Toward Highly Available, Intelligent Cloud and ML Systems

Chuanxiong Guo
Bytedance

NetAI 2018
Outline

• Background: System/networking meets ML
• Deepview: ML for availability improvement of cloud systems
• RDMA for scalable ML training acceleration
• Summary
Two Different Approaches

• Network/systems are *designed* by following *principles*
• Interfaces are explicitly defined, protocols are explicitly coded, and packets can be traced and explained

• Models in machine learning are *learned* from *data* without explicit programming
• Deep learning made breakthroughs in computer vision and speech
Networking Meets Machine Learning

ML helps to improve system/network availability

Networking/system

Networking to scale and accelerate ML systems

ML
Software Rules the Clouds

![Diagram showing the relationship between Code Repo, Data Repo, Software systems, and global distribution of data centers.]

- Code Repo
- Data Repo
- Deployment/provisioning
- Config/Management
- Monitoring
- Resource mgmt

[Image of data centers around the world]
Microsoft Azure Storage Issues Caused by Two Incidents

By Chris Burt on Thursday, March 16, 2017

Summary of the Amazon S3 Service Disruption in the Northern Virginia (US-EAST-1) Region

We’d like to give you some additional information about the service disruption that occurred in the Northern Virginia (US-EAST-1) Region on the morning of February 28th. The Amazon Simple Storage Service (S3) team was debugging...

Google Compute Engine Incident #161

Connectivity Issues in all regions

Incident began at 2016-04-11 18:25 and ended at 2016-04-12 01:25

DATE TIME DESCRIPTION


6月27日阿里云故障说明

6月27日下午，我们运营团队的一个团队协作，导致一些客户的阿里云服务(包括数据库)出现延时和不可用情况。已查明原因，并在第一时间采取措施处理。

经过紧急技术处理，故障已于今日16:00前被处理完毕。

关于此次事件，我们表示深刻的歉意，会进一步排查系统，以确保此问题不再发生。

受影响范围包括阿里云数据库产品，以及Nginx、NFS、OSS等产品功能。

对于此次事件，我们深感抱歉。我们将不断提升技术水平和稳定性，以确保您的业务不受影响。

Folks please do not call the police because #facebookdown we are as upset as you are but we cannot fix facebook. #sorry #wetried #tecpolice

RETWEETS 121  LIKES 69
System Availability is Plagued by Incidents

$$A = \frac{\Sigma T_u}{\Sigma T_u + \Sigma T_d}$$

5 min downtime per year

53 min downtime per year
Incident Handling Practice

- Code Repo
- Data Repo

Software systems:
- Deployment/provisioning
- Config/Management
- Monitoring
- Resource mgmt

Lessons learned
Deepview for Virtual Disk Failure Diagnosis
-- A case where ML helps system availability
VM Availability

• IaaS is one of the largest cloud services today

• High VM availability is a key performance metric

• Yet, achieving 99.999% VM uptime remains a challenge

1. What is the VM availability bottleneck?
2. How to eliminate it?
IaaS Architecture

- Compute and storage clusters with a Clos-like network

- **Compute-storage Separation**
  - VMs and Virtual Hard Disks (VHDs) provisioned from different clusters
  - Hypervisor transparently redirects disk access to remote storage

- Keep data available during localized power failure to a rack
A New Type of Failure: VHD Failures

• Infra failures can disrupt VHD access
• Hypervisor can retry, but not indefinitely
• Hypervisor will crash the VM to surface failures to customer
• Allow customers to take actions to keep their app-level SLAs

How much do VHD failures impact VM availability?
Availability Bottleneck

- VHD failure localization is the bottleneck
  - 52% of unplanned VM downtime
  - Take 10s minutes to hours to localize

- This talk: quick and accurate failure localization

Breakdown of Unplanned VM Downtime in a Year

- SW Failure: 41%
- HW Failure: 6%
- Unknown: 1%
- VHD Failure: 52%
Failure Triage was Slow and Inaccurate

• SREs from each team check their subsystem for anomalies to match the incident
  • e.g. compute host heart-beats, storage perf-counters, network link discards

• Incidents get ping-ponged among different teams due to false positives
  • Inaccurate diagnosis and delayed mitigation

• Gray failures in network and storage are hard to catch
  • Troubled but not totally down, e.g. performance issues or software bugs
  • Only fail a subset of VHDs requests
  • Can take hours to localize
Deepview Approach: Global View

- Isolate failures by examining interactions between subsystems
  - Instead of alerting every SRE team to check if their subsystem is at fault
- Bipartite model
  - Compute Clusters (left) : Storage Clusters (right)
  - VMs are provisioned from compute/storage cluster pair
  - Edge weight = VHD failure rate
Our Approach: Global View

Example Compute Cluster Failure

Example Storage Cluster Failure
Challenges

Remaining challenges:
1. Need to pinpoint network failures
2. Need to handle gray failures
3. Need to be near-real-time

Generalized model to include network devices
Lasso regression/Hypothesis testing algorithm
Streaming data pipeline

Summary of our goal:
A system to localize VHD failures to underlying failures in compute, storage or network subsystems within a time budget of 15 minutes

