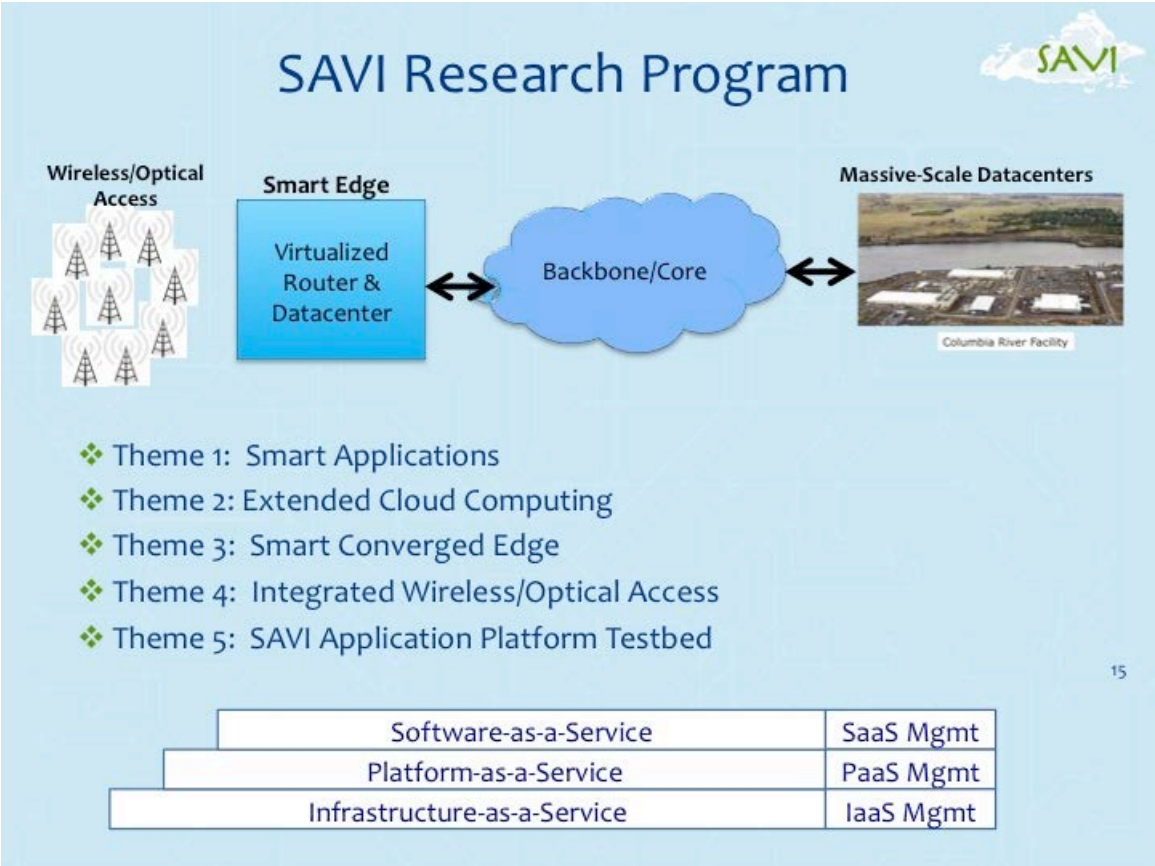


SAVI 2013 Annual General Meeting (AGM) Student Posters

NSERC Funded Canadian Strategic Network NET GP397724-10
Smart Applications on Virtual Infrastructure ©2013
August 2013

SAVI Network Progress Years 1 and 2

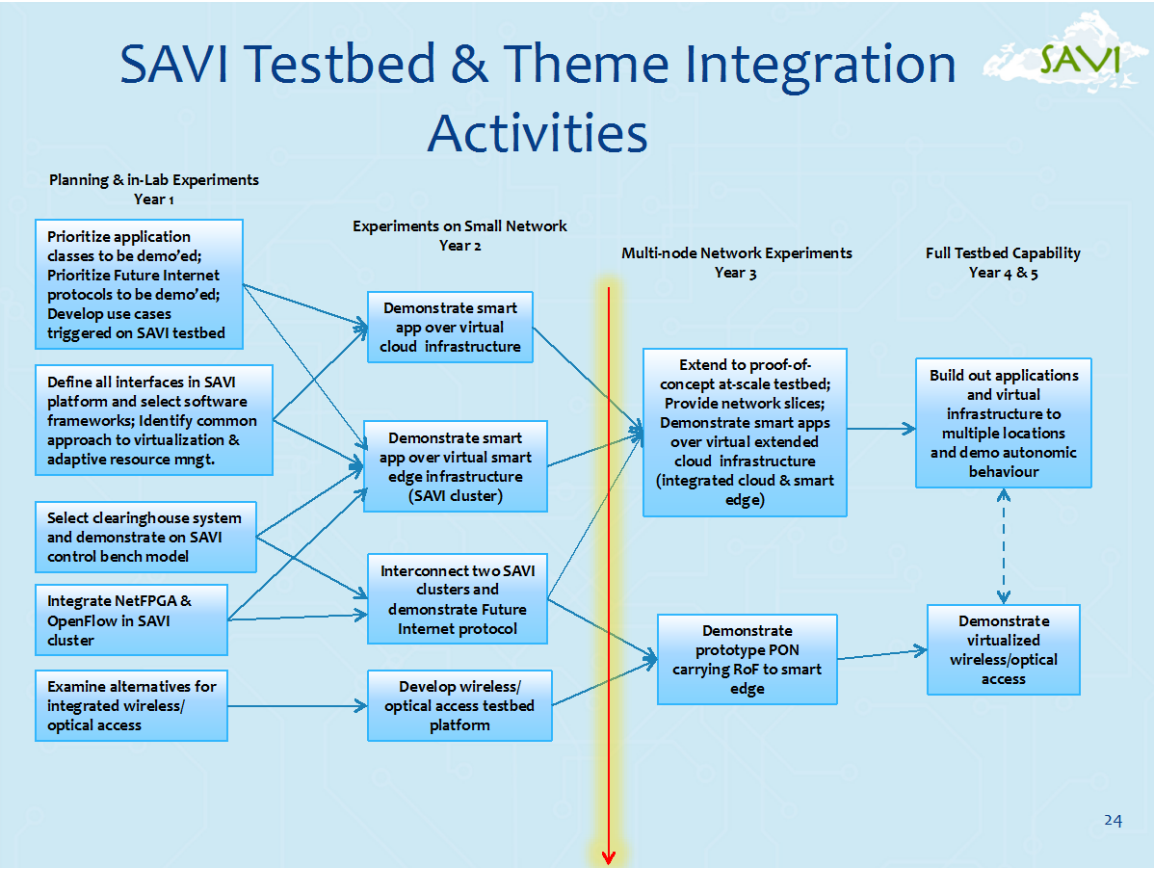


Fig. 1 Timeline --- We are here.

The posters in this booklet have been developed by students working under the umbrella of the NSERC funded SAVI strategic network. These posters were presented at the year 2 AGM, held on July 5th 2013 in Toronto and attended by a large audience of SAVI partners, affiliates, students, researchers and key theme leads. The timeline picture above shows the points that SAVI had (and still has) to intercept to enable the research goals set for the network. As you work through these posters you will see the great progress that has been made in examining the hypothesis of SAVI. You will also be able to observe the level of integration that has occurred and the progress made to date in all aspects of the network.

This booklet has been put together to become a permanent record for all SAVI engaged partners showing the scope and reach so far attained. As we go into year 3 of SAVI, we plan even greater levels of integration across themes and between academia and industry partners.

Alberto Leon-Garcia
SAVI Scientific Director
August 2013

NSERC Funded Canadian Strategic Network NET GP397724-10
Smart Applications on Virtual Infrastructure ©2013
August 2013

Below is the 'List of Student Posters' presented at the SAVI AGM held on July 5th, 2013 at the University of Toronto. The poster numbers are shown along with the theme, the name of the theme lead and their affiliated university within the SAVI Network.

Poster Number and Title	Professor(s)	University
THEME 1 - FUTURE APPLICATIONS		
Team Members: Profs. Li (Lead), Muller and Stroulia		
1. Optimizing Video Delivery over SAVI	Li	Toronto
2. A Software-defined Application-layer Networking Platform	Li	Toronto
3. MobileView - Mobile Video Conferencing over Bellini and SAVI Testbed	Li	Toronto
4. Design Patterns for Spatio-temporal Applications on HBase	Stroulia	Alberta
5. SAVI Billboard Post-it Message Center	Muller	Victoria
6. SAVI Context Based Face Recognition	Muller	Victoria
7. SAVI PALTask - Personal Context Assistant	Muller	Victoria
7A. SAVI NewsCast – Customized Video Feed	Muller	Victoria
THEME 2 - RESOURCE CONTROL & MANAGEMENT		
Team Members: Profs. Litoiu (Lead), Chinnek, Woodside and Salem		
8. Supporting Application Development with Structured Queries in the Cloud	Litoiu	York
9. Model Predictive Resource Management for Cloud	Litoiu	York
10. Pattern-based Deployment Service for Next Generation Clouds	Litoiu	York
11. Towards Cloud-aware Multi Class Adaptive Models	Litoiu	York
12. Algorithms for Task Allocation considering Communication Delays	Chinneck/Woodside	Carleton
13. Impact of Network Latency on Cloud Performance	Chinneck/Woodside	Carleton
14. Simulation of Core-edge Cloud Networks	Chinneck/Woodside	Carleton
THEME 3 - SMART CONVERGED EDGE		
Team Members: Profs. Boutaba (Lead), Leon-Garcia and Ganjali		
15. Dynamic Controller Provisioning in Software-defined Networks	Boutaba	Waterloo
16. Greenhead - Virtual Data Center Embedding Across Distributed Infrastructures	Boutaba	Waterloo
17. HARMONY - Dynamic Heterogeneity-aware Resource Provisioning in the Cloud	Boutaba	Waterloo
18. On Achieving High Survivability in Virtualized Data Centers V06	Boutaba	Waterloo
19. VDC Planner - Dynamic Migration-aware Virtual Data Center Embedding for Clouds	Boutaba	Waterloo
20. Reliable Virtual Data Center Embedding in Clouds	Boutaba	Waterloo
21. Network-aware Job Placement in Data Center Environments	Williamson	Calgary

Poster Number and Title	Professor(s)	University

THEME 4 - INTEGRATED WIRELESS/OPTICAL ACCESS
Team Members: Profs. Le Ngoc (Lead), Rusch, Ghaderi and Williamson

22. Wireless Access Point Virtualization Testbed	LeNgoc	McGill
23. Wireless Virtualization Perspectives and Framework	LeNgoc	McGill
24. Full-duplex In-band Channel OOK-DL OFDM-UL Transmission	Rusch	Laval
25. Optical Integration of Radio Services using Tandem Optical SSB Modulation and DSP-assisted Coherent Detection	Rusch	Laval
26. Distributed Base-station Activation for Energy-efficient Operation of Cellular Networks	Ghaderi	Calgary

THEME 5 - SAVI APPLICATIONS PLATFORM TESTBED
Team Members: Profs. Leon-Garcia (Lead), Boutaba, Ganjali, Chow/Steffan and Litoiu

27. Graph-based Network Topology Management for SAVI Testbed	Leon-Garcia	Toronto
28. SAVI Testbed Control/ Management of Converged Virtual ICT Resources	Leon-Garcia	Toronto
29. Software-defined Networking (SDN) in SAVI Testbed Converged Cluster	Leon-Garcia	Toronto
30. NetThreads - Virtualizing FPGA Accelerators for SAVI	Steffan/ Chow	Toronto

THEMES 1 TO 5 - SAVI SYSTEM WIDE CONCEPTS
Profs. Leon-Garcia, Boutaba, Li, Litoiu and Rusch

31. Identity Management in Software Defined Infrastructure	Leon-Garcia	Toronto
32. Information Centric Data Management for Smart Transportation	Leon-Garcia	Toronto
33. Scalable Content Networking over SAVI	Leon-Garcia	Toronto
34. Software-Defined Infrastructure and the Future CO	Leon-Garcia	Toronto

Theme 1 Smart Applications

Research Team

Baochun Li (lead), University of Toronto

Hausi Müller, University of Victoria

Eleni Stroulia, University of Alberta

NSERC Funded Canadian Strategic Network NET GP397724-10
Smart Applications on Virtual Infrastructure ©2013
August 2013

1. Airlift: Optimizing Video Delivery over SAVI

Yuan Feng, Zimu Liu, **Baochun Li**, U. Toronto

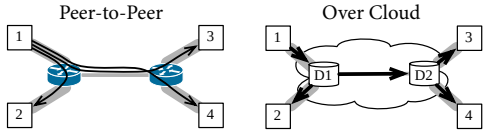
In this poster, we propose that video conferencing, even with its stringent delay constraints, should also be provided as a SAVI service, taking full advantage of the SAVI core and edge platform. We design Airlift, a new protocol designed for the SAVI network, tailored to the needs of a cloud-based video conferencing service. Airlift delivers packets in live video conferences to their respective destination SAVI nodes, with the objective of maximizing the total throughput across all conferences, yet without violating end-to-end delay constraints. In order to simplify our protocol design in Airlift, we use intra-session network coding and the concept of conceptual flows, such that the optimization problem that can be conveniently formulated as a linear program. Our real-world implementation of Airlift has been deployed with the assistance of Bellini, our application-layer networking platform. We have shown that Airlift delivers a substantial performance advantage over state- of-the-art peer-to-peer solutions. This work has been published as a full research paper at the IEEE International Conference on Network Protocols (ICNP 2012).

Airlift: Optimizing Video Delivery over SAVI

Yuan Feng, Zimu Liu, Baochun Li
Department of Electrical and Computer Engineering, University of Toronto

Motivation: Conferencing over the Cloud

Preliminary experiments to investigate the benefits of video delivery over the cloud:



Cloud/P2P	Throughput (Mbps)	Delay (msec)
Toronto-Beijing	2.202/0.179	171.6/148.8
Cambridge-San Paulo	1.687/1.432	103.4/204.6
Seoul-Moscow	7.189/1.103	201.7/436.9

When data are routed through the cloud, all three pairs have enjoyed higher throughput values, yet with similar or even shorter end-to-end delays, as compared to the overlay link in a P2P solution.

Design Objectives

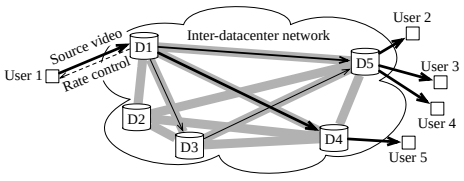
➢ ***Performance.*** We should achieve the higher video flow rates, while still maintaining an acceptable end-to-end delay, so that user experience can be significantly improved using *Airlift*.

➢ ***Simplicity.*** *Airlift* should be conceptually simple to use, and work as a full-service broker. A participating user in a conference should only need to connect to the *Airlift*, and to start transmitting video packets.

➢ ***Scalability.*** *Airlift* should support a large number of concurrent conferences, so that can be routed through the inter-datacenter network efficiently and without much fanfare.

Maximizing Aggregated Multicast Throughput

Aggregated multicast sessions: All the video flows from different participants need to be *aggregated*, provided that these participants connect to the *same source*, and are destined to the *same subset of destinations*, which, in turn, are responsible for delivering them to all other participants in each of the conferences. Each of these aggregated sessions is inherently a *multicast* session.



Feasible paths satisfying a delay bound: To ensure that video packets are delivered within certain time constraints, we need to exclude paths that violate the delay bound in a multicast session, and only consider the set of *feasible paths*:

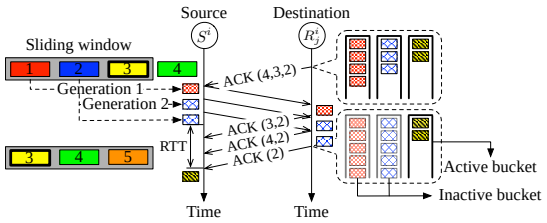
$$\mathcal{P}_j^i = \{p \mid p \text{ is an acyclic path from } S^i \text{ to } R_j^i \text{ s.t. } \sum_{e \in p} d(e) \le D_{\max}\}.$$

Conceptual flows maximization: Instead of maximizing multicast tree using traditional *Steiner tree packing*, an NP-complete algorithm, we propose to use the concept of *conceptual flows*, in the context of network coding. The problem of maximizing throughput can be solved nicely as a linear program.

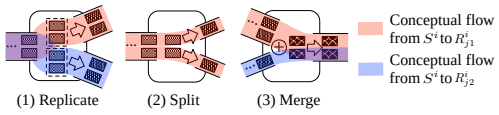
$$\begin{aligned}
\max \quad & \mathcal{X} \\
\text{s.t.} \quad & \mathcal{X} \le \sum_{p \in \mathcal{P}_j^i} x_j^i(p)/w_i, \ \forall i,\ j = 1, \ldots, k_i \\
& \sum_{p \in \mathcal{P}_j^i(e)} x_j^i(p) \le x^i(e), \ \forall i,\ j = 1, \ldots, k_i \\
& \sum_i x^i(e) \le C(e), \ \forall e \in \mathcal{E} \\
& x_j^i(p) \ge 0,\ x^i(e) \ge 0,\ \mathcal{X} \ge 0, \\
& \forall p \in \mathcal{P}_j^i,\ \forall i,\ j = 1, \ldots, k_i,\ \forall e \in \mathcal{E}.
\end{aligned}$$

Protocol Design with Network Coding

Network coding has been adopted in the *Airlift*, to execute the conceptual flow multicasting. Furthermore, since random linear codes are rateless erasure codes, we can easily decode the original data, as long as sufficient linearly independent packets are received from different paths in the multicast tree.



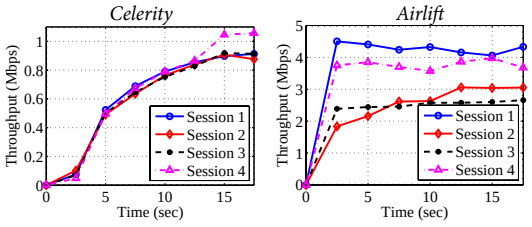
To realize the conceptual flows in *Airlift*, we need to carefully handle coded packets in an intermediate node: (1) packets are to be replicated to multiple outgoing edges; (2) packets are to be forwarded to different edges with different portions; (3) packets are to be merged and forwarded to an bottleneck edge.



Performance Evaluation

We compare our *Airlift* protocol with an state-of-the-art peer-to-peer protocol, called *Celerity*, in real-world networks.

Achievable Throughput



End-to-End Delay

Airlift/Celerity	Delay (msec)
Toronto-Beijing	169.8/142.3
Vancouver-Berlin	137.2/104.9
Seoul-Rio	228.6/203.5

FIGURE 1.1: *Airlift: Optimizaing Video Delivery over SAVI from Yuan Feng, Zimu Liu and Baochun Li at University of Toronto (Theme 1)*

2. Bellini: a Software-Defined Application-Layer Networking Platform

Zimu Liu, Yuan Feng, **Baochun Li**, U. Toronto

Implementing and deploying resource-hungry smart applications, such as streaming and conferencing, on the SAVI platform requires a flexible and reusable software framework that provides customizable elements shared by these applications, such as choices of transport protocols, routing policies, and rate allocation strategies. In this poster, we present Bellini, a flexible and reusable software framework supporting a wide variety of prototype implementations targeting different application scenarios. Customized protocols and algorithms can be plugged in using the provided delegate interface, or loaded from a configuration file with ease. Besides flexibility, Bellini is also optimally designed for efficient resource utilization in the VM-based SAVI core and edge nodes. We are currently working on decoupling the control plane from the data plane in Bellini, so that an OpenFlow-like central controller can be used to control the routing policies in Bellini. With case studies on video conferencing, we demonstrate the benefits of Bellini with respect to supporting and simplifying the development process of SAVI smart applications. A research paper on Bellini has just been accepted to IEEE Globecom 2013.

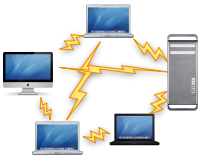
Bellini: A Software-defined Application-layer Networking Platform

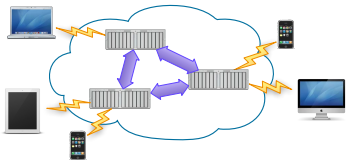
Zimu Liu, Yuan Feng, Baochun Li
Department of Electrical and Computer Engineering, University of Toronto

The Evolution of Application Networking

Client-Server: Users are served by one or a set of servers. Limited server capacity and reliability. Hard to serve geo-distributed users.

Peer-to-Peer: Users have to contribute a part of their bandwidths to alleviate the server workload. Complex algorithms needed.

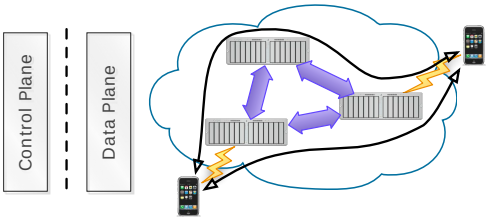




Cloud: Excellent connectivity between users and nearby datacenters. Dedicated high-capacity links are used between datacenters.

Motivation

Implementing and deploying resource-hungry smart applications, such as streaming and conferencing, on the SAVI platform requires a flexible and reusable software platform that provides customizable elements shared by these applications, such as choices of transport protocols, routing policies, and rate allocation strategies. To achieve this, we have designed and implemented Bellini, a flexible and reusable software framework supporting a wide variety of prototype implementations targeting different application scenarios. Customized protocols and algorithms can be plugged in using the provided interface, or loaded from a configuration file with ease. We are also working on decoupling the control plane from the data plane in *Bellini*, so that an OpenFlow-like central controller can be used to control the different policies in *Bellini*.



Design Objectives

➢ ***Flexibility and reusability.*** *Bellini* should support shared elements of a variety of media-rich smart applications. Applications will able to use different transport protocols, routing policies, and rate control strategies.

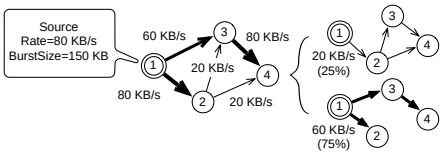
➢ ***Decoupling between data plane and control planes.*** *Bellini* should provide an easy-to-use control plane to configure different QoS parameters, routing policies, and etc. Afterwards, data plane will execute these decisions accurately.

➢ ***Performance.*** *Bellini* should achieve the best possible performance to utilize virtualized resources efficiently, so that incurred costs are minimized. When the system scales up, *Bellini* should be capable of handling a large number of concurrent transmission sessions.

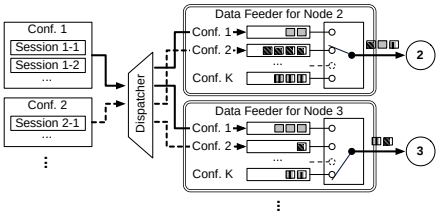
➢ ***Instrumentation.*** *Bellini* is also designed to validate and evaluate new media transmission protocols and algorithms using deployments in SAVI. It should instrument application performance at runtime, and report the results for further analysis.

Implementation Highlights

Flexible source routing and adjustable flow rates. To support various smart applications in SAVI, *Bellini* is designed to allow sophisticated routing policies. Beside direct transmission, a packet can be forwarded at relays in a multi-hop path, or duplicated in any intermediate node in a multicast path. Furthermore, *Bellini* supports dynamic routing at the granularity of packets, so that the rate of each sub-flow can be adjusted dynamically.



Concurrent media conferences and sessions. In order to support a large number of concurrent media sessions, *Bellini* components are carefully designed to administer all ongoing transmission sessions. Among multiple concurrent sessions, each session is allowed to have its own set of routing and other policies.



Instrumentation and supporting scripts. *Bellini* includes an extensive set of instrumentation modules and deployment scripts that provides "turn-key" solutions when it comes to deploying multiple executable instances and configuration files to their respective datacenters, launching them for execution, and collecting logs after they are terminated.

Performance Evaluation

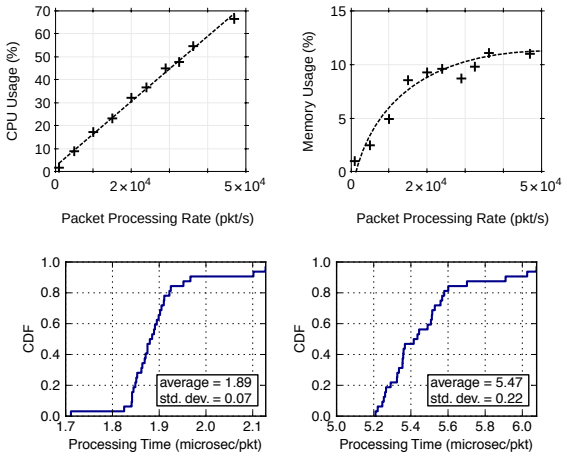


FIGURE 1.2: *Bellini - A Software-defined Application-layer Networking Platform from Zimu Liu, Yuan Feng and Baochun Li at University of Toronto (Theme 1)*

3. MobileView: Mobile Video Conferencing over Bellini and the SAVI Testbed

Zimu Liu, **Baochun Li**, U. Toronto

In order to demonstrate the power and customizability of the Bellini software framework over the SAVI testbed, we have developed a video conferencing application on the iPhone (iOS) platform, that uses Bellini to send and receive traffic via the SAVI testbed. A video conferencing application is bandwidth demanding and delay sensitive, and will be used to test the viability and effectiveness of new protocols to control the deployment of new smart applications in SAVI.

FIGURE 1.3: *MobileView - Mobile Video Conferencing over Bellini and SAVI Testbed from Zimu Liu and Baochun Li at University of Toronto (Theme 1)*

4: Design patterns for Spatio-temporal Applications on HBase

Dan Han, **Eleni Stroulia**, U. Alberta

Cloud-based infrastructures enable applications to collect and analyze massive amounts of data. Whether these applications are newly developed or they are being evolved from existing RDBMS-based implementations, NoSQL databases offer an attractive platform with which to address this challenge. However, development methodologies and tools for these platforms are practically non-existing. Focusing on spatio-temporal applications, as big-data systems with distinct data types and usage patterns in need of scalability, this thesis develops general guidelines for the design of HBase storage, and a specific three-dimensional H-Base "schema" for geospatial applications. These guidelines have been evaluated with the migration of an existing geospatial application to the cloud.

