of the impact region. The population density information is updated when new census information becomes available.

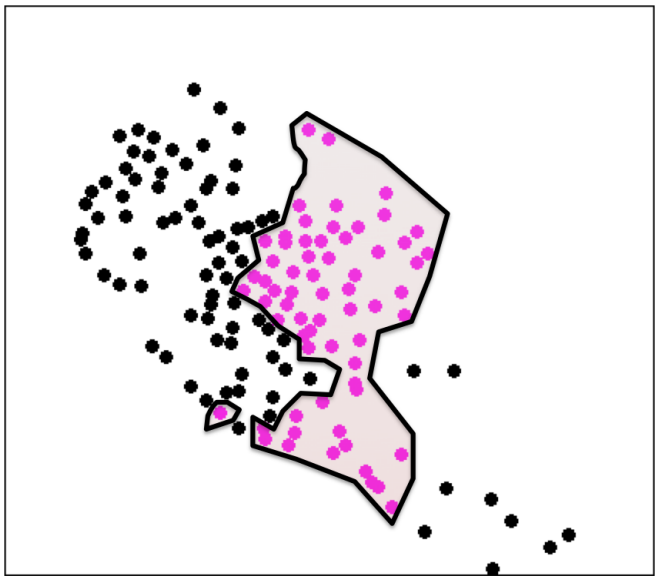


Figure 5.7: Two Regions are extracted based on Figure 5.6.

- **affected primary residing customers** — This metric pertains to the customer population as opposed to the general population. Mobile customers are attributed to the nearest tower(s) based on their home/work address maintained in their profile. The total primary residing (either home or work) customers count is computed by summing up the number of customers over all towers in the impact region. The customer-to-tower mapping can be updated weekly or monthly.

- / loss of 3G mobile users — This metric is derived using the total number-of-UEs in the impact region by comparing the observed value against the expected value without outage. The values are obtained from UMTS counter at a 15-minute update interval.

- **performance degradations** — drop in RRC connection setup success rate for data/voice calls, drop in RAB assignment success rate for data/voice calls, drop in retainability rate for data/voice calls, increase in dropped call rate for data/voice calls, drop in radio link layer throughput, drop in MAC (Medium Access Control)

layer throughput — Similar to the above, these metrics are also derived from UMTS counter values, quantifying various aspects of service quality degradation.

- **increase in customer complaints** — This metric is based on mobile users' feedback through a smartphone application that notifies the mobility service provider of their bad user experience (e.g., dropped calls). The timestamp and tower information are attached in the feedback, which can be used to associate the complaints to the impact region.

While the affected population and affected primary residing customers are associated with rather static information regarding the towers within the impact region, other metrics in the above list are dynamically changing overtime. For the dynamic ones, TowerScan first computes the hourly time series from the counter values pertaining to the impact region from historical data – 30 days in the default setting. TowerScan then utilizes the time series decomposition technique as described in Section 5.2.2 and combines the secular trend and the seasonal components as the expected counter value during the outage. Depending on the metrics of interest, either the average or the total difference between the observed value and the expected value (i.e., the noise component) during the outage duration is used as the quantifier of the service impact. The same significance test as in Section 5.2.2 can be applied here to declare that there is a significant degradation in the metrics of interest if all data points during the outage are marked as anomalous.

5.3 TowerScan Evaluation

In this section, we evaluate TowerScan using both simulation and operational data from a large 3G operational network. We focus on demonstrating that TowerScan is able to identify the impact regions accurately. In order to evaluate the accuracy of TowerScan, it would be ideal to know exactly how individual users move and are assigned to towers.

However, this type of user-level information is hard to access in reality due to both privacy and technical issues. We address this issue by using simulation. We also evaluate TowerScan with operational data from a large 3G operational network.

5.3.1 Evaluation of TowerScan via Simulation

In the simulation, we examine how users are assigned to different towers due to outages by comparing two parallel simulation runs: one without outages and another with injected outages. An accurately identified impact region should mostly consist of users that are assigned to different towers due to outages.

5.3.1.1 Simulation Setup

Our simulation needs to capture three key aspects.

- User Mobility: $20,000$ users are placed on a 5000 by 5000 square map. We adopt the well-known modified Random Waypoint model [60] to generate the mobility trace for $20,000$ users by using a uniform speed distribution $[1, 20]$ and a 2-second pause time.

- Tower Placement: We place 100 towers on the same 5000 by 5000 square map with the towers evenly spaced.

- User-Tower Association: Users are typically in range of several towers and various conditions are considered to select a tower. We introduce an association function that captures the key aspects.The association function shown in Eq.(5.2) calculates the cost of assigning user u to tower t, where $dist(u, t)$ denotes the distance between u and t, $n(t)$ is the number of current users on tower t and $c(t)$ is the capacity of tower t. The smaller the cost, the more preferred the tower. Despite its simplicity, it takes the two most important factors into account: the distance and the tower load. The smaller α is, the load factor is weighted more.

In our simulation, α is set as 0.05, and the capacity of each tower $c(t)$ is randomly chosen from a uniform distribution $[200, 300]$.

$$A(u,t) = \alpha \times dist(u,t) + (1-\alpha) \times \frac{1}{1-\frac{n(t)}{c(t)}} \tag{5.2}$$

With these three basic components, the simulation is ready to run. However, in order to evaluate TowerScan, we still need to figure out how to inject tower outages into the simulation. The most straightforward way is to randomly pick n towers to fail at a given time in the simulation. But in reality the tower outages in operational 3G networks are more likely to be clustered in space due to common shared-risk components (e.g.: upstream device) other than pure random. Therefore, we use the "Gravity" model to fail towers.

In the gravity model, a tower is more likely to be picked to fail if many of its neighboring towers are already failed. The gravity model is an iterative process that the failure probability of all towers are updated after a tower is picked to fail. Specifically, the initial failure probabilities of all the towers is the same $\frac{1}{N}$, where N is the number of towers. After a tower x is picked to fail, the failure probabilities of all the neighboring towers within a 10-mile radius are incremented by $\frac{1}{d(x,i)^2}$, where $d(x,i)$ is the distance between tower x and its neighbor i.

We compare the gravity model and the random approach with operational data, we first get a list of thousands of tower outages observed during a 3-month time window from a large operational 3G network. Then we generate another two lists of tower outages by replacing the towers in the operational data with the towers picked using the random approach and the gravity model. Figure 5.8 shows the PDF of spatial cluster size for all the three lists of tower outages. It shows clearly that the gravity model is far closer to the operational data. Thus, we use the gravity model to generate tower outages in our simulation.

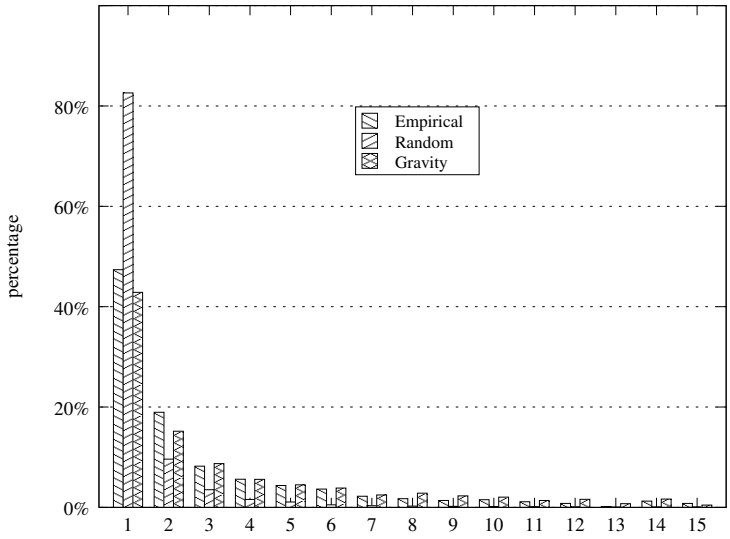


Figure 5.8: Distribution of outage cluster sizes

5.3.1.2 Evaluation Strategy

To evaluate TowerScan thoroughly, we compare the accuracy of identifying impact regions across three approaches:

- Fixed-radius approach: identifies the impact region by simply including all the neighboring towers within the n hops radius of all failed towers. For example. for a tower in the center of the map, the 1 hop radius includes 4 neighboring towers and 1.414 hop radius covers 8 neighboring towers. We vary the parameter n from 1 to 5.

