A Decomposition-Based Architecture for Distributed Virtual Network Embedding

Flavio Esposito
Advanced Technologies
exegy
St.Louis, MO

Ibrahim Matta
Computer Science Department
BOSTON UNIVERSITY
Imagine a world without knobs
We have knobs to tune what’s important to us

House’s Behavior (Electrical)
oven, air conditioning, ...

Car’s Behavior (Mechanical)
speed, music volume, ...

Organ’s Behavior (Biological)
diabetes, dopamine, hormones, ...
Wouldn’t it be great to have knobs for protocols too?

**Algorithmic tradeoffs**
e.g., converge speed vs performance optimality

**Design tradeoffs**
e.g., centralized or distributed

**Implementation tradeoffs**
e.g., support for delay or BW sensitive app
Ad-hoc protocol development prevents quick adaptation

Small Cloud-based **Startup** Adaptation is the key to survive
code to customers needs quickly

Medium-size **Enterprise** **Slower growth**: can’t afford clean-slate
too much to change: this one works!
...till when?

Big Cloud **Provider** **Ossification** (see Internet)
only few dominate the cloud market
Need for a Cloud easier and cheaper to manage

“Management is responsible for 80% of IT budget”
responsible for 62% of outages”

Millions of dollars are spent in support of many protocols and apps with different requirements
Virtual Network Embedding is a Constrained Graph Matching

Shared Infrastructure

Virtual networks

Possibly multiple:
Service Providers (SPs)
Infrastructure Providers (InPs)
Use our architecture to customize distributed VN embedding solutions

Apply decomposition techniques to VN Embedding solved as network utility maximization problem.

Obtain distributed algorithms with different goals that suit needs of specific applications
Can you model your problem as Network Utility Maximization?

- Different instantiation of the optimization problem
- Different policies
- Different implications

Alternative VN Embedding problem representations

- Different decompositions
  - Embedding algorithms
  - Engineering tradeoffs
Our Contributions (Talk Outline)

- **To separate policies from VN embedding mechanisms**: To adapt embedding to providers’ goals
- **Unifying VN Embedding Architecture**: Identifying the invariances of the problem
- **Design new and subsume embedding solutions**: Instantiating different decomposition policies
- **Tradeoff analysis over simulation & prototype**: Implemented over a local Linux-based testbed
- **To enable experiments with new VN embedding policies**
Identification of Mechanisms

To separate policies from VN embedding mechanisms

Unifying VN Embedding Architecture
Identifying the invariances of the problem

Design new and subsume embedding solutions
Instantiating different decomposition policies

Tradeoff analysis over simulation & prototype implemented over a local Linux-based testbed
An architecture identifies the invariances in a problem

What is an architecture?
1. Identify mechanisms (invariances)
2. Identify who does what (separation of functionalities)

What is a policy? (variant aspect of a mechanism)
Provider policies to host virtual nodes & links
Three common mechanisms interfaced by binding constraints
Three common mechanisms interfaced by binding constraints

Maximize

subject to:

Discovery

Discovery p-nodes

Discovery p-paths
Three common mechanisms interfaced by binding constraints

Maximize

subject to:

Discovery

Discovery p-nodes

Discovery p-paths

Mapping

Mapping all v-nodes

Mapping all v-links

Map v-nodes to p-nodes

Map v-links to p-paths
Three common mechanisms interfaced by binding constraints

Maximize

subject to:

- Discovery
- Mapping
- Allocation

- Discovery p-nodes
- Discovery p-paths
- Mapping all v-nodes
- Mapping all v-links
- Map v-nodes to p-nodes
- Map v-links to p-paths
- P-nodes capacity
- P-links capacity
Three common mechanisms interfaced by binding constraints

Maximize

subject to:

Discovery

Discovery p-nodes

Discovery p-paths

Mapping

Mapping all v-nodes

Mapping all v-links

Map v-nodes to p-nodes

Map v-links to p-paths

Allocation

P-nodes capacity

P-links capacity

Binding Constraints
Different decompositions for different distributed (VN embedding) solutions

