VDC Planner: Dynamic Migration-Aware Virtual Data Center Embedding for Cloud

Mohamed Faten Zhani[*], Qi Zhang[*], Gwendal Simon[†] and Raouf Boutaba[*]

[*]University of Waterloo, Canada
[†]Telecom Bretagne, Institut Mines Telecom, France

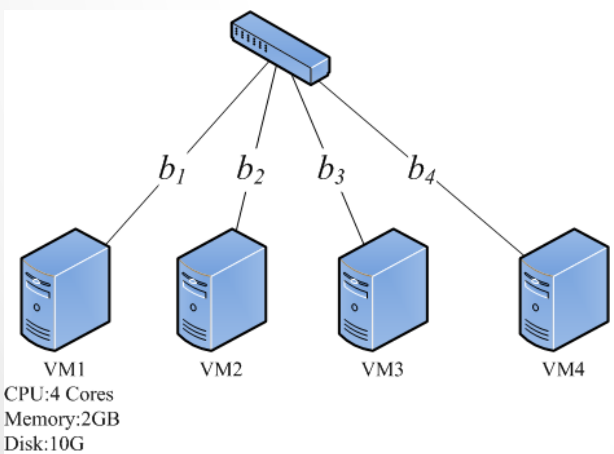
IFIP/IEEE International Symposium on Integrated Network Management (IM 2013)
Ghent, Belgium
27-31 May 2013

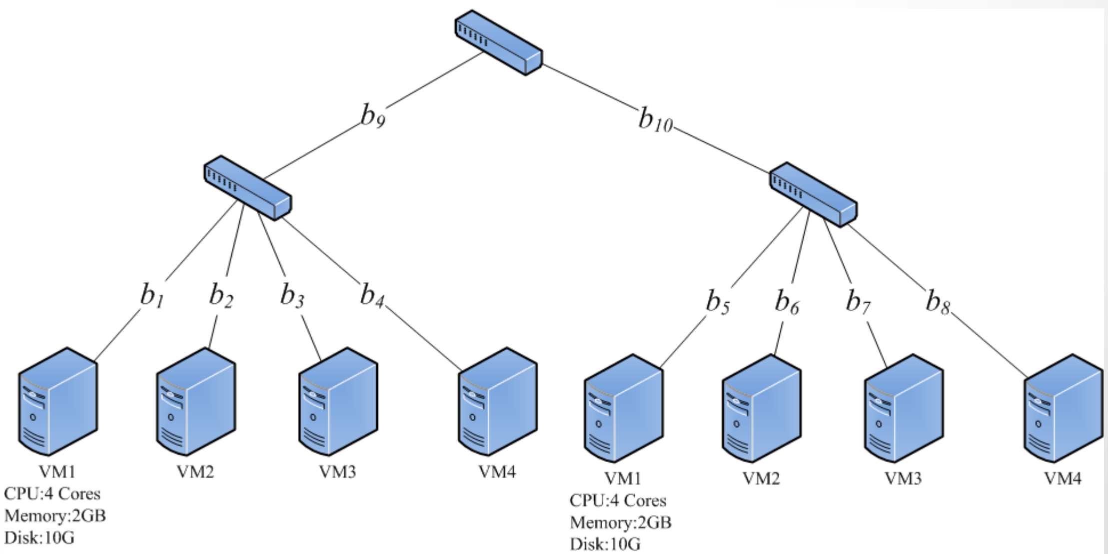
Outline

- Introduction
- VDC Planner architecture
- Problem formulation
- Proposed algorithms
- Experiments
- Conclusions
- Future work

Introduction

- Currently cloud providers provides only computing resources but do no provide any guaranteed network resources
- Goal: Providing both guaranteed computing and network resources
  - Virtual Data Centers (VDCs): virtual machines, routers, switches and links

