University of Waterloo

Venice: Reliable Virtual Data Center Embedding in Clouds

Qi Zhang, Mohamed Faten Zhani, Maissa Jabri and Raouf Boutaba

University of Waterloo

IEEE INFOCOM

Toronto, Ontario, Canada April 29, 2014

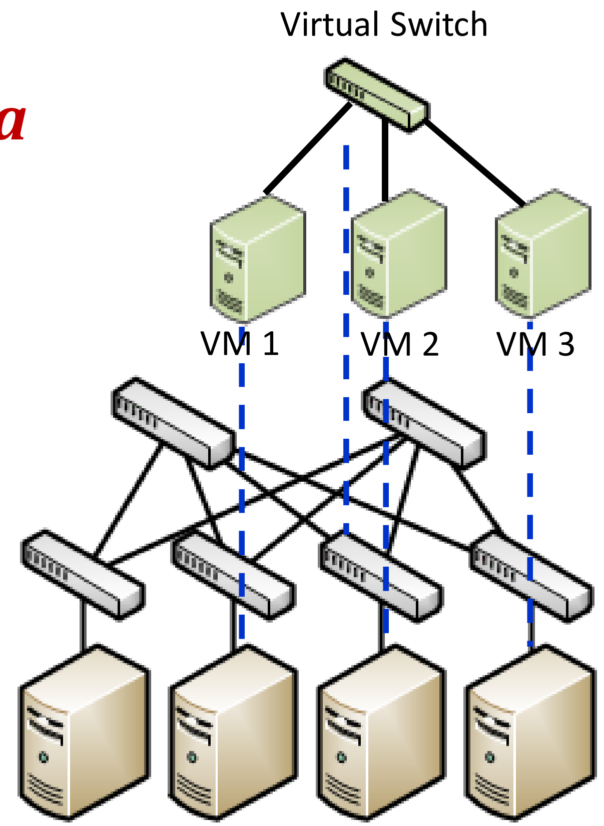
Introduction

- Cloud Computing has become a popular model for hosting online services
  - A **Cloud provider** allocates resources to service providers
  - A **service provider** uses the resources to run services
- Traditional resource allocation approach:
  - Server virtualization only
  - No bandwidth reservation
- Lack of network bandwidth reservation can hurt application performance

Virtual Data Centers

- **A better approach**: Allocating resources in the form of ***Virtual Data Centers*** (VDCs)
  - VMs connected by virtual networks

- VDC scheduling problem
  - Achieving server consolidation
  - Improving communication locality



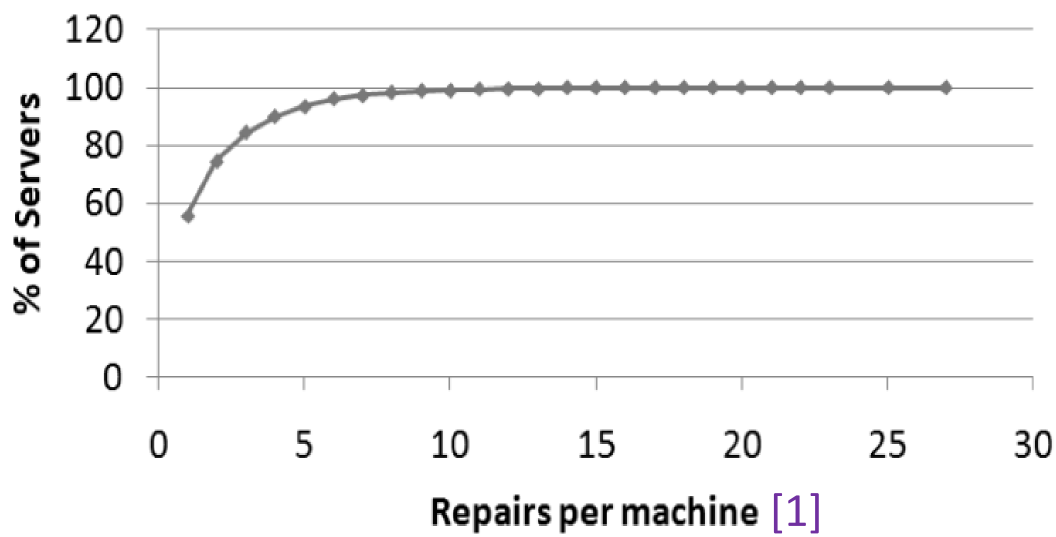
Motivation

- Reliability is a major concern of service providers
  - A service outage can potentially incur high penalty in terms of revenue and customer satisfaction