To separate policies from VN embedding mechanisms
Unifying VN Embedding Architecture identifying the invariances of the problem

To adapt embedding to providers’ goals
Design new and subsume embedding solutions instantiating different decomposition policies

Enable experiments with new embedding policies
Tradeoff analysis over simulation & prototype implemented over a local Linux-based testbed
The structure of the problem is rich so many decompositions are possible

**Primal** Decomposition

**Dual** Decomposition
The structure of the problem is rich so many decompositions are possible

Solve fixing **Variables:**  Primal Decomposition

Solve relaxing **Constraints:**  Dual Decomposition
The structure of the problem is rich so many decompositions are possible.

Solve fixing **Variables:** Primal Decomposition

**Master:** assigns ph.resources to subproblems

**Subproblems:** embed given available resources

Solve relaxing **Constraints:** Dual Decomposition

**Master:** sets prices of all virtual resources

**Subproblems:** embed given resource prices
Primal: decompose fixing decision variables

Fix nodes & links decision variables
Optimize virtual nodes then links using the optimal node embedding
see e.g. [9,20]
Primal: decompose fixing decision variables

Fix nodes & links decision variables  \[\rightarrow\] Optimize virtual nodes then links using the optimal node embedding

Fix sub-VN decision variables  \[\rightarrow\] Optimize nodes & links of first VN partition then nodes & links of the next VN partition

see e.g. [9,20]

see e.g. [14,15]
Primal: decompose fixing decision variables

- Fix nodes & links decision variables
- Optimize virtual nodes then links using the optimal node embedding
  see e.g. [9,20]

- Fix sub-VN decision variables
- Optimize nodes & links of first VN partition then nodes & links of the next VN partition
  see e.g. [14,15]

- Fix subproblems decision variables
- Discovery and/or Mapping and/or Allocation
  Optimize only some embedding mechanisms
  see e.g. [6,18]
Dual: decompose relaxing constraints

Relax binding constraints between Discovery and VN Mapping phase
see e.g. [2,12]
Dual: decompose relaxing constraints

Relax binding constraints between 
Discovery and VN Mapping phase

see e.g. [2,12]

Relax binding constraints between
VN Mapping & Allocation phase

see e.g. [14]
Dual: decompose relaxing constraints

Relax binding constraints between Discovery and VN Mapping phase
see e.g. [2,12]

Relax binding constraints between VN Mapping & Allocation phase
see e.g. [14]

Relax any non-binding constraints
[your papers here]
Use primal and dual to design your VN embedding solution

Service Provider instantiates according to its policies
Use primal and dual to design your VN embedding solution

Service Provider instantiates according to its policies

Infrastructure Providers Solve it (decomposing)
Use primal and dual to design your VN embedding solution

Service Provider instantiates according to its policies

Infrastructure Providers Solve it (decomposing)

Optimal are returned to SP that releases next VN
Testing different decompositions over our VN embedding Prototype

To separate policies from VN embedding mechanisms
Unifying VN Embedding Architecture
Identifying the invariances of the problem

To adapt embedding to providers’ goals
Design new and subsume embedding solutions
Instantiating different decomposition policies

Enable experiments with new embedding policies
Tradeoff analysis over simulation & prototype
Implemented over a local Linux-based testbed
8 Years of real Emulab VN requests used in our CPLEX Simulations

Virtual Topologies
  Range of VN size [1, 100]

Physical Topologies
  Waxman and Barabasi-Albert with BRITE

Performance Metrics
  Computational Time, Optimality

62,000 Emulab requests ’01 to ’09
Primal has lower computation time
Dual has lower optimality gap
We release our VINEA testbed for VINO objects instantiation

VNIrtual NNetwork Embedding Architecture

VNIrtual NNetwork Objects (VINO)
We release our **VINEA** testbed for VINO objects instantiation

**Virtual Network Embedding Architecture**

VINEA nodes join a private overlay after authentication and policy exchange

InPs run the VN Mapping Protocol asynchronous consensus-based

Virtual resources are reserved using vSwitch and Linux traffic control
We release our VINEA testbed for VINO objects instantiation

**Virtual Network Embedding Architecture**

**Discovery**
VINEA nodes join a private overlay after authentication and policy exchange

**VN Mapping**
InPs run the VN Mapping Protocol asynchronous consensus-based

**Allocation**
Virtual resources are reserved using vSwitch and Linux traffic control
The embedding protocol runs on a private overlay.