Time budget set by production team to meet availability goals
Deepview Model: Include the Network

- Need to handle multipath and ECMP
- Simplify Clos network to a tree by aggregating network devices
- Can model at the granularity of clusters or ToRs
Deepview Model: Estimate Component Health

\[
\text{Prob(path i is healthy)} = \prod_{j \in \text{path}(i)} \text{Prob(component j is healthy)}
\]

Component j is healthy with
\[
p_j = \exp(\beta_j)
\]
- \(\beta_j = 0\), clear component j
- \(\beta_j \ll 0\), may blame it

\[
1 - \frac{e_i}{n_i} = \prod_{j \in \text{path}(i)} p_j
\]

\[
\log \left(1 - \frac{e_i}{n_i}\right) = \sum_{j \in \text{path}(i)} \log p_j
\]

*Assume independent failures
\[
e_i = \text{num of VMs crashed}
\]
\[
n_i = \text{num of VMs}
\]

System of Linear Equations
\[
y_i = \log \left(1 - \frac{e_i}{n_i}\right)
\]
\[
\beta_j = \log p_j
\]
\[
\varepsilon_i = \text{measurement noise}
\]
Deepview Algorithm: Prefer Simpler Explanation via Lasso

\[ y_i = \sum_{j=1}^{N} \beta_j x_{ij} + \varepsilon_i \]

- Potentially #unknowns > #equations
- Traditional least-square regression would fail
- But multiple simultaneous failures are rare
- How to encode this domain knowledge mathematically?
- Equivalent to prefer most \( \beta_j \) to be zero
- **Lasso regression** can get sparse solutions efficiently

Example:

\[
\begin{align*}
y_1 &= \beta_{c1} + \beta_{net} + \beta_{s1} + \varepsilon_1 \\
y_2 &= \beta_{c1} + \beta_{net} + \beta_{s2} + \varepsilon_2 \\
y_3 &= \beta_{c2} + \beta_{net} + \beta_{s1} + \varepsilon_3 \\
y_4 &= \beta_{c2} + \beta_{net} + \beta_{s2} + \varepsilon_4
\end{align*}
\]

Lasso Objective Function:

\[
\hat{\beta} = \text{argmin}_{\beta \in \mathbb{R}^N, \beta \leq 0} \|y - X\beta\|^2 + \lambda \|\beta\|_1
\]

Sparsity
Deepview Algorithm: Principled Blame Decision via Hypothesis Testing

- Need a binary decision (flag/clear) for each component
- Ad-hoc thresholds do not work reliably
- Can we make a principled decision?

- If estimated failure probability worse than average, then likely a real failure
- Automate this empirical decision criterion using a hypothesis test:

  \[ H_0(j): \beta_j = \bar{\beta} \quad \text{vs.} \quad H_A(j): \beta_j < \bar{\beta} \]

- Reject \( H_0(j) \) means blame component \( j \)
- Otherwise, clear component \( j \)
Deepview System Architecture: NRT Data Pipeline

**Ingestion Pipeline**
- **Real-time**
  - VHD Failure
- **Non-RT**
  - VM Info
  - StorageAcct
  - Net Topo
  - Kusto Engine

**Raw Data**
- RAW DATA
- SLIDING WINDOW OF INPUT
- RUN ALGO

**Actions**
- ALGO
- Output
- Alerts
- Vis

**Near-realtime Scheduler**
Some Statistics

• Analyzed Deepview results for one month
  • Daily VHD failures: hundreds to tens of thousands

• Detected 100 failures instances
  • 70 matched with existing tickets, 30 were previously undetected

• Reduced unclassified VHD failures to less than a max of 500 per day
  • Single-host failures or customer mistakes (e.g. expired storage accounts)
Case Study 1: Unplanned ToR Reboot

- Unplanned ToR reboot can cause VMs to crash
- We knew this can happen, but not where and when

- Deepview can flag those ToRs
- The figure shows a ToR down in one small region
  - Blamed the right ToR among 288 components

- Associate VM downtime with ToR failures
- Quantify the impact of ToR as a single-point-of-failure on VM availability
Case Study 2: Storage Cluster Gray Failure

- Impact only a subset of VMs

- A storage cluster was brought online with a bug that puts some VHDs in negative cache

- Deepview flagged the faulty storage cluster almost immediately while manual triage took 20+ hours
Deepview Insight: ToR as a Single Point of Failure

• **Reduced Network Cost vs. Availability cost for using a single ToR per rack**
• Unplanned ToR failures: soft failures (recoverable by reboot) vs. hard failures

**ToR Availability**

\[
ToR\text{ Availability} = 1 - \frac{(% \text{ soft } \times \text{ soft dur.} + % \text{ hard } \times \text{ hard dur.}) \times \text{ frac. rebooted ToRs per month}}{\text{total time in a month}}
\]

\[
= 1 - \frac{(90\% \times 20 \text{ min} + 10\% \times 120 \text{ min}) \times 0.1\%}{30 \times 24 \times 60 \text{ min}}
\]

\[
= 99.99993\%
\]

• Dependent services (ToRs) need to provide one extra nine to target service (VMs)