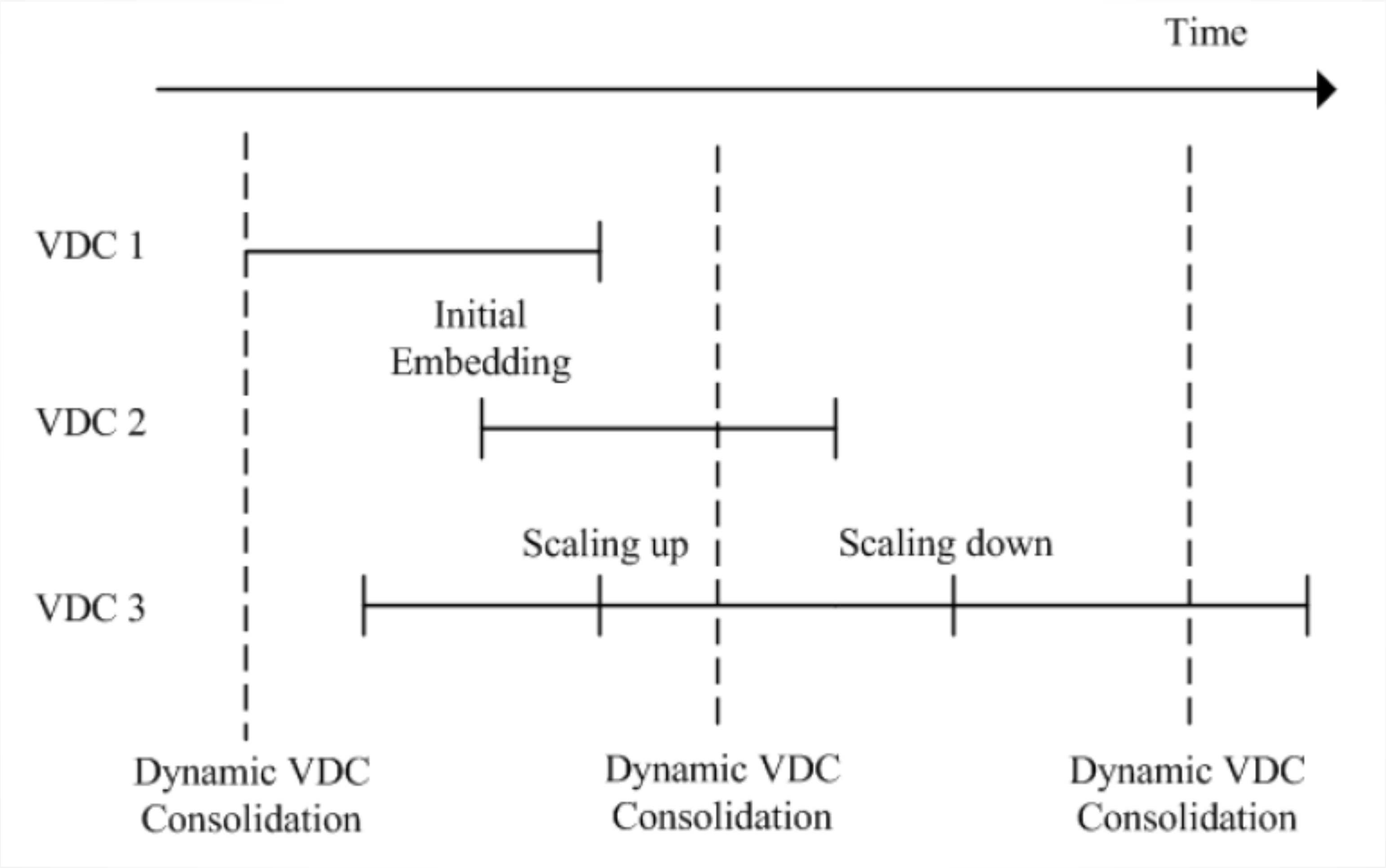




Introduction (Cont'd)

- Objectives
  - Map VDCs onto physical infrastructure (Computing + networking resources)
  - Maximize acceptance ratio/revenue
  - Minimize energy costs
  - Minimize the scheduling delay
  - Achieve all of the above objectives **dynamically** over-time
- Our solution: VDC Planner
  - A migration-aware virtual data center embedding framework
  - VDC embedding, VDC scaling
  - Dynamic VDC consolidation.