VINEA nodes join a private overlay after authentication and policy exchange.

InPs run the VN Mapping Protocol as asynchronous consensus-based.

Virtual resources are reserved using vSwitch and Linux traffic control.
Resource Discovery: SP and InPs processes enroll in a private overlay.

New InP process
Physical Node (PN)

Overlay Manager

InP overlay with private addressing
Resource Discovery: SP and InPs processes enroll in a private overlay

1. Authentication
   user and pwd

New InP process
Physical Node (PN)

Overlay Manager

InP overlay with private addressing
Resource Discovery: SP and InPs processes enroll in a private overlay

1. Authentication
   user and pwd

2. Policy exchange

Overlay Manager

New InP process
Physical Node (PN)

InP overlay with private addressing
Resource Discovery: SP and InPs processes enroll in a private overlay

1. Authentication user and pwd
2. Policy exchange
3. Establish connections

Overlay Manager

New InP process Physical Node (PN)

InP overlay with private addressing
Resource Discovery: SP and InPs processes enroll in a private overlay

1. Authentication user and pwd
2. Policy exchange
3. Establish connections
4. Fork monitoring process for new PN

Overlay Manager

PN

New InP process
Physical Node (PN)

InP overlay with private addressing
VN Mapping: physical nodes decide which hosts paying the lowest price

Virtual Network Embedding Architecture

Discovery
VINEA nodes join a private overlay after authentication and policy exchange

VN Mapping
InPs run the VN Mapping Protocol asynchronous consensus-based

Allocation
Virtual resources are reserved using vSwitch and Linux traffic control
VN Mapping: physical nodes decide which hosts paying the lowest price

$\ x_{ij} = ?\$

Service Provider

Releases VN request & embedding policies

VN to embed

Hosting InP 1

Hosting InP n
VN Mapping: physical nodes decide which hosts paying the lowest price

Service Provider

$x_{ij} = ?$

VN to embed

Lagrangian (prices)

Hosting InP 1

 Hosting InP n

x_1 = \{1, 1, 0, 0\}

x_2 = \{0, 0, 1, 1\}

Distributed consensus on the lowest price & winner on virtual resources (nodes and links)
VN Mapping: physical nodes decide which hosts paying the lowest price

Distributed consensus on the lowest price & winner on virtual resources (nodes and links)
Allocation: User-space vHost, Kernel vSwitches & BW is reserved with Linux tc

Virtual Network Embedding Architecture

Discovery
VINEA nodes join a private overlay after authentication and policy exchange

VN Mapping
InPs run the VN Mapping Protocol asynchronous consensus-based

Allocation
Virtual resources are reserved using vSwitch and Linux traffic control
Allocation: User-space vHost, Kernel vSwitches & BW is reserved with Linux tc
Partitioning lowers the allocation ratio and increases the network overhead.

![Graph showing VN Allocation Ratio vs. Virtual Network Size]

- **Single Partition**
- **Nv/2 Partitions**

![Bar chart showing Overhead vs. Virtual Network size]

**VN embedding Prototype**

What to remember from this talk?

Ad-hoc protocols prevent adaptation
What to remember from this talk?

Ad-hoc protocols prevent adaptation

Decomposition is a useful tool for distributed cloud algorithm design and tradeoff analysis
What to remember from this talk?

Ad-hoc protocols prevent adaptation

Decomposition is a useful tool for distributed cloud algorithm design and tradeoff analysis

Use our VINEA testbed to test your own embedding policies
http://csr.bu.edu/vinea
A Decomposition-Based Architecture for Distributed Virtual Network Embedding

Flavio Esposito

Ibrahim Matta

http://csr.bu.edu/vinea

Computer Science Research

Virtual Network Embedding Architecture

Boston University