Beyond the Stars: Revisiting Virtual Cluster Embeddings

Matthias Rost
Technische Universität Berlin

February 2, 2016, Eötvös Loránd University

Joint work with *Carlo Fuerst, Stefan Schmid*
Published in ACM SIGCOMM CCR Volume 45 Issue 3, July 2015

Data Center Applications

Virtualization has changed our view on complex computations

- Virtual machines (VMs) can be spawned within minutes and ‘anywhere’ on the world (Microsoft Azure, Amazon EC2, . . .)
- Complex tasks can quite easily be distributed to tens or hundreds of servers using frameworks like MapReduce

Insight: Application performance does depend on bandwidth availability

- Facebook traces show that network transfers account for 33% of the execution time [4]
- Data centers exhibit oversubscription factors of upto 1:240 [6]
- Customer’s application execution time can hardly be estimated!

Data Center Applications

Insight

Performance does depend on bandwidth availability

How to achieve resource isolation?

- New topologies limiting the oversubscription factor: Fat trees, MDCubes, ...

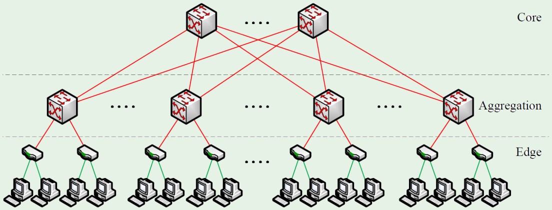


Figure : Fat tree topology [1]

Data Center Applications

Insight

Performance does depend on bandwidth availability

How to achieve resource isolation?

- New topologies limiting the oversubscription factor: Fat trees, MDCubes, . . .

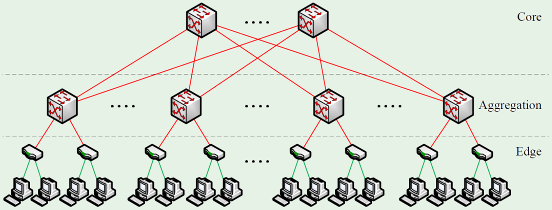
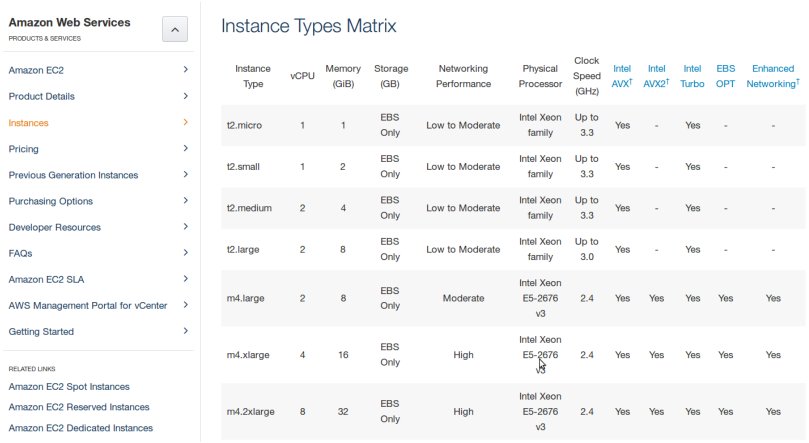


Figure : Fat tree topology [1]

- *Novel descriptions of applications → embedding algorithms*

‘Application Specification’: Case Study Amazon AWS

- Customers can select from a huge number of ‘instances’



Amazon Web Services
PRODUCTS & SERVICES

- Amazon EC2
- Product Details
- Instances
- Pricing
- Previous Generation Instances
- Purchasing Options
- Developer Resources
- FAQs
- Amazon EC2 SLA
- AWS Management Portal for vCenter
- Getting Started

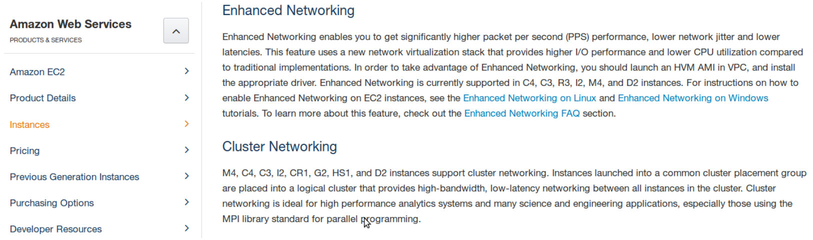
RELATED LINKS
- Amazon EC2 Spot Instances
- Amazon EC2 Reserved Instances
- Amazon EC2 Dedicated Instances

Instance Types Matrix

Instance Type	vCPU	Memory (GiB)	Storage (GB)	Networking Performance	Physical Processor	Clock Speed (GHz)	Intel AVX†	Intel AVX2†	Intel Turbo	EBS OPT	Enhanced Networking†
t2.micro	1	1	EBS Only	Low to Moderate	Intel Xeon family	Up to 3.3	Yes	-	Yes	-	-
t2.small	1	2	EBS Only	Low to Moderate	Intel Xeon family	Up to 3.3	Yes	-	Yes	-	-
t2.medium	2	4	EBS Only	Low to Moderate	Intel Xeon family	Up to 3.3	Yes	-	Yes	-	-
t2.large	2	8	EBS Only	Low to Moderate	Intel Xeon family	Up to 3.0	Yes	-	Yes	-	-
m4.large	2	8	EBS Only	Moderate	Intel Xeon E5-2676 v3	2.4	Yes	Yes	Yes	Yes	Yes
m4.xlarge	4	16	EBS Only	High	Intel Xeon E5-2676 v3	2.4	Yes	Yes	Yes	Yes	Yes
m4.2xlarge	8	32	EBS Only	High	Intel Xeon E5-2676 v3	2.4	Yes	Yes	Yes	Yes	Yes

'Application Specification': Case Study Amazon AWS

- Customers can select from a huge number of 'instances'
- Customers can select 'enhanced networking' or 'cluster networking':



Enhanced Networking

Enhanced Networking enables you to get significantly higher packet per second (PPS) performance, lower network jitter and lower latencies. This feature uses a new network virtualization stack that provides higher I/O performance and lower CPU utilization compared to traditional implementations. In order to take advantage of Enhanced Networking, you should launch an HVM AMI in VPC, and install the appropriate driver. Enhanced Networking is currently supported in C4, C3, R3, I2, M4, and D2 instances. For instructions on how to enable Enhanced Networking on EC2 instances, see the Enhanced Networking on Linux and Enhanced Networking on Windows tutorials. To learn more about this feature, check out the Enhanced Networking FAQ section.

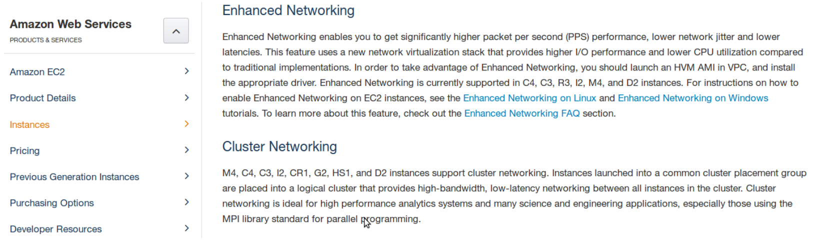
Cluster Networking

M4, C4, C3, I2, CR1, G2, HS1, and D2 instances support cluster networking. Instances launched into a common cluster placement group are placed into a logical cluster that provides high-bandwidth, low-latency networking between all instances in the cluster. Cluster networking is ideal for high performance analytics systems and many science and engineering applications, especially those using the MPI library standard for parallel programming.

Instances launched into a common cluster placement group are placed into a logical cluster that provides high-bandwidth, low-latency networking between all instances in the cluster.

'Application Specification': Case Study Amazon AWS

- Customers can select from a huge number of 'instances'
- Customers can select 'enhanced networking' or 'cluster networking':



Enhanced Networking

Enhanced Networking enables you to get significantly higher packet per second (PPS) performance, lower network jitter and lower latencies. This feature uses a new network virtualization stack that provides higher I/O performance and lower CPU utilization compared to traditional implementations. In order to take advantage of Enhanced Networking, you should launch an HVM AMI in VPC, and install the appropriate driver. Enhanced Networking is currently supported in C4, C3, R3, I2, M4, and D2 instances. For instructions on how to enable Enhanced Networking on EC2 instances, see the Enhanced Networking on Linux and Enhanced Networking on Windows tutorials. To learn more about this feature, check out the Enhanced Networking FAQ section.

Cluster Networking

M4, C4, C3, I2, CR1, G2, HS1, and D2 instances support cluster networking. Instances launched into a common cluster placement group are placed into a logical cluster that provides high-bandwidth, low-latency networking between all instances in the cluster. Cluster networking is ideal for high performance analytics systems and many science and engineering applications, especially those using the MPI library standard for parallel programming.

Instances launched into a common cluster placement group are placed into a logical cluster that provides high-bandwidth, low-latency networking between all instances in the cluster.

Resource Isolation?

Best-effort!

Level of Specification?

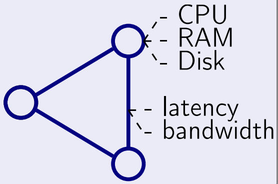
Grouping of instances!