A Framework for Big Data Analytics on HBase

Dan Han and Eleni Stroulia

University of Alberta

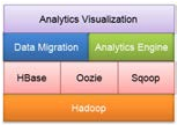
{dhan3, stroulia} @ualberta.ca

The Problem

- Big Data originates from
  - Mobile applications
  - Ubiquitous sensors
  - New breed of archive-style applications
- Big-Data Challenges
  - **Volume**: the size of the data
  - **Velocity**: the latency of data processing
  - **Variety**: the diversity of sources, formats, and structures
- Existing and new applications can now take advantage of more data
  - By collecting additional data
  - By analyzing existing historical data in new ways
- Big-Data Analytics requires
  - Scalable data-management system
  - Support for multiple data types
  - Support for batch processing and/or real-time data streams

Architecture and Data Model

- Various Data Sources
  - Data from applications in SAVI Edge
  - Data from public Cloud
  - Data from applications outside of Cloud
- Various Data Formats
  - Data in file format (txt, csv, xml, etc)
  - Data in RDBMS
- Component Level Architecture
  - Hadoop, HBase, Oozie, Sqoop
  - Data Migration
  - Analytics Engine
  - Analytics Visualization



NoSQL Challenges

- Schema Design
  - No systematic way to organize the data in NoSQL databases
  - Data schema substantially impacts query performance
- Query Implementation
  - Few and Simple APIs (Get and Scan)
  - Co-processors
- Aggregating Data from Different Sources
  - Migrate the data incrementally without any performance impact
  - Schedule the data migration process for different sources or edges

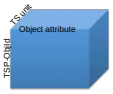
HBase Schema Design

Examples of Datasets from Spatio-Temporal Applications

Dataset / Dimension	Space Size	Time Length	Object Variety	Object Behavior	Object Complexity
Cosmology (N-Body Simulation)	Large	Small	Small	Yes	Low
Bixi (bike-station data in Montreal)	Small	Long	Small	No	Low
Smart Condo (home sensor)	can be Large	Large	Large	No	Medium
HCA-T (mobile app for home-care aides)	Small	Long	Large	Yes	High

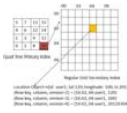
Three-Dimensional Data Model for Time-Series Data [1]

- Row Key: Timestamp period–Object ID
  - A coarse granularity of time, e.g. day, week, month
- Column Name: Object attribute
- Version: Timestamp unit
  - A fine granularity of time, e.g. second, minute, hour



The HGrid Data Model for Geospatial Data [2]

- HGrid Index Construction
  - A new hybrid spatial index structure
- Row key: QTId-RowId
  - Z-value of quad-tree tile in the 1st tier index
  - Row index of its regular-grid cell in the 2nd tier
- Column Name: ColumnId-ObjectId
  - Column index of regular-grid cell in the 2nd tier
- Version: Object Attributes



An appropriate data schema relies on the data usage pattern

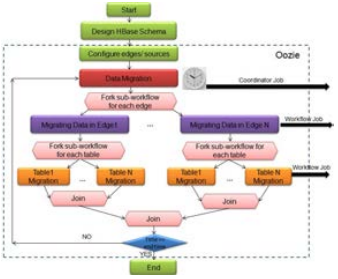
How much unrelated data is examined in a query matters a lot

Performance can be improved with data locality from 3rd dimension

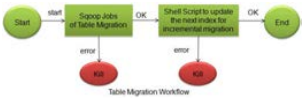
HBase Best Practices [2]

- Data Organization
  - Short Row key and Column names
  - Better to have a single Column Family
  - Not "too large" amount of data in one row
  - The 3rd dimension should also be used
  - The Bloom Filter should be configured to prune rows and columns
- Query Processing
  - "Scan" is better than "Get" for retrieving discontinuous keys, in spite of the irrelevant data scanned
  - "Scan" is preferable for small queries, while Coprocessor for large queries
  - Better to split one large query into multiple sub-queries than use one query with row filter mechanism

Workflow for Data-Aggregation on HBase



- Apache Sqoop
  - A tool for efficiently transferring bulk data between Apache Hadoop and structured data stores such as relational databases
  - **However**, it does not natively support customized HBase Schema
- Apache Oozie
  - A workflow scheduler system to manage Hadoop jobs
  - **Workflow jobs**: Directed Acyclical Graphs (DAGs) of actions
  - **Coordinator Jobs**: Recurrent Oozie Workflow jobs triggered by time (frequency) and data availability
  - **Bundle Jobs**: a set of coordinator jobs
  - Scalable, Reliable and Extensible



Framework in Action

- HCA-T Application
  - a mobile application that communicates with a scheduling service to tell home-care aides where to go and what to do
  - HCAs visit the patients based on the appointment and update the service status scheduled in the appointment
- Dataset
  - 100,000 patients, 22,000 HCAs in each province
  - Each day, four working blocks (AM, Noon, PM, Supper)
  - In each appointment, 10 to 20 services should be done, and 5-10 pictures/notes will be uploaded
  - In a year, for one province/edge, the meta data (excluding pictures) is around 180GB
- Environment Setup
  - Core: 8-node Hadoop and HBase Cluster, 1 node for HCAT migration with Oozie and Sqoop, 1 node for HCAT Analytics Engine, 1 node for HCAT Analytics Console
  - Edges: Two edges with HCAT server installed on it

Related Publications

[1] Dan Han, Eleni Stroulia, A three-dimensional data model in HBase for large time-series dataset analysis. MESOCA 2012: 47-56

[2] Dan Han, Eleni, Stroulia, HGrid: A Data Model for Large Geospatial Data Sets in HBase. (To appear) CLOUD 2013

Future Work

- How to configure the Hadoop and HBase cluster based on a certain amount of data
- How to make HBase work in real-time data processing

Acknowledgments: NSERC (SAVI), IBM, Cybera, U of Alberta

FIGURE 1.4: *Design Patterns for Spatio-temporal Applications on HBase from Dan Han and Eleni Stroulia at University of Alberta (Theme 1)*

5. SAVI Billboard

Sudhakar Ganti, **Hausi A. Müller**, U. Victoria

The objective of Billboard is to provide a virtual public post-it message center for users to post Text Messages on. The concept is identical to public peg-boards. Each virtual Billboard is associated with a geographical location. Users who are geographically near the virtual Billboard can read and post messages on the board.

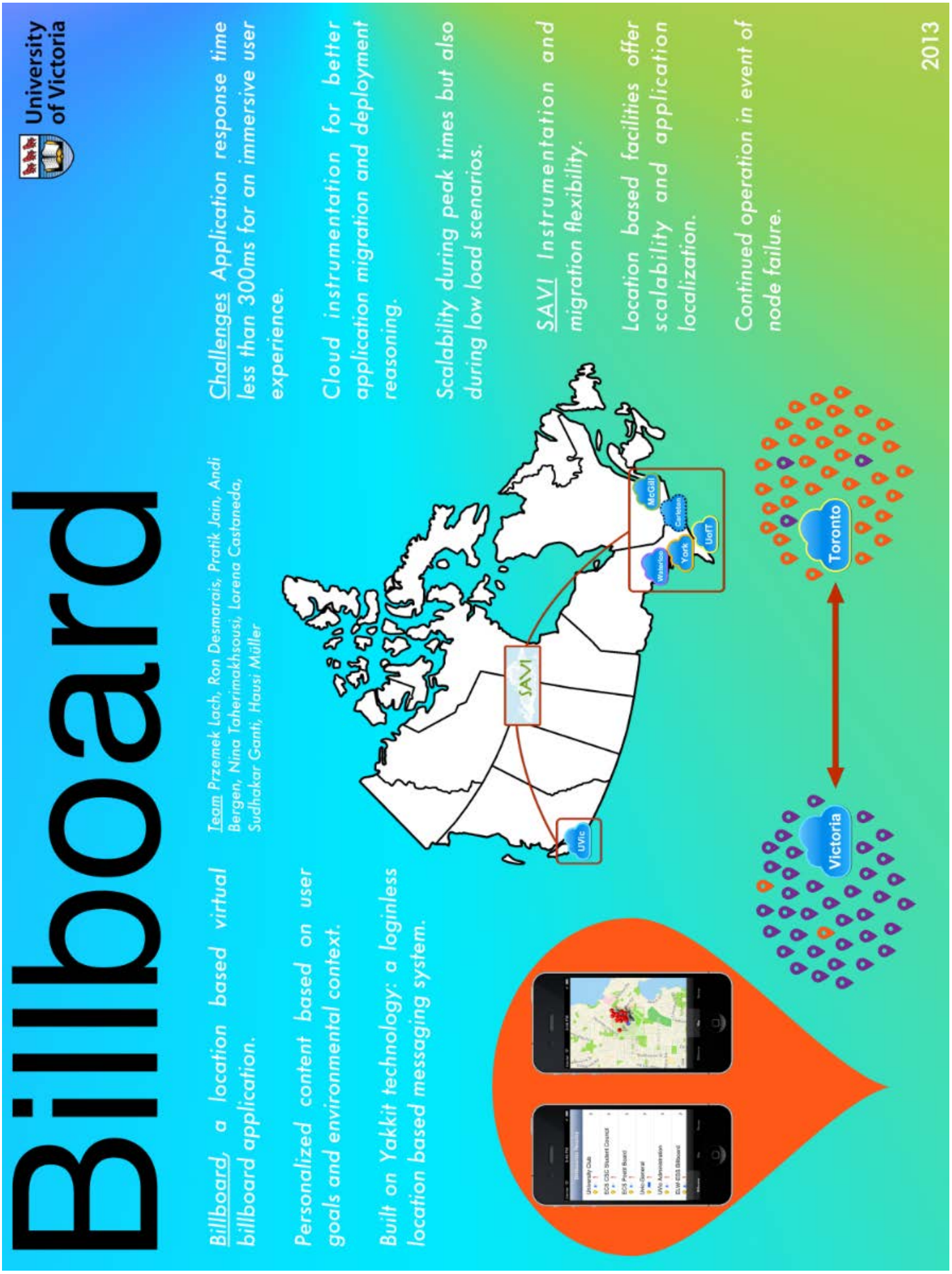


FIGURE 1.5: *SAVI Billboard Post-it Message Center from Przemek Lach, Ron Desmarais, Pratik Jain, Andi Bergen, Nina Taherimakhsousi, Lorena Castaneda, Sudhakar Ganti and Hausi Müller at University of Victoria (Theme 1)*

6. SAVI Context Based Face Recognition

Nina Taheri, Sudhakar Ganti, **Hausi A. Müller**, U. Victoria

During the past several years, face recognition in video has received significant attention. For the video monitoring and artificial vision, real time face recognition has very important meaning. The current method is still very susceptible to the illumination condition, non-real time and very common to fail to track the target face especially when partly covered or moving fast. In this work, we propose to use AdaBoost Cascade for face detection and then in order to recognize the candidate faces, they will be analyzed by the PCA. The implementation shows that the algorithm has quite good performance in terms of real-time and the tracking procedure is triggered accurately.

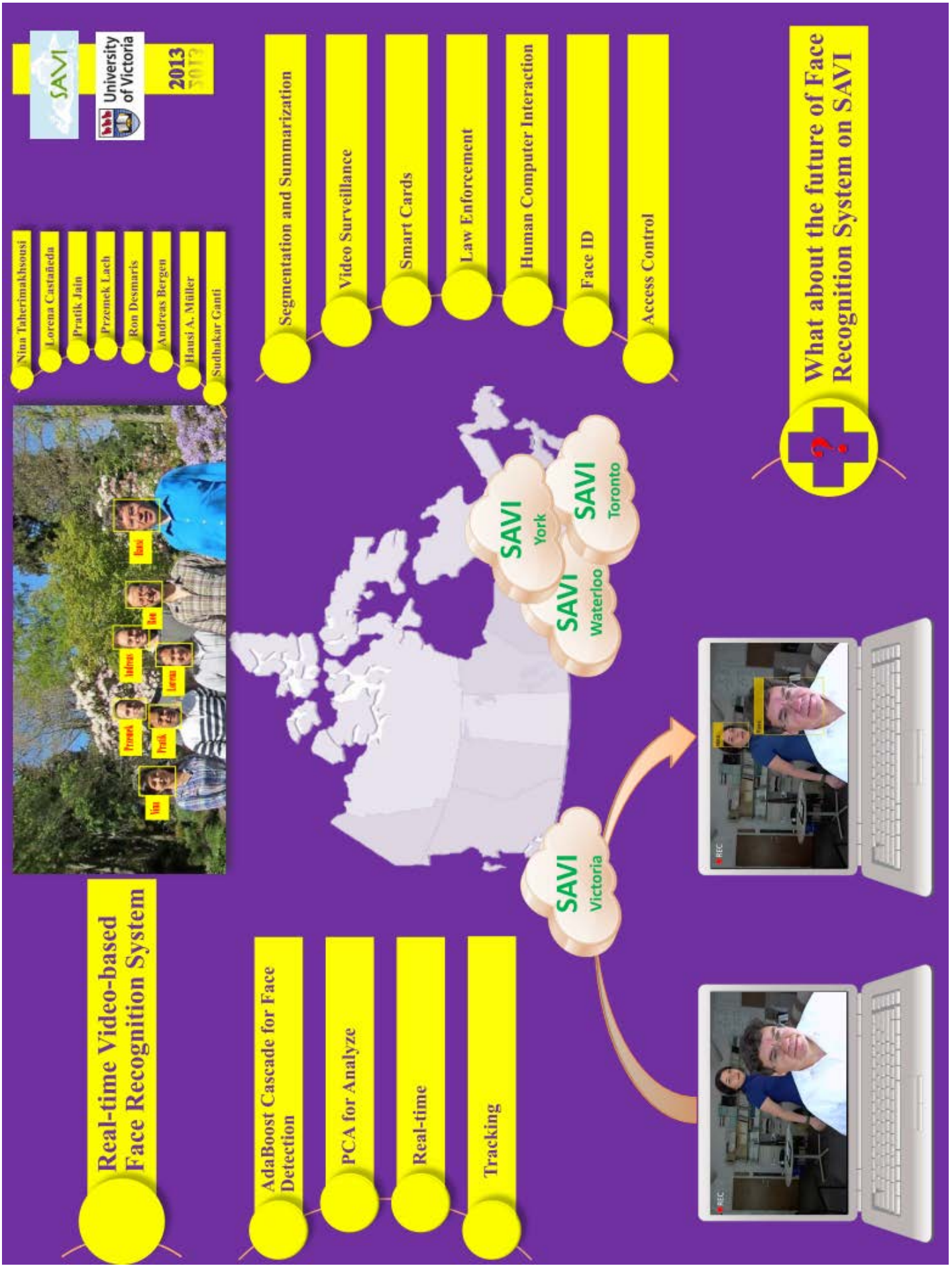


FIGURE 1.6: *SAVI Context Based Face Recognition System from Nina Taherimakhsousi, Lorena Castaneda, Pratik Jain, Przemek Lach, Ron Desmarais, Andreas Bergen, Hausi Müller and Sudhakar Ganti at University of Victoria (Theme 1)*

7. SAVI PALTask

Andreas Bergen, Lorena Castañeda, Pratik Jain, Sudhakar Ganti, **Hausi A. Müller**, U. Victoria

PALTask is a chat based tool which integrates personalized context in order to perform automation of web tasks such as: simultaneously searching for resources on the internet (video, websites, images, audio files) among different providers; facilitating sharing of web resources. To achieve this personalized automated search, a user's personal context-sphere is taken into account, as well as an analysis of the chat context. The concept is similar to existing chat applications where two users enter into communication with each other. Each instance of the chat client application is associated with a fixed geographic location. However, chat client users can benefit from dynamically migrating instances of the chat server application to a location that improves the users' experience (e.g. via improved latency and/or bandwidth). In principle this is similar to Skype's approach of nodes and supernodes. However, while Skype's nodes are positioned at fixed geographic locations and user requests are redirected to connect to specific nodes, PALTask dynamically migrates lightweight instances of the server to locations which improve the experience of its users on a case by case basis. The locations of lightweigth PALTask server instances are determined dynamically based on the geographic location of its users.

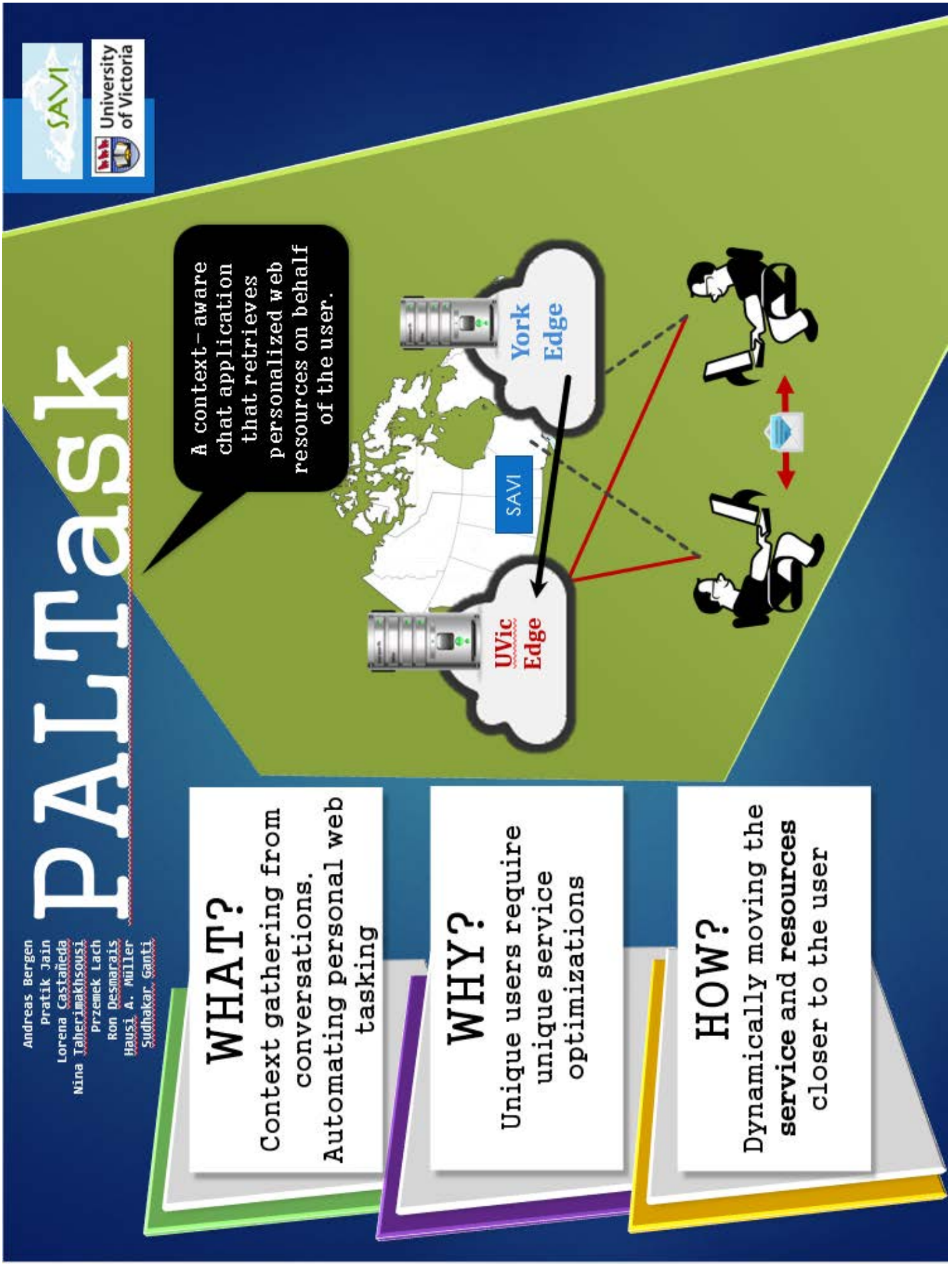


FIGURE 1.7: *SAVI PALTask - Personal Context Assistant from Andreas Bergen, Pratik Jain, Lorena Castañeda, Nina Taherimakhsousi, Przemek Lach, Ron Desmarais, Hausi A. Müller and Sudhakar Ganti at University of Victoria (Theme 1)*

7A. SAVI Newscast

Ron Desmarais, Przemek Lach, Sudhakar Ganti, **Hausi A.Müller**, U. Victoria

The objective of the Newscast application is to provide a customized video feed for each region of the country. The newscast is recorded in front of a green screen and stored at the core. When the newscast is broadcast the video is streamed from the core to each edge. The edge removes the green screen and replaces it with a background that is suitable for the given location. For example, viewers in Vancouver will have a backdrop of Vancouver while viewers in Calgary will have a Calgary backdrop.

FIGURE 1.8: *SAVI NewsCast - Customized Video Feed from Dr. Hausi Muller, Ron Desmarais, Przemek Lach, and Sudhakar Ganti at University of Victoria (Theme 1)*

Page for Notes

Theme 2 Extended Cloud Computing

Research Team

Marin Litoiu (lead), York University

John Chinneck, Carleton University

Kenneth Salem, University of Waterloo

Murray Woodside, Carleton University

Michael Smit, Post-Doctoral Fellow, York University

NSERC Funded Canadian Strategic Network NET GP397724-10
Smart Applications on Virtual Infrastructure ©2013
August 2013

8: Supporting Application Development with Structured Queries in the Cloud

Michael Smit, Bradley Simmons, Mark Shtern and **Marin Litoiu**, York U.

We present a structured query language for the cloud, Cloud SQL, along with a system and methodology for acquiring and organizing information from cloud providers and other entities in the cloud ecosystem such that it can be queried. It allows developers to run queries on data organized based on their semantic understanding of the cloud.

Supporting Application Development with Structured Queries in the Cloud

Michael Smit, Bradley Simmons, Mark Shtern, and Marin Litoiu
York University, Toronto, Canada

YORK UNIVERSITÉ UNIVERSITY

Motivation

- Infrastructure-as-a-service (IaaS) providers have varying offerings, terminology, and pricing models (semantics) as well as different APIs and user interfaces (syntax): steep learning curve
- Creating a common ontology a) is difficult and b) requires learning an evolving ontology and provider-specific exceptions.
- We believe a declarative structured language will reduce development costs and make the multi-cloud accessible to a broader set of developers, like SQL did for data.
- Requires not just a query language, but also a **system** and **methodology** for acquiring and organizing information from cloud providers (and other entities in the cloud ecosystem) such that it can be queried.

Data Model

- Data model maps from user knowledge to provider semantics.
  - ⋆ A provider is any entity in the cloud ecosystem.
- *Provider data tables* are defined rows and columns containing API data, and can be constructed automatically or by the community.
- *Views* also have a row/column structure defined by the user.
- A view is built by joining and/or aggregating provider data tables (PDTs); values are populated at runtime automatically.
- A view may specify data transformations and must specify supported operations (INSERT, DELETE, SELECT, etc.)

Related Work

- SEQUEL proposed to "bring the non-professional user into effective communication" with a data source.
  D. D. Chamberlin and R. F. Boyce, "SEQUEL: A structured English query language," in *Proceedings of the 1974 ACM SIGFIDET (now SIGMOD) workshop on Data description, access and control*. New York, NY, USA: ACM, 1974, pp. 249–264.
- Using queries to manage unstructured data is precedented
  A. Bozzon, M. Brambilla, S. Ceri, and P. Fraternali, "Liquid query: multi-domain exploratory search on the web," in Proceedings of the 19th international conference on World wide web, ser. WWW '10. New York, NY, USA: ACM, 2010, pp. 161–170.
  D. Konopnicki and O. Shmueli, "Information gathering in the world- wide web: the W3QL query language and the W3QS system," ACM Trans. Database Syst., vol. 23, no. 4, pp. 369–410, Dec. 1998.
  E. Spertus and L. A. Stein, "Squeal: a structured query language for the web," Computer Networks, vol. 33, no. 16, pp. 95 – 103, 2000.
- Global ontologies and abstraction layers are temporary band-aids
  L. Youseff, M. Butrico, and D. Da Silva, "Toward a unified ontology of cloud computing," in *Grid Computing Environments Workshop, 2008. GCE'08*. IEEE, 2008, pp. 1–10.
  P. Pawluk, B. Simmons, M. Smit, M. Litoiu, and S. Mankovski, "Introducing STRATOS: A cloud broker service," in *IEEE 5th International Conference on Cloud Computing (CLOUD)*, 2012, pp. 891 –898.

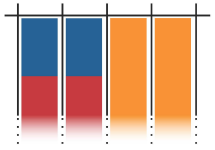
Perception: Using "the cloud" is magical and wonderful, and using the multi-cloud is like using the cloud, but with more magic and wonder.
Reality: Using cloud infrastructure providers can be painful, involving various features, limitations, semantics, etc.; multi-clouds multiply the pain.

CloudSQL aims to replace provider-specific semantics and syntaxes with a standard syntax for managing cloud resources paired with user-specified semantics, seamlessly and **automatically**, with or without provider support.

1. Provider APIs are represented in a structured table-like format (see "Data Model"). Providers can be cloud infrastructure providers, or any provider with an API.

2. The application developer identifies data they are interested in from these tables.

3. A view is created by joining and aggregating these tables. The view maps user semantics to provider terminology, resulting in unified user-driven semantics.



4. This view is queried and manipulated using a SQL-like language (this is the only step where the application developer must be involved).

```
SELECT instance_name, cost, memory FROM running_instances WHERE cost > 1.50;
```

Case Study

- Built one PDT based on JClouds for information about running virtual machines on various providers, and a second based on the CloudyMetrics.com API for information on pricing.
- Built a View to show instances and hourly cost. Joins based on JClouds::ListNodes::size and CloudyMetrics::name; the joined field is renamed to the developer's semantics: InstanceSize. Columns from specific providers are included and also renamed.

InstanceSize	CostPerHour	Status	OSFlavour
m1.small	0.08	RUNNING	linux
m1.medium	0.23	RUNNING	mswin
m1.large	0.32	SUSPENDED	linux
t1.micro	0.02	RUNNING	linux

Joined CloudyMetrics JClouds

- We constructed views and queries to replicate the functionality of previous cloud tools we authored (a cloud monitoring aggregation service and a cloud broker); CloudSQL required fewer lines of code and substantially less cloud-specific knowledge.

Cloud SQL

- `SELECT ... FROM ... WHERE ...`: Acquire the named columns (or *) from named views, when given conditions are met.
- `SELECT CACHED ... FROM ... WHERE ...`: Like SELECT, but uses results from the last time the view was populated.
- `INSERT INTO ... (col)+ VALUES (val)+`: Add a record to a view. This operation is only defined for certain views.
- `DELETE FROM ... WHERE ...`: Delete matching records from a view. Again only applies if this is possible in the PDT.

Implementation

- Allows PDTs to be defined using JSON, mapping return values from API calls to a table structure, or programmatically.
- An XML-based view specification language is used to specify views, their properties and transformations, and how PDTs are joined and aggregated.
- A Query Processor accepts SELECT queries on views, builds the view based on the PDTs, and returns the result table.

NSERC CRSNG Ontario IBM Centers for Advanced Studies

Web: http://www.ceraslabs.com
Email: {msmit,bsimmons,mlitoiu}@yorku.ca, mark@cse.yorku.ca

FIGURE 2.1: *Supporting Application Development with Structured Queries in the Cloud from Michael Smit, Bradley Simmons, Mark Shtern and Marin Litoiu at York University (Theme 2)*

9: Model Predictive Resource Management for Cloud

Hamoun Ghanbari, Brad Simmons and **Marin Litoiu**, York U.