- TowerScan without diffusion: is described in Section 3.2 but we intendedly skip the step 1 (diffusion low-pass filter described in Section 5.2.2.2) to illustrate the importance of diffusion. We vary the critical values in 5.1 from 0.1 to 4. For the diffusion low-pass filter, we use the Gaussian kernel with σ of 2.5 (hops).

- TowerScan: is exactly the same as described in Section 3.2; in other words the complete approach. The parameters used in this approach are the same as the "TowerScan without diffusion" approach.

To verify the results from different approaches, we first identify which users are assigned to different towers as a result of the tower outages (and not due to normal mobility). Specifically, we generate the mobility trace of $20,000$ users for $200,000$ seconds using the parameters mentioned in Section 5.3.1.1. At the end of each 1000 seconds interval, the user-tower association is reset and each user is assigned to the best tower according to Eq.(5.2) one by one. At the $100th$ interval, we keep two versions of user-tower association. One is without failures while another is injected with three tower outages according to the gravity model. By comparing the two versions of user-tower association at the $100th$ interval, we know exactly the set of users that are assigned to different towers due to tower outages.

Each user on the map is either labeled as positive (P) class if he/she is assigned to a different tower due to the outages or labeled as negative (N) class otherwise. A positive user inside the identified impact region is classified as a true positive (TP). A negative user inside the identified impact region is classified as a false positive (TP). The accuracy of an identified impact region can then be measured using the false positive rate ($\frac{FP}{N}$) and true positive rate($\frac{TP}{P}$). A good approach should achieve high true positive rate while keeping false positive rate low.

5.3.1.3 Evaluation Results

In this simulation, we repeat the gravity-model based failure injection 100 times and calculate the average false positive rate and true positive rate. We use a receiver operating characteristic (ROC) curve [65] to present the relation between false positive rate and true positive rate across three approaches in one plot. The left-up corner in ROC curve is the golden area, where the false positive rate is low and true positive rate is high. Figure 5.9 shows the ROC curves for the three approaches respectively. The overall trends in the three curves exhibit the fundamental tradeoff between false positive rate and true positive rate. TowerScan achieves the best tradeoff as it always gives the best true posi-

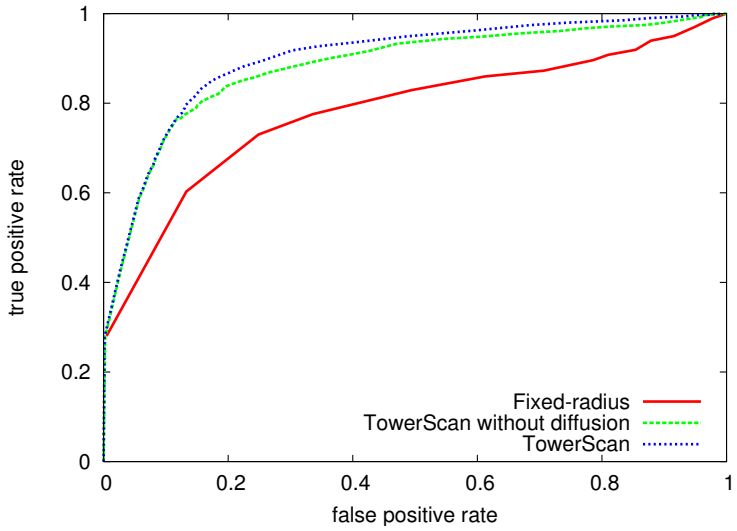


Figure 5.9: A comparison of three approaches in ROC curves

tive rate for any given false positive rate compared with the other two approaches. Both TowerScan and "TowerScan without diffusion" outperforms the fixed-radius approach as they keep track of the dynamic user redistribution more accurately. Interestingly, the diffusion starts to take effect after false positive rate is larger than 0.1 (the two curves for TowerScan and "TowerScan without diffusion" start to diverge when x-axis is larger than 0.1). For very conservative parameters (low false positive rate and low true positive rate), very few towers are detected with significant changes (Section 5.2.2.1) in number of users and thus diffusion makes no difference.

5.3.2 Evaluation of TowerScan using Operational Data

In this section, we first discuss how we use operational data to evaluate the accuracy of TowerScan and then present the evaluation results.

5.3.2.1 Evaluation Strategy

With operational data, we verify the accuracy of TowerScan by comparing the performance metrics on two sets of towers close to the border: one set is the towers INSIDE

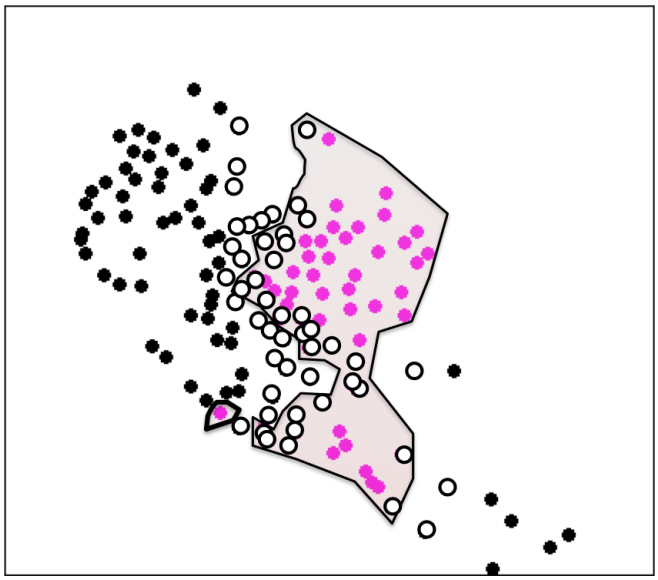


Figure 5.10: Compare the unfilled circles inside the region and the unfilled circles outside the region to verify the accuracy of identified impacted region

the identified region while another set are the towers OUTSIDE the identified region. In other words, we compare the towers are on both sides of the identified region's border. For example. in order to verify one (the larger one) of the two identified impacted regions in Figure 5.7, we compare the set of unfilled circles inside the region with another set of unfilled circles outside the region as show in Figure 5.10. We call them inner towers and outer towers in short.

An impact region is considered to be identified accurately when the aggregate performance metrics (discussed in Section 5.2.3) degrade significantly on the inner ring but not on the outer ring . As shown in Figure 5.11, on all of the four performance metrics, the inner ring indeed manifests significant degradation during the outage (shown as the black dots) while the outer ring does not – bringing confidence to our result.