ToRs are not on critical path for VMs to achieve five-nines availability.
Deepview Insight: VMs and their Storage Co-location

• For load balancing, VMs can mount VHDs from any storage cluster in the same region
• Some VMs have storage that are further away
• **Can longer network paths impact VM availability?**
• At Azure, 52% two-hop, 41% three-hop
• Compute daily VHD failure rates: \( r_0 \) (two-hop), \( r_1 \) (three-hop)
• Average over 3-months
• **Yes!** \( \frac{r_1 - r_0}{r_0} = 11.4\% \) increase

Some benefit to co-locate VM and their VHDs
Deployment
Provisioning
Monitoring
Resource mgmt
Admin

Design
implement
Dev

Incident prevention
Incident resolution, mitigation
Incident localization detection

OPS
RDMA for ML Training Acceleration
-- A case where networking helps ML to scale
Background
Content Understanding using DNN
DNN Training: BP

Forward

\[ x_0^0 \rightarrow w_{0,0}^0 \rightarrow w_{0,1}^0 \rightarrow y_0^1 \rightarrow \cdots \rightarrow x_2^1 \rightarrow w_{1,0}^1 \rightarrow y_1^2 \rightarrow \cdots \rightarrow y_0^2 \]

\[ e = \frac{1}{2} \sum_{i=0,1} (y_i^2 - y_i^f)^2 \]

Backward

\[
\begin{bmatrix}
\frac{\partial e}{\partial w_{0,0}} \\
\frac{\partial e}{\partial w_{0,1}} \\
\frac{\partial e}{\partial w_{0,2}} \\
\frac{\partial e}{\partial w_{1,0}} \\
\frac{\partial e}{\partial w_{1,1}} \\
\frac{\partial e}{\partial w_{1,2}} \\
\frac{\partial e}{\partial w_{2,0}} \\
\frac{\partial e}{\partial w_{2,1}} \\
\frac{\partial e}{\partial x_0} \\
\frac{\partial e}{\partial x_1} \\
\frac{\partial e}{\partial x_2}
\end{bmatrix} = \begin{bmatrix}
\frac{\partial e}{\partial x_0^1} \\
\frac{\partial e}{\partial x_1^1} \\
\frac{\partial e}{\partial x_2^1}
\end{bmatrix} \begin{bmatrix}
x_0^0 \\
x_0^1 \\
x_1^0 \\
x_1^1 \\
x_2^0 \\
x_2^1
\end{bmatrix}
\]

\[
w = w - \eta \nabla w
\]
Distributed Training Acceleration

• GPU, with mini-batch
• Distributed training (data parallel)
Arnold Training System

- Compute (GPU, CPU, FPGA, ASIC)
- Network (RDMA)
- Storage (CephFS)
- Web UI
- Arnold SDK
- Tensorboard
- Arnold API
- Metis
- Arnold agent
- Nvidia Docker
- SCM
- Mesos
- Tensorflow
- MxNet
- Caffe
- Arnold agent
- Tensorboard
- Arnold SDK
- Web UI
When Communication Becomes Bottleneck
RDMA/RoCEv2 background

- RDMA addresses TCP’s latency and CPU overhead problems
  - RDMA offloads the transport layer to the NIC
  - RDMA needs a lossless network
- RoCEv2: RDMA over commodity Ethernet
  - PFC for hop-by-hop flow control
  - DCQCN for connection-level congestion control [sigcomm15]
  - Many issues addressed [sigcomm16, conext17]
RDMA Cluster for Arnold Training

100GbE RDMA Network

- 100Gbps throughput between any servers
- Micros-second e2e latency
- Minimal CPU overhead for packet processing

- Many models spend large amount of time on communication
  - Poor TCP performance
  - Low network bandwidth

- 100GbE RDMA network
  - Much higher bandwidth
  - Reduces communication time
  - Scales the cluster to thousands of GPU cards
RDMA Many-To-One

Sending servers

100GbE

100GbE

switch

Throughput

0 Gbps
25 Gbps
50 Gbps
75 Gbps
100 Gbps
125 Gbps

12:00 14:00 16:00 18:00

PFC

100GbE

Receiving server

ECN
RDMA for ML Training Acceleration (CNN)

Batch size: 32

Batch size: 64
RDMA for ML Training Acceleration (RNN)
When RDMA Acceleration Helps

• Big models
  • ResNet50 (98MB), VGG19 (548MB)

• Communication/computation ratio is large
  • Layers with large parameter size
  • Small minibatch size
  • When TCP is slow
Summary

• ML will be a core part for building highly available systems
  • Deeper availability understanding
  • Automatic incident localization, mitigation, prevention
  • Intelligent system/network design

• System/networking for ML
  • Scalable ML systems
  • Hardware, systems, ML services integrated design
Acknowledgement

- Deepview (nsdi18): Qiao Zhang, Guo Yu, Yingnong Dang, Nick Swanson, Xinsheng Yang, Randolph Yao, Murali Chintalapati, Arvind Krishnamurthy, and Thomas Anderson
- ByteDance Networking Team
- Bytedance ML System Team