Application Abstractions: The VC Abstraction

The Right Level of Abstraction

Early 2000s: Virtual Network Embedding Problem

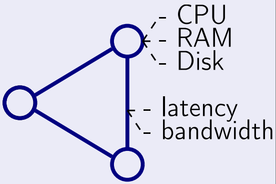
- Requests are specified as graphs
  - Nodes represent VMs
  - Edges represent inter-VM links



The Right Level of Abstraction

Early 2000s: Virtual Network Embedding Problem

- Requests are specified as graphs
  - Nodes represent VMs
  - Edges represent inter-VM links



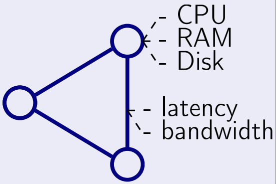
Pro

- *Concise specification*

The Right Level of Abstraction

Early 2000s: Virtual Network Embedding Problem

- Requests are specified as graphs
  - Nodes represent VMs
  - Edges represent inter-VM links



Pro

- *Concise specification*

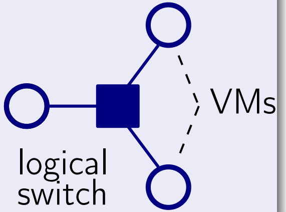
Contra

- Do customers know their requirements?
- Generally: Challenging NP-hard problem!

The Right Level of Abstraction

2011: Virtual Cluster (VC) Abstraction [3]

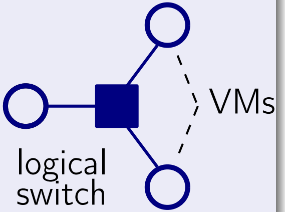
- Allows only for 'star'-shaped graphs
- VMs are connected to logical switch



The Right Level of Abstraction

2011: Virtual Cluster (VC) Abstraction [3]

- Allows only for ‘star’-shaped graphs
- VMs are connected to logical switch

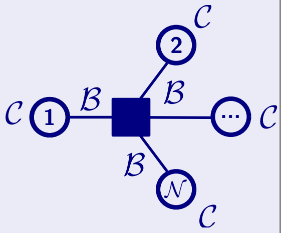


The Right Level of Abstraction

2011: Virtual Cluster (VC) Abstraction [3]

- Allows only for ‘star’-shaped graphs
- VMs are connected to logical switch
- Requests are specified by three parameters:

$\mathcal{N} \in \mathbb{N}$	number of virtual machines
$\mathcal{C} \in \mathbb{N}$	size of virtual machines
$\mathcal{B} \in \mathbb{R}^+$	bandwidth towards logical switch

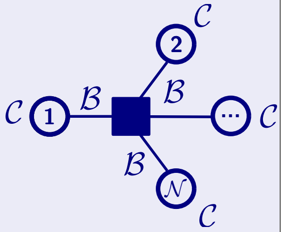


The Right Level of Abstraction

2011: Virtual Cluster (VC) Abstraction [3]

- Allows only for 'star'-shaped graphs
- VMs are connected to logical switch
- Requests are specified by three parameters:

$\mathcal{N} \in \mathbb{N}$	number of virtual machines
$\mathcal{C} \in \mathbb{N}$	size of virtual machines
$\mathcal{B} \in \mathbb{R}^+$	bandwidth towards logical switch



Pro

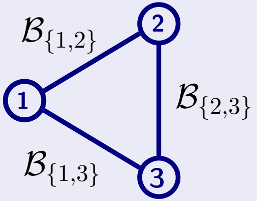
- *Simple specification!*
- *Well-performing heuristics for data-center topologies [3, 8]*

Contra

- The VM size and the amount of bandwidth are dictated by the maximum $\rightarrow$ wasteful

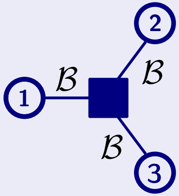
On Traffic Matrices

Graph Abstraction



- Allows for any traffic matrix M, where the bandwidth for edge $\{i,j\}$ is less than $\mathcal{B}_{\{i,j\}}$.

VC Abstraction



- Allows for any traffic matrix M, where for any VM the sum of outgoing and incoming traffic is less than $\mathcal{B}$

Outlook

Previous works . . .

- only considered (fat) trees
- only considered heuristics

Ballani et al.: 'Oktopus' [3]

"allocating virtual cluster requests on graphs with bandwidth-constrained edges is NP-hard"

Xie et al.: 'Proteus' [8]

"[Our algorithm] picks the first fitting lowest-level subtree out of all such lowest-level subtrees."

Outlook

Previous works . . .

- only considered (fat) trees
- only considered heuristics

Ballani et al.: ‘Oktopus’ [3]

“allocating virtual cluster requests on graphs with bandwidth-constrained edges is NP-hard”

Xie et al.: ‘Proteus’ [8]

“[Our algorithm] picks the first fitting lowest-level subtree out of all such lowest-level subtrees.”

Main Questions

Is the VC embedding problem really NP-hard to solve?

Formal Definition of the VC Embedding Problem

VC Embedding Problem Defintion

VC request: $\mathcal{N}, \mathcal{B}, \mathcal{C}$

- $\mathsf{VC} = (V_{\mathsf{VC}}, E_{\mathsf{VC}})$,
- $V_{\mathsf{VC}} = \{1, 2, \ldots, \mathcal{N}, \mathsf{center}\}$
- $E_{\mathsf{VC}} = \{\{i, \mathsf{center}\} | 1 \leq i \leq \mathcal{N}\}$

Physical Network (Substrate)

- $\mathsf{S} = (V_{\mathsf{S}}, E_{\mathsf{S}}, \mathsf{cap}, \mathsf{cost})$,
- $\mathsf{cap} : V_{\mathsf{S}} \cup E_{\mathsf{S}} \to \mathbb{N}$
- $\mathsf{cost} : V_{\mathsf{S}} \cup E_{\mathsf{S}} \to \mathbb{R}_{\geq 0}$

Task: Find a mapping of . . .

- VMs onto substrate nodes $\mathsf{map}_V : V_{\mathsf{VC}} \to V_{\mathsf{S}}$, and
- VC edges onto paths in the substrate $\mathsf{map}_E : E_{\mathsf{VC}} \to \mathcal{P}(E_{\mathsf{S}})$

VC Embedding Problem Defintion

VC request: $\mathcal{N}, \mathcal{B}, \mathcal{C}$

- $\mathsf{VC} = (V_{\mathsf{VC}}, E_{\mathsf{VC}})$,
- $V_{\mathsf{VC}} = \{1, 2, \ldots, \mathcal{N}, \mathsf{center}\}$
- $E_{\mathsf{VC}} = \{\{i, \mathsf{center}\} | 1 \leq i \leq \mathcal{N}\}$

Physical Network (Substrate)

- $\mathsf{S} = (V_{\mathsf{S}}, E_{\mathsf{S}}, \mathsf{cap}, \mathsf{cost})$,
- $\mathsf{cap} : V_{\mathsf{S}} \cup E_{\mathsf{S}} \to \mathbb{N}$
- $\mathsf{cost} : V_{\mathsf{S}} \cup E_{\mathsf{S}} \to \mathbb{R}_{\geq 0}$

Task: Find a mapping of ...

- VMs onto substrate nodes $\mathsf{map}_V : V_{\mathsf{VC}} \to V_{\mathsf{S}}$, and
- VC edges onto paths in the substrate $\mathsf{map}_E : E_{\mathsf{VC}} \to \mathcal{P}(E_{\mathsf{S}})$, such that

  1. $\mathsf{map}_E(\{u, v\})$ connects $\mathsf{map}_V(u)$ and $\mathsf{map}_V(v)$ for $\{u, v\} \in E_{\mathsf{VC}}$

VC Embedding Problem Defintion

VC request: $\mathcal{N}, \mathcal{B}, \mathcal{C}$

- $\mathsf{VC} = (V_{\mathsf{VC}}, E_{\mathsf{VC}})$,
- $V_{\mathsf{VC}} = \{1, 2, \ldots, \mathcal{N}, \mathsf{center}\}$
- $E_{\mathsf{VC}} = \{\{i, \mathsf{center}\} | 1 \leq i \leq \mathcal{N}\}$

Physical Network (Substrate)

- $\mathsf{S} = (V_{\mathsf{S}}, E_{\mathsf{S}}, \mathsf{cap}, \mathsf{cost})$,
- $\mathsf{cap} : V_{\mathsf{S}} \cup E_{\mathsf{S}} \to \mathbb{N}$
- $\mathsf{cost} : V_{\mathsf{S}} \cup E_{\mathsf{S}} \to \mathbb{R}_{\geq 0}$

Task: Find a mapping of ...

- VMs onto substrate nodes $\mathsf{map}_V : V_{\mathsf{VC}} \to V_{\mathsf{S}}$, and
- VC edges onto paths in the substrate $\mathsf{map}_E : E_{\mathsf{VC}} \to \mathcal{P}(E_{\mathsf{S}})$, such that

1. $\mathsf{map}_E(\{i, j\})$ connects $\mathsf{map}_V(i)$ and $\mathsf{map}_V(j)$ for $\{i.j\} \in E_{\mathsf{VC}}$
2. $\displaystyle\sum_{\substack{v' \in V_{\mathsf{VC}} \setminus \{\mathsf{center}\} \\ v = \mathsf{map}_V(v')}} \mathcal{C} \leq \mathsf{cap}(v)$ and $\displaystyle\sum_{\substack{e' \in E_{\mathsf{VC}} \\ e \in \mathsf{map}_E(e')}} \mathcal{B} \leq \mathsf{cap}(e)$ for $v \in V_{\mathsf{S}}, e \in E_{\mathsf{S}}$

VC Embedding Problem Defintion

Task: Find a mapping of ...