Based on the insight from the above example, we define a "contrast ratio" metric to characterize the quality of the computed impact regions. Given a performance metric, we compute the aggregate value for the towers in the inner and outer rings respectively. We then divide the higher of the two values by the lower one and refer to the quotient

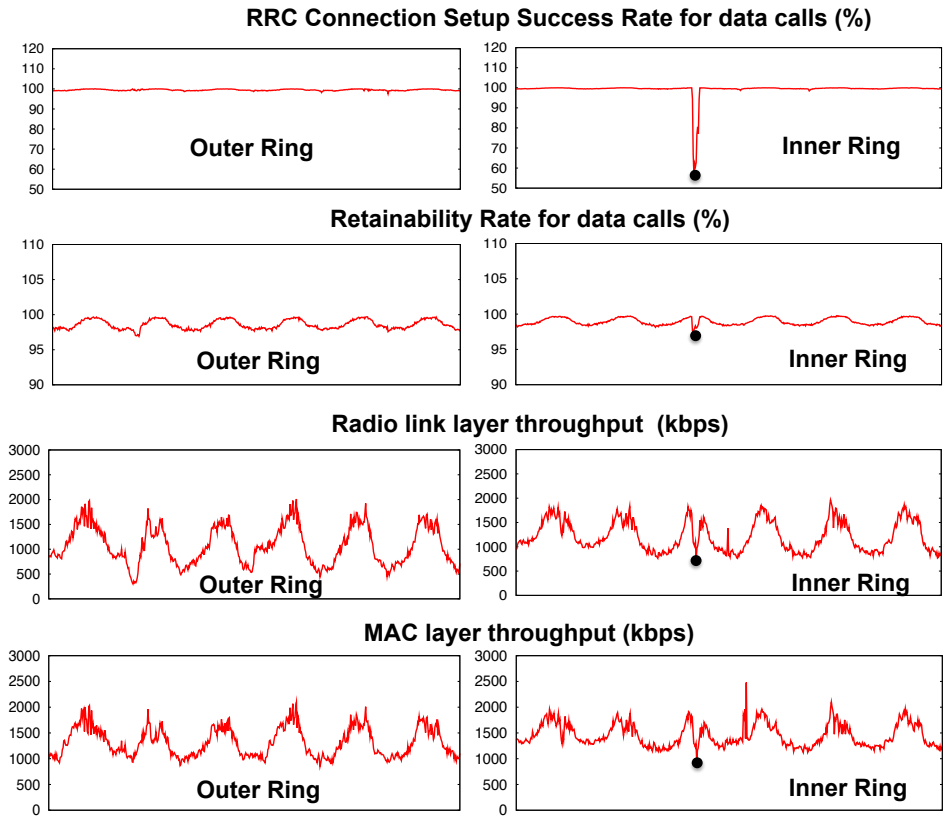


Figure 5.11: Performance comparisons between inner towers and outer towers from Figure 5.10

as the contrast ratio (as the direction of improvement differs across metrics – larger value being better versus lower being better). The higher the contrast ratio is, the more accurate the identified impact region is.

5.3.2.2 Evaluation Results

We first apply TowerScan on three months' worth operational data and TowerScan identifies hundreds of impact regions from thousands of tower outages. Then we run the fixed-radius approach on the same data set. Similar to the fixed-radius approach used in simulation, here the fixed-radius approach identifies the impact region by simply including all the neighboring towers within the n miles radius of all failed towers. For TowerScan we use the default parameters mentioned in Section 5.2.2 to identify the regions

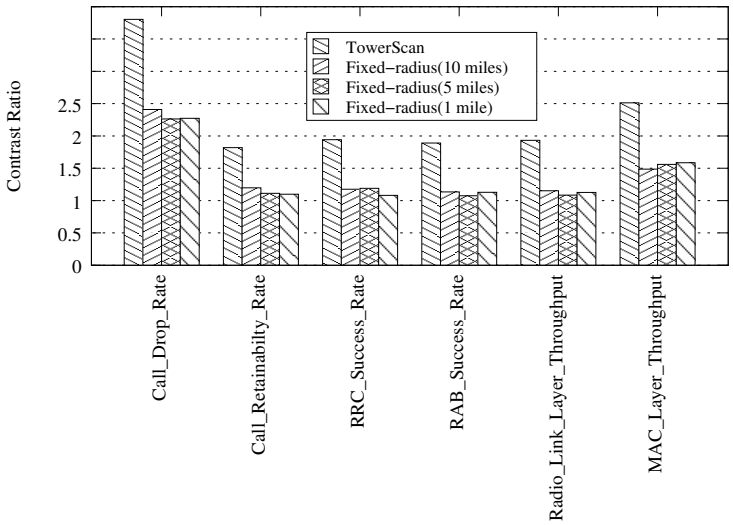


Figure 5.12: Average "contrast ratio" comparison of TowerScan and the fixed-radius approach

while we try three different radiuses 10 miles, 5 miles and 1 mile in the fixed-radius approach. Figure 5.12 shows the average "contrast ratio" on six different performance metrics described in Section 5.2.3. TowerScan approach clearly achieves better "contrast ratio" in comparison to the fixed-radius approaches, indicating that TowerScan can scope the impact due to tower outages more accurately and tightly.

5.4 TowerScan Operational Experiences

In this section, we describe two interesting case studies regarding two clusters of tower outages. We applied TowerScan to quantify the service impact in both cases and compared the result with the impact analysis reports manually generated by operators. Specifically, we found the manual analysis wrongly prioritized their relative impact compared to TowerScan result and operators concurred with our findings after examining the TowerScan output.

5.4.1 Case study 1: Severe User Impact

In this first case study, we focus on a cluster of tower outages that have a severe user impact according to TowerScan. TowerScan detected a cluster of 18 tower outages that lasted for about two and half hours. Figure 5.13 shows the 18 towers (rectangles) under outage and the status of their neighboring towers generated by TowerScan. Besides the tower under complete outage, TowerScan also found many towers (circles) under "partial outage" (not all sectors were out of service), which means they also experienced significantly decreased number of UEs. The towers marked with triangles are those with significantly increased number of UEs. This indicates the "workload" of the tower under outage or "partial outage" was redistributed to nearby towers.

Based on the annotation of number-of-UEs, TowerScan then identified the impact region by considering all nearby towers with significant changes in the number of UEs. Figure 5.14 illustrates how the impact region is scoped through diffusion, thresholding and region extraction. The identified impact region is much larger than the set of towers under outages. Further, TowerScan looked into the identified impact region (the bottom plot in Figure 5.14) to quantify the impact on different metrics discuss in Section 5.2.3.

In this case, TowerScan declared the daily numbers of affected primary residing customers inside this region were 80815 and 80832 for work hours and non-work hours respectively. As the outage occurred during the non-work hours, potentially 80815 users could be impacted. Regarding to the dynamic metrics, Figure 5.15 shows the details of various dynamic metrics on the aggregate region level before and after the tower failures. We can see clearly from the top plot of Figure 5.15 that more than 1000 users were dropped off from the 3G network during the outage, which should be considered as a severe impact. For the users remain on the 3G network during the outage, TowerScan declared that performance degradation is statistically significant for all metrics regarding accessibility, retainability and call quality as mentioned in Section 5.2.3. In particular,

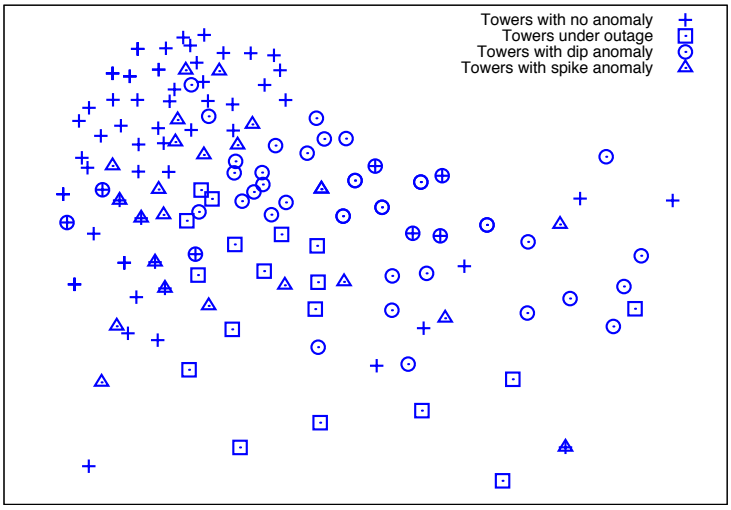


Figure 5.13: number-of-UEs based annotated graph of the surrounding towers to a cluster of tower outages. X-axis and Y-axis are longitude and latitude respectively.