We present a method of optimizing the cloud infrastructure for a set of applications. In this context, a data center provider attempts to meet all service level objectives (SLO) with its clients (i.e., for offered services) while minimizing its own infrastructural cost. Meeting these objectives can be expressed as an Optimal Service Placement (OSP) problem that we address by using Model Predictive Control (MPC).

Model Predictive Resource Management for Cloud

***Hamoun Ghanbari*[1], *Brad Simmons* [1], *Marin Litoiu* [1]**

[1]*Dept. of Computer Science, York University, Toronto*

Introduction

We present a method of optimizing the cloud infrastructure for a set of applications. In this context, a data center provider attempts to meet all service level objectives (SLO) with its clients (i.e., for offered services) while minimizing its own infrastructural cost. Meeting these objectives can be expressed as an Optimal Service Placement (OSP) problem or Optimal Autoscaling (OA) problem that we address by using Model Predictive Control (MPC).

Tracking Adaptive Performance Models using Dynamic Clustering of User Classes

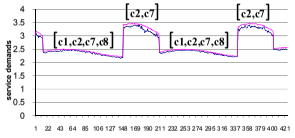
Models enable the controller to project system's behavior and state under different actions in the future. To do this projection one needs to estimate the current state of the system using the data obtained from the system (or environment) in in ongoing fashion.

$$\underset{x_0,\ldots,x_T}{\text{minimize}}\ (x_0-\hat{x}_0)\Pi^{-1}(x_0-\hat{x}_0)+\sum_{t'=0}^{T-1} w_k^T Q_k^{-1} w_k+\sum_{t'=0}^{T} v_k^T R_k^{-1} v_k$$

subject to:

$$\begin{aligned}
&\hat{x}_{t'} = [\vec{\hat{d}}_{t'}, \vec{\hat{N}}_{t'}, \vec{\hat{Z}}_{t'}]^T \\
&u_{t'} = [\vec{\tilde{d}}_{t'}, \vec{\tilde{N}}_{t'}, \vec{\tilde{Z}}_{t'}, \vec{\tilde{\Omega}}_{t'}, \vec{\tilde{\rho}}_{t'}]^T \\
&z_{t'} = [\vec{R}_{t'}, \overrightarrow{U_{h,k',t'}}, \overrightarrow{X_{s,h,t'}}]^T \\
&x_{t'+1} = x_{t'} + w_{t'} \\
&z_{t'} = LQM_{u_{t'}}(\hat{x}_{t'}) + v_{t'} \\
&d_{c,s,k} > 0 && \text{for each } c,k \\
&0 < U_k < 1 && \text{for each } k
\end{aligned}$$

aiming at finding a low complexity model yet with enough accuracy. We propose a combination of clustering algorithm and tracking filter for effective grouping of classes of services.



Optimal Service Placement and Replication for Fixed Number of Hosts via Model Predictive Control

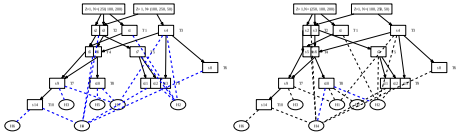
The problem we targeted here was choosing a sequence of optimal placement actions $(u_0^s,\ldots,u_N^s)$ that minimizes the long term resource cost while trying to satisfy SLOs over time, without trashing. We propose an algorithm that solves the optimization problem through MPC by iterative convex optimization. The solution finds the number of service replicas, and optimal deployment of replicas on available hardware at each step. In addition, it considers constraints on the number of possible service migrations, and different types of servers at each time interval in the overall optimization.

$$\underset{u_0,\ldots,u_{T'-1}}{\text{minimize}}\ E\left[r_1\sum_{t'=1}^{T'}\sum_{h} c_h U_{h,t'} + r_2\sum_{t'=1}^{T'}\sum_{h}\sum_{s}|u_{t'}|\right]$$

subject to:

$$\begin{aligned}
&W_{t'} = A\,W_{t'-1} \\
&\theta_{t'} = \theta_{t'-1} + u_{t'} \\
&d_{c,s,h,t'} = d_{c,s}\theta_{s,h,t'} \\
&R_{t'} = \mathrm{LQM}_{\Omega,\sigma}(\overrightarrow{d_{c,s,h,t'}}, W_{t'}) \\
&R_{c,t'} > R_c^{\mathrm{SLA}} && \text{for each } t',c \\
&N_{c,t'} < N_c^{\mathrm{SLA}} && \text{for each } t',c
\end{aligned}$$

where T' denotes the service centre lifetime, $(u_0,\ldots,u_{T'-1})$ is the placement changes performed during this lifetime, $E[\ldots]$ represents the expected value, and cost includes the cost of cloud provider, $\{W_0,\ldots,W_{T'-1}\}$ is the stochastic workload (composed of $N_{t'}$,$Z_{t'}$,$V_{c,s,t'}$) over time.



Optimal Autoscaling in a PaaS Managed Environment

The infrastructure as a service (IaaS) layer solves the placement problem for virtual machines on available hosts. At PaaS layer, we can decide how many resources to add, where and when to add them.

$$\underset{(m_{cl,t'}),(\beta_{s,cl,t'})}{\text{minimize}}\ J_{t'}^{\text{cost}} + J_{t'}^{\text{trashing}}$$

$$= E\left[r_1\sum_{t'=1}^{T'}\sum_{cl=1}^{Cl} p_{cl,t'} m_{cl,t'} + r_2\sum_{t'=1}^{T'}\sum_{cl}^{Cl}(\Delta m_{cl,t'})^2\right]$$

subject to:

$$\begin{aligned}
&m_{cl,t'+1} = m_{cl,t'} + \Delta m_{cl,t'} && \text{for each } cl,t' \\
&d_{c,s,cl,t'} = d_{c,s}\cdot\beta_{s,cl,t'} && \text{for each } s,cl,t' \\
&\beta_{s,cl,t'}\cdot\beta_{s,cl}^{\text{deployed}} = \beta_{s,cl,t'} && \text{for each } s,cl,t' \\
&\vec{R}_{t'} = \mathrm{LQM2}(\overrightarrow{m_{cl,t'}}, \overrightarrow{d_{c,s,cl,t'}}, \vec{W}_{t'}) \\
&R_{c,t'} > R_c^{\mathrm{SLA}} && \text{for each } t',c \\
&N_{c,t'} < N_c^{\mathrm{SLA}} && \text{for each } t',c
\end{aligned}$$

where,

$\mathbf{p_{cl,t'}}$ is the cost of adding a virtual machine to a cluster cl which depends on the type of virtual machine used and the time-step.

LQM2 is, at each t', a LQM with:

(i) C hardware delay centres ($H_c^c|c=1\ldots C$) (i.e. infinite multiplicity), C software centres ($S_c^c|c=1\ldots C$), and C classes with population of N_c to model the workload, with demand matrix of $d_{c,c,c}=Z_c$ otherwise 0

(ii) One hardware queuing centre with multiplicity $m_{cl,t'}$ for each cluster $cl=1\ldots Cl$, a software queuing centre for each service replica, and demand matrix who's elements are $d_{c,s,h}=d_{c,s,h,t'}$.

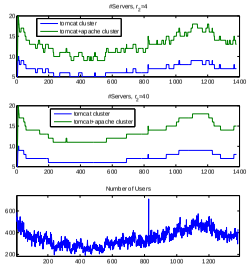


FIGURE 2.2: *Model Predictive Resource Management for Cloud from Hamoun Ghanbari, Brad Simmons and Marin Litoiu at York University (Theme 2)*

10: Pattern-based Deployment Service for Next Generation Clouds

Hongbin Lu, Mark Shtern, Bradley Simmons, Michael Smit and **Marin Litoiu**, York U.

We present a flexible deployment service for cloud computing. The service facilitates the specification and the execution of cloud deployment plans for applications. An application is described through a pattern, an abstract view that captures the logical view of the application and its mapping into cloud resources. The service instantiates the pattern in the cloud and allows for runtime updates of the deployment.

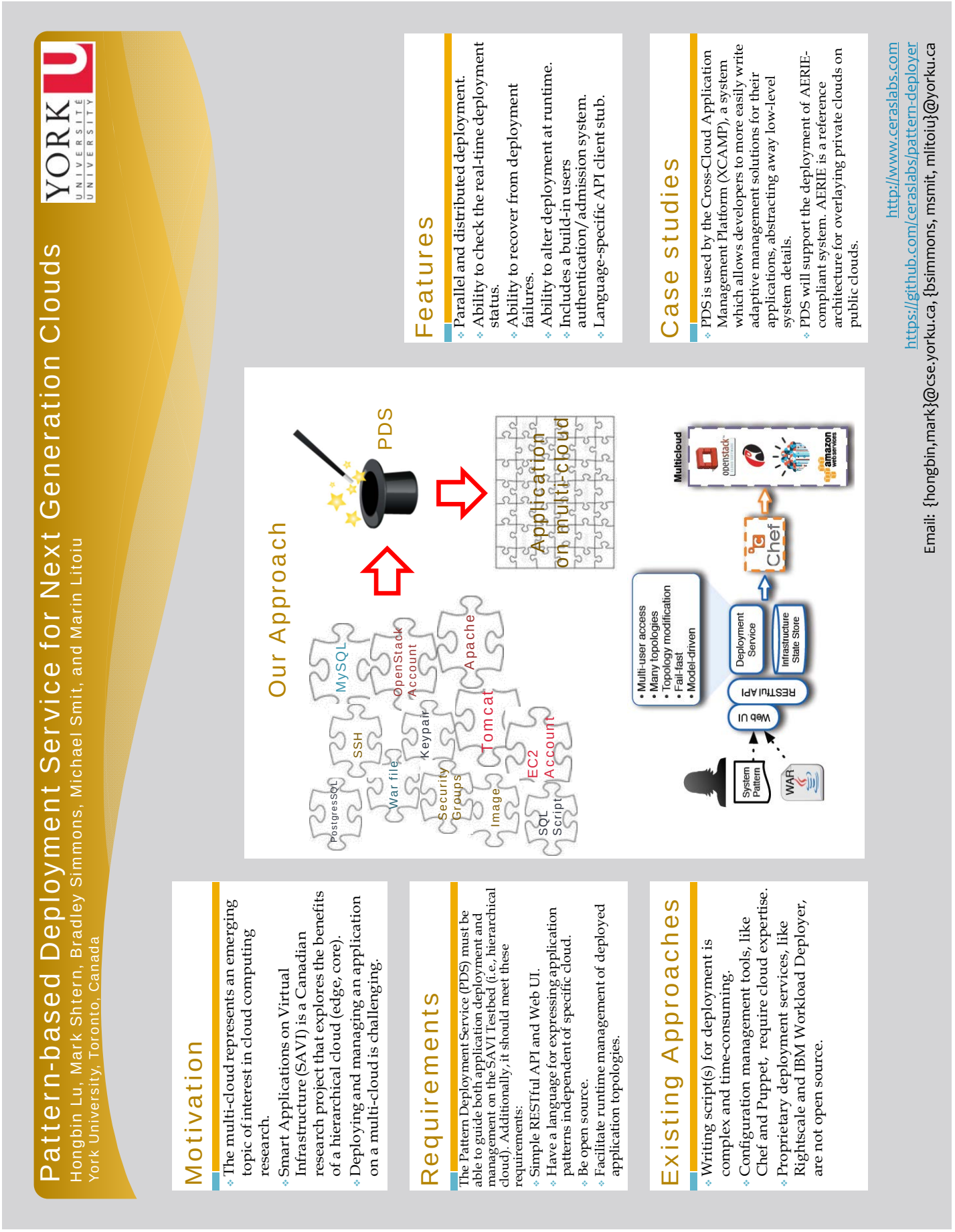


FIGURE 2.3: *Pattern-based Deployment from Marin Litoiu at York University (Theme 2)*

11: Towards Cloud-aware Multi Class Adaptive Models

Cornel Barna, Mark Shtern and **Marin Litoiu**, York U.

We present a method to automatically synchronize and keep synchronized a multi-class performance model with an application deployed in the SAVI cloud environment. The model can be used for runtime management decisions by an autonomic manager.

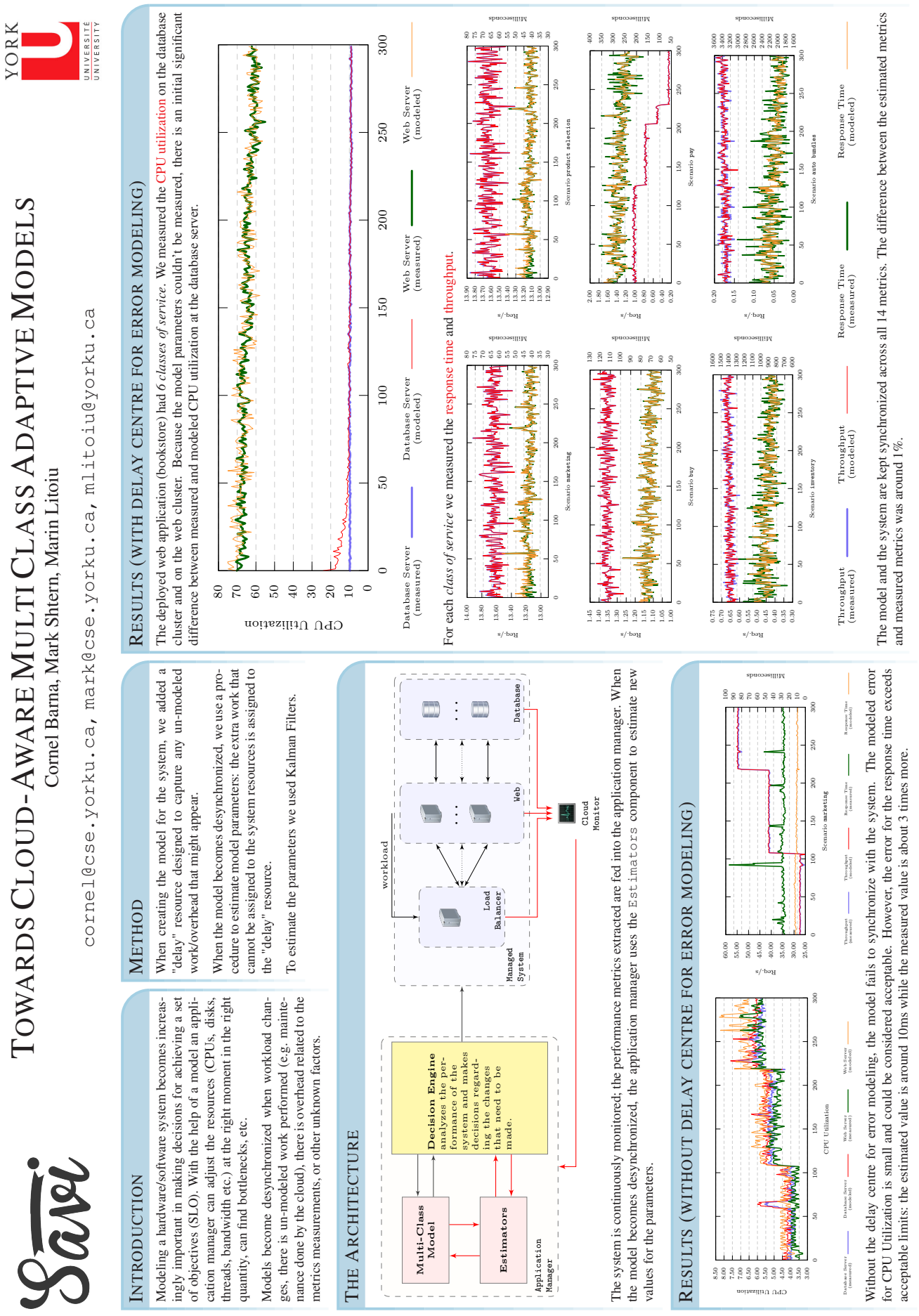


FIGURE 2.4: *Adaptive Model from Cornel Barna and Marin Litoiu at York University (Theme 2)*

12. Algorithms for Task Allocation Considering Communication Delays

Ravneet Kaur, **John Chinneck** and **Murray Woodside**, Carleton U.

In a distributed computing system like a cloud, tasks should be assigned taking into account communication latencies as well as execution or power costs. We are exploring graph algorithms for allocating tasks to the core or edge while considering communication delays due to inter-task communication.

Algorithms for Task Allocation considering Communication Delays

Ravneet Kaur, Dr. John Chinneck, Dr. Murray Woodside
System and Computer Engineering
Carleton University, Canada

SCOPE

Tasks should be assigned to the edge or core to minimize the response time, which includes
- inter-task communication delays
- task execution times.

Our goal is to develop algorithms for optimally allocating tasks on the core and edge clouds while taking into account delays due to inter-task communication using graph cutting. Our initial approach will use the min cut - max flow algorithm to achieve a good task assignment by manipulating weights and moving nodes towards core.

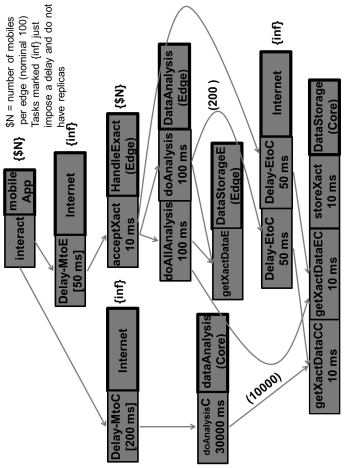
HCAT

HCAT (Home Care Aides Technology): SAVI test application. The HCAT architecture has been re-interpreted here to allow distributing the functions over the edge and the core.
The resulting modules, some of them representing collections of HCAT functional modules, are deployed in the mobile terminal, edge cloud and core cloud. The video/conferencing service is not considered here.

Simplified LQN Model for HCAT

Includes additional network delay pseudo-tasks for delays between mobile and edge (MtoE), mobile and core (MtoC), edge-to-core (EtoC).
A user transaction originates from mobile and goes to front-end transaction HandlXact that hands it off to be processed by applications represented here by DataAnalysis (assumed resident in the edge). It is either entirely performed here or is handed back to a core cloud process dataAnalysis, for heavy-duty processing.
- Task mobileApp represents HCAT application running on the mobile device.
- Through the interact entry, the mobile makes requests
  - HandleXact with entry acceptXact
  - DataAnalysisE with enteries doAnalysis and doAllAnalysis
  - DataStorageE with entry getXactDataE
  - DataAnalysisC with entry doAnalysisC
  - DataStorageC with entries StoreXact,getXactDataCC and getXactDataEC

LQN Model



Graph Representation

Edge Host and core host are each modeled as a single 'host' entity

Edge Weights & Table

Weight = total latency in one user response
= (latency per request) x (requests per user response).
Latency incurred with communication among nodes, nodes and hosts and among the hosts for each transaction. Frequency of sending requests: Multiple requests can be made among the nodes.
If the nodes are co-residing we assume the weight to be 0.
Let 'i' 'j' be two communicating nodes:
C(i,j) be the communication delay
f(i,j) be the frequency at which requests are sent.
Weight on Edge W(i,j)=C(i,j)* f(i,j)
Assumptions: *Processing weights are not included, If the nodes are co-residing we assume the weight to be 0.*

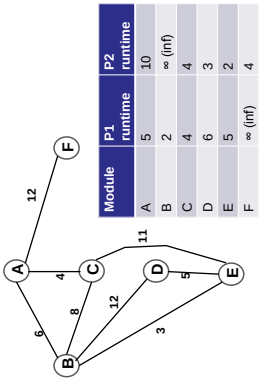
	Node	Node	Weight
W1	acceptXact	doAllAnalysis	0.6
W2	doAllAnalysis	getXactDataE	1.2 (0.6*2=12)
W3	acceptXact	doAnalysisE	0.2
W4	doAnalysisE	getXactDataEC	$nSAD (0.2*10=2)
W5	acceptXact	storeXact	0.2
W6	acceptXact	doAnalysisC	0.01
W7	doAnalysisC	getXactDataCC	$nBAD (100)

Graph Cutting

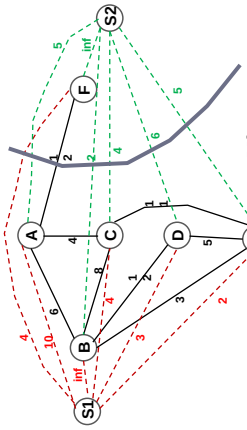
Cut = a set of edges whose removal partitions the graph into two disjoint subsets.
Min Cut = Non trivial partition of the graph vertices into two parts such that the cut weight, the sum of the weights of the edges connecting the two parts is minimum.
Min-cut minimizes sum of:
- **Latency (cost on arcs between task nodes)**
- **Execution time (cost on arcs between tasks and hosts)**

Cheap process and takes into account the performance constraints.
Weaknesses: Does not take into account the capacity constraints.

Min Cut Methodology (STONE)



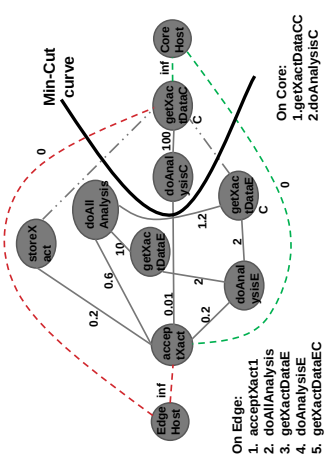
Module	P1 runtime	P2 runtime
A	5	10
B	2	∞ (inf)
C	4	4
D	6	3
E	5	2
F	∞ (inf)	4



Stone: Multiprocessor Scheduling

Min cost cutline

Example: Min-Cut on Simplified HCAT Model



Overall Optimization Approach

- Min cut does not account for capacity constraints
- May overload the edge or core hosts or their memory/storage capacities
- Overloading the core may not affect the system as much as the edge host

Approach
- Analyse the constraints (capacity and performance)
- Find a strategy to modify the weights, to shift some deployments.
- Iterate

Carleton University

SAVI Workshop: July 2013, University of Toronto

FIGURE 2.5: *Task Allocation from Ravneet Kaur, John Chinneck and Murray Woodside at Carleton University (Theme 2)*

13. Impact of Network Latency on Cloud Performance

Adnan Faisal, Dorina Petriu, **Murray Woodside**, Carleton U.

Cloud computing provides computing resources (both hardware and software) as a utility. However cloud data centers could be located at physically distant locations from the end-users, which might cause longer than expected response times. A way to reduce this response time is to partition some applications and data and place them in edge clouds that are located near to the end-users, possibly in the same city. In reality, the extra network latency added by the edge clouds might reduce the overall system performance quite unexpectedly in some cases. The impact of network latency on cloud performance (using edge clouds) is studied, with the help of a health-care related case study.

Effect of Network Latency on Cloud Performance

Adnan Faisal, Dorina Petriu and Murray Woodside
Dept. of Systems and Computer Eng.
Carleton University
{faisal|petriu|cmw}@sce.carleton.ca

July 2013

- In cloud computing, an "edge cloud" may be introduced close to some of the end users, to give faster service for very demanding applications.
- Parts in the core and edge may then have to communicate, introducing associated network latencies.
- We use a Layered Queueing Network performance model to explore the impact of network latency and some possible deployment choices on the responsiveness of a distributed health-care application called HCAT.

1

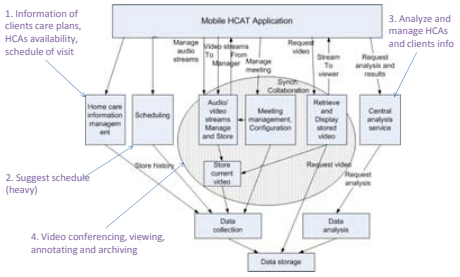
The context of the work

- Evaluate the impact of communication mechanisms on the performance in a distributed application
- A range of mechanisms exists
  - Network latency: different deployments
  - Transport protocol: TCP / UDP
  - Middleware support: CORBA/ RMI / SOAP
- Larger goal:
  - Build a library of performance models for communication mechanism
  - automate the process of composing mechanism to performance models
  - run the composed model to take informed decision about choices that meet performance requirements.
- In this poster we show the impact of network latency for HCAT when it is deployed in edge and core clouds

2

HCAT (Home Care Aides Technology):

a distributed health-care application with 4 services



3

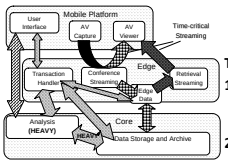
Issues in the present study

1. **The impact of network latency on performance**
   - Performance impact of latency variation (between user and edge, between user and core and between edge and core) is analyzed
2. **Compare 3 deployments**
   - Deploy to both edge and core
   - Deploy fully to edge
   - Deploy fully to core
3. **The impact of data partitioning**
   - Data are moved between edge and core
   - This affects the core data accesses from the edge

4

From Functional Model to Performance Model (1)

- In the performance model, we do not separate classes based on their functions but rather on their service demands
- We have **regrouped** the four services to create **four classes of traffic**



Two Transaction classes

1. **Heavy transactions:**
   - Represents Central Analysis Service
   - large service demand
2. **Light transactions:**
   - Represents all light transaction services
   - Includes home care information management and scheduling

Two Video Streaming classes

1. **Conference video streaming**:
   - Transmit/view conference video
   - Time critical
2. **Video retrieval streaming**
   - View archived video
   - Videos might be available for review
   - Added by us to HCAT capabilities

5

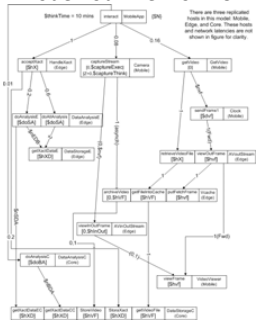
Functional Model to Performance Model (2)

Mapping HCAT services to LQN tasks

- **Heavy transactions** (doAnalysisC)
  - Represented by *DataAnalysis* task
  - Requires access *to DataStorage* task
- **Light transactions handling** (acceptXact)
  - Represented using front-end task *HandleXact*, supported by a back-end *DataAnalysis* task.
  - Requires access to *DataStorage* task
- **Conference video steaming** (captureStream)
  - Modeled by the *AVInOutStream* task
  - This task receives a video stream from one mobile and redirects it to a set of (*$nVC-1)* other mobiles
  - *$nVC* = average number of participants in a video conference.
- **Video retrieval streaming** (getVideo)
  - modeled by the task *AVOutStream* task
  - This task retrieves a video file from storage and sends it to one mobile that receives it via the *GetVideo* task.