Possible scenarios



VDC planner Architecture

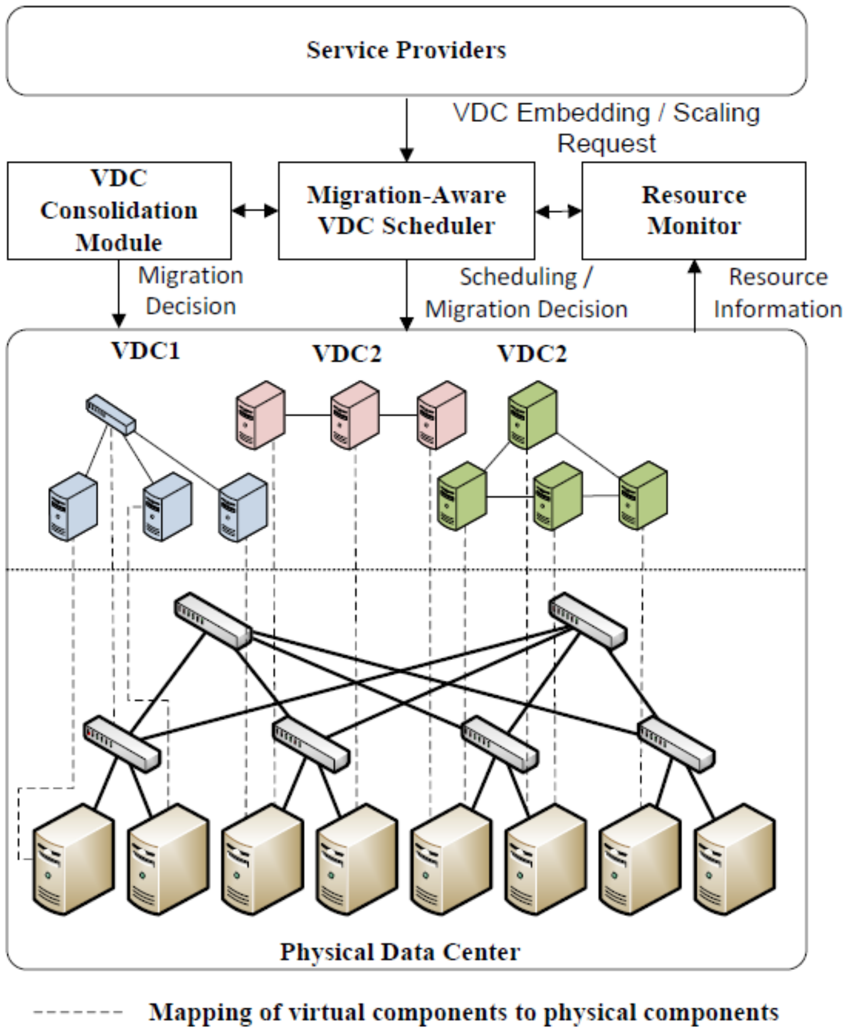


Figure 1: VDC Planner Architecture

Problem formulation

- Objective function

$$\min \sum_{\bar{n}\in\bar{N}} y_{\bar{n}} p_{\bar{n}} + \sum_{i\in I}\sum_{n\in N^i}\sum_{\bar{n}\in\bar{N}} \gamma_n x^i_{n\bar{n}} g^i_{n\bar{n}}$$

Operational costs — embedding cost

$y_{\bar{n}}$ a Boolean that indicates that $\bar{n}$ is active
$x^i_{n\bar{n}}$ a Boolean that indicates that n is embedded in $\bar{n}$

- The embedding cost

$$g^i_{n\bar{n}} = \begin{cases} mig(n,\bar{m},\bar{n}) & \text{if } \bar{n} \neq \bar{m} \\ 0 & \text{if } \bar{n} = \bar{m} \\ 0 & \text{if } n \text{ is currently not embedded} \end{cases}$$