"RRC connection setup success rate" degraded from almost 100% to less than 40% and " RAB assignment success rate" decreased from around 100% to 90% for data calls. Thus, this cluster of tower outages caused severe user impact for two and half hours.

5.4.2 Case study 2: Moderate User Impact

In the second case study, we focus on a cluster of tower outages that have a moderate impact according to TowerScan. TowerScan detected a cluster of 32 tower outages that lasted for about three and half hours. Similar to case study 1, TowerScan first checked the status of their neighboring towers using the key metric number-of-UEs as shown in Figure 5.16. Based on the map of the status of number-of-UEs (Figure 5.16), TowerScan then identified the impact region through diffusion, thresholding and region extraction, as shown in Figure 5.17 . The identified impact region contains not only the towers under outage but also the neighboring towers with significant changes in the number of UEs.

As the final step, TowerScan looked into the identified impact region to quantify the

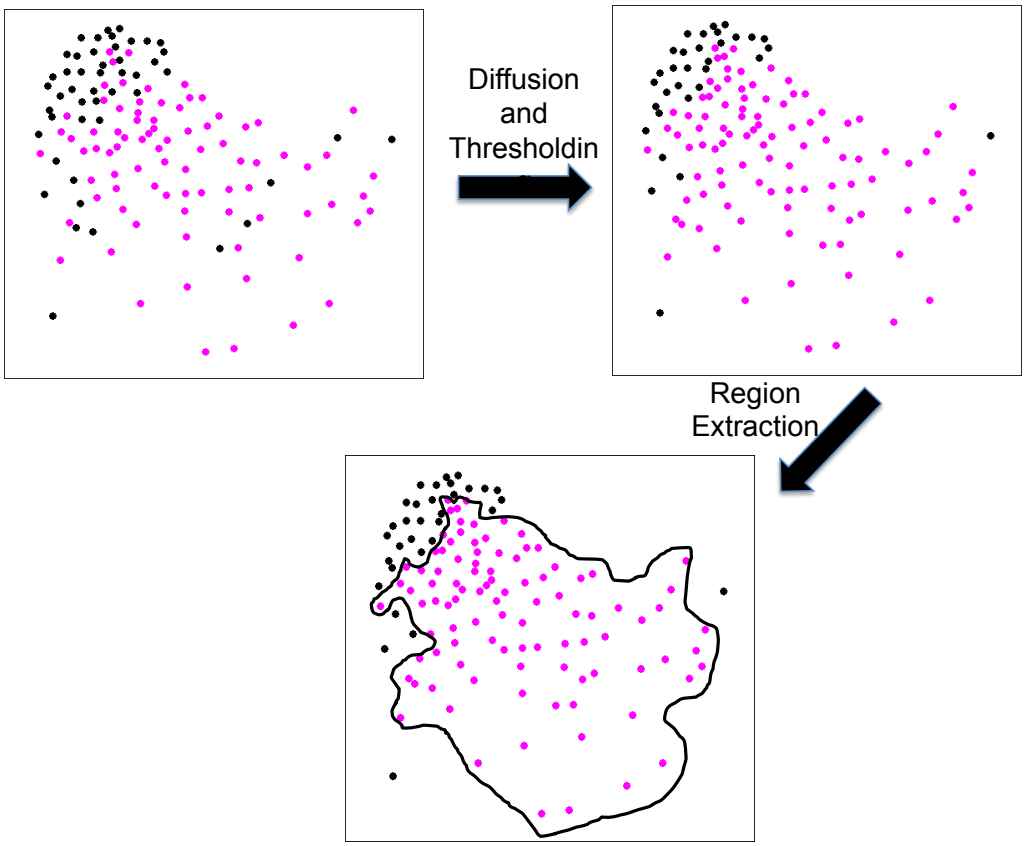


Figure 5.14: The source heat vector, result heat vector and extracted region. X-axis and Y-axis are longitude and latitude respectively.

impact on different static and dynamic metrics. The daily numbers of primary residing customers inside this region were 24441 and 27540 for work hours and non-work hours respectively. As the outage also occurred during the non-work hours, 24441 customers were potentially impacted. By comparing this static metric, this incident has less severe impact compared with the one in case study 1. The details of dynamic metrics indicate the same conclusion as shown in Figure 5.18. Interestingly, the number of 3G users didn't change in this case, which is different from what we have seen in case study 1. This means most of the users were staying on the 3G network as opposed to failing over to 2G. Although TowerScan declared the performance degradation on this region for all other dynamic metrics too, the degradation here is not as severe as in case study 1. For

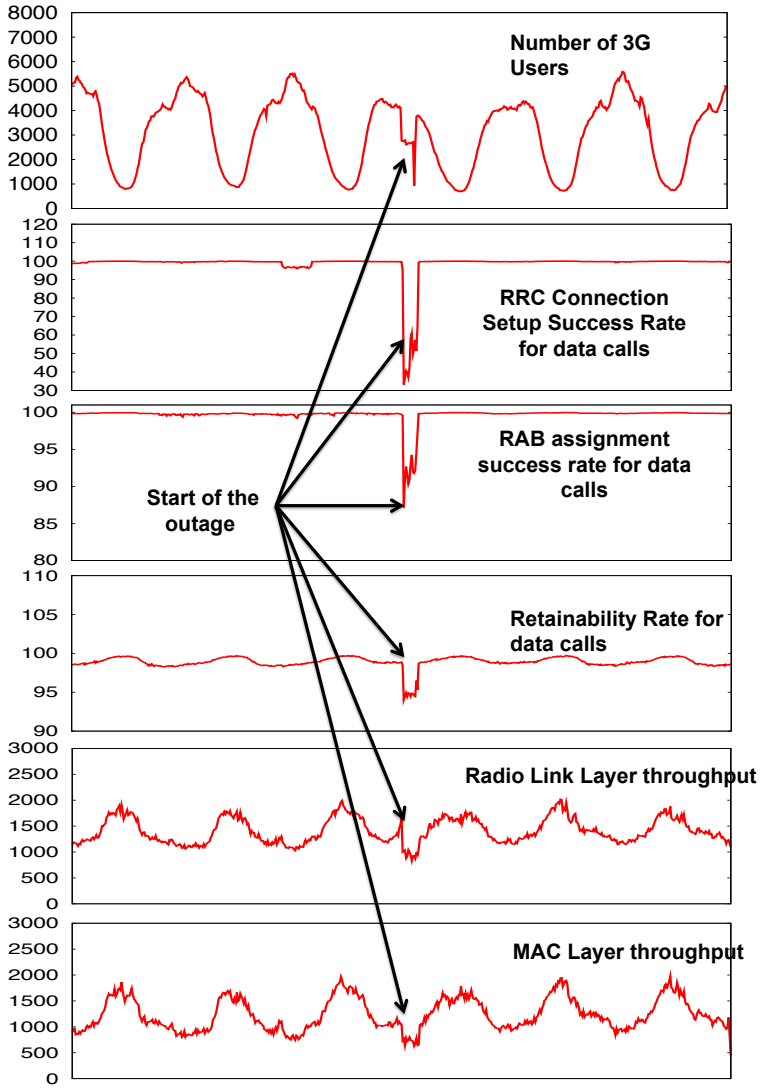


Figure 5.15: Time series of various user-impact metrics on the aggregate region level.

example, in this case, "RRC connection setup success rate" degraded from almost 100% to about 95% for data calls while in case study 1 it degraded to less than 40%. Combining the above observations, TowerScan concluded that this cluster of tower outages caused moderate user impact.