- *Availability* is a common reliability metric specified in SLA

- VDC availability is dependent on
  - Service priority
  - VDC topology and replication groups
  - Hardware availability

Understanding Data Center Failures



- Heterogeneous server failure rates
  - Server that has experienced a failure is likely to fail again in the near future

[1] Vishwanath et al. “Characterizing Cloud Computing Hardware Reliability”, ACM SoCC 2010

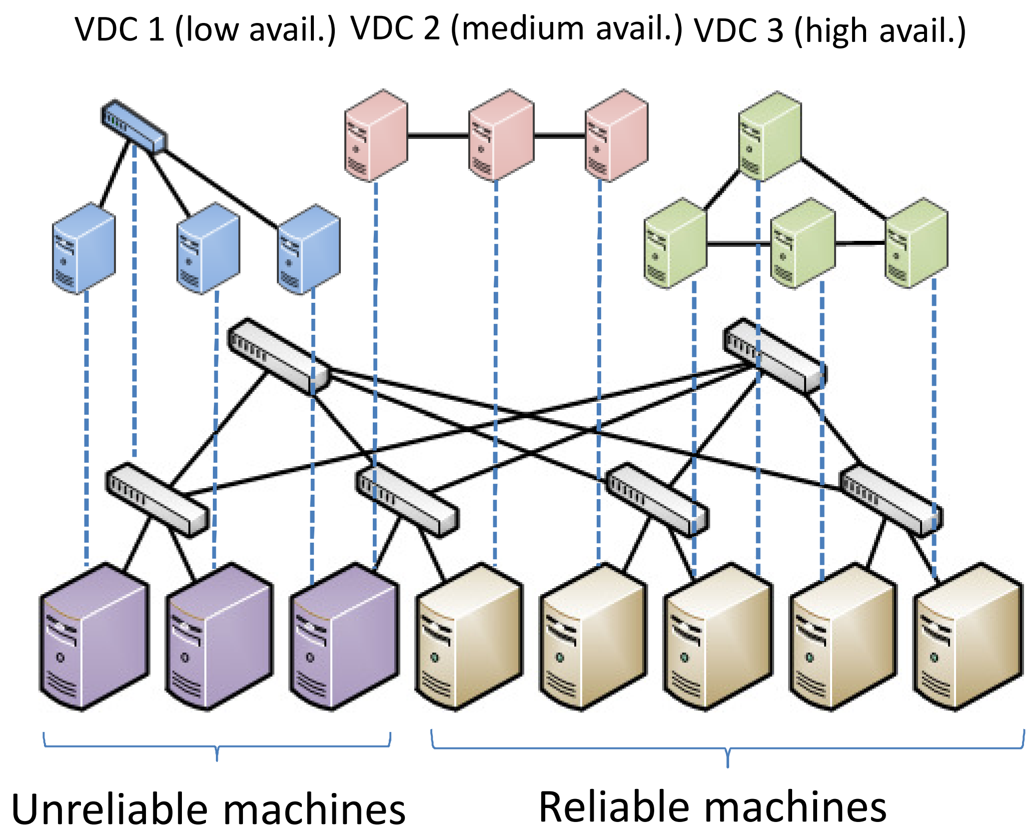
Understanding Data Center Failures

- Network failure characteristics [1][2]
  - Failure rates of network equipment is type-dependent
    - Load balancers have high probability of failure (≥20%),
    - Switches often have low failure probability (≤5%).
  - Number of failures are *unevenly distributed* across equipment of the same type
    - E.g. Load balancer failures dominated by few failure prone devices
  - Correlated network failures are rare
    - More than 50% of link failures are single link failures, and more than 90% of link failures involve less than 5 links [1]

[1] Gill et. al. "Understanding network failures in data centers: measurement, analysis, and implications", SIGCOMM, 2011.
[2] Wu et. al, "Netpilot: automating datacenter network failure mitigation" SIGCOMM 2012.