6

LQN Model: our view of HCAT



7

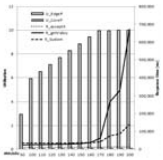
Model variables

$N = number of mobiles per edge (nominal 150)
$hX = demand to **handle a transaction** (nominal 10 ms)
$doSA = demand for a **small transaction** backend operation (e.g., a scheduling query, small analysis query) (nominal 100 ms)
$doBA = demand for a **large business intelligence-type operation** (nominal 0.5 min)
$hXD = demand to **get or store data for a simple transaction** in the archive (nominal 200 ms)
$hvf = demand to **handle a video frame** (compress/decompress, display) (nominal 10 ms)
$hVF = demand to **get and send or receive and store** a 5-minute **video file** (nominal 12 s)
$nvf (videoFrames) = number of **frames** in a **video** (nominal 7200 ms)
$dvf = **duration** of a single **video frame** (nominal 42 ms)
$captureExec**=$nvf*$hvf, **execution time taken to **capture a video file** (nominal 72 s)
$captureTotal=Total time a video is captured (nominal video conference duration 5 min)
$captureThink = $captureTotal - $captureExec. Camera's wait time between processing two consecutive frames, so a frame can be viewed by the mobile users. (nominal 228 s)
$nSDA = number of data transaction per **small analysis** (nominal 200)
$nBDA = number of data transactions per **big analysis** (nominal 10,000)
$nVC = the **number** of **participants** in a **video conference** (nominal 3)
$hInOut = Time taken to process a frame at cloud. Formula ($nVC - 1)*$hvf (nominal 20 ms)
$nEDA = No. of edge data access for a computation that takes place partially in edge and core (nominal 10)
$me = Mobile to edge latency (nominal 50 ms)
$mc = Mobile to core latency (nominal 200 ms)
$ec = Edge to core latency (nominal 50 ms)

8

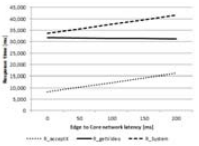
Experimental results(1)

Point of saturation



- We modeled one edge and one core cloud
- All 10 of the edge processors are saturated at 170 mobiles.
- Core is very lightly utilized

Impact of network latency



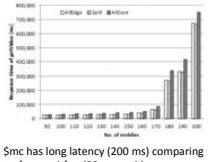
- Varying $me and $mc has a small impact
- Changing **$ec has a large impact on performance** as there are many edge to core requests
- *R_acceptX* increases as we increase *$me*
- *R_getVideo* slightly decreases because *acceptXact* uses fewer common resources due to higher congestion

9

Experimental results(2):

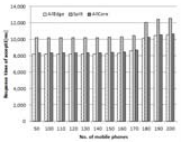
Different deployments

Response time of video retrieval



- $mc has long latency (200 ms) comparing to $me and $ec (50 ms each)
- Before saturation, *AllEdge* deployment is fastest due to its lowest latency
- Before saturation, *AllEdge* is only slightly better than *Split* deployment
- From saturation, *AllEdge* is worse than *Split* deployment, because Split deployment forwards some requests to core, which helps reducing response time.

Response time of light transactions



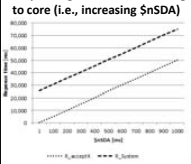
- Similar to *getVideo* class, *AllEdge* deployment gives the lowest response time for *R_acceptX*
- *Split* deployment is the worst of the 3 deployments, which is **contrary to our expectation of performance using the edge**
  - Too many edge to core requests cause the split deployment to perform worst

10

Experimental results(3):

Data partition effect

Requesting more data from edge to core (i.e., increasing $nSDA)



- Edge can not accommodate a huge database
- A small edge database causes an increment of $nSDA.
  - which increases R_acceptX and R_System.
- Data distribution requires a balance

Moving data from core to edge

Xact data access ratio between Edge:Core	R_accept [ms]	R_System [ms]
10:200	10,238.7	35,599
200:10	746.098	26,266

- We show the impact of having large edge database.
- We exchange the values of $nEDA and $nSDA from (10:200) to (200:10)
- **Having a large edge database has huge impact on performance** on *acceptXact* class (e.g., R_acceptX reduces from 10 s to 0.7 s)

11

Conclusion

- We created a model for a distributed health-care application to check whether it meets response time requirements.
- Edge-core latency and deployment were considered

Summary of Results:

- The default assumption that a *Split* deployment is better (in terms of performance) than *AllCore* or *AllEdge* deployments is over-ambitious.
- To gain performance using the edge, it needs to be sufficiently powerful.
- Edge should have large enough database to process locally the queries submitted to it. Too much edge-cloud interaction causes performance degradation.
- The most sensitive latency is Edge to Core. This is especially true for cases when the edge requires accessing the core for retrieving large amount of data.

Future works for SAVI:

- Evolve and validate the model against actual HCAT data.
- Analyze scalability of deployed HCAT system.
- Integrate the model with the strategy to optimize deployment.

12

FIGURE 2.6: *Effect of Network Latency from Adnan Faisal, Dorina Petriu and Murray Woodside at Carleton University (Theme 2)*

14. Simulation of Core-Edge Cloud Networks

Derek Hawker, **John Chinneck** and **Murray Woodside**, Carleton U.

A Cloud simulator (DCSim) is being evaluated for modeling the proposed SAVI core-edge cloud architecture. Several modifications will be required to represent latencies introduced by the geographical distance between core and edge cloud networks. The goals of simulation are to provide testing of Virtual Machine allocation policies on a variety of deployed network configurations and scales.

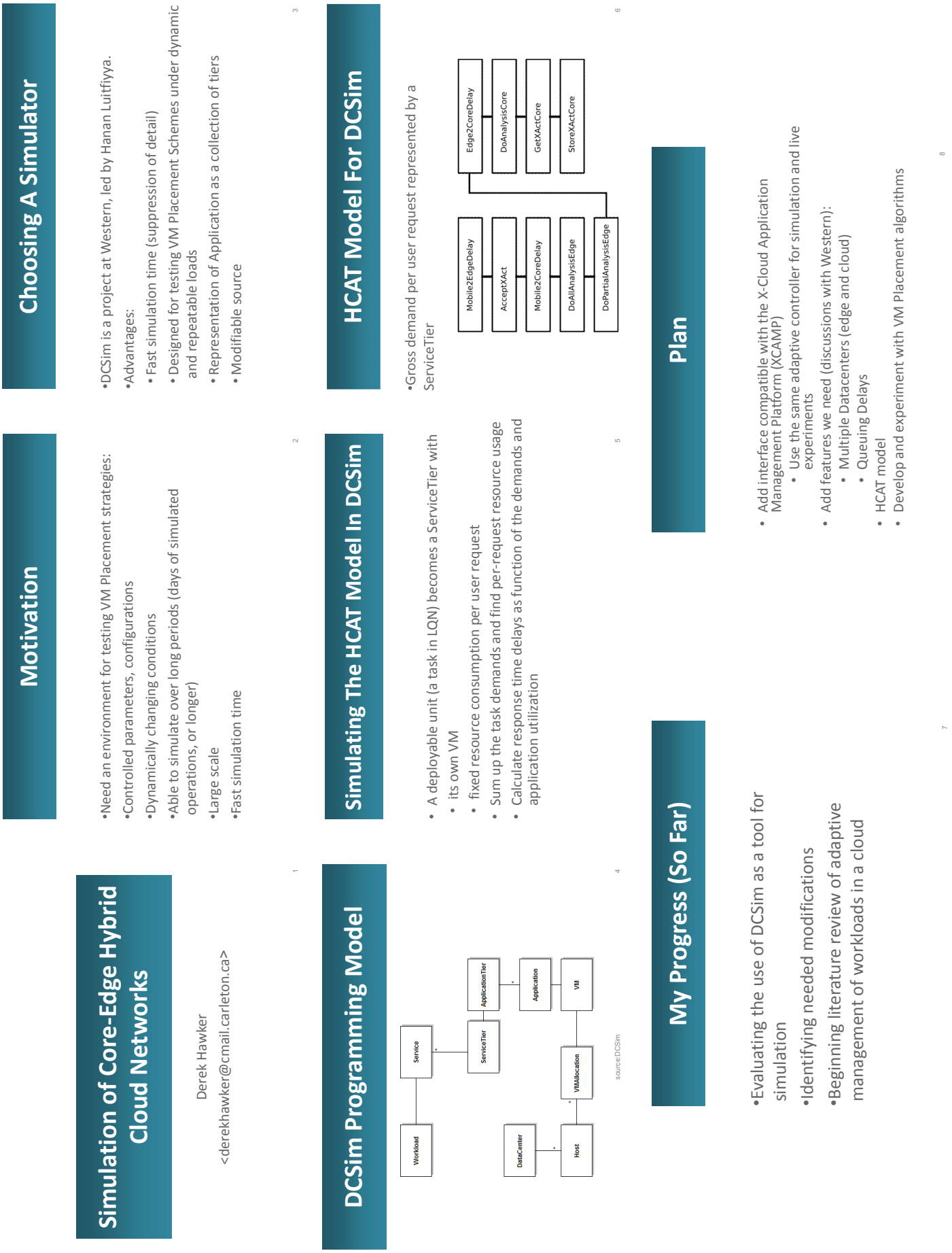


FIGURE 2.7: *Simulation of Core-Edge Hybrid Cloud Networks from Derek Hawker at Carleton University (Theme 2)*

Page for Notes

Theme 3 Smart Converged Edge

Research Team

Raouf Boutaba (lead), University of Waterloo

Paul Chow, University of Toronto

Yashar Ganjali, University of Toronto

Alberto Leon-Garcia, University of Toronto

NSERC Funded Canadian Strategic Network NET GP397724-10
Smart Applications on Virtual Infrastructure ©2013
August 2013

15. Dynamic Controller Provisioning in Software Defined Networks

Submitted to IEEE/ACM/IFIP Conference on Network and Service Management (CNSM 2013)

Md Faizul Bari, Arup Raton Roy, Shihabur Rahman Chowdhury, Qi Zhang, Mohamed Faten Zhani, Reaz Ahmed and **Raouf Boutaba**, U. Waterloo

Software Defined Networking (SDN) has emerged as a new paradigm that offers the programmability required to dynamically configure and control a network. A traditional SDN implementation relies on a central controller that runs the control plane. However, in a large-scale WAN deployment, this rudimentary centralized approach has several limitations related to performance and scalability. To address these issues, recent proposals have advocated deploying multiple controllers that work cooperatively to control a network. Nonetheless, this approach drags in an interesting problem, which we call the Dynamic Controller Provisioning Problem (DCPP). DCPP dynamically adapts the number of controllers and their locations with changing network conditions, in order to minimize flow setup time and communication overhead. In this work, we propose a framework for deploying multiple controllers within a WAN. Our framework dynamically adjusts the number of active controllers and delegates each controller with a subset of Openflow switches according to network dynamics while ensuring minimal flow setup time and communication overhead. To this end, we formulate the optimal controller provisioning problem as an Integer Linear Program (ILP) and propose two heuristics to solve it. Simulation results show that our solution minimizes flow setup time while incurring very low communication overhead.

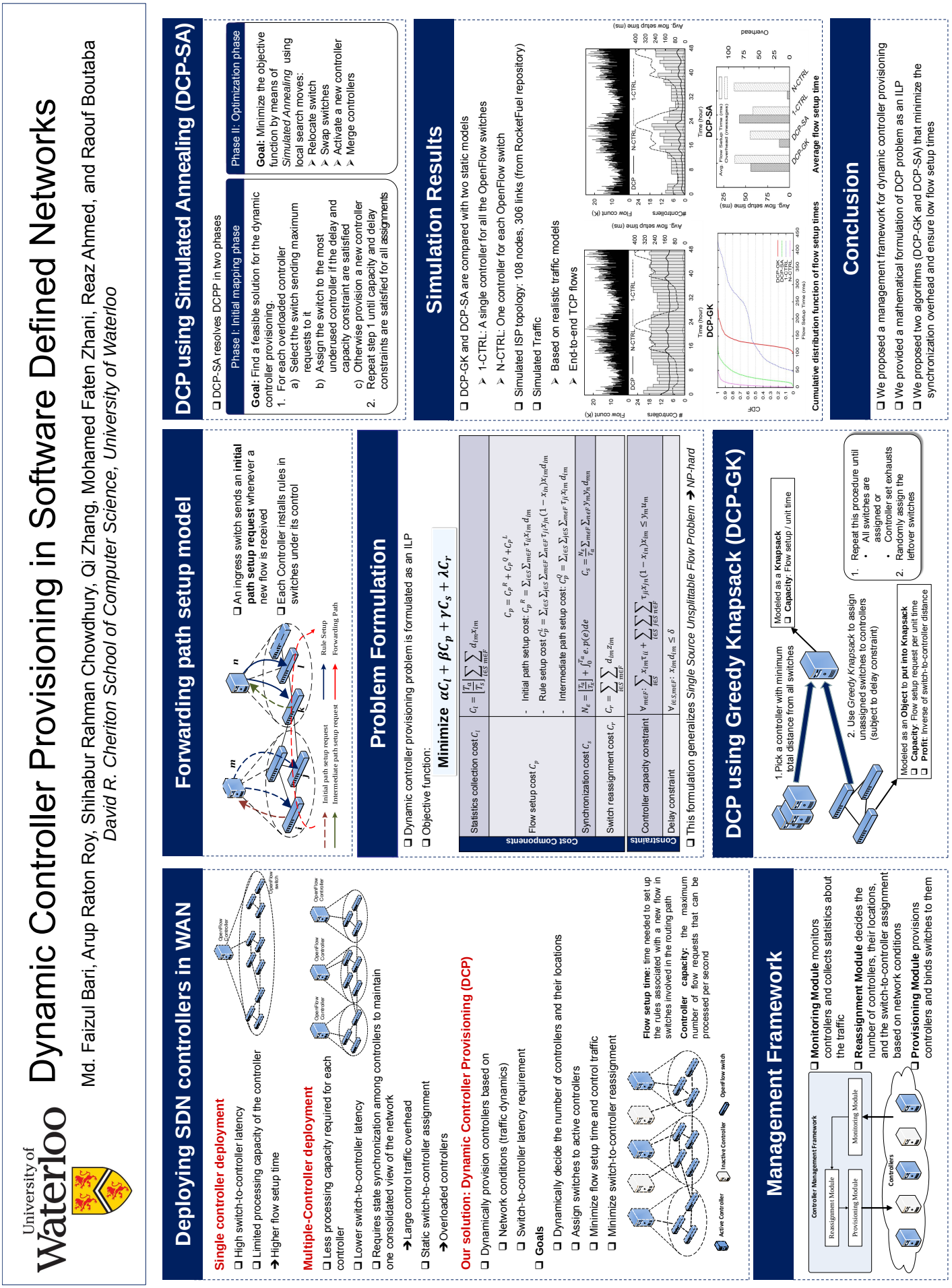


FIGURE 3.1: *Dynamic Controller Provisioning in Software Defined Networks from Raouf Boutaba at University of Waterloo (Theme 3)*

16 Greenhead: Virtual Data Center Embedding Across Distributed Infrastructures
Transactions on Cloud Computing (Under revision)
Ahmed Amokrane, Mohamed Faten Zhani, Rami Langar, **Raouf Boutaba** and Guy Pujolle, U. Waterloo

Cloud computing promises to provide on-demand computing, storage and networking resources. However, most cloud providers simply offer virtual machines (VMs) without bandwidth and delay guarantees, which may hurt the performance of the deployed services. Recently, a number of proposals have suggested remediating such limitation by offering Virtual Data Centers (VDCs) instead of providing only VMs. However, they have only considered the case where the VDC components are embedded within a single data center. In practice, infrastructure providers should have the ability to provision requested VDCs across their distributed infrastructure to leverage the abundance of resources they possess and to achieve multiple goals including revenue maximization, operational costs reduction, energy efficiency and green IT, or to simply satisfy geographic location constraints of the VDCs.

In this work, we propose Greenhead, a holistic resource management framework for embedding VDCs across geographically distributed data centers connected through a backbone network. The goal of Greenhead is to find the best trade-off between maximizing the revenue, reducing energy costs and respecting environmental considerations as well. To evaluate the effectiveness of our proposal, we conducted extensive simulations for four data centers connected through the NSFNet topology. Results show that the proposed framework improves requests' acceptance ratio and revenue by up to 40% while ensuring high usage of renewable energy and minimal carbon footprint.

Greenhead: Virtual Data Center Embedding Across Distributed Infrastructures

University of Waterloo

Ahmed Amokrane[1], Mohamed Faten Zhani[2], Rami Langar[1], Raouf Boutaba[2] and Guy Pujolle[1]
[1]Pierre and Marie Curie University, Paris VI, [2]University of Waterloo

UPMC SORBONNE UNIVERSITÉS

Introduction

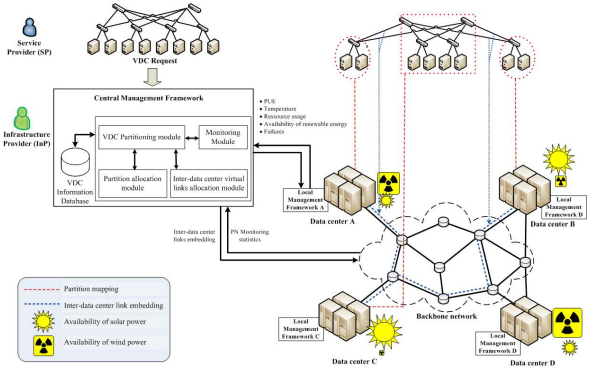
Context

- **Infrastructure Providers (InPs)** build geographically distributed data centers
- InPs sell resources in the form of **Virtual Data Centers (VDCs)**
  - Virtual Machines (VMs) (CPU, memory and disk)
  - Virtual links (bandwidth and propagation delay)
- Sources of energy for data centers:
  - Locally-available renewable energy sources (e.g., solar, wind)
    - Limited and dependent on location and weather conditions
  - Electric grid
    - Fluctuating prices of electricity and high carbon footprint
    - Price and carbon footprint differ from one location to another

Objectives

- Embed VDCs across a distributed infrastructure in order to
  - Maximize infrastructure provider's revenue
  - Minimize energy costs
  - Ensure a green and environment-friendly infrastructure
  - Minimize inter-data center communication traffic

The Greenhead Framework



- **A local controller** manages internal resources in each data center
- **A central controller** manages the entire infrastructure
  - The Partitioning Module: splits a VDC request into partitions
  - The Partition Allocation Module: assigns partitions to data centers
  - Monitoring Module: collects information about resource utilization, electricity prices and availability of renewable energy
  - The Inter-data center Virtual Link Allocation Module: allocates virtual links in the backbone network

Problem Formulation

System model:

- The infrastructure is modeled as a graph $G(V \cup W, E)$, where V is the set of data centers, W and E are the sets of nodes and links of the backbone, respectively.
- A VDC request is modeled as a graph $G'(V', E')$, where V' and E' are the sets of VMs and virtual links, respectively.

ILP decision variables:

$$x_{ik}^{j} = \begin{cases} 1 & \text{If the VM } k \text{ of the VDC } j \text{ is assigned to data center } i \\ 0 & \text{Otherwise.} \end{cases} \qquad f_{e,e'} = \begin{cases} 1 & \text{If the physical link } e \in E \text{ is used to embed the virtual link } e' \in E^{j} \\ 0 & \text{Otherwise.} \end{cases}$$

Objective function: $\boldsymbol{Maximize\ R^{j} - (D^{j} + P^{j})}$

R^{j}: Revenue for VDC request j

P^{j}: Embedding cost for request j in data centers

D^{j}: Embedding cost of the request j in the backbone network

VDC Partitioning and Embedding

- **Step 1 - VDC Partitioning:** Location Aware *Louvain* Algorithm
  - Split the VDC requests into partitions with high intra-partition bandwidth demand and low inter-partition bandwidth demand
  - ⇨ Minimize inter-data center traffic
- **Step 2 – Partition embedding:** Greedy partition assignment to data centers
  - Assign each partition to the data center that
    - Satisfies location and delay constraints
    - Minimizes electricity costs and carbon footprint

Simulation Settings

- Physical infrastructure:
  - 4 data centers located in New York, California, Illinois and Texas
  - Backbone network: NSFNet network

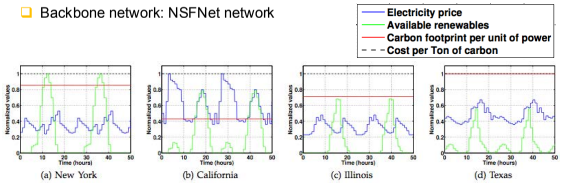
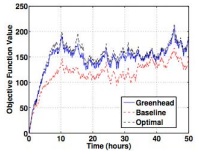


Fig: Electricity price, renewable availability, carbon per unit of power and cost per ton of carbon

- VDC requests:
  - Number of VMs randomly generated between 5 and 10 for small-sized VDCs and between 20 and 100 for large-sized VDCs
  - Virtual links randomly created with a bandwidth demand between 10 and 50 Mbps
  - Poisson arrivals (8 requests per hour) and exponential lifetime (average 6 hours)
  - 15 % of the VMs have location constraint
- Baseline approach: Greenhead without partitioning

Simulation Results

- Greenhead provides near-optimal solution within a reasonable time frame

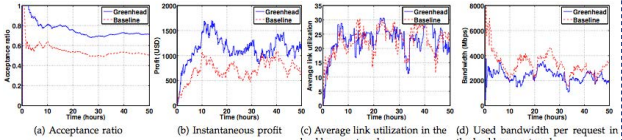


Cumulative objective function

Computation Time (ms)

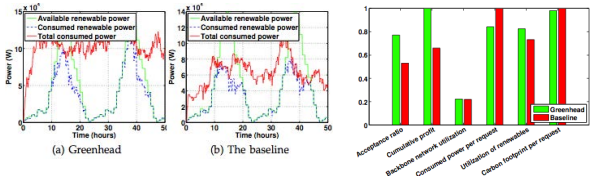
VDC size	Greenhead			Baseline	ILP Solver
	Partitioning	Embedding	Total		
5-10 VMs	0.214	0.061	0.079	0.275	13540
20-100 VMs	31.41	0.28	31.69	2.2	-

- Greenhead achieves higher acceptance ratio, higher revenue and better backbone network utilization



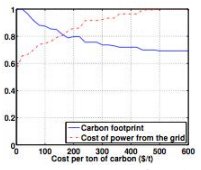
(a) Acceptance ratio (b) Instantaneous profit (c) Average link utilization in the backbone network (d) Used bandwidth per request in the backbone network

- Greenhead maximizes the utilization of available renewable power

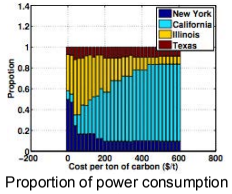


(a) Greenhead (b) The baseline

- Electricity cost reduction comes at the expense of higher carbon footprint



Power cost vs. carbon footprint for variable carbon costs



Proportion of power consumption in different data centers for variable carbon costs

Conclusion

- Greenhead: a holistic resource management framework for embedding VDCs across a geographically distributed infrastructure
  - High acceptance ratio and revenue
  - High utilization of renewables
  - Low inter-data center communication traffic

FIGURE 3.2: *Greenhead: Virtual Data Center Embedding from Raouf Boutaba at University of Waterloo (Theme 3)*

17. HARMONY: Dynamic Heterogeneity-Aware Resource Provisioning in the Cloud

In the 33rd International Conference on Distributed Computing Systems (ICDCS), Philadelphia (USA), July 08-11, 2013

Qi Zhang, Mohamed Faten Zhani, **Raouf Boutaba**, J. L. Hellerstein, U. Waterloo

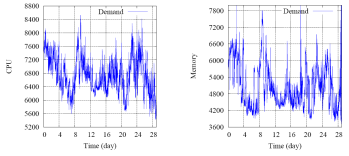
Data centers today consume tremendous amount of energy in terms of power distribution and cooling. Dynamic capacity provisioning is a promising approach for reducing energy consumption by dynamically adjusting the number of active machines to match resource demands. However, despite extensive studies of the problem, existing solutions for dynamic capacity provisioning have not fully considered the heterogeneity of both workload and machine hardware found in production environments. In particular, production data centers often comprise several generations of machines with different capacities, capabilities and energy consumption characteristics. Meanwhile, the workloads running in these data centers typically consist of a wide variety of applications with different priorities, performance objectives and resource requirements. Failure to con- sider heterogeneous characteristics will lead to both sub-optimal energy-savings and long scheduling delays, due to incompatibility between workload requirements and the resources offered by the provisioned machines. To address this limitation, in this work we present HARMONY, a Heterogeneity-Aware Resource Management System for dynamic capacity provisioning in cloud computing environments. Specifically, we first use the K-means clustering algorithm to divide the workload into distinct task classes with similar characteristics in terms of resource and performance requirements. Then we present a novel technique for dynamically adjusting the number of machines of each type to minimize total energy consumption and performance penalty in terms of scheduling delay. Through simulations using real traces from Google's compute clusters, we found that our approach can improve data center energy efficiency by up to 28% compared to heterogeneity-oblivious solutions.