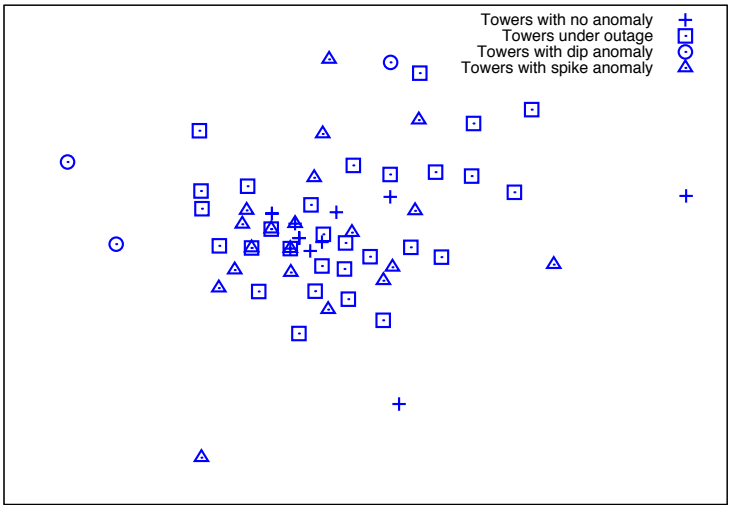


Figure 5.16: number-of-UEs based annotated graph of the surrounding towers to a cluster of tower outages. X-axis and Y-axis are longitude and latitude respectively.

5.4.3 Comparison with manual impact analysis reports from operators

The same two incidents were analyzed manually, but the conclusions of the impact analysis reports were the opposite to TowerScan. More specifically, the manual reports concluded that the incident in case study 2 had more severe user impact that case study 1, and hence assigned the incident a higher fix priority. This erroneous conclusions were due to the focus on just the failed number of towers, outage duration and population density of the region under outage. More specifically, the incident in case study 2 was flagged as more severe due to (a) more towers under outage (32 vs 18), (b) longer duration (3.5 hours vs 2.5 hours) and (c) denser population (1452 vs 1142 per square mile).

In a contrast, TowerScan generates the opposite result. All static and dynamic metrics support this conclusion as we discussed in the above two case studies. These results suggest that without considering user mobility, service redundancy, neighboring towers and fine granularity metrics, the estimation of user impact could be off, or even wrong.

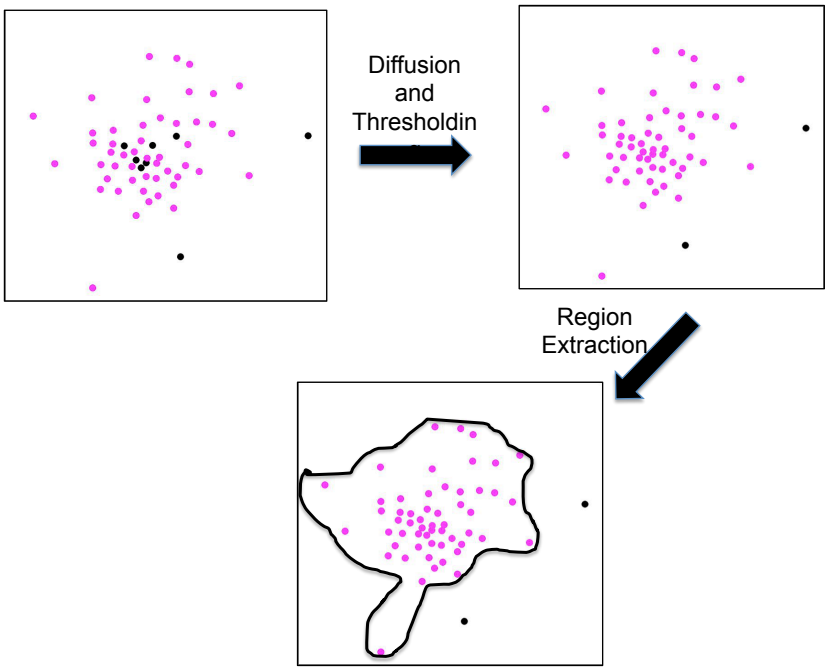


Figure 5.17: The source heat vector, result heat vector and extracted region. X-axis and Y-axis are longitude and latitude respectively.

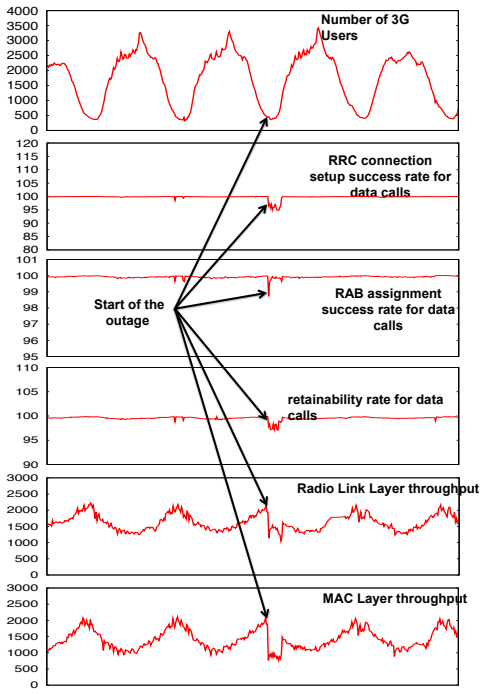


Figure 5.18: Time series of various user-impact metrics on the aggregate region level.

Chapter 6

Future Work

In the chapter, we will discuss the future work in managing user-perceived quality of Internet-based service along both top-down and bottom-up direction.

6.1 Detecting Service Anomaly Events and Estimating their User Impacts

Our future work in this step can be divided into two directions.

- First, we plan to extend the current univariate anomaly detection on a single metric to the multi-variate anomaly detection on multiple correlated metrics. In this dissertation, we have been focusing on a single metric in this step, such negative feedbacks from users and RTTs passively measured from traffic. The next intriguing research question is how to gain more insights into the nature of the service degradations by analyzing multiple different metrics.

  The most natural approach is to first detect anomalies in multiple metrics separately in parallel and then analyze the detected anomalies across different metrics in a post-processing manner. In this way, more insights can be obtained regarding the nature of the service quality issues. For example, if anomalies are detected for RTT and throughput simultaneously, they are more likely to be caused by routing changes or detours. If anomalies are detected for tcp retransmission rate

and throughput simultaneously, they are more likely to be link congestion/error related.

Another alternative direction is to combine the information from different metrics in the anomaly detection phase. Instead of doing anomaly detection on a univariate metric time series as we discussed in Argus, it is may desirable to conduct anomaly detection on a multi-variate metric time series to increase the sensitivity of anomaly detection while maintaining false alarm rate. For example, By analyzing the multi-variate metric time series, one can detect the anomalies that are not visible by looking at the different metrics separately such as the changes in the relationships among different metrics (e.g.: Throughput decreases when RTT increases) or changes that are significant enough only if we look at the different metrics simultaneously.

- Secondly, we plan to incorporate the probability model into user-group hierarchy to estimate the impact scope more accruately. For example, all the users from the same BGP prefixes may take different AS paths to reach the servers in a CDN service at different times due to the dynamics in the routing or load balancing.

  The current event localization logic in Argus is based on a static user-group hierarchy (as shown Figure 4.13) and is not able to handle the uncertainties in the above example. In the static user-group hierarchy, each BGP prefix user-group only has a single AS path user-group as the parent node. To model the uncertainties, we can possibly let the BGP prefix user-group have multiple AS path user-groups, each of which is associate with certain probabilities. Then we also need to change the ELP formulation and the heuristic to handle the probabilities defined in the user-group hierarchy.

6.2 Identifying the Root Causes of Service Quality Degradations

Our work in this step can be extended in several directions. First, we plan to make the temporal joining rules less sensitive for robust root cause analysis and deal with the cyclic causal relationship in diagnosis rules. Second, we plan to refine the inference algorithm and simplify its configuration to further improve its usability. Third, we want to support realtime root cause applications. The main challenge in conducting root cause analysis in realtime is how to deal with different delays in different data sources.