- Placement constraint

$$x^i_{n\bar{n}} \leq \tilde{x}^i_{n\bar{n}} \qquad \forall i \in I, n \in n, \bar{n} \in \bar{N} \quad \tilde{x}^i_{n\bar{n}} = \begin{cases} 1 & \text{if node } n \text{ of VDC } i \text{ can be embedded in } \bar{n} \\ 0 & \text{otherwise} \end{cases}$$

Migration-Aware VDC Embedding Heuristic

- Sort the VMs by their size

$$size_n^i = \sum_{r \in R} w^r c_n^{ir}$$

- Compute the embedding cost (for each VM and physical node)

$$\begin{aligned} cost^i(n, \bar{n}) &= \gamma_n(mig(n, \bar{m}, \bar{n}) + MigOther(n, \bar{n})) \\ &+ \sum_{n' \in N^i : (n', n) \in L^i} d(\bar{n'}, \bar{n}) \cdot b_{(n', n)} \quad (19) \end{aligned}$$

- Embed the VM in the physical machine with the minimal embedding cost

Dynamic VDC Consolidation Algorithm

- Sort the physical nodes in increasing order of their utilizations

$$U_{\bar{n}} = \sum_{r \in R} \sum_{i \in I} \sum_{n \in N^{i}: n \in loc(\bar{n})} \frac{w^r c_n^{ir}}{c_{\bar{n}}^r},$$

- Migrate the VMs hosted in low-utilization machines (using Algorithm 1)

- If all VMs are successfully migrated, the machine is turned off.

Experiments

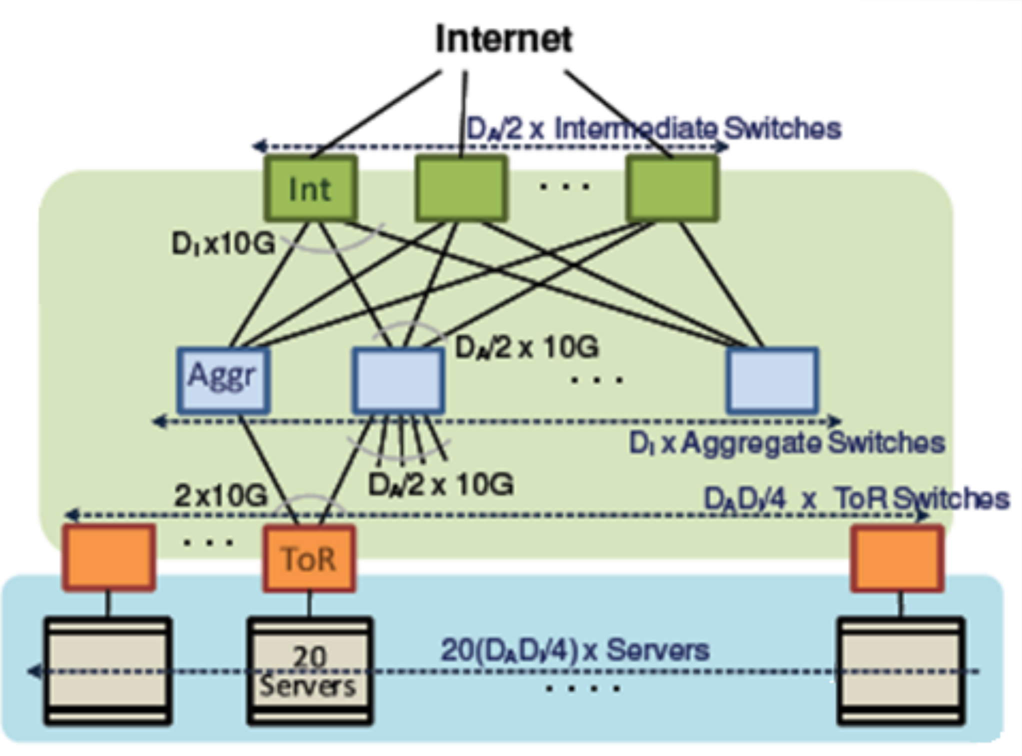
- Physical data center:

4 core switches

4 aggregation switches

4 top-of-rack switches

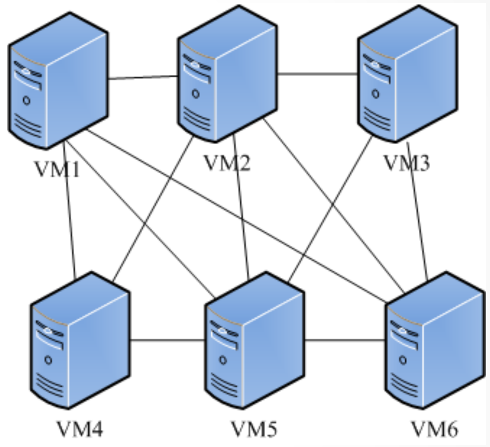
400 physical machines
(8 Cores, 8GB, 100 GB disk).



The VL2 Topology
(Greenberg et al., 2009)

Experiments

- VDC requests:
  - Number of VMs/VDC: [1-20]
  - VM requirements:
    - 1 – 4 cores
    - 1 – 2GB of RAM
    - 1 – 10GB of disk space
  - Virtual link capacity: [1-10 Mbps]



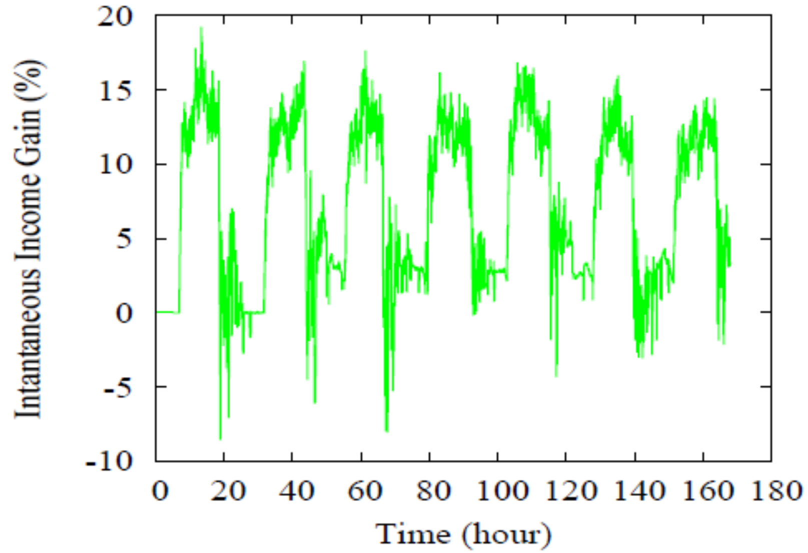
  - Arrival: Poisson distribution
    - 0.01 request/second during night time
    - 0.02 request/second during day time
  - VDC lifetime: exponential distribution (~3 hours)
  - Maximum waiting time: 1 hour

Experiments

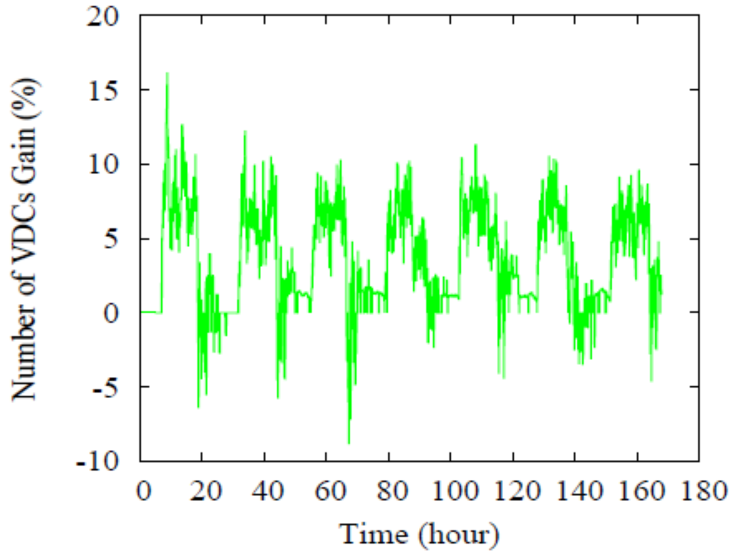
- Comparison metrics:
  - Gain in acceptance Ratio $A_{m/n} = 100 \times \frac{A_m}{A_n} - 100$
  - Gain in revenue $G_{m/n} = 100 \times \frac{R_m}{R_n} - 100$
  - Gain in number of active machines