HARMONY: Dynamic Heterogeneity-Aware Resource Provisioning in the Cloud

Qi Zhang, Mohamed Faten Zhani and Raouf Boutaba
David R. Cheriton School of Computer Science, University of Waterloo

Introduction

- Energy cost is an important concern in data centers
  - Accounts for 12% of data center operational cost [Gartner Report 2010]
  - Governments policies for building energy-efficient computing platforms
- An idle server consumes as much as 60% of its peak energy demand
- The fluctuation of resource demand in data centers creates opportunities for dynamically turning on and off servers



CPU Demand over 30 days Memory Demand over 30 days

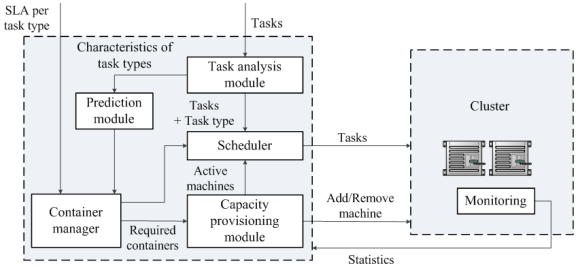
Total resource demand in Google's Cluster Data Set

- However, to dynamically control data center capacity, one must consider the following factors:
  - Heterogeneity of machines
  - Heterogeneity of task size and duration
  - Heterogeneity of task arrival rate
  - Workload performance requirement (e.g., scheduling delay)
  - The cost of turning on and off servers
  - The fluctuating energy prices

Solution

- Classify tasks based on their size and duration using k-means clustering algorithm
- Capture the run-time workload composition in terms of arrival rate for each task class
- Predict the arrival rate of each type of tasks
- Define container as a logical allocation of resources to a task that belongs to a task class
- Use containers to reserve resources for each task class
- Using task arrival rate to estimate the number of required containers of each type of task

System architecture



Problem formulation

Optimal capacity provisioning can be formulated as the following integer program:

$$\max_{a_t^{in}, u_t^i, y_t^i, a_t^{in}} \quad R_T = \sum_{t=1}^{T} U_t^{pref} - E_t - C_t^{sw}$$

where

$$U_t^{perf} = \sum_{n \in N} f^n\Big(\sum_{m \in M} \sum_{i \in N_t^m} a_t^{in}\Big) \qquad \text{(Performance objective)}$$

$$E_t = p_t \sum_{m \in M} \sum_{i \in N_t^m} y_t^i \left(E^{idle,m} + \sum_{r \in R} \alpha^{mr} u_t^{ir} \right) \qquad \text{(Energy cost)}$$

$$C_t^{sw} = \sum_{m \in M} \sum_{i \in N_t^m} q^m |u_t^i| \qquad \text{(Switching cost)}$$

Subject to constraints

$$y_t^i \geq a_t^{in} \quad \forall m \in M, i \in N_t^m, n \in N, t \in \mathcal{T} \qquad \text{(Machine state constraint)}$$

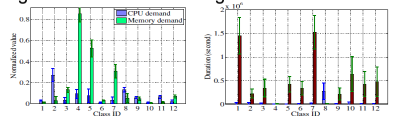
$$\sum_{n \in N} a_t^{ni} c^{nr} \leq C^{mr} \quad \forall m \in M, i \in N_t^m, t \in \mathcal{T} \qquad \text{(Capacity constraint)}$$

Proposed algorithms

- Optimal Capacity Provisioning is NP-hard to solve
- Two approaches
  - Container-Based Scheduling (CBS)
    - Statically allocate containers in physical machines
    - At run-time, schedule tasks in containers
  - Container-Based Provisioning (CBP)
    - Use the estimated number of containers to provision machines
    - At run-time, schedule tasks using existing task scheduling algorithms such as first-fit (FF)

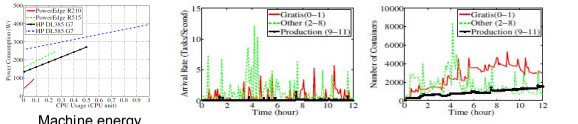
Simulation Results

- Workload traces collected from a production compute cluster in Google over 29 days (~ 12,000 machines, ~2,012,242 jobs, 25,462,157 tasks)
- 12 task priorities divided into 3 priority groups
  - Gratis (0-1): low priority batch jobs (e.g., MapReduce jobs)
  - Other (2-8) : medium priority tasks (e.g., monitoring)
  - Production (9-11) : high priority applications (e.g., user facing)
- Task classification
  - Classify tasks based on size
  - Categorize into short and long tasks

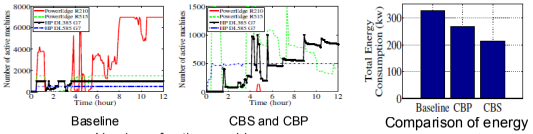


Class size (gratis) Task duration (gratis)

- Experimental results

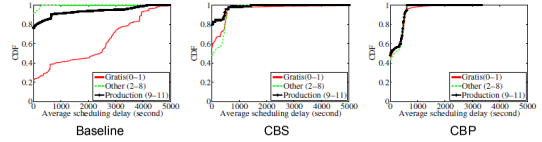


Machine energy consumption model Task arrival rates Number of required containers



Baseline CBS and CBP Comparison of energy consumption

Number of active machines



Baseline CBS CBP

CDF of task scheduling delay

Conclusion

- Dynamic capacity provisioning requires to consider the heterogeneity of both workload and machine hardware found in production environments.
- Harmony can lead to up to 28 % energy savings while significantly improving task scheduling delay

FIGURE 3.3: *Harmony: Dynamic Resource Provisioning in the Cloud from Qi Zhang, Mohamed Faten Zhani and Raouf Boutaba at University of Waterloo (Theme 3)*

18. On achieving high survivability in virtualized data centers
Submitted to IEICE Transactions on Electronics.
Md Golam Rabbani, Mohamed Faten Zhani, **Raouf Boutaba**, U. Waterloo

As businesses are increasingly relying on the cloud to host their services, cloud providers are striving to offer guaranteed and highly-available resources. To achieve this goal, recent proposals have advocated offering both computing and networking resources in the form of Virtual Data Centers (VDCs). Subsequently, several attempts have been made to improve the availability of VDCs through reliability-aware resource allocation schemes and redundancy provisioning techniques. However, the research to date has not considered the heterogeneity of the underlying physical components. Specifically, it does not consider recent findings showing that failure rates and availability of cloud data center equipments can vary significantly depending on various parameters including their types and ages. To address this limitation, in this paper we propose an availability-aware management framework that takes into account the heterogeneity of data center equipments to dynamically provision backup resources in order to ensure the required VDC availability. Hence, we first propose a technique to compute the availability of a VDC that considers both (1) the heterogeneity of the networking and computing elements of the data center in terms of failure rates and availability, and (2) the number of redundant virtual nodes and links provisioned as backups. We then leverage this technique to propose an allocation scheme that jointly provisions resources for VDCs and backups of virtual components with the goal of achieving the required VDC availability while minimizing operational costs (notably energy costs). Through simulations, we demonstrate the effectiveness of our framework compared heterogeneity-oblivious solutions.

University of Waterloo

On achieving high survivability in virtualized data centers

Md Golam Rabbani, Mohamed Faten Zhani, and Raouf Boutaba
David R. Cheriton School of Computer Science, University of Waterloo

Introduction

Context

- Cloud Providers (CPs) lease their resources to Service Providers (SPs)
- SPs request Virtual Data Centers (VDCs)
- A VDC request specifies
  - Computing, networking and storage resources
  - Required availability for the resources

Motivations

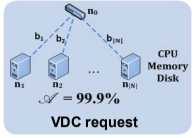
- Minimize service disruption
  - Reduce SP's revenue loss
  - Preserve SP's reputation
- Avoid SLA violation penalty
  - Amazon EC2 pays 10% of the bill if availability falls below 99.95%

Goals

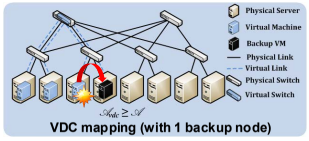
- Ensure requested availability and resource requirements
- Minimize energy costs
- Maximize VDC acceptance ratio
- Increase CP's revenue

Challenges

- Embed VDCs considering equipment availability
- Estimate number and placement of backups to ensure required availability
- Compute VDC availability considering equipment heterogeneity



VDC request



VDC mapping (with 1 backup node)

Computation of VDC Availability

- Assumption: B backups are provisioned

⇨The VDC is available if at most B Virtual Machines (VMs) are unavailable

- Goal: Compute the availability of a VDC

$$\mathscr{A}_{vdc} = \sum_{k=0}^{B} Pr(k) = Pr(0) + \sum_{k=1}^{B} Pr(k)$$

$$Pr(0) = \prod_{\bar{n}:y_{\bar{n}}=1} y_{\bar{n}}\mathscr{A}_{\bar{n}}$$

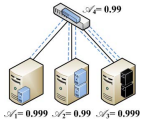
$$Pr(k) \geq \sum_{\bar{n}:g_{\bar{n}}=k} ((1-\mathscr{A}_{\bar{n}}) \prod_{\bar{t}\in\bar{N}\setminus\{\bar{n}\}:y_{\bar{t}}=1} y_{\bar{t}}\mathscr{A}_{\bar{t}})$$

$Pr(k)$: Probability that k VMs fail
$\mathscr{A}_{\bar{n}}$: Availability of physical node $\bar{n} \in \bar{N}$
$g_{\bar{n}}$: Number of VMs disconnected if $\bar{n}$ fails
$y_{\bar{n}}$: Indicates whether $\bar{n}$ is used either to embed a VM, or to embed a virtual link (as an intermediate node in the physical path).

A lower bound on the availability can be given by:

$$\mathscr{A}_{vdc} \geq \prod_{\bar{n}:y_{\bar{n}}=1} y_{\bar{n}}\mathscr{A}_{\bar{n}} + \sum_{k=1}^{K} \left(\sum_{\bar{n}:g_{\bar{n}}=k} ((1-\mathscr{A}_{\bar{n}}) \prod_{\bar{t}\in\bar{N}\setminus\{\bar{n}\}:y_{\bar{t}}=1} y_{\bar{t}}\mathscr{A}_{\bar{t}}) \right)$$

- Example



$$Pr(0) = \mathscr{A}_1\mathscr{A}_2\mathscr{A}_3\mathscr{A}_4 = 0.978141$$
$$Pr(1) = (1-\mathscr{A}_2)\mathscr{A}_1\mathscr{A}_3\mathscr{A}_4 = 0.000979$$
$$Pr(2) = (1-\mathscr{A}_3)\mathscr{A}_1\mathscr{A}_2\mathscr{A}_4 + (1-\mathscr{A}_3)\mathscr{A}_1\mathscr{A}_2\mathscr{A}_4 = 0.010859$$
$$\mathscr{A}_{vdc} \geq 0.989979$$

Problem Formulation

- Minimize operational, communication and backup costs

$$Min\ (\alpha \sum_{\bar{n}\in\bar{N}} y_{\bar{n}}p_{\bar{n}} + \beta \sum_{l\in L}\sum_{\bar{l}\in\bar{L}} f_{\bar{l}l}b_l + \gamma \sum_{\bar{n}\in\bar{N}}\sum_{n\in N} x_{n\bar{n}}c_n)$$

- Subject to capacity and availability constraints $B \leq B_{max}$ and $\mathscr{A}_{vdc} \geq \mathscr{A}$

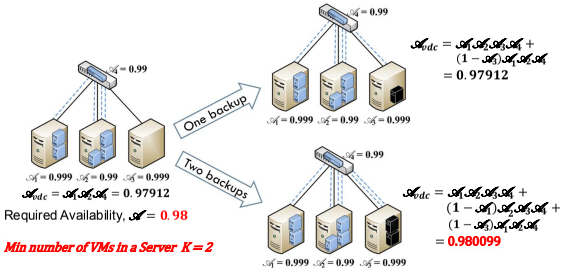
$f_{\bar{l}l}$: indicates whether $\bar{l}$ is used to embed virtual link l
$x_{n\bar{n}}$: indicates whether $\bar{n}$ is used to embed virtual machine n
c_n: size of virtual machine n
B_{max}: maximum number of backups set by the CP
b_l: bandwidth requirement of virtual link l
$p_{\bar{n}}$: cost of turning on machine $\bar{n}$

Solution

- Availability-aware VDC embedding problem is NP-hard
- Our Solution: Hi-VI (High availability Virtual Infrastructure Embedding)
  - Heuristic consisting of two phases
    - VM Mapping
    - Switch and Link Mapping

Phase 1: VM Mapping

- Physical servers are sorted by their statuses (active or inactive) and availability
- VMs are sorted by size and then embedded sequentially
- Once all VMs are embedded, if VDC availability is met, go to Phase 2, else
  - Repeat : Add K backups where K is the minimum number of VMs embedded in a single physical server
  - Until B_{max} is reached or the required availability is met



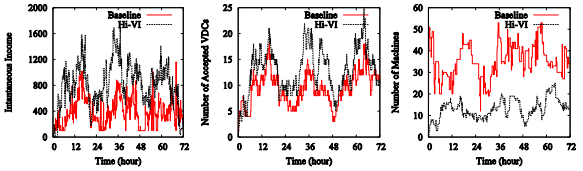
Impact of the number of backups and their placement on the VDC availability

Phase 2: Virtual Switch and Link Mapping

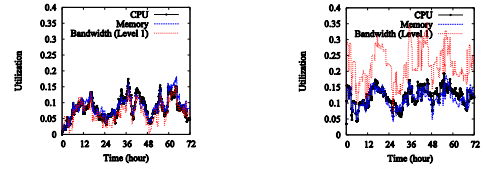
- Compute communication cost for each possible placement for the switch
  - Each of the servers hosting the VMs
  - Physical switches on the shortest paths between those servers
- The switch is placed where both
  - The communication cost is minimized
  - The VDC availability is satisfied

Simulation Results

- Physical data center of 120 nodes organized into 4 racks (VL2 topology)
- Physical equipment availabilities are randomly selected from set {0.99, 0.999, 0.9999, 0.99999}
- VDC requests arrive following Poisson distribution
- VDC availability is generated randomly from set {0.99, 0.999, 0.9999}
- Baseline: spread VMs across the data center then add backups
- Comparison of instantaneous income and number of used machines:

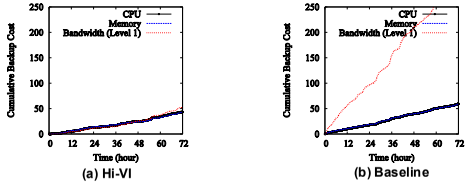


- Data center utilization:



(a) Hi-VI (b) Baseline

- Cumulative backup cost:



(a) Hi-VI (b) Baseline

Conclusions

- We proposed a technique to compute VDC availability considering
  - Heterogeneity of data center equipment in terms of availability
  - Number of redundant virtual nodes and links
- We presented a mathematical formulation of availability-aware VDC embedding considering backup provisioning
- We proposed Hi-VI, an availability aware VDC allocation scheme that
  - Ensures VDC availability through backup provisioning
  - Minimizes communication costs
  - Minimizes energy cost (number of active machines)

FIGURE 3.4: *On Achieving High Survivability in Virtual Data Center from Md Golam Rabbani, Mohamed Faten Zhani and Raouf Boutaba at University of Waterloo (Theme 3)*

19. VDC Planner: Dynamic Migration-Aware Virtual Data Center Embedding for Clouds

In Proceedings IFIP/IEEE Integrated Network Management Symposium (IM 2013), Ghent, Belgium, May 2013.

Mohamed Faten Zhani, Qi Zhang, Gwendal Simon, and **Raouf Boutaba**, U. Waterloo

Cloud computing promises to provide computing resources to a large number of service applications in an on- demand manner. Traditionally, cloud providers such as Amazon only provide guaranteed allocation for compute and storage resources, and fail to support bandwidth requirements and performance isolation among these applications. To address this limitation, recently, a number of proposals advocate providing both guaranteed server and network resources in the form of Virtual Data Centers (VDCs). This raises the problem of optimally allocating both servers and data center networks to multiple VDCs in order to maximize the total revenue, while minimizing the total energy consumption in the data center. However, despite recent studies on this problem, none of the existing solutions have considered the possibility of using VM migration to dynamically adjust the resource allocation, in order to meet the fluctuating resource demand of VDCs. In this demo, we present VDC Planner, a migration-aware dynamic virtual data center embedding framework that aims at achieving high revenue while minimizing the total energy cost over-time. Our framework supports various usage scenarios, including VDC embedding, VDC scaling as well as dynamic VDC consolidation.

In this poster, we show how cloud users can define their VDC specifications and monitor their resources through a Web interface. We implemented VDC Planner as a plugin module to OpenStack to efficiently allocate computing and networking resources for VDCs while achieving the intended objectives including revenue maximization, dynamic server consolidation and energy costs minimization.

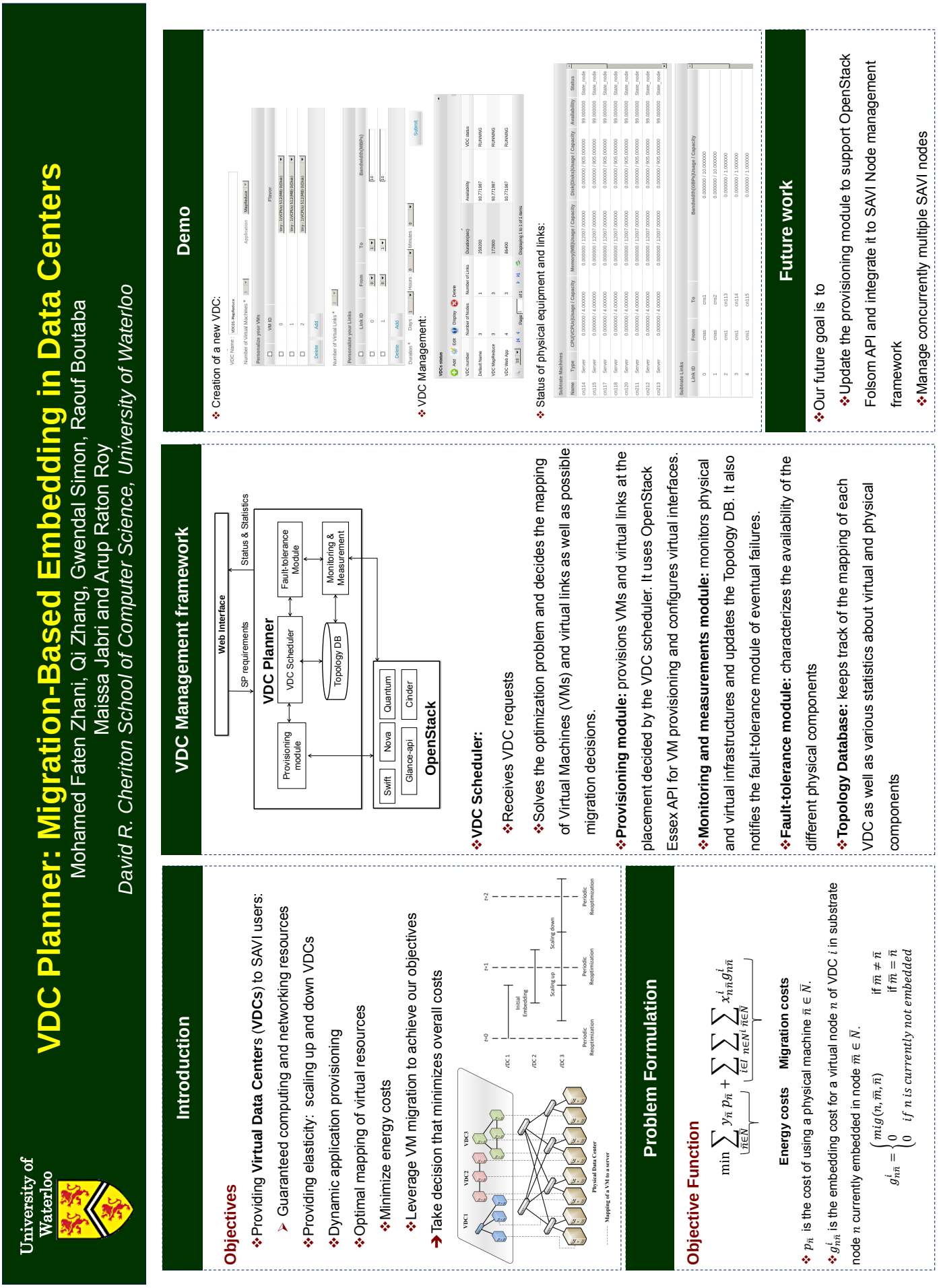


FIGURE 3.5: *VDCPlanner: Migration-based Embedding in Data Centers from Mohamed Faten Zhani, Qi Zhang, Md Golam Rabbani, Raouf Boutaba, Raissa Jabri and Arup Raton Roy at University of Waterloo (Theme 3)*

20. Venice: Reliable Virtual Data Center Embedding in Clouds
Submitted to IEEE/ACM/IFIP Conference on Network and Service Management (CNSM 2013)
M. F. Zhani, Q. Zhang, M. Jabri, **R. Boutaba**, U. Waterloo

Cloud computing has become a cost-effective model for deploying online services in recent years. To improve the Quality of Service (QoS) of the provisioned services, recently, a number of proposals have advocated to provision both guar- anteed server and network resources in the form of Virtual Data Centers (VDC-s). However, simply guaranteeing the quantity of provisioned resources is not sufficient for achieving high service quality, as the service quality is also dependent on (1) the failure characteristics of physical components on which the service is hosted, (2) the impact of individual failure on service availability, given the complex dependencies among the virtual components. However, despite efforts towards designing efficient VDC allocation algorithms, existing work has neither considered the heterogeneous failure characteristics of physical components in data centers, nor the dependencies among VMs in different failure scenarios. As a result, existing VDC embedding algorithms are often insufficient for achieving high VDC availability in practical settings.

To address this limitation, in this work we first develop a technique for computing VDC availability that takes into consideration heterogeneous hardware failure rates as well as dependencies among virtual components. We then propose Venice, an availability-aware virtual data center embedding framework for achieving VDC availability while minimizing total operational costs over time. Through realistic simulations, we show Venice can significantly improve the availability of VDCs while incurring the same energy costs as availability-oblivious solutions.

University of Waterloo

Venice: Reliable Virtual Data Center Embedding in Clouds

Mohamed Faten Zhani, Qi Zhang, Maissa Jabri and Raouf Boutaba
David R. Cheriton School of Computer Science, University of Waterloo

Introduction

- **Availability** is a major concern of **Service Providers** (**SPs**)
  - Many Internet service applications require high availability
  - A service outage can potentially incur high penalty in terms of revenue and customer satisfaction
- SPs lease resources from **Infrastructure Providers** (**InPs**) and specify their requirements in terms of
  - Computing and networking resources in the form of a **Virtual Data Center** (**VDC**) (i.e., virtual machines and virtual links with guaranteed bandwidth)
  - Resource availability (e.g., 99.99%)

Goal: Allocate resources for VDCs and satisfy their availability requirements while minimizing energy costs.

We need to

- Estimate VDC availability in data center environments
- Consider dependencies among VMs in different failure scenarios
- Consider heterogeneous failure characteristics of the data center components

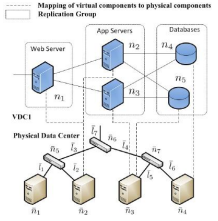
Computing VDC Availability

- The availability of a physical component j is given by

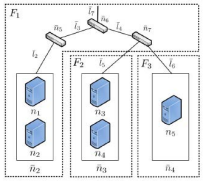
$$A_j = \frac{MTBF_j}{MTBF_j + MTTR_j}$$

$MTBF_j$: Mean Time Between Failures for component j
$MTTR_j$: Mean Time To Repair for component j

- Example: a 3-tier Application



Embedding of a 3-tier Application (VDC_1)



Analyzing the availability of VDC_1
3 failure scenarios : F_1, F_2,and F_3

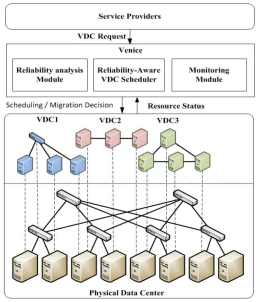
Using conditional probability, the availability of VDC_1 can be computed as:

$$A_{VDC_1} = \sum_{i=1}^{3} P(F_i) A_{F_i}$$

$P(F_i)$: probability of failure scenario F_i
A_{F_i}: availability of VDC_1 under failure scenario F_i

i	$P(F_i)$	A_{F_i}
1	$1 - \prod_{i \in F_1} A_i$	0
2	$(\prod_{i \in F_1} A_i)(1 - \prod_{i \in F_2} A_i)$	$\prod_{i \in F_3} A_i$
3	$\prod_{i \in F_1 \cup F_2} A_i$	1

Venice Architecture



- **Monitoring module** monitors and detects failures in the physical infrastructure.
- **Reliability analysis module** characterizes the availability of the data center components.
- **VDC Scheduler** embeds VDCs in the data center.

Problem Formulation

Objective Function: $\min C_E + C_M + C_A$

- **Energy cost**

$$C_E = \sum_{\bar{n} \in \bar{N}} y_{\bar{n}}\, p_{\bar{n}}$$

- **Migration cost**

$$C_M = \sum_{i \in I} \sum_{n \in N^i} \sum_{\bar{n} \in \bar{N}} x^i_{n\bar{n}} g^i_{n\bar{n}}$$

- **Unavailability cost** $C_A = C^u + C^r$
  - Unavailability penalty $C^u = \sum_{i \in I} (1 - A_i)\pi_i$
  - Restoration cost $C^r = \sum_{\bar{n} \in \bar{N}} F_{\bar{n}} C^{restore}_{\bar{n}} + \sum_{\bar{n} \in \bar{L}} F_{\bar{l}} C^{restore}_{\bar{l}}$

- $p_{\bar{n}}$ cost of using physical node $\bar{n} \in \bar{N}$
- $g^i_{n\bar{n}}$ embedding/migration cost of virtual node n of VDC i in/to substrate node $\bar{n}$
- π_i unit SLA penalty due to resource unavailability
- $C^{restore}_{\bar{n}}$ cost of restoring node $\bar{n}$
- $F_{\bar{n}}$ failure rate of node $\bar{n}$

Decision variables

- $x^i_{n\bar{n}}$ is 1 if virtual node n of VDC i is embedded in $\bar{n} \in \bar{N}$
- $f^i_{l\bar{l}}$ is 1 if a physical link $\bar{l}$ is used to embed virtual link l of VDC i.
- $y_{\bar{n}}$ is 1 if the node $\bar{n}$ is active

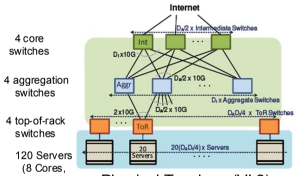
Availability-aware Embedding Algorithm

- For each received VDC
  - **Initial embedding:** embed a VM from each replication group.
  - **Repeat** $EmbedNode\ (n)$
  - **Until** the VDC availability is achieved or all nodes are embedded
  - Embed the remaining components greedily in machines with low availability.
- $EmbedNode\ (n)$: the optimal embedding for a VM n is $\bar{n}$ given by

$$\bar{n} = \arg\min_{n' \in \bar{N}} \{cost(n, n') + (1 - A_i)\pi_i - (1 - A_i(n, n'))\pi_i\}$$

Performance Evaluation

- Simulation setup



- VDC request format
  - 3 replication groups
  - From 1 to 10 VMs per group

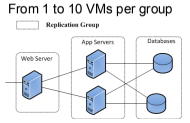
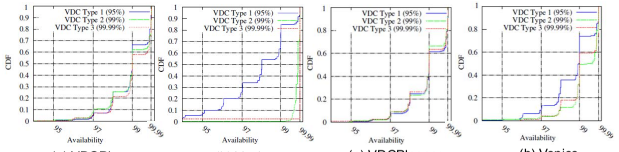


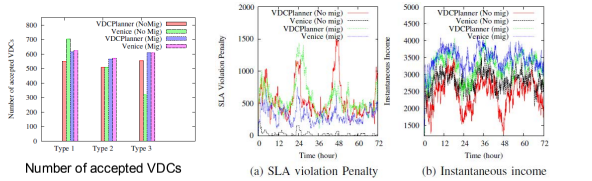
TABLE I: VDC Availability requirements

VDC Type	Minimum Required Availability (%)	Acceptable daily downtime
1	95.00	1h:12mn
2	99.00	14mn:2s
3	99.99	08.64s

- We compared Venice to VDCPlanner (availability-oblivious embedding scheme)
- Simulation results:
  - 35 % more request satisfying the availability requirements
  - 10-30% less SLA violation penalty
  - 15-20 % more net income and acceptance rate



(a) VDCPlanner (b) Venice — Availability without VM migration
(a) VDCPlanner (b) Venice — Availability using VM migration



Number of accepted VDCs (a) SLA violation Penalty (b) Instantaneous income

Conclusion

- We proposed a technique to compute VDC availability that considers heterogeneous failure characteristics of the data center components
- We proposed an availability-aware VDC embedding scheme
- Benefits of the proposed framework Venice:
  - Increases the number of VDCs satisfying availability requirement by up to 35%
  - Increases the net income by up to 15%.