6.3 Understanding the Impact of Network Events on Service Quality

In the future, we plan to extend TowerScan in following two directions. First, we plan to extend TowerScan to better understand the cascading "cross-network" user impact when users are migrated from 4G/LTE to 3G to 2G. For example, existing 2G users might have degrading performance or even fall out of the 2G network when the 3G users on the failed 3G towers are migrated into 2G network. Secondly, once TowerScan understands the "cross-network" user impact, we may be able to provide key information for operators to optimize the failover strategy. If TowerScan reports that failover from LTE to 3G has caused significant impact on both LTE and 3G users, Operators may rethink about the failover strategy. For example, instead of push all the LTE users on the failed LTE towers to their 3G network, may be they should migrate some of them to other providers' LTE network to avoid this kind of cascading impact. Secondly, TowerScan currently simply presents all service and performance metrics separately to users. But in order to use TowerScan to prioritize incidences in a more efficient manner, a single impact index by combining all service and performance metrics is desirable. Thirdly, we will extend TowerScan to a generic infrastructure for various services by

leveraging the lessons learned in TowerScan regarding how to map the network events to user-perceived service quality by taking into account service redundancy.

Chapter 7

Conclusion

The Internet has become the mainstay of many networked services (e.g., content distribution network (CDN), VoIP, VPN, IPTV and mobile phone). User-perceived service quality is the most important metric in evaluating these Internet-based services and largely decides the reputation and revenue for the service providers. In this dissertation, we explored two directions in managing user-perceived quality of Internet-based services.

7.1 Top-Down Direction Summary

The top-down approach starts with user-view measurements (e.g.: complains from Twitter, customer calls and download throughput) to first detect the service quality degradations, then aims to understand the user impact (e.g.: who are impacted? by how much? for how long?), and finally finds the root causes (e.g.: link congestion and router CPU overload) that can be extracted from network-view measurements. On the other hand, the bottom-up approach starts with network-view measurements to detect network events (e.g.: link congestion and router CPU overload) that may have been the root causes of some service quality issues.

Two systems are designed and implemented to detect service anomaly events, estimate their user impacts and identity the their root causes.

- Argus is a generic system to detect service anomaly events and estimate their user impacts base on the user-view service quality measurements collected from individual users. It turns the service quality measurements from individual users into localized and prioritized service anomaly events in streaming fashion.

  Argus has been deployed in the CDN service and mobility service managed by the same service provider. Argus has proven itself effective and accurate in detecting and localizing real service quality issues before user complain. Service operator started to receive real-time alarms from Argus and react to these alarms on a regular basis. We evaluate the accuracy of Argus through comparison with a set of "labelled anomaly events" from 2 independent data sources (Keynote and DNS) provided by the CDN service operation team. Out of 310 labelled anomaly events from Keynote during a two-month period, Argus successfully detected 91% (281 events) of them and missed only 29 events (9%). We also obtained a list of "labelled anomaly events" from DNS for 15 days. Out of 68 labeled anomaly events from DNS, Argus successfully detected 72% (49 events) and missed 13%. The rest 15% is hard to decide due to lack of data.

- G-RCA is a generic root cause analysis system that is designed to automate troubleshooting and trending the root causes of service quality issues. It starts with imperfect domain knowledge that can be gradually improved. Given the domain knowledge G-RCA first retrieves and correlates networks events given symptoms. Then it identifies the root causes based on the correlated events. It is generic to support diverse services running on top of the same service provider's network by separating common and service-specific components. The common components are implemented once and well, such as capabilities for spatial/temporal correlation and reasoning/inference. Service-specific knowledge captured by event/rule definitions.

The key advantage of G-RCA is its capability to be rapidly customized for various services built on top of the same network. We have customized G-RCA for diagnosing service quality issues in CDN, DNS, mobility service, multicast VPN and customer BGP service. In the CDN service, G-RCA identified the root causes of 25.17% of the RTT anomaly event detected by Argus in one month as network events within the service provider's network (e.g., interface flap, link congestion and CDN assignment policy change) or from outside events that are visible in our network (such as BGP routing changes announced by other ISPs). For the rest of the RTT anomaly event, G-RCA didn't find any evidence from inside our network, which suggests that those RTT degradations may be caused by other ISPs on the end-to-end path.

In the example of customer BGP service, G-RCA identified the root causes of 90.05% BGP flaps over several hundred eBGP sessions established with customer routers. Specifically, 63.94% of them are caused by interface flap and 6.46% of them are caused by high CPU load. As another example, G-RCA has proved to be extremely useful in classifying root causes of PIM adjacency losses and in guiding operators and engineers to a better understanding of actual MVPN performance in the network. For each day, G-RCA is able to identify the root causes for more than 98% of PIM neighbor adjacency changes. For example, a large number (69.21%) of PIM adjacency changes are found to be due to link flaps between the customer and provider routers, which typically are caused by customer activities.

7.2 Bottom-Up Direction Summary

The bottom-up approach needs to determine if a network event has actually caused any service quality issues or not given the service/network redundancy (e.g.: When the primary router fails, the secondary router may work just fine without affecting user per-

ceived service quality.). Finally if the network event has caused service quality issues, similar to the top-down approach, the bottom-up approach needs to understand the user impact (e.g.: who are impacted? by how much? for how long?).

We focus on the 3G mobility service and design a system called TowerScan to quantify the service quality impact of cell tower failures on users. Although our work in the bottom-up direction is not as generic as the top-down direction, TowerScan reveals the common challenges and complexities in the bottom-up direction. It is crucial for service operators to prioritize outages and better schedule fixes with limited resources. If a cell tower fails, mobility and redundancy may result in no perceived user impact if users are migrated to nearby towers with spare capacity. But if nearby towers are heavily loaded, users maybe significantly impacted in terms of accessibility or throughput of data calls. The situation becomes more complex in multiple tower outages. TowerScan first identifies the impacted regions of each cluster of outages by analyzing the number of users served by a tower. For each impacted region, TowerScan provides various user impact estimations at different granularity levels.

We evaluate TowerScan using both simulation and operational data from a large 3G mobility network. In the simulation, we examine how users are assigned to different towers due to outages by comparing two parallel simulation runs: one without outages and another with injected outages. An accurately identified impact region should mostly consist of users that are assigned to different towers due to outages. Each user is either labeled as positive (P) class if he/she is assigned to a different tower due to the outages or labeled as negative (N) class otherwise. A positive user inside the identified impact region is classified as a true positive (TP). A negative user inside the identified impact region is classified as a false positive (TP). TowerScan achieves the best tradeoff as it always gives the best true positive rate for any given false positive rate compared with the fixed-radius approach. For example, when the false positive rate is 0.2, TowerScan

can achieve roughly 0.9 true positive rate while the fixed-radius approach only has 0.6 true positive rate.

With operational data, we verify the accuracy of TowerScan by comparing the performance metrics on two sets of towers close to the border: one set (inner ring) is the towers INSIDE the identified region while another set (outer ring) are the towers OUTSIDE the identified region. In other words, we compare the towers are on both inner ring and outer ring of the identified region's border. We define the contrast ratio metric to characterize the quality of the identified impact regions. Given a performance metric, we compute the aggregate value for the towers in the inner and outer rings respectively. We then divide the higher of the two values by the lower one and refer to the quotient as the contrast ratio (as the direction of improvement differs across metrics larger value being better versus lower being better). The higher the contrast ratio is, the more accurate the identified impact region is. We first apply TowerScan on three months' worth operational data and TowerScan identifies hundreds of impact regions from thousands of tower outages. Then we run the fixed-radius approach on the same data set. TowerScan approach achieves better "contrast ratio" in comparison to the fixed-radius approaches for all six performance metrics.

7.3 Top-Down v.s. Bottom-Up

The top-down approach directly focuses on the user-view measurements such as end-to-end download throughput, complains from Twitter and customer calls. User-view measurements directly and accurately reflect the use-perceived service quality. The main advantage of the Top-Down approach against the bottom-up approach is that we don't need to infer the impact of network events (e.g. router failure, link congestion and routing change) on the use-perceived service quality. But the disadvantage of the top-down approach is that it manages the service quality perceived in a reactive manner.