- VMs onto substrate nodes $\mathsf{map}_V : V_{\mathsf{VC}} \to V_{\mathsf{S}}$, and
- VC edges onto paths in the substrate $\mathsf{map}_E : E_{\mathsf{VC}} \to \mathcal{P}(E_{\mathsf{S}})$, such that

1. $\mathsf{map}_E(\{u,v\})$ connects $\mathsf{map}_V(u)$ and $\mathsf{map}_V(v)$ for $\{u,v\} \in E_{\mathsf{VC}}$
2. $\sum_{\substack{v' \in V_{\mathsf{VC}} \setminus \{\mathsf{center}\} \\ v = \mathsf{map}_V(v')}} \mathcal{C} \leq \mathsf{cap}(v)$ and $\sum_{\substack{e' \in E_{\mathsf{VC}} \\ e \in \mathsf{map}_E(e')}} \mathcal{B} \leq \mathsf{cap}(e)$ for $v \in V_{\mathsf{S}}, e \in E_{\mathsf{S}}$
3. minimizing the cost $\mathcal{C} \cdot \sum_{v \in V_{\mathbf{VC}} \setminus \{\mathsf{center}\}} \mathsf{cost}(\mathsf{map}_V(v)) + \mathcal{B} \cdot \sum_{\substack{e' \in E_{\mathsf{VC}} \\ e \in \mathsf{map}_E(e')}} \mathsf{cost}(e)$.

VC-ACE Algorithm

Key Insights

Lemma

We can assume $\mathcal{B} = \mathcal{C} = 1$.

Proof idea.

If $\mathcal{B} \neq 1$, $\mathcal{C} \neq 1$, we transform the substrate by scaling capacities and costs:

- $\text{cap}_{S'}(u) = \lfloor \text{cap}(u)/\mathcal{C} \rfloor$ for $u \in V_S$
- $\text{cap}_{S'}(e) = \lfloor \text{cap}(e)/\mathcal{B} \rfloor$ for $e \in E_S$
- $\text{cost}_{S'}(u) = \text{cost}(u) \cdot \mathcal{C}$ for $u \in V_S$
- $\text{cost}_{S'}(e) = \text{cost}(e) \cdot \mathcal{B}$ for $e \in E_S$

$\square$

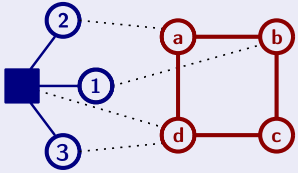
Key Insights

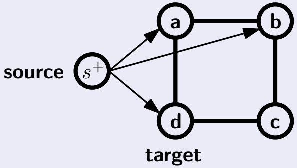
Lemma

We can solve the edge embedding problem if all nodes are placed.

Proof.

1. Construct extended graph with additional node s^+ and (parallel) edges: $\{(s^+, \mathrm{map}_V(i)) | i \in \{1, \ldots, \mathcal{N}\}\}$ of capacity 1 and cost 0
2. Compute a minimum cost flow of value $\mathcal{N}$ from s^+ to $\mathrm{map}_V(\text{center})$.
3. Perform a path-decomposition to obtain mapping for edges.





□

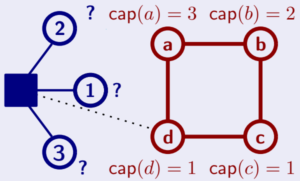
Key Insights

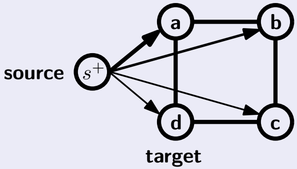
Lemma

We can solve the embedding problem if the logical switch is placed.

Proof.

1. Construct extended graph with additional edges $\{(s^+, u) | u \in V_S\}$, $\text{cap}(s^+, u) = \text{cap}(u)$ and $\text{cost}(s^+, u) = \text{cost}(u)$ for $u \in V_S$
2. Compute a minimum cost flow of value $\mathcal{N}$ from s^+ to $\text{map}_V(\text{center})$.
3. Perform path-decomposition to obtain mapping for *nodes and edges*.





□

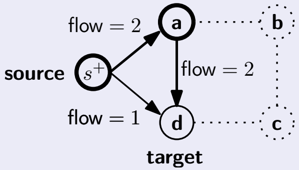
Key Insights

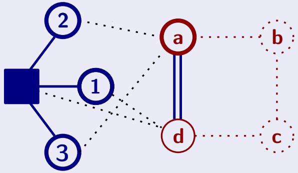
Lemma

We can solve the embedding problem if the logical switch is placed.

Proof.

1. Construct extended graph with additional edges $\{(s^+, u) | u \in V_S\}$, $\text{cap}(s^+, u) = \text{cap}(u)$ and $\text{cost}(s^+, u) = \text{cost}(u)$ for $u \in V_S$
2. Compute a minimum cost flow of value $\mathcal{N}$ from s^+ to $\text{map}_V(\text{center})$.
3. Perform path-decomposition to obtain mapping for *nodes and edges*.





□

VC-ACE Algorithm

Idea

Simply iterate over possible locations for the center.

Algorithm 1: The VC-ACE Algorithm

Input: Substrate $S = (V_S, E_S)$, request $(\mathcal{N}, \mathcal{B}, \mathcal{C})$
Output: Optimal VC mapping $\text{map}_V, \text{map}_E$ if feasible

```
$(\hat{f}, \hat{v}) \leftarrow (\texttt{null}, \texttt{null})$
for $v \in V_S$ do
    $V_{S'} = V_S \cup \{s^+\}$ and
    $E_{S'} = E_S \cup \{(s^+, u) | u \in V_S\}$
    $\text{cap}_{S'}(e) = \begin{cases} \lfloor \text{cap}(e)/\mathcal{B} \rfloor & \text{, if } e \in E_S \\ \lfloor \text{cap}(u)/\mathcal{C} \rfloor & \text{, if } e = (s^+, u) \in E_S \end{cases}$
    $\text{cost}_{S'}(e) = \begin{cases} \text{cost}(e) \cdot \mathcal{B} & \text{, if } e \in E_S \\ \text{cost}(u) \cdot \mathcal{C} & \text{, if } e = (s^+, u) \in E_S \end{cases}$
    $f \leftarrow$
    $\texttt{MinCostFlow}(s^+, v, \mathcal{N}, V_{S'}, E_{S'}, \text{cap}_{S'}, \text{cost}_{S'})$
    if $f$ is feasible and $\text{cost}(f) < \text{cost}(\hat{f})$ then
        $(\hat{f}, \hat{v}) \leftarrow (f, v)$
if $\hat{f} = \texttt{null}$ then
    return null
return $\texttt{DecomposeFlowIntoMapping}(\hat{f}, \hat{v})$
```

VC-ACE Algorithm

Idea

Simply iterate over possible locations for the center.

Theorem

Correctness follows from the lemma on the previous slide.

Algorithm 2: The VC-ACE Algorithm

Input: Substrate $S = (V_S, E_S)$, request $(\mathcal{N}, \mathcal{B}, \mathcal{C})$
Output: Optimal VC mapping $\text{map}_V, \text{map}_E$ if feasible

```
$(\hat{f}, \hat{v}) \leftarrow (\texttt{null}, \texttt{null})$
for $v \in V_S$ do
    $V_{S'} = V_S \cup \{s^+\}$  and
    $E_{S'} = E_S \cup \{(s^+, u) | u \in V_S\}$
    $\text{caps}_{S'}(e) = \begin{cases} \lfloor \text{cap}(e)/\mathcal{B} \rfloor & \text{, if } e \in E_S \\ \lfloor \text{cap}(u)/\mathcal{C} \rfloor & \text{, if } e = (s^+, u) \in E_S \end{cases}$
    $\text{cost}_{S'}(e) = \begin{cases} \text{cost}(e) \cdot \mathcal{B} & \text{, if } e \in E_S \\ \text{cost}(u) \cdot \mathcal{C} & \text{, if } e = (s^+, u) \in E_S \end{cases}$
    $f \leftarrow \texttt{MinCostFlow}(s^+, v, \mathcal{N}, V_{S'}, E_{S'}, \text{caps}_{S'}, \text{costs}_{S'})$
    if $f$ is feasible and $\text{cost}(f) < \text{cost}(\hat{f})$ then
        $(\hat{f}, \hat{v}) \leftarrow (f, v)$
if $\hat{f} = \texttt{null}$ then
    return null
return $\texttt{DecomposeFlowIntoMapping}(\hat{f}, \hat{v})$
```

VC-ACE Algorithm

Theorem

The runtime is $\mathcal{O}\left(\mathcal{N}(n^2 \log n + n \cdot m)\right)$ with $n = |V_S|$ and $m = |E_S|$, when using the successive-shortest path for the flow computation.

Corollary.