FIGURE 3.6: *Venice: Reliable Virtual Data Center Embedding in Clouds from Mohamed Faten Zhani, Qi Zhang, Maissa Jabri and Raouf Boutaba at University of Waterloo (Theme 3)*

Page for Notes

Theme 4 Integrated Wireless/Optical Access

Research Team

Leslie Rusch (lead), University of Laval

Majid Ghaderi, University of Calgary

Tho LeNgoc, McGill University

Carey Williamson, University of Calgary

NSERC Funded Canadian Strategic Network NET GP397724-10
Smart Applications on Virtual Infrastructure ©2013
August 2013

21. Network-Aware Job Placement in Data Center Environments

Yang Li, **Carey Williamson**, Mea Wang, U. Calgary

This project focuses on strategies for efficient placement and execution of jobs in data center environments. Since network bandwidth is sometimes a scarce resource in multi-tenant data centers, several virtualization abstractions have been proposed to avoid oversubscription of data center networks, and to provide predictable QoS performance for users. In our work, we have built a simulation environment to study job placement and scheduling strategies in data center networks. In particular, we model the Time-Interleaved Virtual Clusters (TIVC) algorithm, and study its sensitivity to different job characteristics. Preliminary results indicate that careful management of a delay queue for pending jobs can help achieve high system utilization. Ongoing work is exploring the impacts of job migration, VM consolidation, and dynamic wireless bandwidth augmentation.

Network-Aware Job Placement in Data Center Environments

Yang Li, Carey Williamson, and Mea Wang

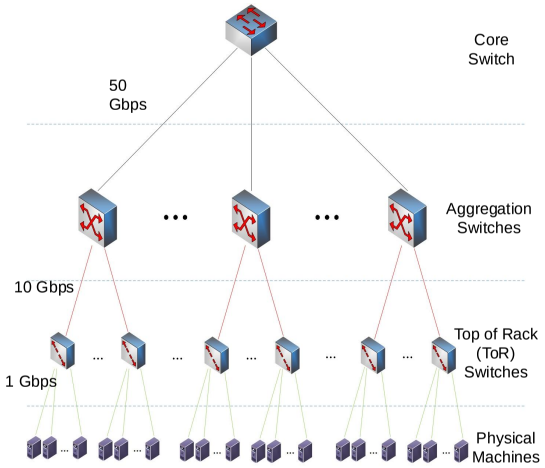
UNIVERSITY OF CALGARY

Problem Statement

How to optimize the placement of jobs for execution in a data center environment?

Challenges:
- Diverse Job Sizes
- QoS Needs
- Real-time Deadlines for Jobs
- System "Fragmentation" Issues

Goals:
- High Utilization
- QoS Guarantees for Jobs
- Fairness
- Revenue (happy customers)



Approach

- Simulation (C++, about 4000 lines of code)
- Time-Interleaved Virtual Clusters (TIVC) Algorithm [1]

Input: Datacenter topology tree T
For level l from 0 to height(T)
  For each subtree v at level l
    If (l==0) $L_v[0] = \{0, ..., \#\text{ avail. VMs}\}$ // leaf machine
    Else $L_v[0] = \{0\}$
    For v's child k from 1 to n
      $L_v[k] = \{0\}$
      For each possible value e in v's kth child's M set (M_{v_k})
        For each possible value h in $L_v[k-1]$
          $L_v[k] = L_v[k] \cup \{e+h\}$
          $D_v[k, e+h] = e$
    $M_v = \emptyset$
    For each value h in $L_v[n]$
      If (bandwidth check of v's uplink per Eq. (1) == true)
        $M_v = M_v \cup \{h\}$
    If $N \in M_v$
      Alloc(r, v, N)
      Return true
Return false
function Alloc(r, v, m)**:**
If v is a machine
  allocate m VMs in v
Else
  For v's child k from n to 1
    Alloc(r, v_k, $D_v[k, m]$)
    record bw reservation on v's kth link
    $m = m - D_v[k, m]$

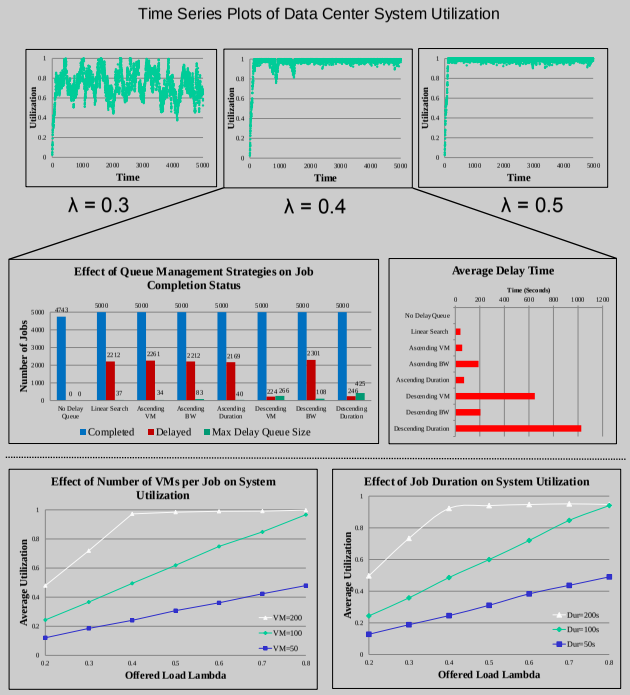
Simulation Factors

Input Workload [2]**:**
- Arrival Rate of Jobs (λ)
- Number of VMs per Job
- Bandwidth Requirement per Job
- Duration of Job (seconds)

Network Topology:
- Number of Levels in Hierarchy
- Branching Factor at each Level
- VMs per Physical Machine
- Bandwidth Capacity for Network Links

Queue Management Strategies

Simulation Results



References

[1] D. Xie, N. Ding, C. Hu, and R. Kompella, "The Only Constant is Change: Incorporating Time-Varying Network Reservations in Data Centers", *Proceedings of ACM SIGCOMM*, pp.199-210, Helsinki, Finland, August 2012.

[2] H. Ballani, P. Costa, T. Karagiannis, and A. Rowstron, "Towards Predictable Datacenter Networks", *Proceedings of ACM SIGCOMM*, pp.242-253, Toronto, Canada, August 2011.

FIGURE 4.1: *Network-Aware Job Placement in Data Center Environments from Yang Li, Carey Williamson and Mea Wang at University of Calgary (Theme 4)*

22. 802.11 Wireless Access Point Virtualization Testbed

Heming Wen, Kevin Han, Michael Smith, **Tho Le-Ngoc**, McGill U.

This poster showcases some of the features integrated in the wireless access point firmware for the SAVI testbed based on the OpenWrt image and the Stanford OpenFlow Wireless image (Debian-based). One of the main features currently supported is OpenFlow. Notably, a mobile handover scenario is demonstrated by using OpenFlow to detect wireless clients on access points and provision a video stream over a fully OpenFlow-enabled network infrastructure. Open VSwitch is used as the wireless gateway on the wireless access point firmware. A flexible control and data path separation is made by using virtual interfaces (veth) and tunneling interfaces (tap). A discussion of some of the integration of other modern WLAN functionalities (VLAN, multiple SSID, RADIUS, etc.) in the context of the SAVI testbed will be provided.

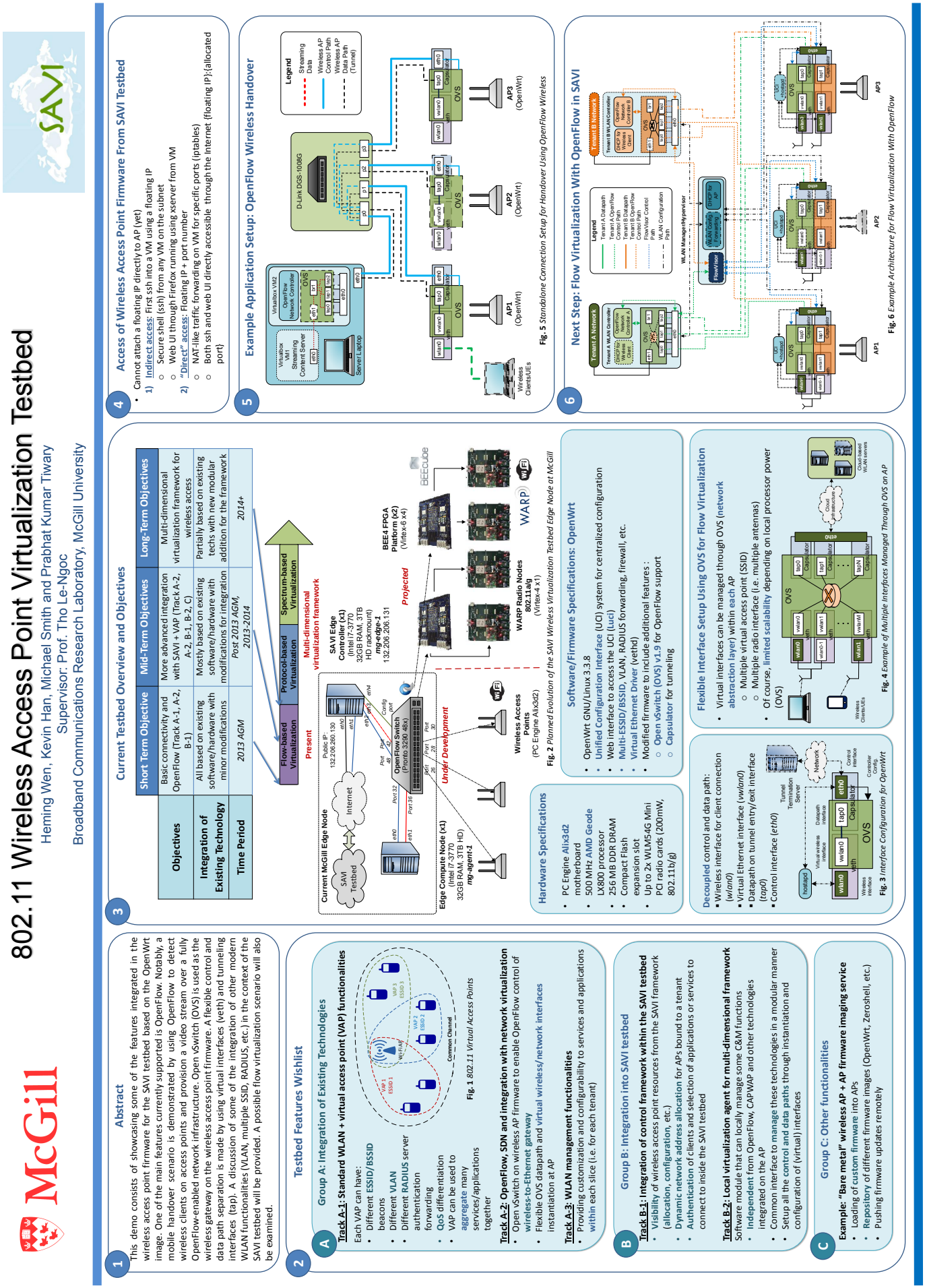


FIGURE 4.2: *802.11 Wireless Access Point Virtualization Testbed from Heming Wen, Kevin Han, Michael Smith, Prabhat Kumar Tiwary and Tho Le-Ngoc at Mcgill University (Theme 4)*

23. Wireless Virtualization Perspectives and Framework

Heming Wen, Prabhat Kumar Tiwary, **Tho Le-Ngoc**, McGill U.

The objective of this poster is to survey and compare recent works in the field of wireless virtualization in order to identify potential applications, common trends and future research directions as well as formulate a sustainable wireless virtualization framework. First, different wireless virtualization architectures as well as both enabling and enabled technologies related to wireless virtualization are explored. Then, a classification of wireless virtualization approaches based on the type of virtualized resources and the depth of slicing is proposed. The three main perspectives identified are the data and flow-based perspective, the protocol-based perspective and the spectrum-based perspective. The challenges and requirements of a sustainable wireless virtualization framework are discussed. Finally, a hypothetical multi-dimensional virtualization framework for wireless virtualization is presented.

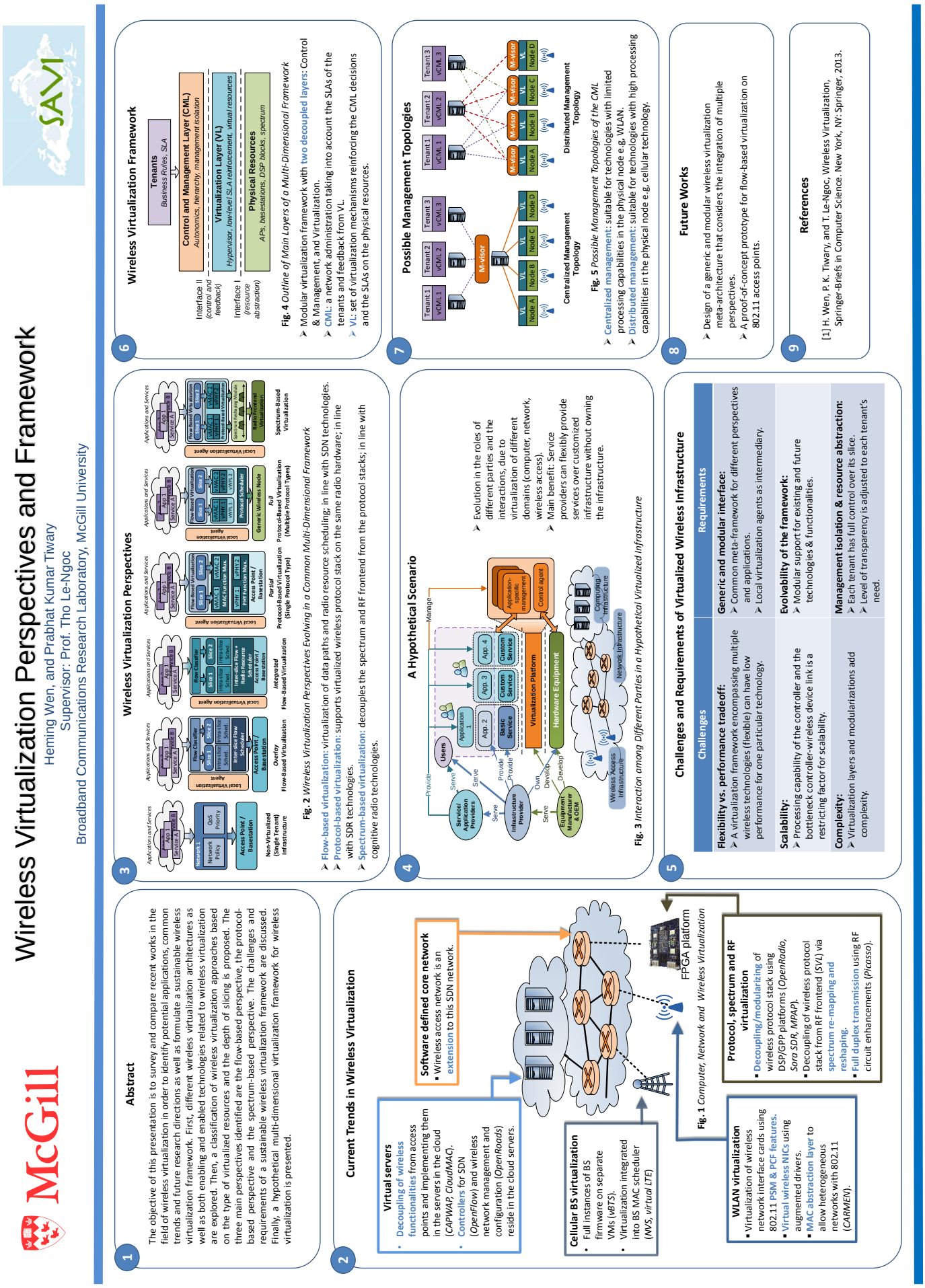


FIGURE 4.3: *Wireless Virtualization Perspectives and Framework from Heming Wen, Prabhat Kumar Tiwary and Tho Le-Ngoc at Mcgill University (Theme 4)*

24. Full-Duplex In-Band Channel OOK-DL/OFDM-UL Transmission up to 80 km of SMF in RSOA-Based Radio-over-Fiber System Using Carrier Reuse and DL Modulation Tuning

Kim Lefebvre, An Nguyen, **Leslie A. Rusch**, U. Laval

We study the working principle of RSOA-based carrier remodulating WDM-PON networks, and apply this architecture to the implementation of a full-duplex digital/analog radio-over-fiber system that is suitable for FTTx applications. Experiment of transmitting full-duplex OOK-DL and WiFi OFDM-UL up to 80 km of SSMF is carried-out to demonstrate the feasibility of the proposed scheme.

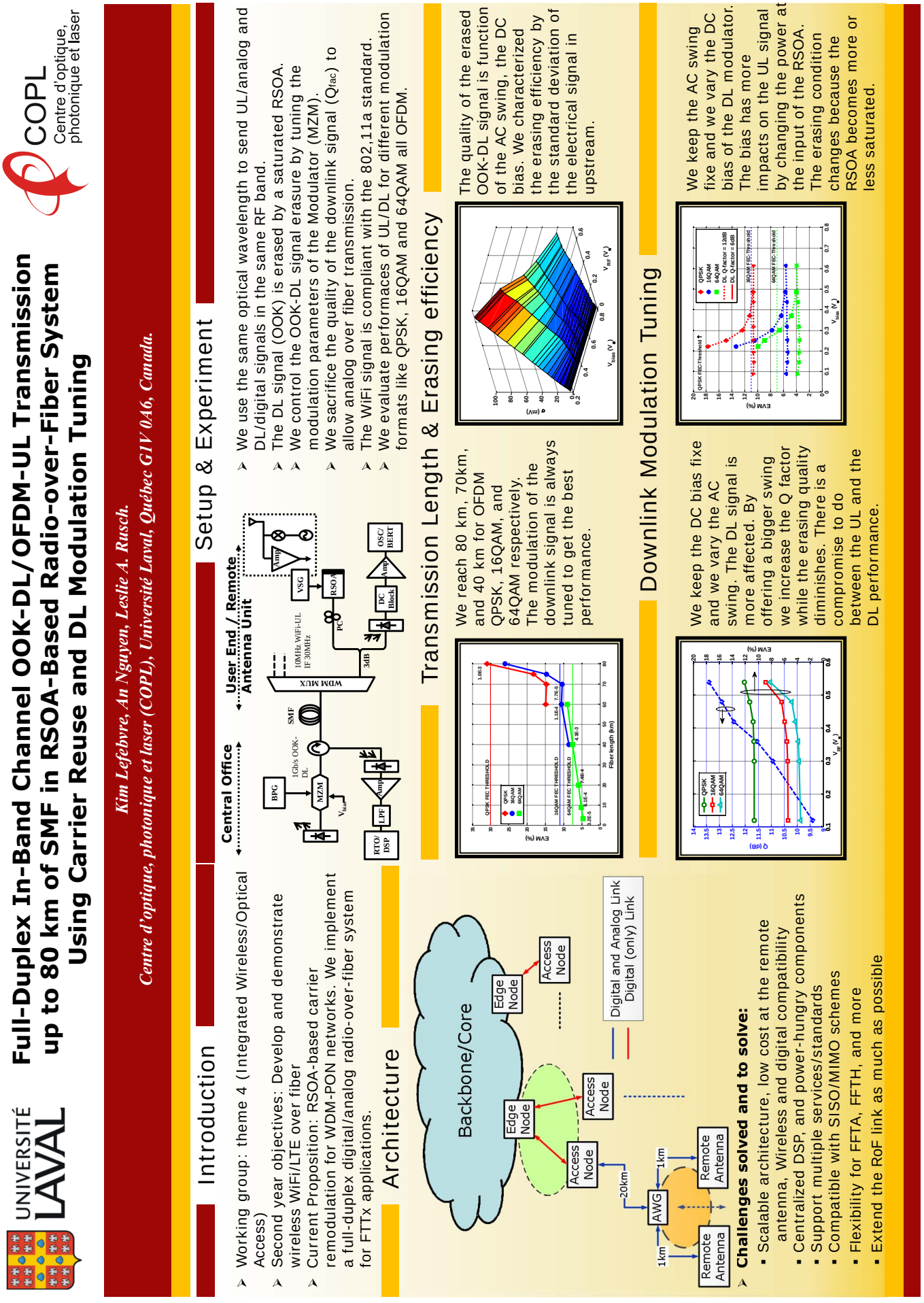


FIGURE 4.4: *Full-Duplex In-Band Channel OOK-DL/OFDM-UL Transmission up to 80 km of SMF in RSOA-Based Radio-over-Fiber System Using Carrier Reuse and DL Modulation Tuning from Kim Lefebvre, An Nguyen and Leslie A. Rusch at Université of Laval (Theme 4)*

25. Optical integration of multiple radio services using tandem optical SSB modulation and DSP-assisted coherent detection

Chul Soo Park, An Nguyen, **Leslie A. Rusch**, U. Laval

Tandem optical single sideband (OSSB) modulation with DSP-assisted coherent optical detection is presented to integrate multiple radio services. 2- and 5-Gbps BPSK signals are tested as a proof of concept. Tandem OSSB modulation technique carries two BPSK signals on each optical sideband. Coherent optical detection with offline digital signal processing is applied to evaluate the system performance. Q-factor above 6 is measured after 80-km single-mode optical fiber transmission.

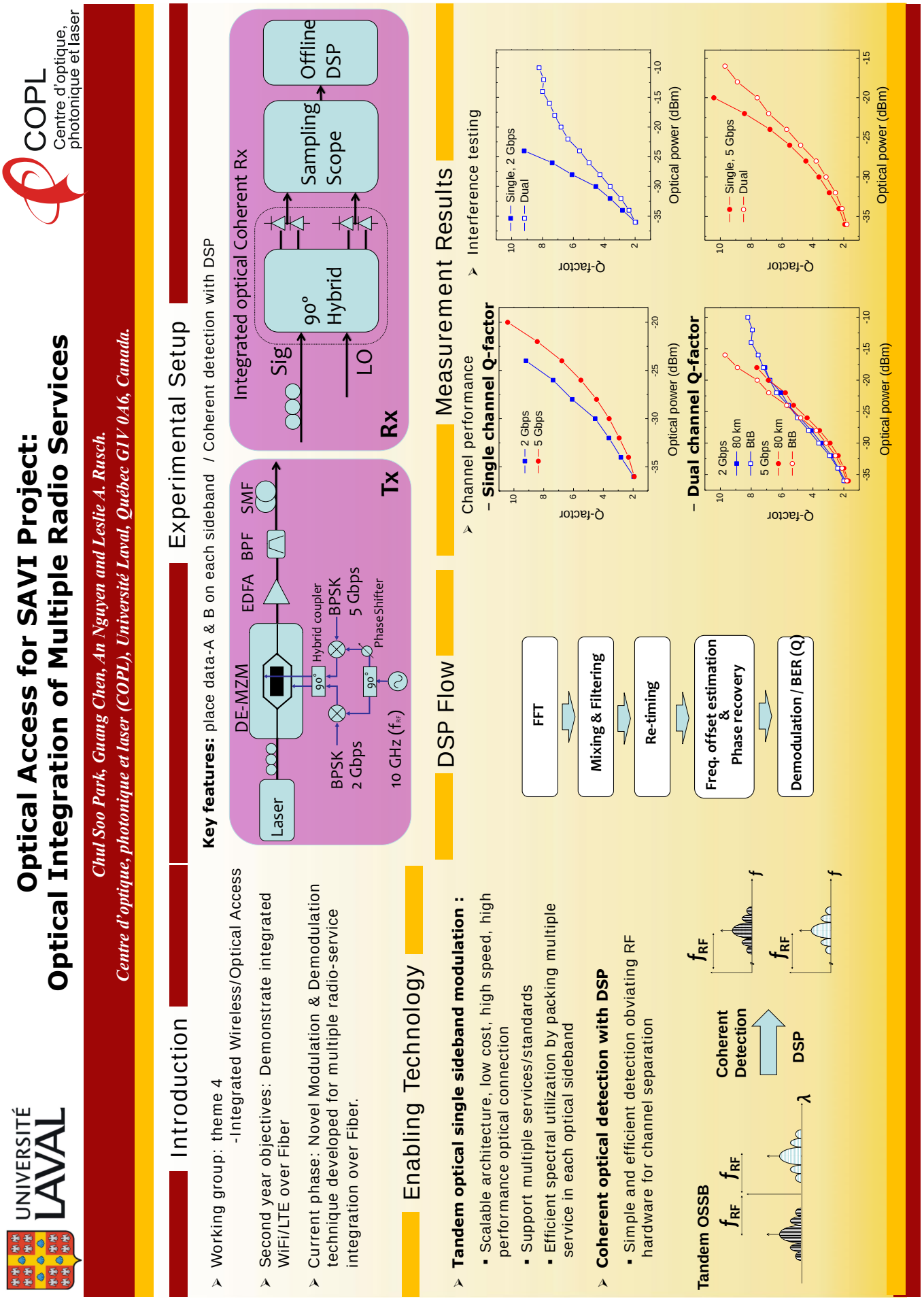


FIGURE 4.5: *Optical Access for SAVI Project: Optical Integration of Radio Services from Chul Soo Park, Guang Chen, An Nguyen and Leslie A. Rusch at Université of Laval (Theme 4)*

26. Distributed Base Station Activation for Energy-Efficiency

Ali Abbasi, **Majid Ghaderi**, U. Calgary

Our work is focused on distributed algorithm design for dynamic base station activation in dense small cell networks. Distributed solutions are particularly desirable in small cell networks due to importance of self-organization and self-optimization in such networks. The goal is to achieve an optimal trade-off between network operator's revenue and operational cost while guaranteeing coverage for network users. In our work, this problem is posed as a network utility maximization aiming to find the optimal activation schedule of each base station. Using the Lagrangian duality technique and controlled message passing among base stations, we design a distributed algorithm for the problem. We have simulated the proposed algorithm to demonstrate its behavior in terms of utility and cost trade-offs in various network scenarios.

Distributed Base-Station Activation for Energy-Efficient Operation of Cellular Networks

Ali Abbasi and Majid Ghaderi

UNIVERSITY OF CALGARY

Motivation

ICT [2]

- produces 2% of total CO2 emission.
- consumes 3% of total electrical energy.

ICT energy consumption increases 15-20% each year. 0.1 of ICT energy is consumed in cellular networks.
Cellular data traffic has massively increased in recent years.
60-80% energy in cellular networks is consumed in base stations.

Energy-efficient base station operation

Features

Current base stations are not energy-proportional.