In other words, user dissatisfaction has been caused in a service when we detect an anomaly (e.g.: slow download throughput and spikes in Twitter complains or customer calls) in user-view measurements.

In the contrast, the bottom-up approach starts from the network-view measurements (e.g.:link congestion, router CPU overload and cell tower outage). The main advantage of the bottom-up approach is that it manages the service quality perceived in a proactive manner. Unlike the top-down approach that reacts to the issues in use-perceived service quality, the bottom-up approach keeps monitoring the network health and determining the impact of a network event on use-perceived service quality. Early detection of network issues could avoid a potential severe user-impacting service quality issue. The disadvantage of the bottom-up approach lies on two facts. On one hand, it is non-trivial to understand the user impact of a network event as we discussed in TowerScan. Understanding the user impact of a network event may vary in different services. Therefore, build a generic bottom-up approach for various services is more challenging than the top-down approach. On the other hand, as there are tons of network events in a large network, understanding the impact of each of them on various services may just not be practical.

Chapter 8

Publications

1. He Yan, Lee Breslau, Zihui Ge, Dan Pei, Jennifer Yates and Dan Massey, G-RCA: A Generic Framework for Rapidly Developing Root Cause Analysis Applications in Large ISP Networks, accepted by IEEE/ACM Transactions on Networking.

2. He Yan, Ashley Flavel, Zihui Ge, Alexandre Gerber, Dan Massey, Christos Papadopoulos, Hiren Shah, Jennifer Yates, "Argus: End-to-End Service Anomaly Detection and Localization From an ISP's Point of View", IEEE INFOCOM (mini-conference), March 2012.

3. Suk-Bok Lee, Dan Pei, MohammadTaghi Hajiaghayi, Ioannis Pefkianakis, Songwu Lu, He Yan, Zihui Ge, Jennifer Yates, Mario Kosseifi, "Threshold Compression for 3G Scalable Monitoring", IEEE INFOCOM, March 2012.

4. He Yan, Benjamin Say, Brendan Sheridan, Dave Oko, Christos Papadopoulos ,Dan Pei and Dan Massey, IP Reachability Differences: Myths and Realities, Global Internet Symposium 2011. Shanhai, China. April 2011.

5. Suk-Bok Lee, Dan Pei, MohammadTaghi Hajiaghayi, Ioannis Pefkianakis, Songwu Lu, He Yan, Zihui Ge, Jennifer Yates and Mario Kosseifi, Scalable Monitoring via Threshold Compression in a Large Operational 3G Network, ACM SIGMETRICS 2011 Poster, June 2011.

6. He Yan, Lee Breslau, Zihui Ge, Dan Pei, Jennifer Yates and Dan Massey, G-RCA: A Generic Framework for Rapidly Developing Root Cause Analysis Applications in Large ISP Networks, ACM CoNext, December 2010.

7. He Yan, Vamsi Kambhampati, Dan Massey and Dan Pei, Analyzing Failures and Attacks in Map & Encap Protocols, in 6th IEEE ICNP Workshop on Secure Network Protocols, October 2010.

8. He Yan, Dan Pei and Dan Massey, Failure Handling in Map&Encap Protocols, AsiaFI Summer School 2009, short paper, August 2009.

9. He Yan, Dan Massey, Earnest McCracken and Lan Wang, BGPMon and NetViews: Real-Time BGP Monitoring System, IEEE INFOCOM, demo, April 2009.

10. He Yan, Ricardo Oliveira, Kevin Burnett, Dave Matthews, Lixia Zhang and Dan Massey, BGPmon: A real-time, scalable, extensible monitoring system, in Proceedings of the Cybersecurity Applications & Technology Conference For Homeland Security, March 2009.

11. Dan Jen, Michael Meisel, He Yan, Dan Massey, Lan Wang, Beichuan Zhang and Lixia Zhang, Towards A Future Internet Architecture: Arguments for Separating Edges from Transit Core, in ACM Workshop on Hot Topics in Networks (HotNets), October 2008.

12. He Yan, Eric Osterweil, Jon Ha jdu, Jonas Acres, and Dan Massey, Limiting Replay Vulnerabilities in DNSSEC, in 4th IEEE ICNP Workshop on Secure Network Protocols, October 2008.

REFERENCES

[1] Akamai, inc. website. http://www.akamai.com/.

[2] A border gateway protocol 4 (bgp-4). http://www.ietf.org/rfc/rfc4271.txt.

[3] Cisco ios bgp command reference. http://www.cisco.com/en/US/docs/ios/iproute/command/reference/irp_bgp1.html#wp1013297.

[4] Emc smarts. http://www.emc.com/products/family/smarts-family.htm.

[5] Gomez, inc. website. http://www.gomez.com/.

[6] Hp openview. http://www.openview.hp.com.

[7] Ibm tivoli. http://www.ibm.com/software/tivoli/.

[8] Keynote systems, inc. website. http://www.keynote.com/.

[9] Limelight, inc. website. http://www.LimelightNetworks.com/.

[10] Overview of Multilink PPP Bundle. http://www.juniper.net/techpubs/software/erx/junose81/swconfig-link/html/mlppp-config2.html.

[11] Overview of Naive Bayes classifier. http://en.wikipedia.org/wiki/Naive_Bayes_classifier.

[12] SONET Automatic Protection Switching. http://www.cisco.com/en/US/tech/tk482/tk606/tsd_technology_support_sub-protocol_home.html.

[13] Using the crowd to monitor the cloud: Detecting network events from edge systems. http://www.aqualab.cs.northwestern.edu/projects/NEWS.html.

[14] P. Bahl, R. Chandra, A. Greenberg, S. Kandula, D. Maltz, and M. Zhang. Towards highly reliable enterprise network services via inference of multi-level dependencies. In *ACM SIGCOMM Computer Communication Review*, volume 37, pages 13–24. ACM, 2007.

[15] P. Bloomfield. *Fourier analysis of time series: an introduction*. Wiley-Interscience, 2004.

[16] A. Blumer, A. Ehrenfeucht, D. Haussler, and M. Warmuth. Occam's razor. *Information processing letters*, 24(6):377–380, 1987.

[17] P. Brockwell and R. Davis. *Time series: theory and methods*. Springer Verlag, 2009.

[18] J. Brutag. Aberrant behavior detection and control in time series for network monitoring. In *Proceedings of the 14th Systems Administration Conference (LISA 2000)*.

[19] C. Chatfield. *The analysis of time series: an introduction*. CRC press, 2004.

[20] M. Chen, E. Kiciman, E. Fratkin, A. Fox, and E. Brewer. Pinpoint: Problem determination in large, dynamic internet services. In *Dependable Systems and Networks, 2002. DSN 2002. Proceedings. International Conference on*, pages 595–604. IEEE, 2002.

[21] X. Chen, M. Zhang, Z. Mao, and P. Bahl. Automating network application dependency discovery: Experiences, limitations, and new solutions. In *Proceedings of the 8th USENIX conference on Operating systems design and implementation*, pages 117–130. USENIX Association, 2008.

[22] I. Cohen, M. Goldszmidt, T. Kelly, and J. Symons. Correlating instrumentation data to system states: A building block for automated diagnosis and control. In *In OSDI*, pages 231–244, 2004.

[23] I. Cohen, S. Zhang, M. Goldszmidt, J. Symons, T. Kelly, and A. Fox. Capturing, indexing, clustering, and retrieving system history. In *Proceedings of the twentieth ACM symposium on Operating systems principles*, pages 105–118. ACM New York, NY, USA, 2005.

[24] P. Corn, R. Dube, A. McMichael, and J. Tsay. An autonomous distributed expert system for switched network maintenance. In *Proceedings of IEEE GLOBECOM88*, pages 1530–1537, 1988.