The VC Embedding Problem is *not* NP-hard. □

Algorithm 3: The VC-ACE Algorithm

Input: Substrate $S = (V_S, E_S)$, request $(\mathcal{N}, \mathcal{B}, \mathcal{C})$
Output: Optimal VC mapping $\text{map}_V, \text{map}_E$ if feasible

```
$(\hat{f}, \hat{v}) \leftarrow (\texttt{null}, \texttt{null})$
for $v \in V_S$ do
    $V_{S'} = V_S \cup \{s^+\}$ and
    $E_{S'} = E_S \cup \{(s^+, u) | u \in V_S\}$
    $\text{caps}_{S'}(e) = \begin{cases} \lfloor \text{cap}(e)/\mathcal{B} \rfloor & \text{, if } e \in E_S \\ \lfloor \text{cap}(u)/\mathcal{C} \rfloor & \text{, if } e = (s^+, u) \in E_S \end{cases}$
    $\text{cost}_{S'}(e) = \begin{cases} \text{cost}(e) \cdot \mathcal{B} & \text{, if } e \in E_S \\ \text{cost}(u) \cdot \mathcal{C} & \text{, if } e = (s^+, u) \in E_S \end{cases}$
    $f \leftarrow$ MinCostFlow($s^+, v, \mathcal{N}, V_{S'}, E_{S'}, \text{caps}_{S'}, \text{cost}_{S'}$)
    if $f$ is feasible and $\text{cost}(f) < \text{cost}(\hat{f})$ then
        $(\hat{f}, \hat{v}) \leftarrow (f, v)$
if $\hat{f} = \texttt{null}$ then
    return null
return DecomposeFlowIntoMapping($\hat{f}, \hat{v}$)
```

We can compute optimal solutions in polynomial-time.

We can compute optimal solutions in polynomial-time.

Can we do even better?

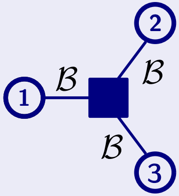
Can we do even better?

Yes, . . .

Hose-Based Virtual Cluster Embeddings

Starting from Scratch

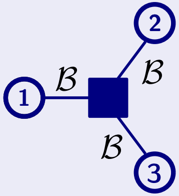
VC Abstraction



- Allows for any traffic matrix M, where for any VM the sum of outgoing and incoming traffic is less than $\mathcal{B}$

Starting from Scratch

VC Abstraction



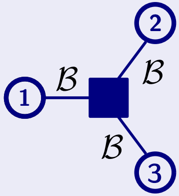
- Allows for any traffic matrix M, where for any VM the sum of outgoing and incoming traffic is less than $\mathcal{B}$

Question:

What is the purpose of the switch?

Starting from Scratch

VC Abstraction



- Allows for any traffic matrix M, where for any VM the sum of outgoing and incoming traffic is less than $\mathcal{B}$

Question:

What is the purpose of the switch?

Ballani et al. 'Oktopus' [3]

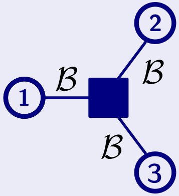
"'Oktopus' allocation algorithms assume that the traffic between a tenant's VMs is routed along a tree."

Answer:

To route the traffic along a tree.

Starting from Scratch

VC Abstraction



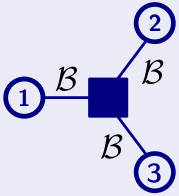
- Allows for any traffic matrix M, where for any VM the sum of outgoing and incoming traffic is less than $\mathcal{B}$

Question:

Can we do without the switch?

Starting from Scratch

VC Abstraction



- Allows for any traffic matrix M, where for any VM the sum of outgoing and incoming traffic is less than $\mathcal{B}$

Question:

Can we do without the switch?

Ballani et al. 'Oktopus' [3]

"Alternatively, the NM [Network Manager] can control datacenter routing to actively build routes between tenant VMs [..]"

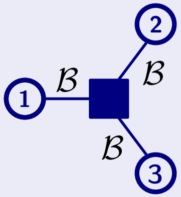
"We defer a detailed study of the relative merits of these approaches to future work."

Answer:

Yes!

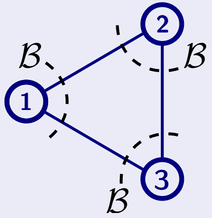
Starting from Scratch

VC Abstraction



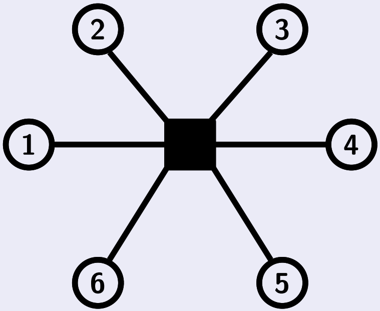
- Allows for any traffic matrix M, where for any VM the sum of outgoing and incoming traffic is less than $\mathcal{B}$

Hose-Based VC Abstraction

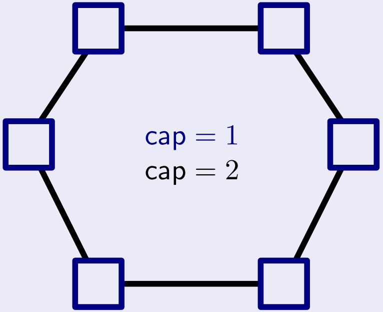


- Allows for any traffic matrix M, where for any VM the sum of outgoing and incoming traffic is less than $\mathcal{B}$

Motivating Example

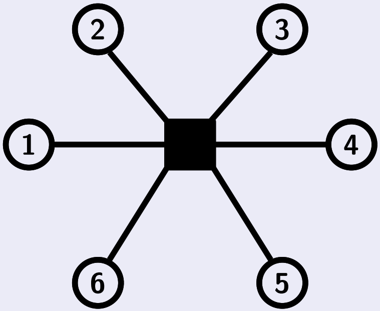


$\mathcal{N} = 6, \ \mathcal{B} = \mathcal{C} = 1$

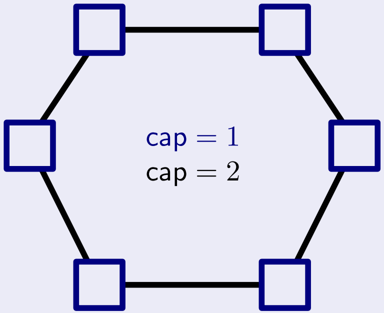


ring substrate

Motivating Example



$\mathcal{N} = 6,\ \mathcal{B} = \mathcal{C} = 1$

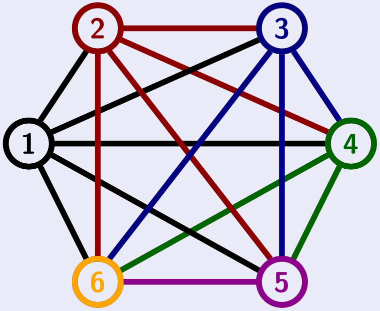


ring substrate

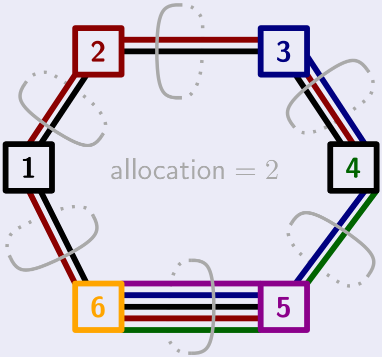
There exists no solution ...

... in the classic VC embedding model.

Motivating Example

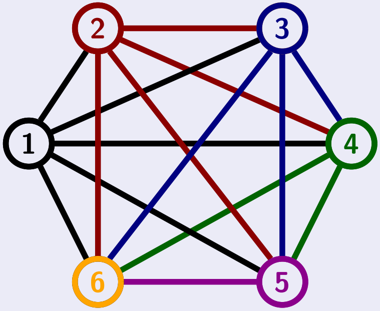


$\mathcal{N} = 6,\ \mathcal{B} = \mathcal{C} = 1$

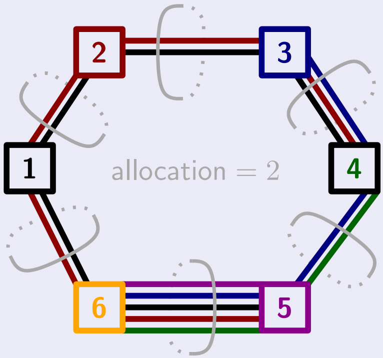


Embedding on the same substrate

Motivating Example



$\mathcal{N} = 6, \ \mathcal{B} = \mathcal{C} = 1$

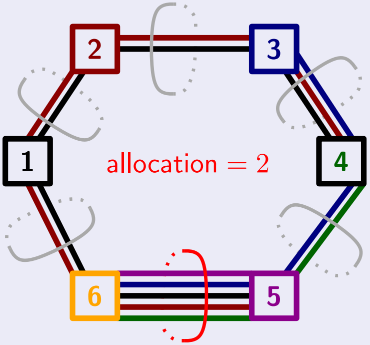


Embedding on the same substrate

There exists a solution ...

... in the hose-based VC model!