Motivation

- VDCs have heterogeneous availability requirements
- Resources have heterogeneous availability characteristics
- Place VDCs with high availability on reliable machines



Outline

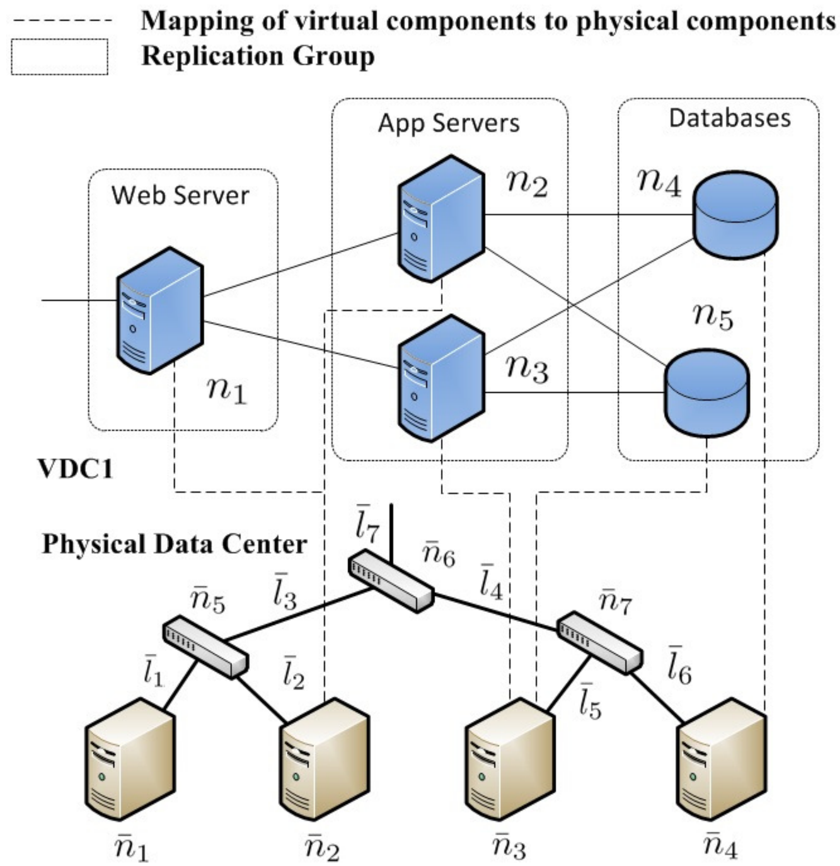
- Introduction
- Motivation
- Computing VDC Availability
- Venice: Reliable VDC Embedding in Clouds
- Experiments
- Conclusion

Computing VDC Availability

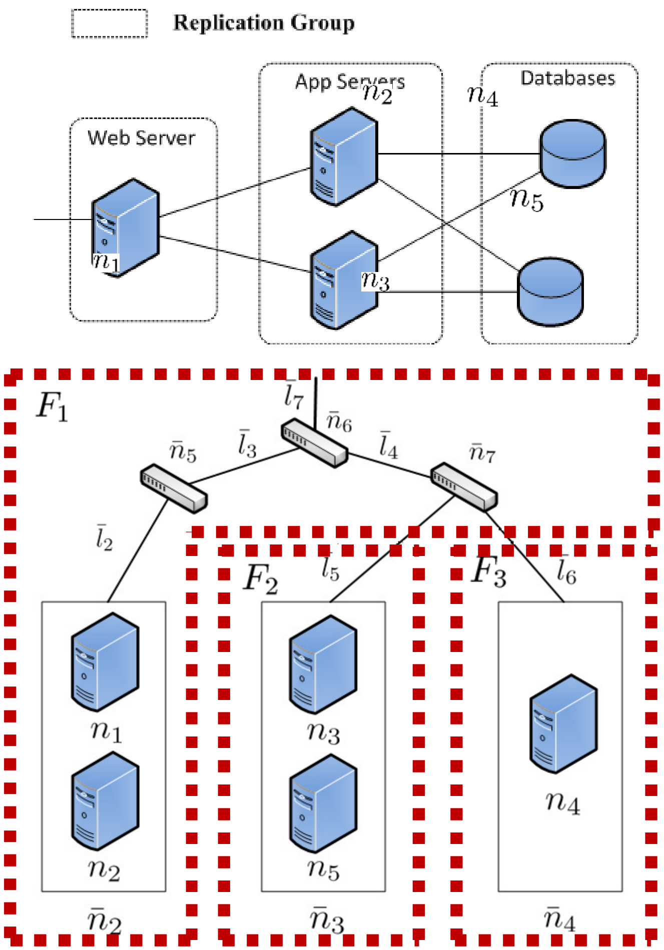
- Example 3-tier application
- Assume physical components $\bar{n}_i$ and $\bar{l}_i$ have availability $A_{\bar{n}_i}$ and $A_{\bar{l}_i}$ respectively, where

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

- How to compute the availability of this VDC?