[25] J. Dilley, B. Maggs, J. Parikh, H. Prokop, R. Sitaraman, and B. Weihl. Globally distributed content delivery. *Internet Computing, IEEE*, 6(5):50–58, 2002.

[26] L. Eriksson, E. Johansson, N. Kettaneh-Wold, and S. Wold. *Multi-and megavariate data analysis*. Umetrics Acad., 2006.

[27] N. Feamster, D. Andersen, H. Balakrishnan, and M. Kaashoek. Measuring the effects of internet path faults on reactive routing. In *Proceedings of the 2003 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, page 137. ACM, 2003.

[28] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. Hypertext transfer protocol–HTTP/1.1, 1999.

[29] D. Forsyth and J. Ponce. *Computer vision: a modern approach*. Prentice Hall Professional Technical Reference, 2002.

[30] L. He, Y. Chao, K. Suzuki, and K. Wu. Fast connected-component labeling. *Pattern recognition*, 42(9):1977–1987, 2009.

[31] D. Johnson. Approximation algorithms for combinatorial problems*. *Journal of Computer and System Sciences*, 9(3):256–278, 1974.

[32] C. Joseph, J. Kindrick, K. Muralidhar, and T. Toth-Fejel. MAP fault management expert system. *Integrated Network Management I, North-Holland, Amsterdam*, pages 627–636, 1989.

[33] C. Kalmanek, I. Ge, S. Lee, C. Lund, D. Pei, J. Seidel, J. van der Merwe, and J. Ates. Darkstar: Using exploratory data mining to raise the bar on network reliability and performance. In *Design of Reliable Communication Networks, 2009. DRCN 2009. 7th International Workshop on*, pages 1–10. IEEE, 2009.

[34] S. Kandula, R. Chandra, and D. Katabi. What's going on?: learning communication rules in edge networks. *ACM SIGCOMM Computer Communication Review*, 38(4):87–98, 2008.

[35] S. Kandula, D. Katabi, and J. Vasseur. Shrink: A tool for failure diagnosis in ip networks. In *Proceedings of the 2005 ACM SIGCOMM workshop on Mining network data*, pages 173–178. ACM, 2005.

[36] S. Kandula, D. Katabi, and J. Vasseur. Shrink: A tool for failure diagnosis in IP networks. In *Proceedings of the 2005 ACM SIGCOMM workshop on Mining network data*, pages 173–178, 2005.

[37] R. Kompella, J. Yates, A. Greenberg, and A. Snoeren. Ip fault localization via risk modeling. In *Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation-Volume 2*, pages 57–70. USENIX Association, 2005.

[38] R. Kompella, J. Yates, A. Greenberg, and A. Snoeren. Detection and localization of network black holes. In *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, pages 2180–2188. IEEE, 2007.

[39] R. R. Kompella, J. Yates, A. Greenberg, and A. C. Snoeren. Ip fault localization via risk modeling. In *NSDI'05: Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation*, pages 57–70, 2005.

[40] R. Krishnan, H. Madhyastha, S. Srinivasan, S. Jain, A. Krishnamurthy, T. Anderson, and J. Gao. Moving beyond end-to-end path information to optimize CDN performance. In *Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference*, pages 190–201. ACM, 2009.

[41] A. Mahimkar, J. Yates, Y. Zhang, A. Shaikh, J. Wang, Z. Ge, and C. Ee. Troubleshooting chronic conditions in large IP networks. In *Proceedings of the 2008 ACM CoNEXT Conference*, 2008.

[42] J. Moy. RFC2328: OSPF Version 2. 1998.

[43] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose. Modeling TCP throughput: A simple model and its empirical validation. In *Proceedings of the ACM SIGCOMM'98 conference on Applications, technologies, architectures, and protocols for computer communication*, page 314. ACM, 1998.

[44] V. Padmanabhan, L. Qiu, and H. Wang. Server-based inference of Internet performance. In *IEEE INFOCOM*. Citeseer, 2003.

[45] A. Pankratz. *Forecasting with univariate Box-Jenkins models*, volume 3. Wiley Online Library, 1983.

[46] M. Pathan, R. Buyya, and A. Vakali. Content Delivery Networks: State of the Art, Insights, and Imperatives. *Content Delivery Networks*, page 1, 2008.

[47] V. Paxson. End-to-end routing behavior in the Internet. *ACM SIGCOMM Computer Communication Review*, 36(5):56, 2006.

[48] D. Percival and A. Walden. *Wavelet methods for time series analysis*, volume 4. Cambridge Univ Pr, 2006.

[49] L. Peterson and T. Roscoe. The design principles of planetlab. *ACM SIGOPS Operating Systems Review*, 40(1):11–16, 2006.

[50] I. Rish. An empirical study of the naive Bayes classifier. In *IJCAI 2001 Workshop on Empirical Methods in Artificial Intelligence*, pages 41–46, 2001.

[51] M. Roughan, A. Greenberg, C. Kalmanek, M. Rumsewicz, J. Yates, and Y. Zhang. Experience in measuring internet backbone trafc variability: Models, metrics, measurements and meaning. In *Proceedings of the International Teletrafc Congress (ITC-18)*, 2003.

[52] S. Saroiu, K. Gummadi, R. Dunn, S. Gribble, and H. Levy. An analysis of internet content delivery systems. *ACM SIGOPS Operating Systems Review*, 36(SI):315–327, 2002.

[53] A. Shaikh and A. Greenberg. OSPF monitoring: Architecture, design, and deployment experience. In *Proc. USENIX/ACM NSDI*, 2004.

[54] H. Song, A. Mahimkar, Z. Ge, J. Wang, J. Yates, and Y. Zhang. Q-score: Proactive service quality assessment in a large iptv system, 2011.

[55] M. Tariq, A. Zeitoun, V. Valancius, N. Feamster, and M. Ammar. Answering what-if deployment and configuration questions with wise. In *ACM SIGCOMM Computer Communication Review*, volume 38, pages 99–110. ACM, 2008.

[56] A. Vakali and G. Pallis. Content delivery networks: Status and trends. *Internet Computing, IEEE*, 7(6):68–74, 2003.

[57] L. Wang, K. Park, R. Pang, V. Pai, and L. Peterson. Reliability and security in the codeen content distribution network. In *Proceedings of the annual conference on USENIX Annual Technical Conference*, pages 14–14. USENIX Association, 2004.

[58] J. Wright, J. Zielinski, and E. Horton. Expert systems development: the ACE system. *Expert Systems Applications to Telecommunications*, pages 45–72, 1988.

[59] T. Yamahira, Y. Kiriha, and S. Sakata. Unified fault management scheme for network troubleshooting expert system. *Integrated Network Management, I. North-Holland: Elsevier Science Publishers BV*, 1989.

[60] J. Yoon, M. Liu, and B. Noble. Random waypoint considered harmful. In *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, volume 2, pages 1312–1321. IEEE, 2003.

[61] M. Zhang, C. Zhang, V. Pai, L. Peterson, and R. Wang. PlanetSeer: Internet path failure monitoring and characterization in wide-area services. In *Proc. USENIX OSDI*, 2004.

[62] S. Zhang, I. Cohen, M. Goldszmidt, J. Symons, and A. Fox. Ensembles of models for automated diagnosis of system performance problems. In *IEEE Conference on Dependable Systems and Networks (DSN)*, pages 31–109, 2005.

[63] Y. Zhang. *Characterizing End-to-End Internet Performance*. PhD thesis, Citeseer, 2001.

[64] C. Zheng and D. Sun. Image segmentation. *Computer vision technology for food quality evaluation*, page 37, 2008.

[65] M. Zweig and G. Campbell. Receiver-operating characteristic (roc) plots: a fundamental evaluation tool in clinical medicine [published erratum appears in clin chem 1993 aug; 39 (8): 1589]. *Clinical chemistry*, 39(4):561, 1993.