Motivating Example

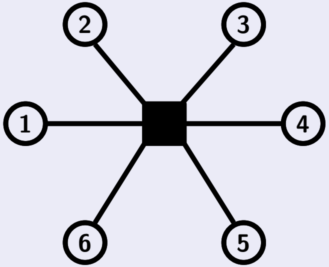


Embedding on the same substrate

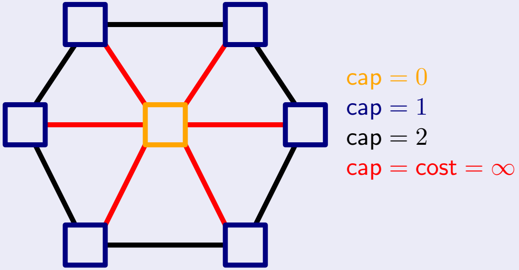
Why allocations of 2 are sufficient:

- Consider edge e between VMs 6 and 5.
- The edge is used by routes $R(e) = \{(1,5),(2,5),(3,6),(4,6),(5,6)\}$.
- Any valid traffic matrix M will respect:
  - $M_{1,5} + M_{2,5} \leq 1$
  - $M_{3,6} + M_{4,6} + M_{5,6} \leq 1$
- Hence $\sum_{(i,j)\in R(e)} M_{i,j} \leq 2$ holds.

Motivating Example II



$\mathcal{N} = 6, \ \mathcal{B} = \mathcal{C} = 1$



extended ring topology

Solution costs …

…can be arbitrarily higher under the classic star-interpretation!

Hose-Based Virtual Cluster Embedding Problem

Hose-Based VC Embedding Problem (HVCEP)

Definition (Clique Graph)

- $V_C = \{1, \dots, \mathcal{N}\}$, $E_C = \{(i,j) | i,j \in V_C, i < j\}$

Task: Find a mapping of . . .

- VC nodes onto substrate nodes $\text{map}_V : V_C \to V_S$, and
- VC routes onto paths in the substrate $\text{map}_E : E_C \to \mathcal{P}(E_S)$, such that

1. *route* $(i,j) \in E_C$ connects $\text{map}_V(i)$ and $\text{map}_V(j)$,
2. the mapping of VMs must not violate node capacities (cf. slide 12),

Hose-Based VC Embedding Problem (HVCEP)

Definition (Clique Graph)

- $V_C = \{1, \dots, \mathcal{N}\}$, $E_C = \{(i,j) | i,j \in V_C, i < j\}$

Task: Find a mapping of . . .

- VC nodes onto substrate nodes $\text{map}_V : V_C \to V_S$, and
- VC routes onto paths in the substrate $\text{map}_E : E_C \to \mathcal{P}(E_S)$, and
- integral bandwidth reservations $l_{u,v} \leq \text{cap}(u,v)$ for $\{u,v\} \in E_S$, s.t.
  1. *route* $(i,j) \in E_C$ connects $\text{map}_V(i)$ and $\text{map}_V(j)$,
  2. the mapping of VMs must not violate node capacities (cf. slide 12),
  3. for all valid traffic matrices M_{ij} – i.e. $\sum_{(j,i) \in \boldsymbol{E_C}} M_{ji} + M_{ij} \leq \mathcal{B}$ holds – the bandwidth reservation is not exceeded on any edge $\{u,v\} \in E_S$: $\sum_{\{i,j\} \in \boldsymbol{E_C} : \{u,v\} \in \text{map}_{\boldsymbol{E}}(\{i,j\})} M_{ij} \leq l_{u,v}$,

Hose-Based VC Embedding Problem (HVCEP)

Definition (Clique Graph)

- $V_C = \{1, \dots, \mathcal{N}\}$, $E_C = \{(i,j) | i,j \in V_C, i < j\}$

Task: Find a mapping of ...

- VC nodes onto substrate nodes $\mathsf{map}_V : V_C \to V_\mathsf{S}$, and
- VC routes onto paths in the substrate $\mathsf{map}_E : E_C \to \mathcal{P}(E_\mathsf{S})$, and
- integral bandwidth reservations $l_{u,v} \leq \mathsf{cap}(u,v)$ for $\{u,v\} \in E_\mathsf{S}$, such that

  1. *route* $(i,j) \in E_C$ connects $\mathsf{map}_V(i)$ and $\mathsf{map}_V(j)$,
  2. the mapping of VMs must not violate node capacities (cf. slide 12),
  3. for all valid traffic matrices M – i.e. $\sum_{(j,i) \in E_\mathbf{C}} M_{ji} + M_{ij} \leq \mathcal{B}$ holds – the bandwidth reservation is not exceeded on any edge $\{u,v\} \in E_\mathsf{S}$: $\sum_{\{i,j\} \in E_\mathbf{C} : \{u,v\} \in \mathsf{map}_\mathbf{E}(\{i,j\})} M_{ij} \leq l_{u,v}$,
  4. minimizing $\mathcal{C} \cdot \sum_{i \in V_\mathbf{C}} \mathsf{cost}(\mathsf{map}_V(i)) + \mathcal{B} \cdot \sum_{e \in E_\mathbf{S}} l_{u,v} \cdot \mathsf{cost}(e)$.

Computational Complexity of HVC Embeddings

Excursion: VPN Embeddings and the VPN Conjecture

Definition (VPN Embedding Problem (VPNEP) [7])

Given:
- Substrate network $G = (V, E)$ with edge costs $\text{cost} : E \rightarrow \mathbb{R}_0^+$
- Set of terminals $W \subseteq V$ with demands $b(i) \in \mathbb{R}^+$ for $i \in W$

Task:
- Find paths $P_{\{i,j\}}$ for all pairs $i, j \in W$, $i \neq j$, and
- bandwidth allocations $x_e \in \mathbb{R}_0^+$ on edges $e \in E$, s.t.
- $\sum_{i,j \in W : i \neq j, P_{\{i,j\}}} M_{\{i,j\}} \leq x_e$ holds for traffic matrices M,
- minimizing the cost $\sum_{e \in E} \text{cost}(e) \cdot x_e$.

Excursion: VPN Embeddings and the VPN Conjecture

Definition (VPN Embedding Problem (VPNEP) [7])

Given:
- Substrate network $G = (V, E)$ with edge costs $\mathrm{cost} : E \to \mathbb{R}_0^+$
- Set of terminals $W \subseteq V$ with demands $b(i) \in \mathbb{R}^+$ for $i \in W$

Task:
- Find paths $P_{\{i,j\}}$ for all pairs $i, j \in W$, $i \neq j$, and
- bandwidth allocations $x_e \in \mathbb{R}_0^+$ on edges $e \in E$, s.t.
- $\sum_{i,j \in W: i \neq j, P_{\{i,j\}}} M_{\{i,j\}} \leq x_e$ holds for traffic matrices M,
- minimizing the cost $\sum_{e \in E} \mathrm{cost}(e) \cdot x_e$.

Theorem

Finding a feasible solution for the capacitated VPNEP is NP-hard [7].

Excursion: VPN Embeddings and the VPN Conjecture

Definition (VPN Embedding Problem (VPNEP) [7])

Given:
- Substrate network $G = (V, E)$ with edge costs $\text{cost} : E \to \mathbb{R}_0^+$
- Set of terminals $W \subseteq V$ with demands $b(i) \in \mathbb{R}^+$ for $i \in W$

Task:
- Find paths $P_{\{i,j\}}$ for all pairs $i, j \in W$, $i \neq j$, and
- bandwidth allocations $x_e \in \mathbb{R}_0^+$ on edges $e \in E$, s.t.
- $\sum_{i,j \in W : i \neq j, P_{\{i,j\}}} M_{\{i,j\}} \leq x_e$ holds for traffic matrices M,
- minimizing the cost $\sum_{e \in E} \text{cost}(e) \cdot x_e$.

Theorem

Finding a feasible solution for the capacitated VPNEP is NP-hard [7].

Theorem (By reduction from the VPNEP)

Finding a feasible solution for the HVCEP is NP-hard.

Excursion: VPN Embeddings and the VPN Conjecture

Definition (VPN Embedding Problem (VPNEP) [7])

Given:
- Substrate network $G = (V, E)$ with edge costs $\text{cost} : E \to \mathbb{R}_0^+$
- Set of terminals $W \subseteq V$ with demands $b(i) \in \mathbb{R}^+$ for $i \in W$

Task:
- Find paths $P_{\{i,j\}}$ for all pairs $i, j \in W$, $i \neq j$, and
- bandwidth allocations $x_e \in \mathbb{R}_0^+$ on edges $e \in E$, s.t.
- $\sum_{i,j \in W : i \neq j, P_{\{i,j\}}} M_{\{i,j\}} \leq x_e$ holds for traffic matrices M,
- minimizing the cost $\sum_{e \in E} \text{cost}(e) \cdot x_e$.

Theorem (VPN Conjecture)

Tree routing and arbitrary routing solutions coincide for the VPNEP on uncapacitated graphs. [5].

Excursion: VPN Embeddings and the VPN Conjecture

Definition (VPN Embedding Problem (VPNEP) [7])

Given:
- Substrate network $G = (V, E)$ with edge costs $\text{cost} : E \to \mathbb{R}_0^+$
- Set of terminals $W \subseteq V$ with demands $b(i) \in \mathbb{R}^+$ for $i \in W$

Task:
- Find paths $P_{\{i,j\}}$ for all pairs $i, j \in W$, $i \neq j$, and
- bandwidth allocations $x_e \in \mathbb{R}_0^+$ on edges $e \in E$, s.t.
- $\sum_{i,j \in W : i \neq j, P_{\{i,j\}}} M_{\{i,j\}} \leq x_e$ holds for traffic matrices M,
- minimizing the cost $\sum_{e \in E} \text{cost}(e) \cdot x_e$.