    $$M_{m/n} = 100 \times \frac{M_m}{M_n} - 100$$
  - Request scheduling delay

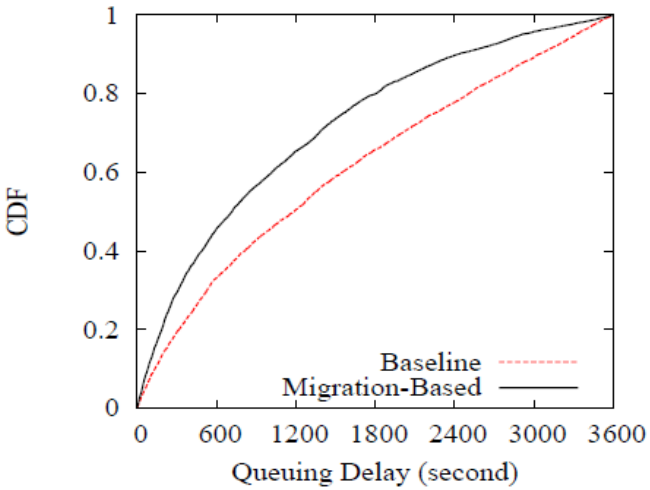
Migration-aware Embedding vs. Baseline



(a) Instantaneous income gains

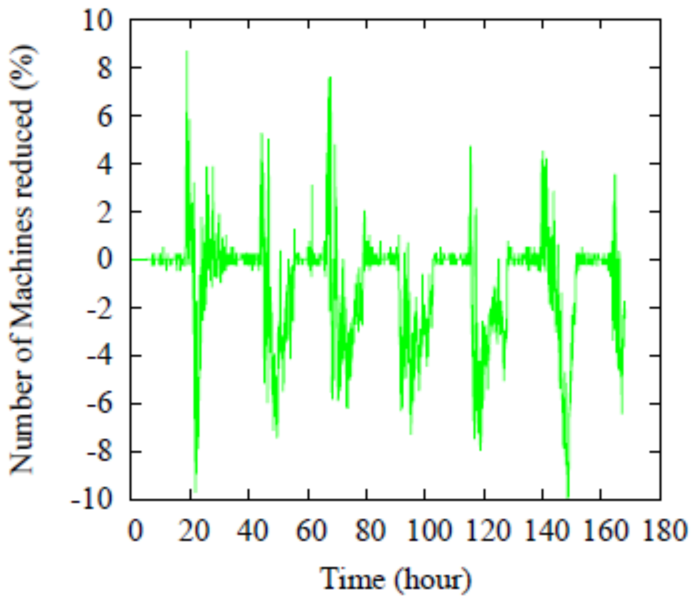


(b) Gain in acceptance ratio

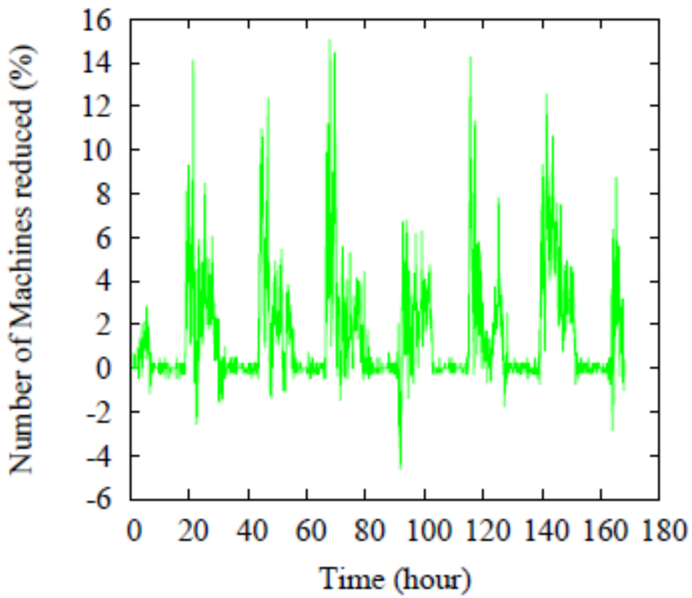


(c) Gain in terms of queuing delay

Migration-Aware embedding + Consolidation



(a) Migration-aware algorithm (Revenue gain up to 17%)