Computing VDC Availability



Case 1: F1 unavailable,

$$A_{F_1} = 0$$

Prob. of occurrence: $P(F_i) = 1 - \prod_{i \in F_1} A_i$

Case 2: F1 available, F2 unavailable

$$A_{F_1} = \prod_{i \in F_3} A_i$$

Prob. of occurrence: $P(F_2) = \left(\prod_{i \in F_1} A_i\right)\left(1 - \prod_{i \in F_2} A_i\right)$

Case 3: F1 available, F2 available

$$A_{F_1} = 1$$

Prob. of occurrence: $P(F_2) = \prod_{i \in F_1 \cup F_2} A_i$

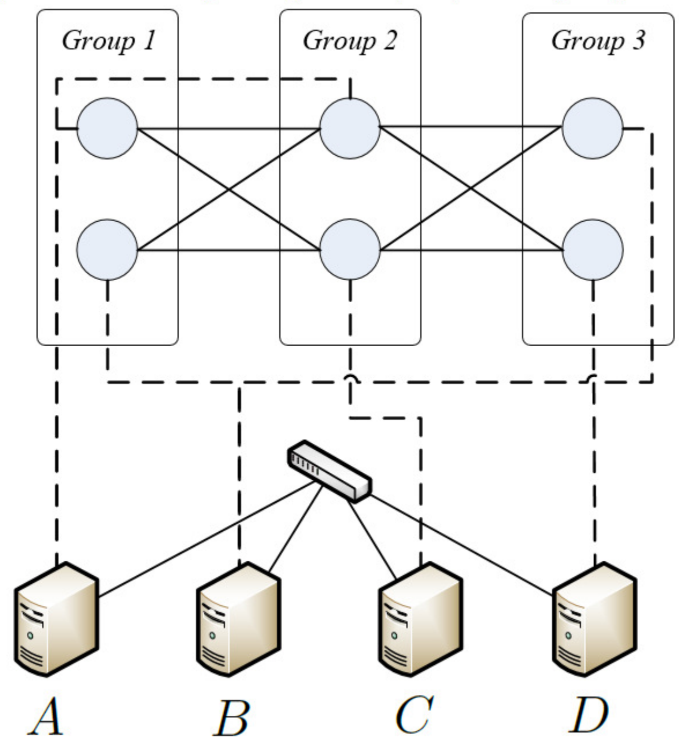
Using conditional probability, the availability of VDC_1 can be computed as:

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

Computing VDC Availability

Theorem 1: VDC availability cannot be computed in polynomial time in the general case

$f(A, B, C, D) = (A \vee B) \wedge (A \vee C) \wedge (B \vee D)$



Proof: Reduction from the counting monotone 2-Satisfiability problem

Need to consider an exponential number of scenarios in the worst case!

Computing VDC Availability

- Observation: it is unlikely to see large simultaneous failures
  - Given 3 nodes, each with availability ≥ 95%, the probability of seeing all 3 nodes fail simultaneously is at most $(1 - 0.95)^3 \leq 0.00013$

- A fast heuristic:
  - Compute availability using scenarios S^k that involve at most k simultaneous failures

- Fast heuristic provides a ***lower bound*** on VDC availability

Computing VDC Availability

- An alternative approach: ***Importance sampling***
  - Consider base-cases in S^k
  - Sampling the remaining cases ($N \in \{0,1\}^n \backslash S^k$) and assign weight $w(s) = P(s)/\bar{P}(s)$

$$\overline{A_{VDC}} = \underbrace{\sum_{s \in S^k} P(s)A(s)}_{\text{base case}} + \underbrace{\frac{1}{|N|}\sum_{s \in N} w(s)A(s)}_{\text{samples}}$$

Define $\overline{S^k} = \{0,1\}^n \backslash S^k$ and $r = |\overline{S^k}| max_{s \in \overline{S^k}}\{P(s)\}$, we can show

$$\Pr(\overline{A_{VDC}} - A_{VDC} > \varepsilon) \leq \exp(-\frac{2|N|\varepsilon^2}{r^2})$$