Theorem (VPN Conjecture)

Tree routing and arbitrary routing solutions coincide for the VPNEP on uncapacitated graphs. [5].

Theorem (Via the VPN Conjecture)

Algorithm VC-ACE solves the HVCEP when capacities are sufficiently large!

Computing (Fractional) HVC Embeddings

A Mixed-Integer Programming Formulation for the HVCEP

Variables

x_u^i	mapping of VM i onto node u
$y_{u.v}^{i,j}$	mapping of link $(i,j) \in V_C$ onto (directed) substrate edge (u,v)
$l_{u.v}$	load on substrate edge $\{u,v\}$
ω_{uv}^i	'dual variable' for allocation of communications of VM i on edge $\{u,v\}$

Mixed-Integer Program 1: HVC-OSPE

$$\min \sum_{i\in V_C, u\in V_S} \text{cost}_u \cdot x_u^i + \sum_{\{u,v\}\in E_S} \text{cost}_{u,v} \cdot l_{uv} \tag{1}$$

$$\sum_{u\in V_S} x_u^i = 1 \qquad \forall i \in V_C. \tag{2}$$

$$\sum_{u\in V_S} \sigma_u \cdot (x_u^i - x_u^{i+1}) \leq 0 \qquad \forall i \in V_C \setminus \{\mathcal{N}\}. \tag{3}$$

$$\sum_{i\in V_C} \mathcal{C} \cdot x_u^i \leq \text{cap}_u \qquad \forall u \in V_S. \tag{4}$$

$$l_{uv} \leq \text{cap}_{uv} \qquad \forall \{u,v\} \in E_S. \tag{5}$$

$$\sum_{(u,v)\in\delta_u^+} y_{uv}^{ij} - \sum_{(v,u)\in\delta_u^-} y_{vu}^{ij} = x_u^i - x_u^j \qquad \forall (i,j) \in E_C, \forall u \in V_S. \tag{6}$$

$$\sum_{i\in V_C} \mathcal{B} \cdot \omega_{uv}^i \leq l_{uv} \qquad \forall \{u,v\} \in E_S. \tag{7}$$

$$y_{uv}^{ij} + y_{vu}^{ij} \leq \omega_{uv}^i + \omega_{uv}^j \qquad \forall (i,j) \in E_C, \forall \{u,v\} \in E_S. \tag{8}$$

A Mixed-Integer Programming Formulation for the HVCEP

Explanation

- (2) - (4) control the VM embedding
- (5) - (8) is adapted from Altin et al. [2] for computing the optimal hose allocations on edges

Mixed-Integer Program 2: HVC-OSPE

$$\min \sum_{i\in V_C, u\in V_S} \text{cost}_u \cdot x_u^i + \sum_{\{u,v\}\in E_S} \text{cost}_{u,v} \cdot l_{uv} \tag{1}$$

$$\sum_{u\in V_S} x_u^i = 1 \qquad \forall i \in V_C. \tag{2}$$

$$\sum_{u\in V_S} \sigma_u \cdot (x_u^i - x_u^{i+1}) \leq 0 \qquad \forall i \in V_C \setminus \{\mathcal{N}\}. \tag{3}$$

$$\sum_{i\in V_C} \mathcal{C} \cdot x_u^i \leq \text{cap}_u \qquad \forall u \in V_S. \tag{4}$$

$$l_{uv} \leq \text{cap}_{uv} \qquad \forall \{u,v\} \in E_S. \tag{5}$$

$$\sum_{(u,v)\in\delta_u^+} y_{uv}^{ij} - \sum_{(v,u)\in\delta_u^-} y_{vu}^{ij} = x_u^i - x_u^j \qquad \forall (i,j) \in E_C, \forall u \in V_S. \tag{6}$$

$$\sum_{i\in V_C} \mathcal{B} \cdot \omega_{uv}^i \leq l_{uv} \qquad \forall \{u,v\} \in E_S. \tag{7}$$

$$y_{uv}^{ij} + y_{vu}^{ij} \leq \omega_{uv}^i + \omega_{uv}^j \qquad \forall (i,j) \in E_C, \forall \{u,v\} \in E_S. \tag{8}$$

A Mixed-Integer Programming Formulation for the HVCEP

Explanation

- (2) - (4) control the VM embedding
- (5) - (8) is adapted from Altin et al. [2] for computing the optimal hose allocations on edges

Observation

There are $\mathcal{O}(\mathcal{N}^2 \cdot |E_S|)$ binary variables for computing paths.

Mixed-Integer Program 3: HVC-OSPE

$$\min \sum_{i \in V_C, u \in V_S} \mathrm{cost}_u \cdot x_u^i + \sum_{\{u,v\} \in E_S} \mathrm{cost}_{u,v} \cdot l_{uv} \quad (1)$$

$$\sum_{u \in V_S} x_u^i = 1 \qquad \forall i \in V_C. \quad (2)$$

$$\sum_{u \in V_S} \sigma_u \cdot (x_u^i - x_u^{i+1}) \leq 0 \qquad \forall i \in V_C \setminus \{\mathcal{N}\}. \quad (3)$$

$$\sum_{i \in V_C} \mathcal{C} \cdot x_u^i \leq \mathrm{cap}_u \qquad \forall u \in V_S. \quad (4)$$

$$l_{uv} \leq \mathrm{cap}_{uv} \qquad \forall \{u,v\} \in E_S. \quad (5)$$

$$\sum_{(u,v) \in \delta_u^+} y_{uv}^{ij} - \sum_{(v,u) \in \delta_u^-} y_{vu}^{ij} = x_u^i - x_u^j \qquad \forall (i,j) \in E_C, \forall u \in V_S. \quad (6)$$

$$\sum_{i \in V_C} \mathcal{B} \cdot \omega_{uv}^i \leq l_{uv} \qquad \forall \{u,v\} \in E_S. \quad (7)$$

$$y_{uv}^{ij} + y_{vu}^{ij} \leq \omega_{uv}^i + \omega_{uv}^j \qquad \forall (i,j) \in E_C, \forall \{u,v\} \in E_S. \quad (8)$$

A Mixed-Integer Programming Formulation for the HVCEP

Observation

There are $\mathcal{O}(\mathcal{N}^2 \cdot |E_S|)$ binary variables for computing paths.

Initial Computational Results

Solving this formulation may take up to 1800 seconds for embedding a 10-VM VC onto a 20 node substrate.

Mixed-Integer Program 4: HVC-OSPE

$$\min \sum_{i \in V_C, u \in V_S} \text{cost}_u \cdot x_u^i + \sum_{\{u,v\} \in E_S} \text{cost}_{u,v} \cdot l_{uv} \tag{1}$$

$$\sum_{u \in V_S} x_u^i = 1 \qquad \forall i \in V_C. \tag{2}$$

$$\sum_{u \in V_S} \sigma_u \cdot (x_u^i - x_u^{i+1}) \leq 0 \qquad \forall i \in V_C \setminus \{\mathcal{N}\}. \tag{3}$$

$$\sum_{i \in V_C} \mathcal{C} \cdot x_u^i \leq \text{cap}_u \qquad \forall u \in V_S. \tag{4}$$

$$l_{uv} \leq \text{cap}_{uv} \qquad \forall \{u,v\} \in E_S. \tag{5}$$

$$\sum_{(u,v) \in \delta_u^+} y_{uv}^{ij} - \sum_{(v,u) \in \delta_u^-} y_{vu}^{ij} = x_u^i - x_u^j \qquad \forall (i,j) \in E_C, \forall u \in V_S. \tag{6}$$

$$\sum_{i \in V_C} \mathcal{B} \cdot \omega_{uv}^i \leq l_{uv} \qquad \forall \{u,v\} \in E_S. \tag{7}$$

$$y_{uv}^{ij} + y_{vu}^{ij} \leq \omega_{uv}^i + \omega_{uv}^j \qquad \forall (i,j) \in E_C, \forall \{u,v\} \in E_S. \tag{8}$$

A Mixed-Integer Programming Formulation for the HVCEP

Further Observations

- The hardness result has shown that the problem is hard even if the VMs are fixed.
- The large number of variables necessary for computing each end-to-end path between VMs renders solving even the linear relaxation – i.e. dropping integrality constraints – computationally hard.

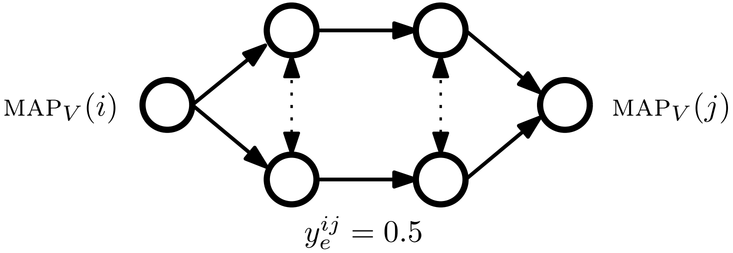
Assumptions for obtaining a ‘solvable’ formulation

- Assume that the VMs are already mapped.
- Assume that the hose-paths are *splittable*, i.e. each VMs are connected by a set of (weighted) paths.