(b) Migration-aware embedding + consolidation (Revenue gain up to 15%)

Conclusions

- The migration-aware embedding can lead to a gain in terms of revenue and acceptance ratio that can reach up to 17%
- Combined with consolidation, VDC planner uses up to 14% less machines than the Baseline.
- Reduce the scheduling delay by up to 25%.

Future work

- Conduct experiments with real traces/real testbed.
- Combine the Migration-Aware embedding with a capacity provisioning technique
  - The provisioning technique provides the optimal number of machines to be turned on.
  - The migration-Aware embedding will maximize the utilization and the revenue

Thank you

Related Work

- SecondNet [8] is a data center network virtualization architecture
  - a greedy heuristic for VDC embedding problem
- Oktopus [1] proposed two abstractions (virtual cluster and virtual oversubscribed cluster)
  - A greedy heuristic for VDC embedding in tree-like topologies
- SecondNet and Oktopus do not consider migration

Migration-Aware VDC Embedding Heuristic

Algorithm 1 Algorithm for embedding VDC request i

```
Sort N̄ based on their states (active or inactive)
S ← N^i
repeat
    Let C ⊆ S be the nodes that are connected to already embedded nodes
    if C == ∅ then
        Sort S according size^i_n defined by equation (18).
        n* ← first node in S
    else
        Sort C according size^i_n defined by equation (18).
        n* ← first node in C
    end if
    for n̄ ∈ N̄ in sorted order do
        Compute embedding cost cost^i(n*, n̄) according to equation (19). If not feasible, set cost^i(n*, n̄) = ∞.
    end for
    if cost^i(n*, n̄) = ∞ ∀n̄ ∈ N̄ then
        return VDC i is not embeddable
    else
        Embed n* on the node n̄ ∈ N̄ with the low cost^i(n, n̄).
        S ← S\n*
    end if
until S == {∅}
```

$$size_n^i = \sum_{r \in R} w^r c_n^{ir}, \quad (18)$$

$$cost^i(n, \bar{n}) = \gamma_n(mig(n, \bar{m}, \bar{n}) + MigOther(r \\ + \sum_{n' \in N^i : (n', n) \in L^i} d(\bar{n'}, \bar{n}) \cdot b_{(n', n}$$

Dynamic VDC Consolidation Algorithm

Algorithm 2 Dynamic VDC Consolidation Algorithm

Let $\bar{S}$ represent the set of active machines
repeat
    Sort $\bar{S}$ in increasing order of $U_{\bar{n}}$ according to equation (21).
    $\bar{n} \leftarrow$ next node in $\bar{S}$
    $S \leftarrow loc(\bar{n})$
    Sort S according to $size_n^i$ defined in equation (18).
    for $n \in S$ **do**
        $n \leftarrow$ next node in S. Let i denote the VDC to which n belongs
        Run Algorithm 1 on VDC i over $\bar{S}\backslash\{\bar{n}\}$.
    end for
    $cost(\bar{n}) \leftarrow$ the total cost according to equation (17)
    if $cost(\bar{n}) \leq p_{\bar{n}}$ **then**
        Migrate all virtual nodes according to Algorithm 1
        Set $\bar{n}$ to inactive
    end if
    $\bar{S} \leftarrow \bar{S}\backslash\{\bar{n}\}$
until $U_{\bar{n}} \geq C_{th}$

$$U_{\bar{n}} = \sum_{r\in R}\sum_{i\in I}\sum_{n\in N^i : n\in loc(\bar{n})} \frac{w^r c_n^{ir}}{c_{\bar{n}}^r},$$