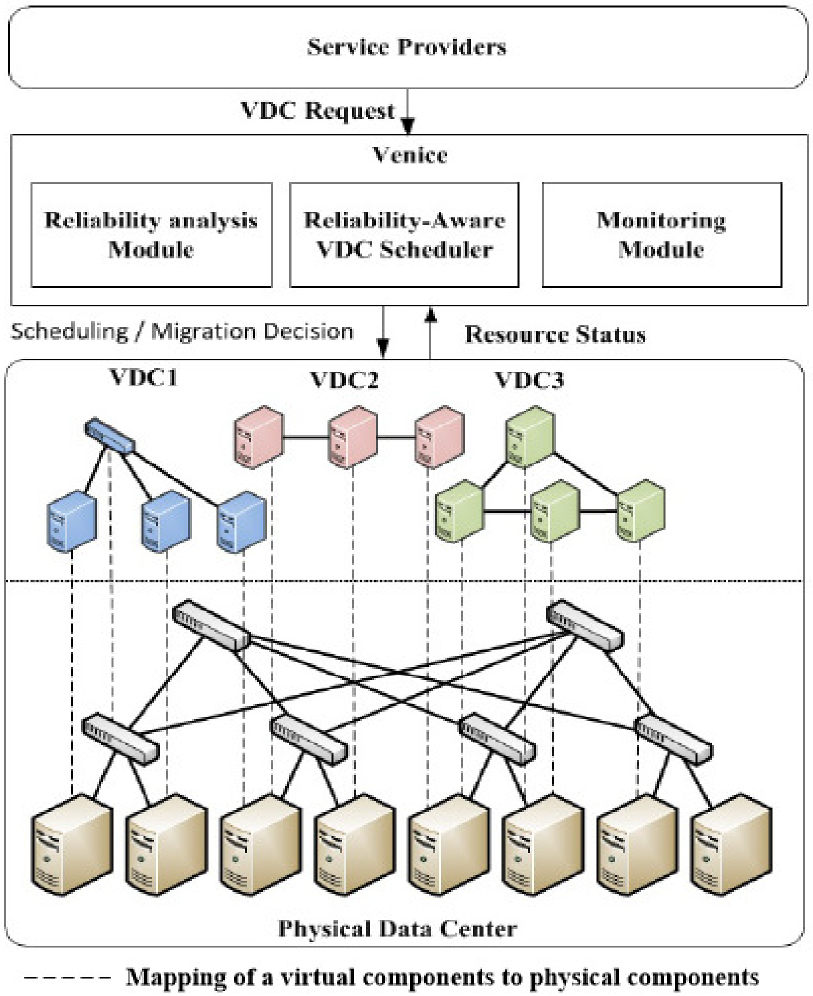
Computing VDC Availability

- Generalizations
  - Replication group that tolerates *k* out of *n* failures
    - E.g. replicated file systems
  - *Partial availability* where failures cause down-graded performance
    - Availability as a continuous value between [0,1]

Outline

- Introduction
- Motivation
- Computing VDC Availability
- Venice: Reliable VDC Embedding in Clouds
- Experiments
- Conclusion

Venice: Reliable VDC Embedding



- 3 Components:
  - Resource Monitor
  - Reliability analysis module
  - VDC Scheduler
- Features
  - Migration-based scheduling
  - Dynamic scaling
  - Periodic consolidation

Problem Formulation

- Objective function: $\min C_E + C_M + C_A$
- Where

$$C_E = \sum_{\bar{n} \in \bar{N}} y_{\bar{n}} p_{\bar{n}}$$ (Resource cost)

$$C_M = \sum_{i \in I} \sum_{n \in N^i} \sum_{\bar{n} \in \bar{N}} \gamma_n x^i_{n\bar{n}} g^i_{n\bar{n}}$$ (Migration cost)

$$\mathcal{C}_A = \sum_{i \in I} (1 - A_i)\pi_i + \sum_{\bar{n} \in \bar{N}} F_{\bar{n}} C^{restore}_{\bar{n}} + \sum_{\bar{l} \in \bar{L}} F_{\bar{l}} C^{restore}_{\bar{l}}$$ (Failure cost)

- Subject to constraints:

$$\sum_{i \in I} \sum_{n \in N^i} x^i_{n\bar{n}} c^{ir}_n \leq c^r_{\bar{n}} \quad \sum_{i \in I} \sum_{l \in L^i} f^i_{l\bar{l}} \leq b_{\bar{l}}$$ (Capacity constraint)

$$\sum_{\bar{l} \in \bar{L}} \bar{s}_{\bar{n}\bar{l}} f^i_{l\bar{l}} - \sum_{\bar{l} \in \bar{L}} \bar{d}_{\bar{n}\bar{l}} f^i_{l\bar{l}} = \sum_{n \in N^i} x^i_{n\bar{n}} s^i_{nl} b_l - \sum_{n \in N^i} x^i_{n\bar{n}} d^i_{nl} b_l$$ (Flow constraint)

$$x^i_{n\bar{n}} \leq \tilde{x}^i_{n\bar{n}} \quad \sum_{\bar{n} \in \bar{N}} x^i_{n\bar{n}} = 1 \quad \sum_{\bar{l} \in \bar{L}} f^i_{l\bar{l}} = b_l$$ (Assignment constraint)