Computing Splittable HVC Embeddings

Assumptions

- Assume that the VMs are already mapped.
- Assume that the hose-paths are *splittable*. arbitrarily many paths.



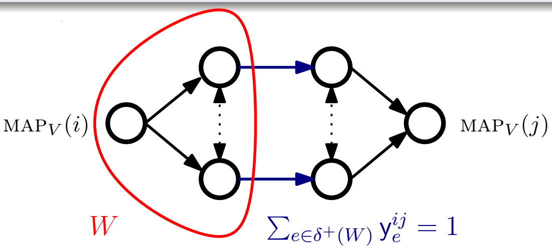
Computing Splittable HVC Embeddings

$$l_{uv} \leq \mathsf{cap}_{uv} \qquad \forall \{u,v\} \in E_S. \quad (5)$$

$$\sum_{(u,v)\in\delta_u^+} y_{uv}^{ij} - \sum_{(v,u)\in\delta_u^-} y_{vu}^{ij} = x_u^i - x_u^j \qquad \forall (i,j) \in E_C, \ \forall u \in V_S. \quad (6)$$

$$\sum_{i\in V_C} \mathcal{B} \cdot \omega_{uv}^i \leq l_{uv} \qquad \forall \{u,v\} \in E_S. \quad (7)$$

$$y_{uv}^{ij} + y_{vu}^{ij} \leq \omega_{uv}^i + \omega_{uv}^j \qquad \forall (i,j) \in E_C, \ \forall \{u,v\} \in E_S. \quad (8)$$



This type of constraint is equivalent to (6).

Computing Splittable HVC Embeddings

$$l_{uv} \leq \mathsf{cap}_{uv} \qquad \forall \{u,v\} \in E_S. \tag{5}$$

$$\sum_{(u,v)\in\delta^+_u} y^{ij}_{uv} - \sum_{(v,u)\in\delta^-_u} y^{ij}_{vu} = x^i_u - x^j_u \qquad \forall (i,j) \in E_C, \forall u \in V_S. \tag{6}$$

$$\sum_{i \in V_C} \mathcal{B} \cdot \omega^i_{uv} \leq l_{uv} \qquad \forall \{u,v\} \in E_S. \tag{7}$$

$$y^{ij}_{uv} + y^{ij}_{vu} \leq \omega^i_{uv} + \omega^j_{uv} \qquad \forall (i,j) \in E_C, \forall \{u,v\} \in E_S. \tag{8}$$

$$\sum_{(u,v)\in\delta^+(W)} y^{ij}_{uv} \geq 1 \qquad \forall (i,j) \in E_C. \forall W \subset V_S : \mathsf{map}_V(i) \in W, \mathsf{map}_V(j) \notin W \tag{6$\star$}$$

Derivation of a new constraint

- Across a cut W, the amount of flow must be greater than 1 (6$\star$).
- By summing up Constraints (8) accordingly, we obtain for $(i,j) \in E_C$ and across any node set $\forall W \subset V_S$ with $\mathsf{map}_V(i) \in W$ and $\mathsf{map}_V(j) \notin W$ that $\sum_{(u,v)\in\delta^+(W)} (\omega^i_{uv} + \omega^j_{uv}) \geq 1$ holds.

Computing Splittable HVC Embeddings

$$l_{uv} \leq \mathrm{cap}_{uv} \qquad \forall \{u,v\} \in E_S. \tag{5}$$

$$\sum_{(u,v)\in\delta_u^+} y_{uv}^{ij} - \sum_{(v,u)\in\delta_u^-} y_{vu}^{ij} = x_u^i - x_u^j \qquad \forall (i,j) \in E_C, \ \forall u \in V_S. \tag{6}$$

$$\sum_{i \in V_C} \mathcal{B} \cdot \omega_{uv}^i \leq l_{uv} \qquad \forall \{u,v\} \in E_S. \tag{7}$$

$$y_{uv}^{ij} + y_{vu}^{ij} \leq \omega_{uv}^i + \omega_{uv}^j \qquad \forall (i,j) \in E_C, \ \forall \{u,v\} \in E_S. \tag{8}$$

$$\sum_{(u,v)\in\delta^+(W)} y_{uv}^{ij} \geq 1 \qquad \forall (i,j) \in E_C . \forall W \subset V_S : \mathrm{map}_V(i) \in W, \ \mathrm{map}_V(j) \notin W \tag{6$\star$}$$

$$\sum_{(u,v)\in\delta^+(W)} (\omega_{uv}^i + \omega_{uv}^j) \geq 1 \qquad \forall (i,j) \in E_C . \forall W \subset V_S : \mathrm{map}_V(i) \in W, \ \mathrm{map}_V(j) \notin W \tag{9}$$

Remarks

- Given (9), we can always (re-)construct the flow variables y_{uv}^{ij} afterwards by breadth-first searches.
- Furthermore, this property does not depend on (6$\star$).

Computing Splittable HVC Embeddings

$$l_{uv} \leq \mathrm{cap}_{uv} \qquad \forall \{u,v\} \in E_S. \tag{5}$$

$$\sum_{(u,v)\in\delta_u^+} y_{uv}^{ij} - \sum_{(v,u)\in\delta_u^-} y_{vu}^{ij} = x_u^i - x_u^j \qquad \forall (i,j) \in E_C, \; \forall u \in V_S. \tag{6}$$

$$\sum_{i\in V_C} \mathcal{B} \cdot \omega_{uv}^i \leq l_{uv} \qquad \forall \{u,v\} \in E_S. \tag{7}$$

$$y_{uv}^{ij} + y_{vu}^{ij} \leq \omega_{uv}^i + \omega_{uv}^j \qquad \forall (i,j) \in E_C, \; \forall \{u,v\} \in E_S. \tag{8}$$

$$\sum_{(u,v)\in\delta^+(W)} y_{uv}^{ij} \geq 1 \qquad \forall (i,j) \in E_C. \forall W \subset V_S : \mathrm{map}_V(i) \in W, \; \mathrm{map}_V(j) \notin W \tag{6$\star$}$$

$$\sum_{(u,v)\in\delta^+(W)} (\omega_{uv}^i + \omega_{uv}^j) \geq 1 \qquad \forall (i,j) \in E_C. \forall W \subset V_S : \mathrm{map}_V(i) \in W, \; \mathrm{map}_V(j) \notin W \tag{9}$$

Remarks

- Therfore Constraints (6), (8), and (6$\star$) are not needed anymore!

Computing Splittable HVC Embeddings

$$l_{uv} \leq \mathrm{cap}_{uv} \qquad \forall \{u,v\} \in E_S. \quad (6)$$

$$\sum_{(u,v)\in\delta_u^+} y^{ij}_{uv} - \sum_{(v,u)\in\delta_u^-} y^{ij}_{vu} = x^i_u - x^j_u \qquad \forall (i,j) \in E_C, \forall u \in V_S. \quad (7)$$

$$\sum_{i\in V_C} \mathcal{B} \cdot \omega^i_{uv} \leq l_{uv} \qquad \forall \{u,v\} \in E_S. \quad (8)$$

$$y^{ij}_{uv} + y^{ij}_{vu} \leq \omega^i_{uv} + \omega^j_{uv} \qquad \forall (i,j) \in E_C, \forall \{u,v\} \in E_S. \quad (9)$$

Algorithm 5: HMPR

$$\min \sum_{\{u,v\}\in E_S} \mathrm{cost}_{u,v} \cdot l_{uv} \quad (10)$$

$$l_{uv} \leq \mathrm{cap}_{uv} \; \forall \{u,v\} \in E_S. \quad (11)$$

$$\sum_{i\in V_C} \mathcal{B} \cdot \omega^i_{uv} \leq l_{uv} \qquad \forall \{u,v\} \in E_S. \quad (12)$$

$$\sum_{(u,v)\in\delta^+(W)} (\omega^i_{uv} + \omega^j_{uv}) \geq 1 \qquad \forall (i,j) \in E_C . \forall W \subset V_S : \mathrm{map}_V(i) \in W, \mathrm{map}_V(j) \notin W \quad (13)$$

Computing Splittable HVC Embeddings

Algorithm 6: HMPR

$$\min \sum_{\{u,v\} \in E_S} \text{cost}_{u,v} \cdot l_{uv} \tag{10}$$

$$l_{uv} \leq \text{cap}_{uv} \quad \forall \{u,v\} \in E_S. \tag{11}$$

$$\sum_{i \in V_C} \mathcal{B} \cdot \omega^i_{uv} \leq l_{uv} \quad \forall \{u,v\} \in E_S. \tag{12}$$

$$\sum_{(u,v) \in \delta^+(W)} (\omega^i_{uv} + \omega^j_{uv}) \geq 1 \quad \forall (i,j) \in E_C . \forall W \subset V_S : \text{map}_V(i) \in W, \ \text{map}_V(j) \notin W \tag{13}$$

Exponential number of constraints, ...

...which can be separated in polynomial time.

