Opening Up Black-Box Networks with CloudTalk

Costin Raiciu, Mihail Ionescu and Dragos Niculescu
University Politehnica Bucharest
Cloud providers keep customers in the dark w.r.t. network-topology

- Not disclosed for commercial and security reasons
- Do the customers need to know anyways?
Cloud Customers want performance

Fast Map Reduce!
Cloud Customers want performance

Static Optimization: A is in the same rack as B, so Run map on A

oversubscribed

MAP? A

MAP? B

MAP? C

Fast Map Reduce!

....
Cloud Customers want performance

Dynamic Optimization: A’s downlink is congested, so
Run map on C

Fast Map Reduce!
The Status Quo

Cloud providers **MUST** keep network information private for business and security reasons.

Customers **NEED** network information to help optimize distributed applications.

**Are we stuck** between a rock and a hard place?
Is network topology information private in the first place?

• We did a brief study of the Amazon EC2 network topology (us-east-1d)
• Rented many VMs
• Between all pairs we ran:
  – Traceroute
  – Record route (ping –R)
  – Used aliasing techniques to group IPs on the same device
EC2 Measurement results

Edge Router (IP)

Top-of-Rack Switch (L2)
EC2 Measurement results
EC2 Measurement results

Diagram:
- Top-of-Rack Switch
- Edge Router
- Connections between Top-of-Rack Switch and Edge Router
EC2 Measurement results

INTERNET

Core Router

Edge Router

Top-of-Rack Switch

....
The **REAL** Status Quo

Cloud providers **can’t** keep network information private

Customers **infer** network information to help optimize distributed applications
- inaccurate
- high overhead

Everyone loses
Moving forward: strawman solutions

The cloud provider shares information publicly.

• The network blueprint
  – This is sensitive information!
  – Does not capture dynamic load

• Aggregate information (IETF ALTO WG):
  – Not accurate enough to be useful
  – Does not capture dynamic information, one-to-many traffic patterns
CloudTalk in a nutshell
CloudTalk in a nutshell

How long would it take to transfer 64MB from B to A?
CloudTalk in a nutshell
CloudTalk in a nutshell

How long would it take to transfer 64MB from B to C?
CloudTalk in a nutshell
CloudTalk in a nutshell

RUN JOB ON A!
CloudTalk is a language that describes network tasks

- Each task consists of a set of flows
- Each flow is specified by:
  - *Flow ID* - unique within a task
  - *Start time* - relative to the start of the task or the finish time of another flow(s)
  - *Size* - in Bytes
  - *Other constraints* (e.g. max bandwidth).
- A task is considered finished when the last of its flows finishes
Applications use CloudTalk tasks to describe alternative ways of doing their job.

Example

The map-reduce scheduler creates two tasks:

Task 1
- **ID**: 0, **B -> A**, **START**: 0, **SIZE**: 64MB

Task 2
- **ID**: 0, **B -> C**, **START**: 0, **SIZE**: 64MB

Network tasks are sent to servers of the cloud provider.
The network servers estimate finish times for the tasks they receive:

- **Using topology information**
  - Task 1 (B->A): 0.65s
  - Task 2 (B->C): 2.6s

- **Using dynamic information**
  - Task 1 (B->A): 6.5s
  - Task 2 (B->C): 0.65s
CloudTalk for other apps

Example: Search

Task 1: aggregate at A
ID: 0, A→B, START: 0, 1KB
ID: 1, A→C, START: 0, 1KB
ID: 2, B→A, START: finish(0), 10KB
ID: 3, C→A, START: finish(1), 10KB

Task 2: aggregate at B
CloudTalk Properties

Expressive  
captures one-to-one, many-to-one, many-to-many traffic patterns

Allows independent evolution of applications and the network  
the language is just an API used by applications and implemented by the network

Minimum information leaked  
Network only reveals finish time estimates  
Applications describe their equivalent traffic patterns (some of these are visible to the network).
Network-side problem to be solved:

Input: network topology information
dynamic flow information
CloudTalk task(s)

Output: estimated finish time

Our prototype:
packet-level simulation with htsim of 1200-node EC2-like network, no dynamic information
Changing applications to use CloudTalk

• Focus on a few “framework” applications
  – MapReduce, Distributed File Systems, etc.

• Preliminary evaluation: synthetic task descriptions
  – Web-search: scather-gather
  – Map-reduce: job placement
Evaluation results

• Map/reduce: where should a map task be placed?
  – Runs just before map task is started
  – 64MB map input
  – Takes 1.2s to evaluate

• Web-search – see paper.
Limitations and Challenges

• CloudTalk is not useful for short transfers
• Using dynamic information is tricky
  – Need to find out what’s going on (e.g. [Hedera]).
  – Expensive to give answers in this context
• Optimizing in a changing network
  – How can we be sure that the estimate is right?
  – What if a new flow arrives in the meantime?
• Scaling network estimation to large tasks:
  – Flow-level simulation?
Conclusions

Cloud applications need network information for optimizations.
Conclusions

Cloud applications need network information for optimizations.

Cloud Networks are **black-boxes** for commercial and security reasons.
Conclusions

Cloud applications need network information for optimizations.

Cloud Networks aren’t black-boxes because topology inference is possible.
Conclusions

Cloud applications need network information for optimizations.

Cloud Networks aren’t black-boxes because topology inference is possible.

The applications and network should break the silence for mutual benefits
Conclusions

CloudTalk is a simple language that allows the network and the applications to communicate.

• Expressive enough to capture common applications
• Reveals little information beyond what is known or can be inferred

Initial prototype proves feasibility, but there are many interesting challenges to solve.
Backup
Optimizing on EC2

• We ran web search on 100 machines (2TB dataset)
  – Need to use aggregators to reduce incast effects

Where should aggregators be placed?
Placing aggregators to be as close to servers as possible reduces delay by a factor of three
Evaluation results: web search

• Pre-deployment analysis of aggregator placement
• Each tasks describes one query running in an idle network with different placement of aggregators:
  – Server processing time is assumed constant
  – Message size is constant
• Qualitatively similar results, correct ranking
• Evaluation speed: 1.5s for each task (10 runs, to capture ECMP routing)
Analyzing possible network implementations

- Packet-level simulation
  - Is very accurate
  - Time scales linearly with #packets
- Flow-level simulation
  - Less accurate (no incast)
  + Time scales with the number of flows

We probably need a hybrid solution...