Greedy Scheduling Algorithm

- For each received VDC request
  - **Initial embedding:** embed one node from each replication group.
  - **Repeat**
    - For each remaining component compute a score as the availability improvement - resource cost
    - Embed the component with the highest score
  - **Until** the VDC availability is achieved or all nodes are embedded
  - Embed the remaining components greedily based solely on resource cost

Outline

- Introduction
- Motivation
- Computing VDC Availability
- Venice: Reliable VDC Embedding in Clouds
- Experiments
- Conclusion

Experiments

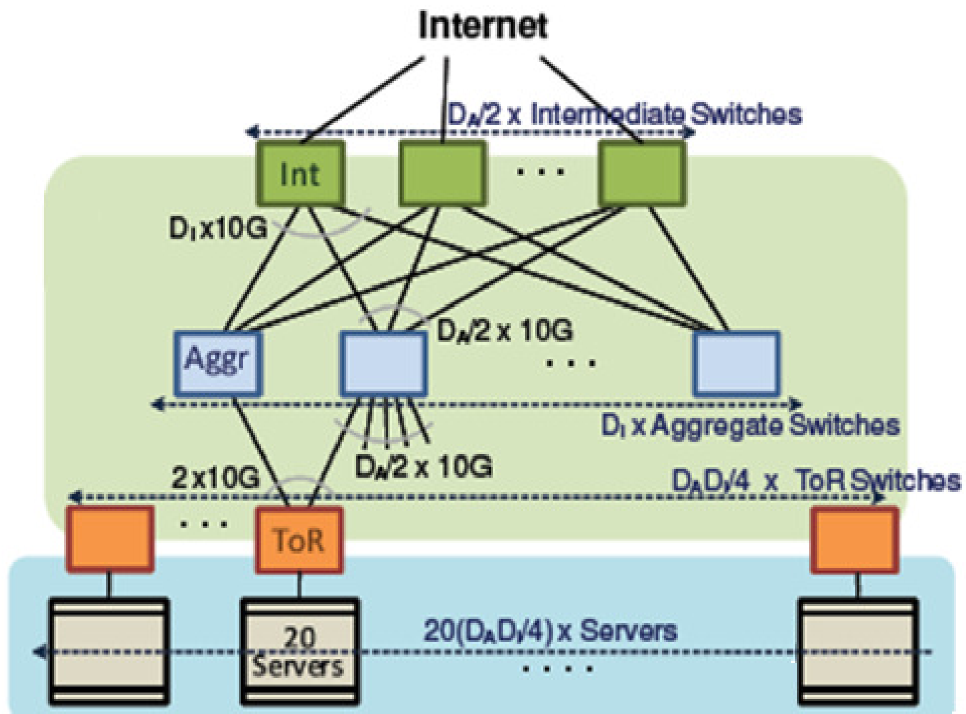
- Data Center Topology

4 core
switches

4 aggregation
switches

4 top-of-rack
switches

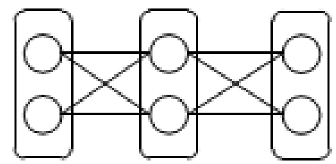
120 Servers
(8 Cores, 8GB Memory,
100 GB disk).



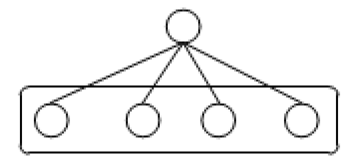
Physical Topology (VL2)

Experiments

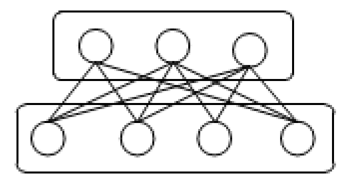
- VDC request formats
  - From 1 to 10 VMs per group
  - Different availability requirements