Computing Splittable HVC Embeddings

Algorithm 7: HMPR

$$\min \sum_{\{u,v\} \in E_S} \mathrm{cost}_{u,v} \cdot l_{uv} \tag{10}$$

$$l_{uv} \leq \mathrm{cap}_{uv} \quad \forall \{u,v\} \in E_S. \tag{11}$$

$$\sum_{i \in V_C} \mathcal{B} \cdot \omega^i_{uv} \leq l_{uv} \quad \forall \{u,v\} \in E_S. \tag{12}$$

$$\sum_{(u,v) \in \delta^+(W)} (\omega^i_{uv} + \omega^j_{uv}) \geq 1 \quad \forall (i,j) \in E_C . \forall W \subset V_S : \mathrm{map}_V(i) \in W, \mathrm{map}_V(j) \notin W \tag{13}$$

Exponential number of constraints, . . .

. . . which can be separated in polynomial time.

Number of variables, . . .

. . . in the order of $\mathcal{O}(\mathcal{N} \cdot |E_S|)$ instead of $\mathcal{O}(\mathcal{N}^2 \cdot |E_S|)$.

Computing Splittable HVC Embeddings

We can compute fractional edge embeddings, . . .

. . . but how to find node locations?

Computing Splittable HVC Embeddings

Heuristic Idea

- without capacities: "VC = HVC"
- reuse VC-ACE algorithm, but allow violation of capacities w.r.t. VC model
- violating capacities induces k times the cost of the original edge

Algorithm 5: The HVC-ACE Embedding Heuristic

```
Input:  Substrate $S = (V_S, E_S)$, request
        VC$(\mathcal{N}, \mathcal{B}, \mathcal{C})$,
          cost factor $k \geq 1$
Output: Splittable HVC-Embedding $\text{map}_V, \text{map}_E$
$E_{S'} \leftarrow \emptyset$
for $e \in E_S$ do
    $E_{S'} = E_{S'} \sqcup \{e, e'\}$
    $\text{cap}_{S'}(e) = \text{cap}(e)$ and $\text{cap}_{S'}(e') = \infty$
    $\text{cost}_{S'}(e) = \text{cost}(e)$ and $\text{cost}_{S'}(e') = \text{cost}(e) \cdot k$
$\text{map}_V, \text{map}_E \leftarrow \texttt{VC-ACE}(V_S, E_{S'}, \text{VC}(\mathcal{N}, \mathcal{B}, \mathcal{C}))$
if $\text{map}_V \neq \texttt{null}$ then
    $\text{map}_E \leftarrow \texttt{HMPR}(\text{VC}(\mathcal{N}, \mathcal{B}, \mathcal{C}), \text{map}_V)$
    if $\text{map}_E \neq \texttt{null}$ then
        return $\text{map}_V, \text{map}_E$
return null
```

Computational Evaluation

What do we get by using HVC-ACE?

Setup

Topologies

- Fat trees with 12 port switches and 432 server overall
- MDCubes consisting of 4 BCubes with 12 port switches and k = 1, such that the topology contains 576 server

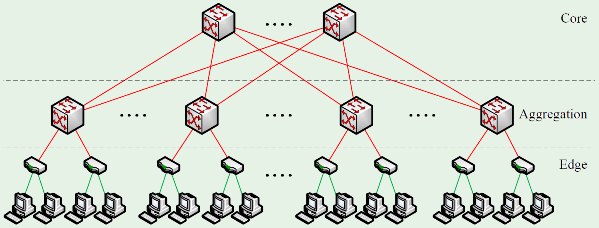


Figure : Fat tree (n=4)

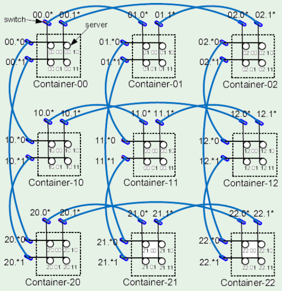


Figure : MDCube (n=2,k=1)

Setup

Generation of Requests

- $\mathcal{N}$ is chosen uniformly at random from the interval $\{10, \dots, 30\}$.
- $\mathcal{B}$ is uniformly distributed in the interval of $\{20\%, \dots, 100\%\}$ w.r.t. to the available capacity of an unused link.
- $\mathcal{C} = 1$ and the capacity of servers are 2.

Generation of Scenarios

- Requests are embedded over time using the VC-ACE algorithm.
- After system stabilization, a single data point is generated by considering the performance of both algorithms on the same substrate state and the same request.

Metrics

Acceptance Ratio

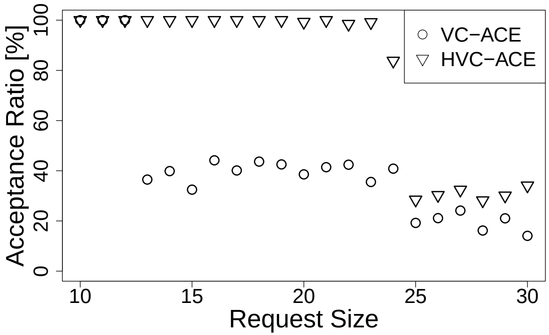
How many requests can VC-ACE embed compared to HVC-ACE?

Footprint Change

Assuming that both algorithms have found a solution, how much resources do we save by using HVC-ACE (compared to VC-ACE using 100%).

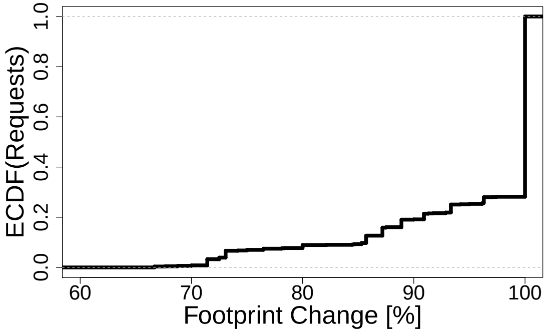
Results

Results on Fat Tree Topology



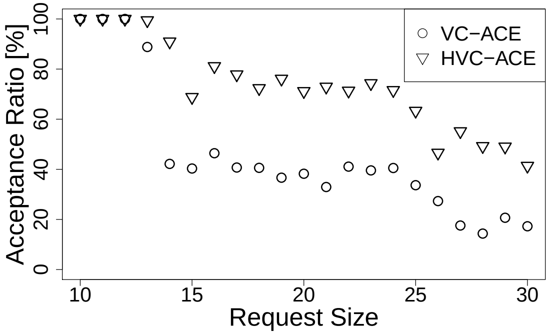
HVC-ACE can improve acceptance ratio dramatically.

Results on Fat Tree Topology



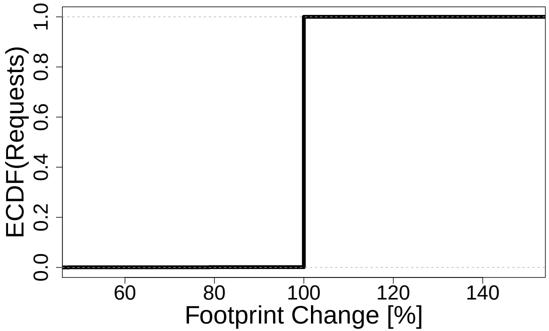
HVC-ACE saves $>$ 10% of resources for more than 20% of the requeusts.

Results on MDCube



HVC-ACE can improve acceptance by around 20% on average.

Results on MDCube



HVC-ACE saves no resources.

Conclusion

Contributions

- Showed how to solve the classic VC embedding problem optimally.
- Defined formally the hose-based VC embedding problem and considered its computational complexity.
- Derived a compact formulation for the splittable hose edge embedding and the HVC-ACE heuristic for the respective embedding problem.
- Showed that by using HVC-ACE one may indeed save a substantial amount of resources and increase the acceptance ratio (on fat trees).

Bottomline

- Complexity of specification can often be traded-off with the complexity of the respective embedding algorithms.
- We need to understand this trade-off better and explore the boundaries of specifications that we can efficiently embed.

References I

[1] M. Al-Fares, A. Loukissas, and A. Vahdat.
A scalable, commodity data center network architecture.
In *ACM SIGCOMM Computer Communication Review*, volume 38, pages 63–74. ACM, 2008.

[2] A. Altin, E. Amaldi, P. Belotti, and M. c. Pinar.
Provisioning virtual private networks under traffic uncertainty.
Networks, 49(1):100–115, Jan. 2007.

[3] H. Ballani, P. Costa, T. Karagiannis, and A. Rowstron.
Towards predictable datacenter networks.
In *Proc. ACM SIGCOMM*, 2011.

[4] M. Chowdhury, M. Zaharia, J. Ma, M. I. Jordan, and I. Stoica.
Managing Data Transfers in Computer Clusters with Orchestra.
In *ACM SIGCOMM*, 2011.

[5] N. Goyal, N. Olver, and F. B. Shepherd.
The VPN conjecture is true.
Proc. ACM STOC, 2008.

References II

[6] A. Greenberg, J. R. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, D. A. Maltz, P. Patel, and S. Sengupta.
VL2: a scalable and flexible data center network.
In *ACM SIGCOMM Computer Communication Review*, volume 39 of *SIGCOMM '09*, pages 51–62. ACM, Acm, 2009.

[7] A. Gupta, J. Kleinberg, A. Kumar, R. Rastogi, and B. Yener.
Provisioning a virtual private network.
In *Proc. ACM STOC*, 2001.

[8] D. Xie, N. Ding, Y. C. Hu, and R. Kompella.
The only constant is change: Incorporating time-varying network reservations in data centers.
In *Proc. ACM SIGCOMM*, pages 199–210, 2012.