- Cellular traffic shows periodic behavior.
- Current BSs consume 90% of their peak energy even in idle mode.
- Power control does not lead to high energy saving.

To satisfy increasing traffic demand, cellular base stations are deployed densely.

- Results in high coverage overlap.

Problem Challenges

Energy-proportional operation can be emulated in a coarser granularity by

- Load-aware BS activation

Idea: Turn off/on base stations dynamically based on current load.
How to select active base stations ?

- Traffic characteristics should be considered.
- A combinatorial problem with complex coverage constraints

Solution approach

Distributed self-organization and self-optimization are emphasized in the design of future cellular networks.
We focus on distributed solutions:

- An activation probability is associated to each BS.
- The problem is relaxed to a convex one.
- Duality is employed to compute optimal activation probabilities.
- Each BS is activated independently based on its activation probability.

User Association

Set of base stations $\mathcal{B}$.
Set of locations $\mathcal{L}$.

- Each location l_j has demand d_j which could be served by multiple covering BSs ($\mathcal{B}_j$).
- Each BS b_i incurs c_i cost of activation.

Received rate at l_j if all resources (channels) of b_i is allocated to it is r_{ij}.
Allocated resources to l_j is x_{ij}.
Total received rate at l_j is $\sum_{b_i \in \mathcal{B}_j} x_{ij} r_{ij}$.

Problem Formulation

Find the optimal trade-off between the system utility and energy cost.
Utility function u_j is associated to location l_j which is assumed to be increasing and concave.
Associating activation probability α_i to BS b_i, the problem is represented as

$$\max_{\boldsymbol{x}} \sum_{l_j \in \mathcal{L}} u_j(r_j - d_j) - \sum_{b_i \in \mathcal{A}} \alpha_i c_i$$

where we have

- $\alpha_i = \sum_j x_{ij}$ and $r_j = \sum_{b_i \in \mathcal{B}_j} x_{ij} r_{ij}$

The problem also has the following constraints

- Coverage constraint: $r_j \geq d_j$
- Resource constraint: $\sum_{l_j \in \mathcal{L}_i} x_{ij} \leq 1$

Solution procedure

Making the objective function strictly concave, the following expression

$$-\sum_{l_j \in \mathcal{L}} \sum_{b_i \in \mathcal{B}} \frac{e_j}{2}(x_{ij} - y_{ij})^2$$

is added to it [4]. Also, Lagrangian of the problem is obtained and decomposed to local BS problems. Then the dual problem is derived as

$$\min_{\boldsymbol{\lambda} \geq 0, \boldsymbol{\nu}} g(\boldsymbol{\lambda}, \boldsymbol{\nu})$$

where $\boldsymbol{\lambda}$ and $\boldsymbol{\nu}$ are Lagrange multipliers associated with the constraints. Dual problem is solved by the following iterations

$$\lambda_j(t+1) = \Big[\lambda_j(t) - \xi\big(\sum_{b_i \in \mathcal{B}_j} x_{ij} r_{ij} - d_j\big)\Big]^+,$$

$$\nu_i(t+1) = \nu_i(t) - \xi\big(1 - \sum_{l_j \in \mathcal{L}_i} x_{ij}\big)$$

Local problem

For given Lagrange multipliers $\boldsymbol{\lambda}, \boldsymbol{\nu}$, each local BS problem (D_i) is solved based on KKT conditions according which we have

$$\frac{\partial D_i}{\partial x_{kj}} = 0$$

Assuming utility functions as follows

$$u_j\big(\sum_{b_k \in \mathcal{B}_j} x_{kj} r_{kj} - d_j\big) = w_j \ln\big(\sum_{b_k \in \mathcal{B}_j} x_{kj} r_{kj} - d_j\big)$$

we have

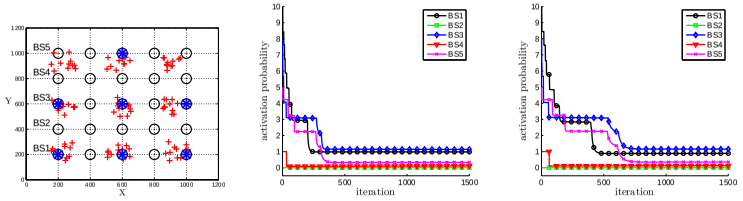
$$\frac{w_j \sum_{b_k \in \mathcal{B}_j} r_{kj}^2}{\sum_{b_k \in \mathcal{B}_j} x_{kj} r_{kj} - d_j} - e_j \sum_{b_k \in \mathcal{B}_j} x_{kj} r_{kj} + \sum_{b_k \in \mathcal{B}_j} \big(e_j y_{kj} r_{kj} + \lambda_j r_{kj}^2 - (c_k + \nu_k) r_{kj}\big) = 0$$

Defining $s_j = \sum_{b_k \in \mathcal{B}_j} x_{kj} r_{kj}$, x_{kj}^* is obtained as follows

$$x_{kj}^* = \Big[\frac{1}{e_j}\big(\frac{w_j r_{kj}}{s_j} + e_j y_{kj} + \lambda_j r_{kj} - (c_k + \nu_k)\big)\Big]_{[0,1]}$$

Simulation

- Activated BSs match more crowded regions.
- Convergence speed of the the distributed algorithm depends on the step-size.



network snapshot convergence with $\xi = 0.02$ convergence with $\xi = 0.01$

Future Direction

Current work

- The model is oversimplified when considering inter-cell interference to allow tractability of the formulation.

More accurate modeling of the problem makes it noncovex integer program.

- Global optimization methods: MCMC
- Challenge: design of state changes

References

[1] C. Peng . Traffic-Driven Power Saving in Operational 3G Cellular Networks. In *MobiCom '11*.
[2] E. Oh . Toward Dynamic Energy-Efficient Operation of Cellular Network Infrastructure. In *IEEE communications magazine '11*.
[3] X. Lin and N. Shroff Utility Maximization for Communication Networks with Multipath Routing In *IEEE Transactions on Automatic Control '06*.

FIGURE 4.6: *Distributed Base-station Activiation for Energy-eficient Operation of Cellular Networks fromAli Abbasi and Majid Ghaderi at University of Calgary (Theme 4)*

Page for Notes

Theme 5: SAVI Application Platform Testbed

Research Team

Alberto Leon-Garcia (lead), University of Toronto
Raouf Boutaba, University of Waterloo
Paul Chow, University of Toronto
Yashar Ganjali, University of Toronto
Baochun Li, University of Toronto
Marin Litoiu, York University
Leslie Rusch, University of Laval
Greg Steffan, University of Toronto
Hadi Bannazadeh, SAVI Architect

NSERC Funded Canadian Strategic Network NET GP397724-10
Smart Applications on Virtual Infrastructure ©2013
August 2013

27. Graph-based Network Topology Management for SAVI Testbed

Joon-Myung Kang, Hadi Bannazadeh, **Alberto Leon-Garcia**, U. Toronto

In SAVI Testbed a network topology consists of many converged heterogeneous resources for computing and networking as a form of physical and virtual. However, the network topology is dynamic due to various physical and logical activities such as addition of new racks, VM addition/deletion/migration, and network creation/modification/deletion. In this research we show a graph-based management approach for the dynamic network topology in SAVI Testbed. A graph is one of the best models for representing converged heterogeneous resources and their associations. We use a graph database to enable us to efficiently store resources and query based on graph traversing and indexing.

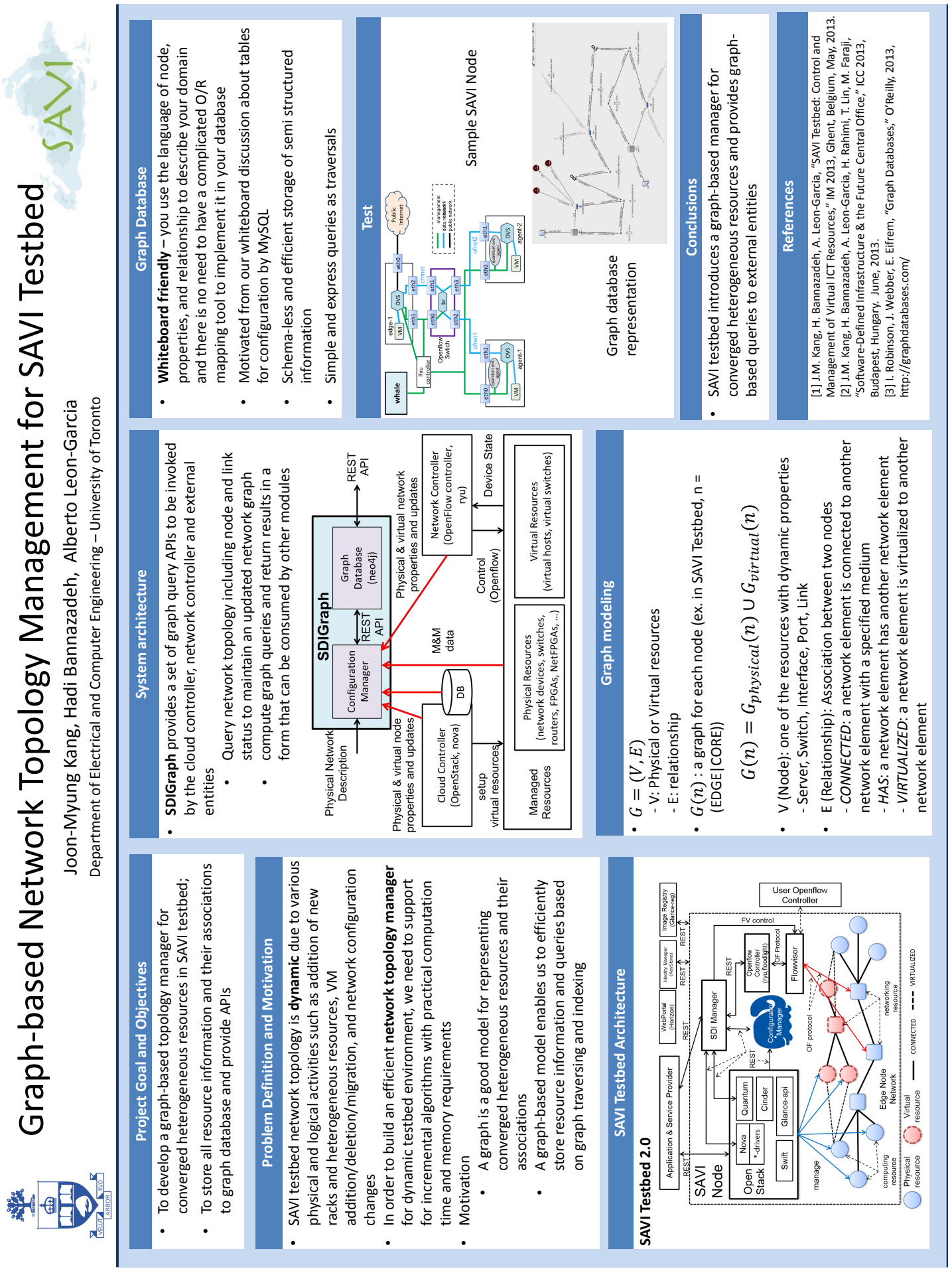


FIGURE 5.1: *Graph-based Network Topology Management for SAVI Testbed from Joo-Myung Kang, Hadi Bannazadeh and Alberto Leon-Garcia at University of Toronto (Theme 5)*

28. SAVI Testbed: Control and Management of Converged Virtual ICT Resources

Joon-Myung Kang, Hadi Bannazadeh, Hesam Rahimi, Thomas Lin, Moe Faraji, **Alberto Leon-Garcia**, U. Toronto

This research introduces the SAVI testbed control and management for converged virtual ICT resources. The SAVI testbed attempts to create an open applications marketplace that operates on converged virtualized computing and networking infrastructures and that supports experimentation with emerging Future Internet protocols and applications. SAVI addresses a major obstacle in experimental network research namely the inability to test and deploy new protocols and applications at scale. In this research we briefly review the Smart Applications on Virtual Infrastructure (SAVI) network and the Virtualized Application Networking Infrastructure (VANI) which are the basis for the SAVI testbed. We then present the system architecture and design of the SAVI testbed. Finally, we report on the current status of development and share our experience with the SAVI testbed.

SAVI Testbed: Control and Management of Converged Virtual ICT Resources

Joon-Myung Kang, Hadi Bannazadeh, and Alberto Leon-Garcia
Department of Electrical and Computer Engineering
University of Toronto, Toronto, ON, Canada
{joonmyung.kang, hadi.bannazadeh, alberto.leongarcia}@utoronto.ca

INTRODUCTION

This paper introduces the *Smart Applications on Virtual Infrastructure (SAVI) testbed (TB)* for control & management of converged virtual ICT resources. The SAVI TB attempts to create an open applications marketplace that operates on converged virtual computing and networking infrastructures and supports experimentation with emerging Future Internet protocols and applications.

SAVI NETWORK AND TESTBED

- Design of future application platforms built on flexible, versatile and evolvable infrastructure that can readily deploy, maintain, and retire the large-scale distributed applications that will be typical in the future applications marketplace. http://www.savinetwork.ca
- SAVI Research by 9 Canadian Universities: Smart applications, Extended Cloud Computing, Smart Converged Edge, Integrated Wireless Optical Access, SAVI Application-Platform TB
- *Virtualized Application Networking Infrastructure (VANI)* TB developed at the University of Toronto, that allows researchers to utilize virtualized resources to rapidly create and deploy networked systems and distributed applications [1].

SAVI TESTBED ARCHITECTURE

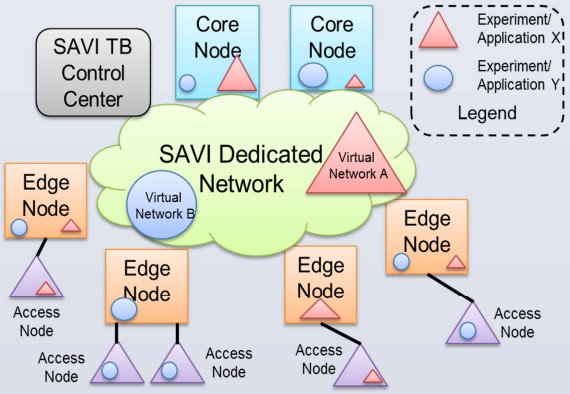


Fig. 1. SAVI TB main entities, hosting two sample applications

- **Core Nodes** provide conventional cloud computing resources (compute, storage, and basic networking)
- **Edge Nodes** provide more advanced and quality-sensitive resources such as reconfigurable hardware resources and are deployed on sites located at participating universities
- **Access Nodes** are deployed on various universities and connected to an edge node

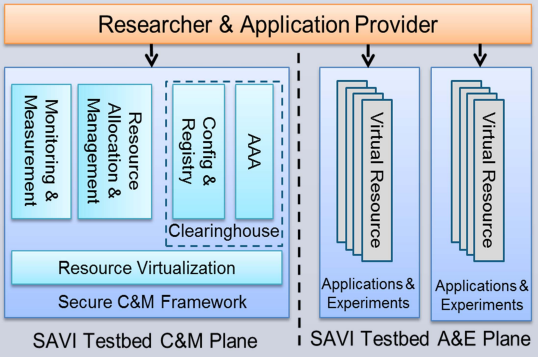


Fig. 2. SAVI TB basic reference architecture

- **C&M Plane**: allocating and securing a slice of virtualized resources to an application: virtualizing resources, allocating resources to an experiment or application, managing allocated resources, monitoring the allocated resources and facilitating measurements, and conducting clearinghouse functionalities
- **A&E Plane**: researchers can run their application on a set of virtualized resources allocated by the C&M plane.

SAVI TESTBED DESIGN

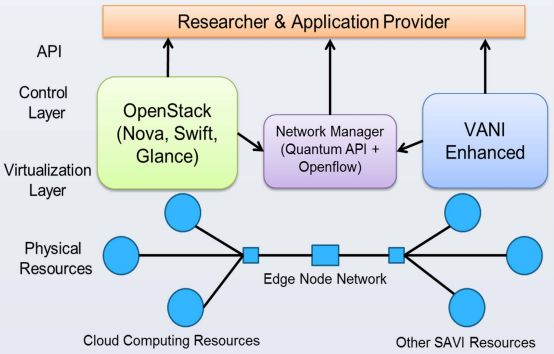


Fig. 3. Conceptual design of SAVI TB for Edge Node

- **Cloud Computing Resources:** conventional compute, storage, and networking resources
- **Edge Node Network:** Software Defined Network (OpenFlow)
- **Other SAVI Resources:** reconfigurable hardware resources (FPGA, NetFPGA), acceleration resources (GPU), and so on

SAVI TESTBED DEPLOYMENT AND TESTING

- SAVI TB Hands-on Workshop for fifty researchers (July 2012)

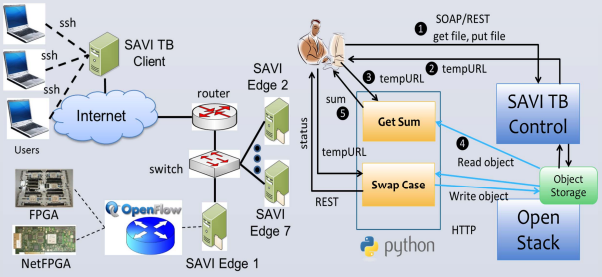


Fig. 4. Deployment of SAVI TB client and SAVI Edge Nodes

Fig. 5. Block diagram for cloud-based applications on SAVI TB

CONCLUSION

This paper has presented a novel approach to controlling and managing a converged computing and networking infrastructure in a multi-tier cloud to support future applications. Currently, we have extended the SAVI TB architecture by focusing on more heterogeneous resources. We have introduced Software-Defined Infrastructure and useful cases for future Central Office such as Content Centric Networking, Smart Transportation, IP Multimedia Subsystem for Next Generation Networks, or other smart applications [2].

REFERENCES

[1] H. Bannazadeh, et al. "Virtualized application networking infrastructure," in TRIDENTCOM, LNCS vol. 46. Springer, 2010, pp. 363–382

[2] J.M. Kang, et al. "Software-Defined Infrastructure and the Future Central Office," IEEE ICC'13 - CNDC 2013, Budapest, Hungary, June 9-13, 2013

FIGURE 5.2: *SAVI Testbed Control and Management of Converged Virtual ICT Resources from Joo-Myung Kang, Hadi Bannazadeh and Alberto Leon-Garcia at University of Toronto (Theme 5)*

29. Software-Defined Networking (SDN) in SAVI Testbed Converged Cluster

Hesam Rahimi, Hadi Bannazadeh, **Alberto Leon-Garcia**, U. Toronto

In this project, we introduce advanced networking features in SAVI Testbed software-defined infrastructure by utilizing SDN approaches. Our goal is to improve SAVI Testbed Converged cluster networking performance by addressing two main issues: 1) Use of proactive openflow rule installations to maximize utilization of networking resources and decreasing rule setup time 2) Control the broadcast messages in the system by handling ARP messages in the SDN controller. In this project, we first show how we can block all broadcast messages and how the controller, knowing the physical and virtual topology of the network, can directly reply to ARP requests, preventing the network from carrying broadcast packets. Furthermore, we present how the controller can proactively install all necessary rules to all OpenFlow switches based on user-specific requirements. We will also discuss major advantages of this approach including higher utilization of resources, more effective fault recovery and better QoS control.

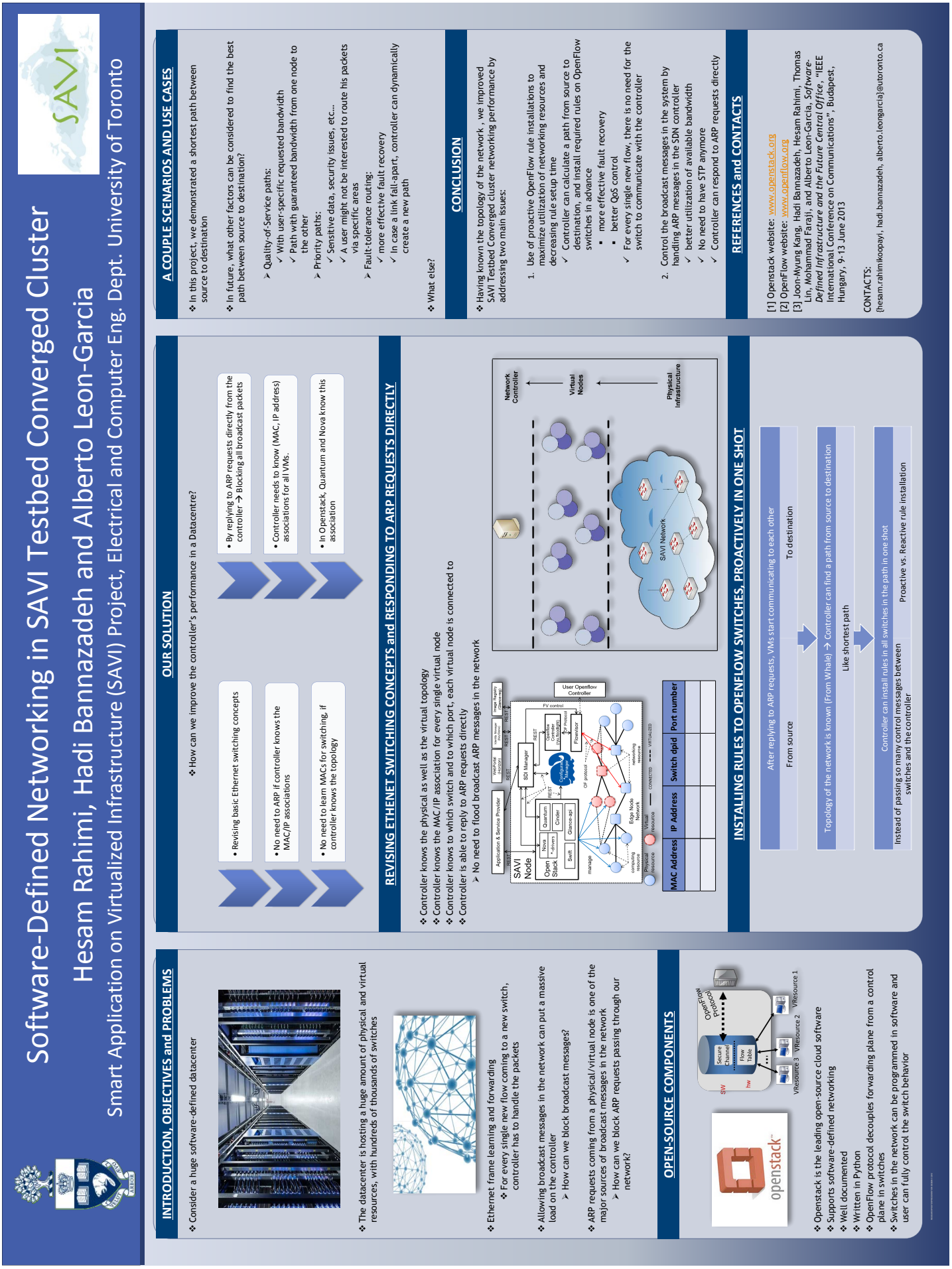


FIGURE 5.3: *Software-defined Networking (SDN) in SAVI Testbed Converged Cluster from Hesam Rahimi, Hadi Bannazadeh and Alberto Leon-Garcia at University of Toronto (Theme 5)*

30. NetThreads - Virtualizing FPGA Accelerators for SAVI

Stuart Byrma, **Greg Steffan**, **Paul Chow**, U. Toronto

FPGAs excel at high speed packet processing applications. However, not all users who require high speed applications have the required skills to develop complicated digital systems; for others, it is too time consuming and error prone. This work aims to provide a framework for the SAVI testbed that virtualizes FPGA resources—giving researchers a relatively quick and painless method of implementing hardware accelerators for packet processing, without the hassle of manually programming an FPGA.

With this system, users can view accelerators as cloud-based resources: accelerators can be set up and torn down quickly and on-the-fly, with the illusion of unlimited resources. We accomplish this using partial reconfiguration of the FPGAs. Accelerators could provide high speed applications such as encryption, protocol conversion, or video or audio stream processing. The virtualization system is designed to interact with the SAVI command line tools, enabling users to boot and delete accelerators similar to virtual machines in the SAVI Testbed.

NetThreads is an FPGA-based multithreaded multicore architecture, designed specifically for packet processing applications. Originally implemented and tested on the NetFPGA, the system was comprised of two MIPS-based cores with four threads each, executing in a round-robin fashion, with hardware-based locks for critical sections. A 10-slot packet input/output buffer holds packets for processing by the system. NetThreads abstracts away the base FPGA architecture, allowing non-hardware expert users to implement fast, gigabit speed packet processing using C programming. We aim to familiarize SAVI researchers with NetThreads: how to use it, and how to write packet processing applications for it. The demo will use NetThreads-10G, a recently ported and upgraded version of NetThreads running on the NetFPGA-10G. Attendees will learn the basic NetThreads architecture and programming model, and how to set up and use the system in the context of the SAVI testbed. A sample application will be developed, compiled and demonstrated.

Virtualizing FPGA Accelerators for SAVI

Stuart Byma, J. Gregory Steffan, Paul Chow

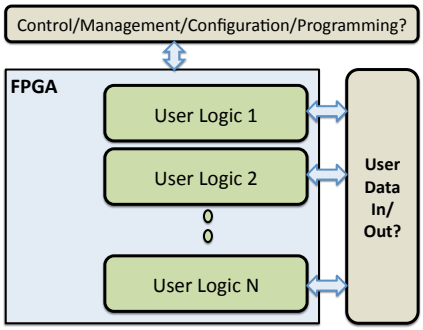
Department of Electrical and Computer Engineering
University of Toronto, Ontario, Canada

Motivation

- SAVI testbed contains FPGA resources, capable of low-latency, fast packet processing - much faster than general purpose machines
- Many flavours of FPGAs and boards - difficult to build portable, scalable, share-able applications
- By virtualizing SAVI FPGA resources, we can afford all users simple, flexible access to cloud-based hardware acceleration

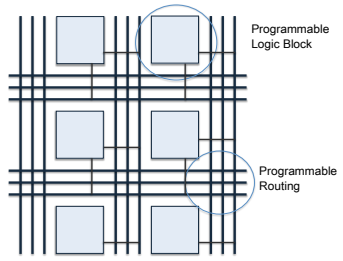
What is Virtualization?

- Adding an abstraction layer
  - e.g. PC hardware virtualization
  - Abstracts physical hardware, allowing "sharing" of hardware by several virtual computers
- Goal: Virtualize FPGAs
  - Allow multiple users to "share" the same FPGA fabric
  - Enable easy, flexible programmability
  - Make FPGA accelerators appear as cloud resources - create and tear down on the fly
- Challenge - How to accomplish this?



Wait, What is an FPGA?