(a) Multi-tiered



(b) Partition-Aggregate



(c) Bipartite

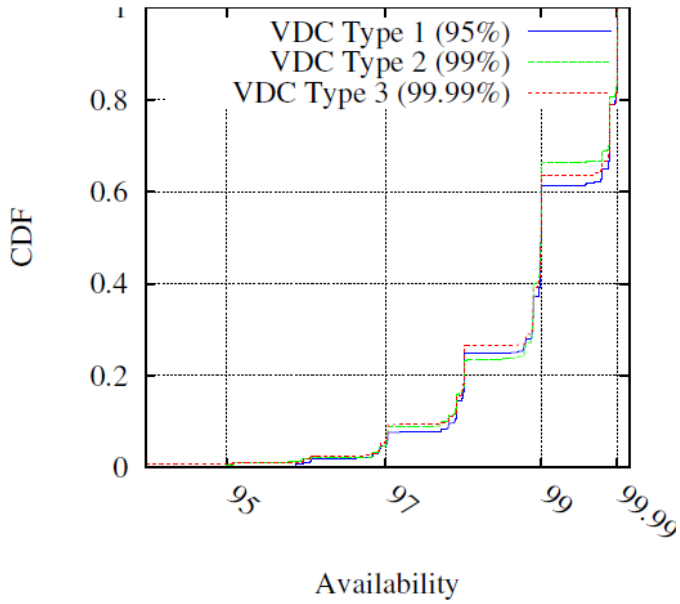
- We use VDC Planner [1] as a baseline for comparison

TABLE I: VDC Availability requirements

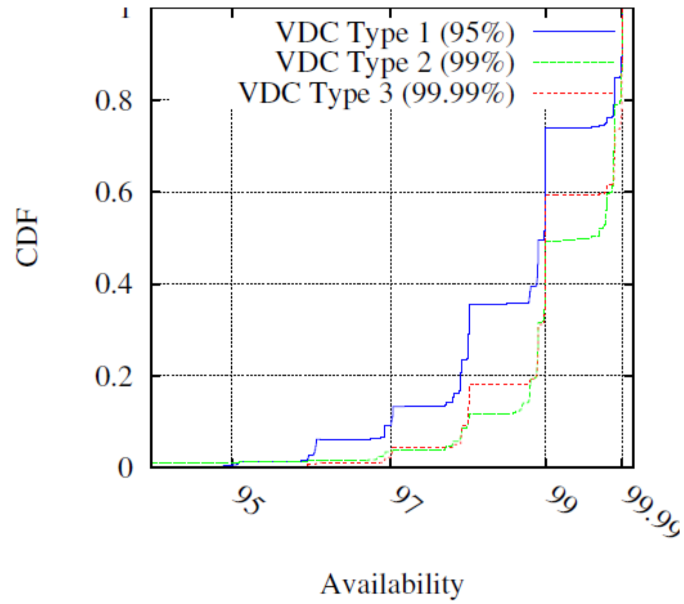
VDC Type	Minimum Required Availability (%)	Acceptable daily downtime
1	95.00	1h:12mn
2	99.00	14mn:2s
3	99.99	08.64s

[1] Zhani et al. “VDC Planner: Dynamic Migration-Aware Virtual Data Center Embedding for Clouds”, IM 2013

Experiments



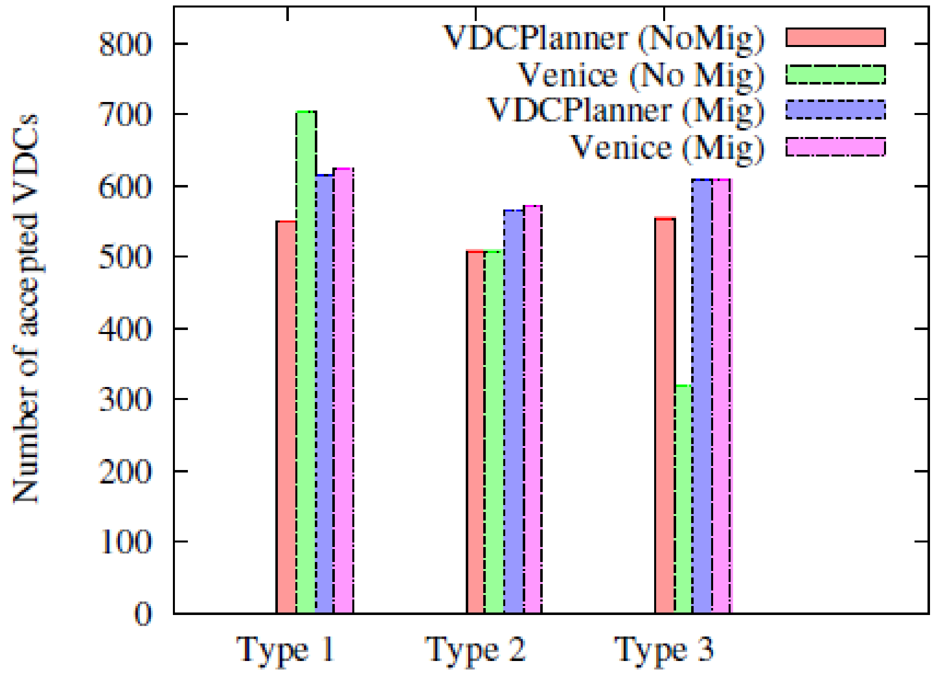
(a) VDC Planner (using migration)



(b) Venice (using migration)

- Venice increases the number of VDCs satisfying availability requirements by up to 35%

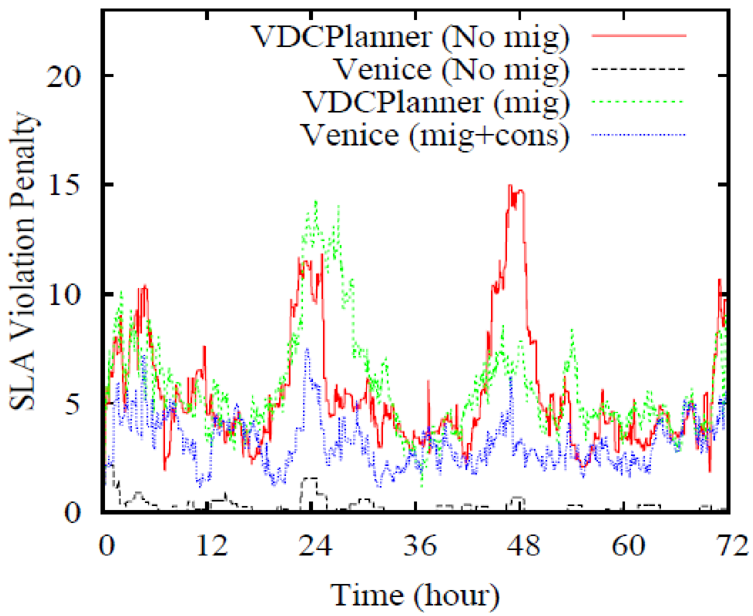
Experiments



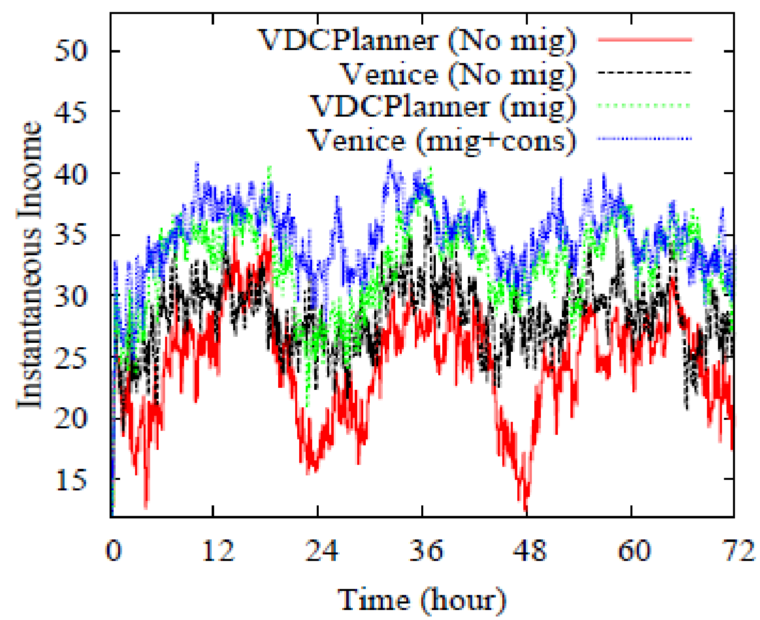
Number of accepted VDCs

- With migration, the number of accepted VDCs is comparable to that of VDC Planner

Experiments



SLA violation Cost



Instantaneous Income rate

- Venice achieves 15% increase in revenue compared to VDC Planner

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 framework called Venice

- Benefits of Venice:
  - Increases the number of VDCs satisfying availability requirement by up to 35%
  - Increases the net income by up to 15%.

Thank you!

Dynamic Workload Consolidation

- Consolidate workload during idle periods while improving VDC availability

- Algorithm
  - Step 1: Improve availability of existing VDCs
    - Select top V VDCs that have highest penalty
    - Try to re-embed each of them to improve solution cost
  - Step 2: Consolidate on fewer machines
    - Iterate C_{th} times
      - Select most under utilized machine $\bar{n}$
      - Re-embed VDCs running on $\bar{n}$ without using the machine $\bar{n}$

Experiments