- Field Programmable Gate Array
  - Tiled array of programmable logic blocks, interconnected by programmable routing architecture
  - Hard block RAMs, DSP Blocks (multipliers), high speed transeivers
  - Result: Fully reconfigurable hardware operating at speeds >500 MHz
- Drawback - Requires complex CAD flow to map design to the device
  - Big learning curve to implement significant designs



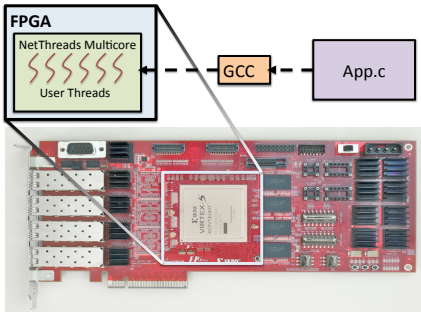
Basic modern FPGA architecture

Software Virtualization

- One or more virtual processors are implemented in the fabric of an FPGA
- Allows programming of the FPGA in software than hardware - much easier, more accessible
- Compiler toolchains enable fast, easy programming
- Generally bus-based, single-core, single-threaded designs.
- Commercial Products - Altera NIOS II, Xilinx MicroBlaze

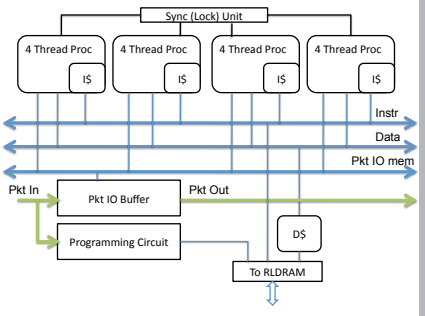
NetThreads: A Soft Multicore for Packet Processing

- Expansion on software virtualization - Gigabit speed packet processing in C
- NetThreads [1] is
  - A soft, multithreaded multicore system for packet processing
  - Closely coupled to Ethernet hardware - no OS or copying overhead as in a PC or VM
  - Programmable in C - a cross compiler toolchain is available
  - Threaded - Users could share the FPGA fabric via separate threads
- However - NetThreads alone can't fully capitalize on the FPGA's strengths - custom accelerators are faster
  - Can we combine them?



NetThreads Architecture

- 4 MIPS processors on Virtex 5 FPGA (NetFPGA-10G), 4 threads per core, round robin execution
- Shared data cache, hardware locks, private instruction caches
- Memory-mapped, shared, 20 packet capacity input output buffer
- 64MB RLDRAM main memory



NetThreads on NetFPGA-10G is Available!

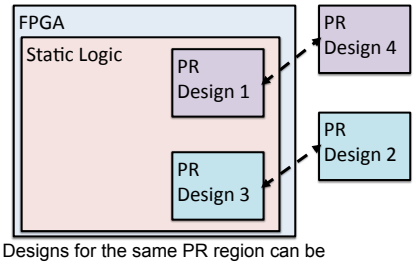
- NetThreads [1] has been ported to the NetFPGA-10G, and is available for use in the SAVI testbed
- We are looking for new applications and collaborators
- Possible applications include
  - Deep Packet Inspection - see demo!
  - Non-IP layer 3 routing
  - Encryption, firewalls
- Email - bymastua@eecg.toronto.edu

Acknowledgements

XILINX
CMC MICROSYSTEMS
NSERC CRSNG

Hardware Virtualization

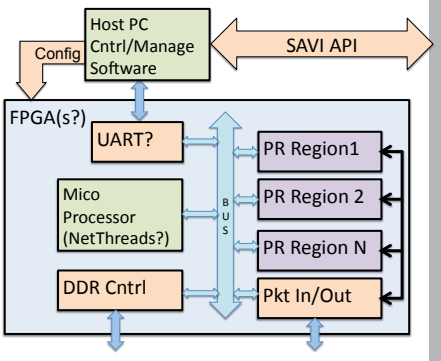
- Soft processors don't allow sharing of FPGA fabric on hardware level - not as fast as custom accelerators
- FPGA Partial Reconfiguration technology can allow sharing of FPGA fabric on the hardware level
- What is Partial Reconfiguration (PR)?
  - An architectural feature of an FPGA - allows reconfiguration of predefined sections of the fabric
  - Allows fast reconfiguration of subcircuits, without affecting other running sections of the FPGA
  - A long-time research topic, recently becoming more commercialized



Designs for the same PR region can be dynamically reconfigured ("swapped") at run time

A PR-Based FPGA Virtualization Framework for SAVI

- We envision a system that will
  - Abstract away the FPGA fabric entirely
    - As well as different devices, boards and vendors
    - Make multiple FPGAs appear as one fluid, flexible resource
  - Use PR regions to run different users' hardware accelerators concurrently on the same device
  - Interface with existing SAVI command line tools
  - Provide a library of accelerators for common packet processing functions
- There are still many questions:
  - How complex will the abstraction layer be? How will it be implemented? How will users, PR regions and PR designs be tracked?
  - How will the framework interface to the SAVI tools?
  - How can user data (packets) be directed to the correct PR region?
  - Could we use NetThreads to implement exceptional path handling for multiple users concurrently (e.g. dynamic thread allocation)?



References

M. Labrecque, J. G. Steffan, G. Salmon, M. Ghobadi, and Y. Ganjali.
Netthreads: Programming netfpga with threaded software.
In *NetFPGA Developers Workshop*, volume 9, 2009.

bymastua@eecg.toronto.edu

FIGURE 5.4: *Virtualizing FPGA Accelerators for SAVI from Stuart Byma, J. Gregory Steffan and Paul Chow at University of Toronto (Theme 5)*

31: "Identity Management in Software-defined Infrastructure"

Moe Faraji, Hadi Bannazadeh, **Alberto Leon-Garcia**, U. Toronto

Software defined infrastructure (SDI) enables applications to create their resource layer through the web service technology. This infrastructure leverages Service Oriented Architecture to decouple application and resource provider e.g. compute, network, storage, etc. Applications use mainly RESTful APIs to send their resource need to providers, where these requests are authenticated and authorized in the first place then proceed with the provider to be addressed. Therefore, identity manager is a key component in this platform which should be scalable, support different authentication mechanism, and more importantly provide a fine-grained access control to maintain the flexibility of platform.

In this work, we introduce a new architecture for Identity Manager (idM) in SDI which has been implemented on Smart Applications Virtual Infrastructure (SAVI) and built upon Openstack Keystone. SAVI IdM protects APIs through a middleware named enforcement layer. This middleware intercepts user or resource initiated requests before passing them to the next element in the pipeline. Consequently, the IdM receives a large number of authorization inquiries that in turn positions the IdM to be the bottleneck of system. SAVI IdM uses Attribute Based Access Control to bring flexibility and ultimate security to applications. In this poster, we also describe caching and load balancing techniques which maintain the IdM performance comparable to similar commercial solutions, while delivering all required functions to SAVI.

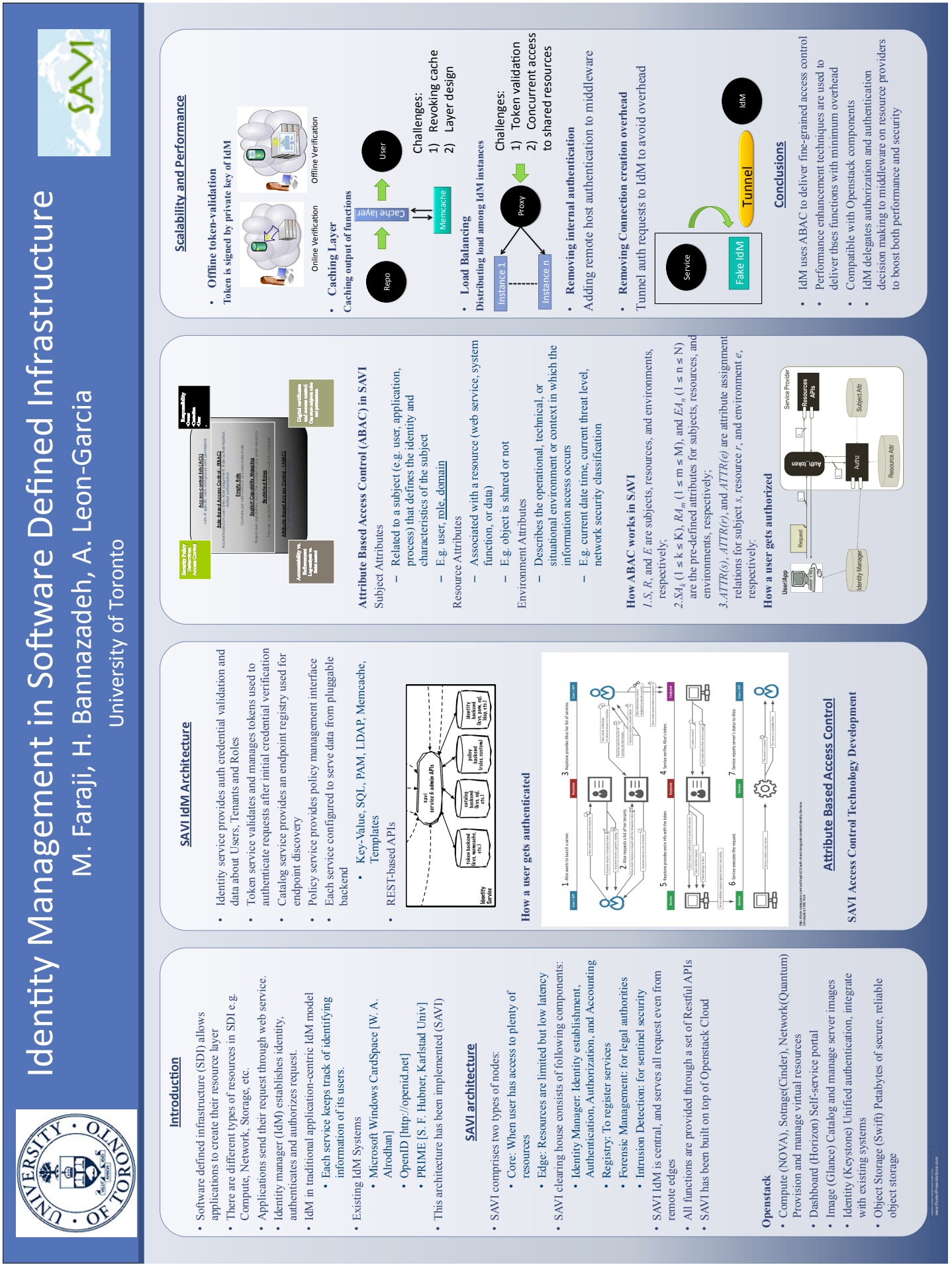


FIGURE 5.5: *Identity Manangement in Software Defined Infrastructure from M. Faraji, H. Bannazadeh and A. Leon-Garcia at University of Toronto (Theme 5)*

32: "Information Centric Data Dissemination Platform for Smart Transportation"

Aakash Nigam, Elman Mansimov, Ali Tizghadam, **Alberto Leon-Garcia**, U. Toronto

Evolving mobile cloud networks in settings such as connected vehicles and smart transportation require both content distribution as well as event notification and processsing support. Content Centric Networking (CCN), built around named data, is a clean slate network architecture for supporting future applications. Due to its focus on content distribution, CCN does not inherently support Publish-Subscribe-based event notification, a fundamental building block in computer-mediated systems and a critical requirement for smart tranportation applications. Semantics of content distribution and event notification requires different functions from the underlying network infrastructure, but can still be united by leveraging similarities in the routing infrastructure. We therefore have proposed a minimalistic extension to CCN architecture to provide a lightweight publish-subscribe service at the network layer. As a proof of concept, we have designed and implemented an open data sensing, aggregation, management and publishing system that gathers multiple streams of sensor data and provides high quality data for a common smart transportation application platform that supports ITS and privately offered applications.

Information Centric Data Dissemination Platform for Smart Transportation

Aakash Nigam, Elman Mansimov, Ali Tizghadam and Alberto Leon-Garcia

Department of Electrical and Computer Engineering – University of Toronto

Problem Statement

To design an open data sensing, aggregation, management and publishing system that gathers multiple streams of sensor data and provides high-quality data for a common smart transportation applications platform that supports ITS and privately-offered applications

Solution Approach for Smart Transportation Data Management

The Data Management Platform needs to handle:

- Transient and Persistent Information
- Delay sensitive and Location-aware information
- Event notification and on-demand content

It should be able to provide:

- Data security and authentication
- Support for high mobility applications

Existing Information Centric Based Approaches:

- *Content Centric Networking (CCN[1])*: Future Internet Proposal built on named data, where a packet address names content rather than location.
- *Publish-Subscribe (Pub-Sub[2])*: Provides decoupling between participants in Space, Time and Synchronization

Proposed Approach:

- To provide event notification (Pub-Sub) as a native service in the existing CCN architecture, leading to an Information Centric Network (ICN) Architecture.
- To support QoS for different types of data traffic flows in
- a smart transportation environment

Our Solution: Extended CCN

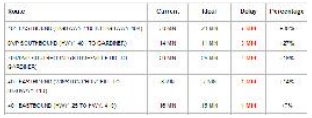
Our system supports both communication paradigms (Pub-Sub, CCN) over shared routing infrastructure by introducing extensions in the CCN architecture:

- Light weight predicate support
- Mechanism to support long interest
- Different service classes for QoS
- ATM-style flow control mechanism for Pub-Sub at Network Layer

Proof of Concept System: CVST (Connected Vehicles and Smart Transportation) Data Management Platform

Current CVST Data Sources (Publishers):

- One-ITS Consortium
  - Provides list of incident/accidents in GTA, OBD Data, Traffic Cameras, Loop Detectors, Traffic Maps
- CP24 Traffic Information
  - Provides travel times for GTA highways



- BIXI biking service
  - Provides station wise information about number of bikes and biking docs available.

Future Data Sources to be integrated:

- GPS mobility traces from Smart Phones
- TTC Information (Street Car, Subways)
- Participatory Sensing Information
- Car Pooling Information
- ZipCar (rental car) info
- Dynamic Tourist Information
- Parking Slot Information
- Twitter

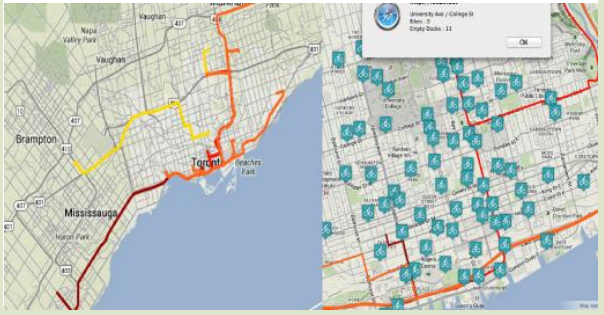
CVST Data Subscribers:

- Archival Storage
  - For trending historical data and forecasting road traffic conditions and urban city behavior.
  - To provide with more robust online routing algorithms for drivers.
- Analytics Engine:
  - For postprocessing data and extracting useful information Example: Analytics engine creates map of impacted area, i.e. road and intersection experiencing atypical traffic due to accidents/incidents.
- Application Platform to support
  - Rich ITS Applications
  - Private Sector Applications

Sample Application: Traffic Map of GTA

Receives data subscribed by Application platform and plots it on a map. Currently:

- Routes are color coded according to delay times
- Information from Bixi station (bikes and docs available) is displayed.



Smart Transportation Applications Enabled by ICN centric Data Management Platform

- Personalized multi-modal itinerary and updates
- Congestion Pricing, Collision Notification
- Enroute Multimodal Assistance
- Adhoc Point of Interest
- Services for people with reduced Mobility
- Continuous Monitoring and disruption handling
- Sharing (car, bike)
- Personalized Routing guidance
- Automated Payment

References

[1]Van Jacobson et al., "Networking Named Content," in *CoNEXT'09 Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies*, Rome, 2009, pp. 1-12.

[2]Jacobsen, H.-arno, Cheung, A., Li, G., Maniymaran, B., Muthusamy, V., & Kazemzadeh, R. S. (n.d.). The PADRES Publish / Subscribe.Carzaniga, A., & Papalini, M. (2011).

[3]Content-based publish/subscribe networking and information-centric networking. *Workshop on Information-centric networking*, Sigcomm, Toronto, 2011

FIGURE 5.6: *Information Centric Data Management for Smart Transportation from Aakash Nigam, Elman Mansimov, Ali Tizghadam and A. Leon-Garcia at University of Toronto (Theme 5)*

33. Scalable Content Centric Networking over SAVI

TTang Tang, **Alberto Leon-Garcia**, U. Toronto

Content Centric Networking (CCN) is a novel network architecture in which host-naming in the current Internet is replaced by content-naming: communication is resolved based on the relevant data transferred instead of locations of the end-hosts. In this project we first evaluated CCNx, an open-source prototype of IP-based CCN under realistic traffic load using SAVI virtualized resources. The performance bottleneck of current CCNx implementation was identified, and a few alternatives towards designing a scalable CCN router over virtualized infrastructure was proposed. We then developed and deployed a preliminary CCNx-based prototype over SAVI, and its performance was analyzed and compared to that of the original CCNx implementation.

Scalable Content Centric Networking over SAVI

Tang Tang, Alberto Leon-Garcia
Department of Electrical and Computer Engineering – University of Toronto

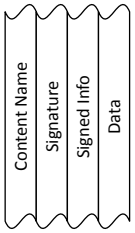
Project Goal and Objectives

- Identify bottlenecks of existing CCNx implementation
- Improve CCNx for scalability
- Test and verify the design, compare its performance with existing CCNx implementation

Introduction to CCN

Idea behind Content Centric Networking (CCN): *Content Naming* instead of *Host Addressing*

Two types of packets [1]:



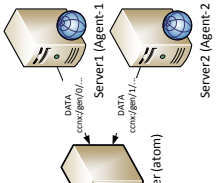
Three main components of a CCN node:

- Content Store (CS)
- Pending Interest Table (PIT)
- Forwarding Information Base (FIB)

CCNx Benchmarking

CCNx: an open-source implementation of CCN

Benchmarking topology:



Benchmarking tool: GNU gprof

Benchmarking results:

Function Name	CPU Usage %
`ccn_skeleton_decode`	52.61
`content_skiplist_findbefore`	5.60
`ccn_buf_advance`	5.15
`ccn_compare_names`	4.81
Others	31.83

Optimizing `ccn_skeleton_decode`

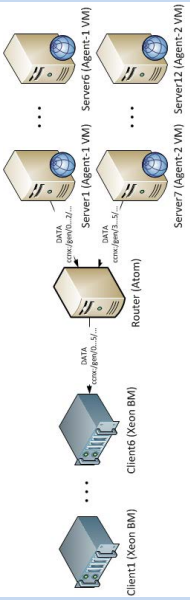
Approach 1: Parallelization

- Two approaches: multi-core CPUs or co-processors (e.g. GPGPU on SAVI)
- Problem: parallelization overhead

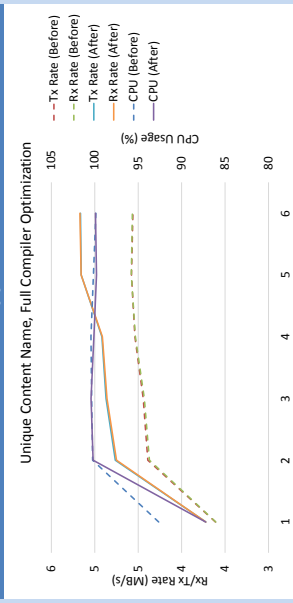
Approach 2: Change branching logic to table look-ups

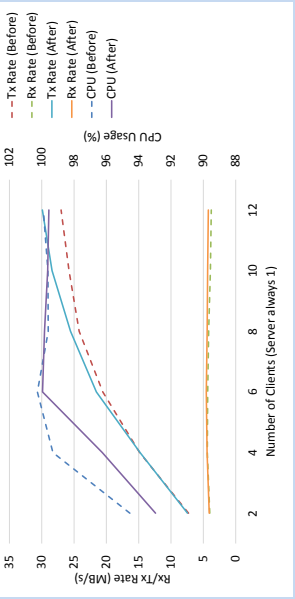
- Preload results in memory for common inputs/outputs
- Change is transparent to other system components

Experiment Setup



Results (Approach 2)

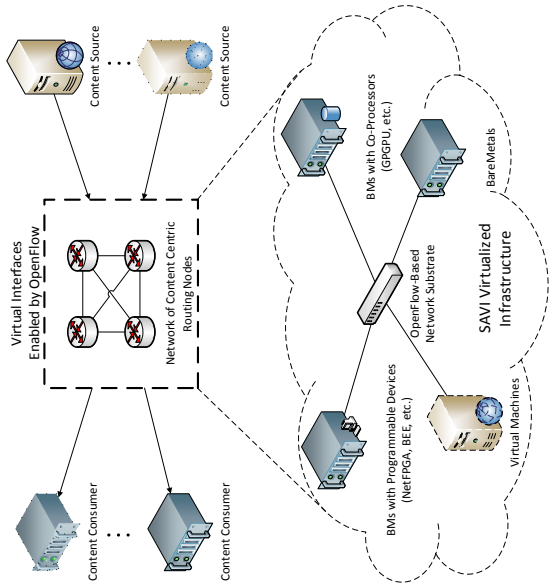




Remarks and Limitations

- 10~15% improvement to throughput of routing node under full CPU load
- However current max throughput (< 250Mbps) is still far from the link capacity (1Gbps)
- To further scale up the system, need more than single thread optimization

Next Step: Content Centric Routing Network



- Idea: distribute PIT and CS to multiple nodes
- Use OpenFlow to create a virtual interface (single IP) and to perform load balancing
- Opens up a new series of questions:
  - In-network routing
  - Table partitioning
  - Dynamic scaling
  - New bottleneck analysis, etc.

References

[1] Van Jacobson et al., "Networking Named Content," in *CoNEXT'09 Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies*, Rome, 2009, pp. 1-12.

FIGURE 5.7: *Scalable Content Networking over SAVI from Tang Tang and A. Leon-Garcia at University of Toronto (Theme 5)*

34. Software-Defined Infrastructure and the Future CO

Joon-Myung Kang, Hadi Bannazadeh, Hesam Rahimi, Thomas Lin, Moe Faraji, **Alberto Leon-Garcia**, U. Toronto

This research discusses the role of virtualization and software-defined infrastructure (SDI) in the design of future application platforms, and in particular the Future Central Office (CO). A multi-tier computing cloud is presented in which resources in the Smart Edge of the network play a crucial role in the delivery of low-latency and data-intensive applications. Resources in the Smart Edge are virtualized and managed using cloud computing principles, but these resources are more diverse than in conventional data centers, including programmable hardware, GPUs, etc. We propose an architecture for future application platforms, and we describe the SAVI Testbed design for the Smart Edge. The design features a novel Software-Defined Infrastructure manager that operates on top of OpenStack and OpenFlow. We conclude with a discussion of the implications of the Smart Edge design on the Future CO.

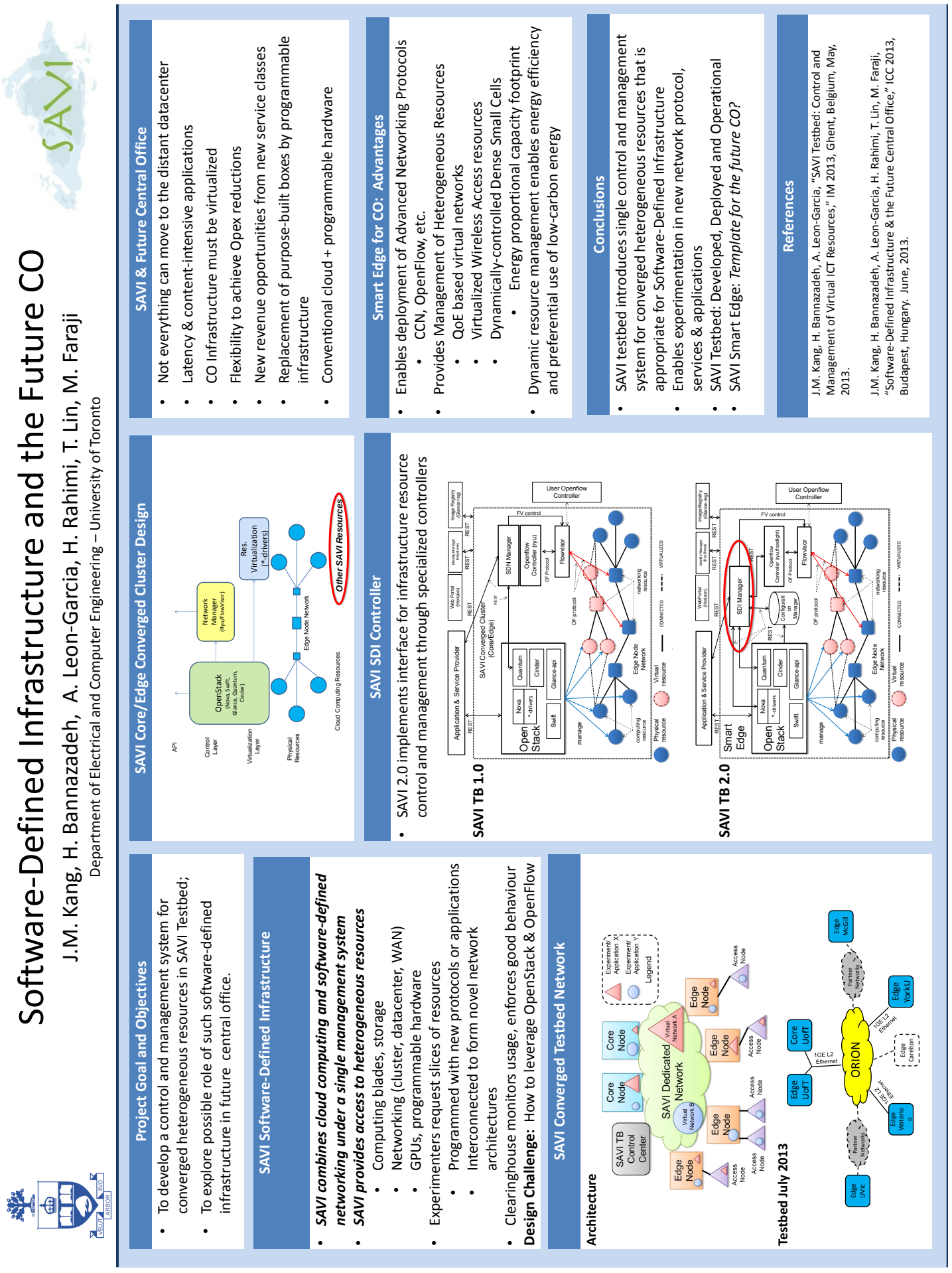


FIGURE 5.8: *Software-Defined Infrastructure and the Future CO from J.M.Kang, H. Bannazadeh and A. Leon-Garcia at University of Toronto (Theme 5)*

Page for Notes