Virtual Network Mapping for Cloud Services Under Probabilistic Regional Failures

M. Pourvali[1], H. Bai[1], F. Gu[2], K. Shaban[3], M. Naeini[4], J. Crichigno[5], M. Hayat[2], S. Khan[6], N. Ghani[1]
[1]University of South Florida, [2]University of New Mexico, [3]Qatar University, [4]Texas Tech University, [5]Northern New Mexico College, [6]North Dakota State University

***Abstract*—Network virtualization is critical for distributing cloud services over expanded distances and improving scalability and responsiveness. However many providers are very concerned about maintaining high availability, particularly under large catastrophic/disaster type conditions causing multiple failures. Now recent studies have looked at network virtualization for disaster recovery. However these efforts have only treated resource-intensive protection strategies and have not incorporated the probabilistic nature of disasters. Hence this paper studies virtual network embedding using a-priori failure state information and proposes various strategies based upon both traffic engineering (resource efficiency) and risk minimization (disaster mitigation) objectives. The performance of these schemes is tested using network simulation and future research directions identified.**

***Index Terms*—Cloud networking services, network virtualization, virtual network mapping, virtual network survivability**

I. INTRODUCTION

Cloud services allow users to outsource many applications and even infrastructure needs to third-party operators. These services are seeing strong traction these days, owing to their high cost-effectiveness and rapid scalability. Nevertheless, as these deployments grow, many users are becoming increasingly concerned about the reliability and availability of their critical processes. These concerns are particularly acute under the threat of larger catastrophic failures affecting multiple system elements across wider geographic regions, e.g., natural disasters, malicious *weapons of mass destruction* (WMD) attacks, cascading power outages, etc [1]-[3].

In order to improve the reliability and scalability of their cloud-based services, many operators are starting to leverage network virtualization techniques [4],[5] to distribute services over wider geographic domains, i.e., using underlying substrate networks to interconnect data-centers. Hence cloud service providers can now setup *virtual network* (VN) overlays with dedicated VN node and VN link resources to support their needs. Here each VN node can be assigned a desired amount of computing and storage resources, and similarly, each VN link can be assigned a desired amount of bandwidth capacity.

Now the problem of mapping VN requests over substrate networks has been well-studied, and a variety of schemes have been proposed with varying objectives, e.g., revenue maximization, cost reduction, etc [4]-[10]. VN mapping survivability schemes have also been proposed to improve support for higher-end *service level agreements* (SLA). For the most part, these efforts focus on single node or link failure recovery and pre-provision (protection) VN nodes and VN links, see [11]-[17]. Recent studies have also looked at VN disaster recovery for larger regional failures, [18]-[20]. These schemes assume pre-defined risk regions and use a variety of techniques to map VN nodes (links) to ensure recovery from single (non-concurrent) disaster events. However these pre-provisioned strategies are quite resource-intensive and ignore the *probabilistic* nature of disaster events, i.e., essentially treating all stressors as equiprobable. As a result, these schemes can yield lower revenues, especially if the relative frequency of regional failures is low.

In light of the above, this paper directly incorporates the stochastic nature of regional disaster events into the VN mapping. Namely, a-priori probabilistic models are used to specify substrate node/link vulnerability, and advanced "risk-aware" VN mappings are computed to limit the damage from large regional faults. However, purely focusing on risks can lead to increased resource usage, as seen in studies for regular connection provisioning [3]. As a result, a more refined strategy is also proposed to improve the balance between competing resource efficiency and risk objectives. Overall, the paper is organized as follows. Section II surveys the latest work in survivable VN mapping design. Section III then presents two novel VN mapping schemes to incorporate the stochastic nature of disaster events. Performance evaluation results are then presented in Section VI along with conclusions and directions for future work in Section VII.

II. BACKGROUND REVIEW

VN mapping (embedding) is a complex NP-hard problem [4] given the high dimensionalities involved. As a result, a variety of constrained schemes have been emerged, including optimization and heuristic strategies. For example various *mixed integer linear programming* (MILP) models have been proposed to minimize substrate network resource usages and/or maximize revenues, see [5],[6]. However, given the high intractability of most MILP formulations, researchers have also studied graph-based heuristics for VN mapping, including separate node/link mapping (two-stage) and joint node/link mapping (single-stage) schemes [4]. The former types map all VN nodes first and then proceed to mapping their respective VN link connections using modified shortest path or k-shortest path algorithms. Examples here include the cluster-based scheme in [7] and the work in [8]. Meanwhile, single-stage algorithms route VN links right after the VN

nodes have been mapped. In general these strategies are more resource efficient and give lower blocking (higher revenues) than two-stage schemes, see [9],[10].

Now many studies have also looked at survivable VN protection mapping, mostly for single link failures and single node failures. For example, [11]-[13] propose a range of link-disjoint path-pair routing schemes to provision protected VN links. Alternatively, [14] proposes a link-level approach by computing backup detour routes for physical substrate links. Meanwhile others have also addressed single node failures. For example, [15] and [16] propose 1- and k-redundant schemes to provide single and multiple backup VN nodes for protecting working VN nodes. The former types are not as effective as they can yield connection-level congestion on links from backup nodes. Meanwhile, [17] details a single backup VN node scheme which tries to re-map *all* all VN nodes after a failure (using bipartite graph techniques). Although this approach can lower resource congestion (blocking), it entails high post-fault switchover complexity.

Finally, multi-failure VN recovery from large regional disaster events has also been studied. For example, [18] and [19] compute multiple backup VN mappings for each potential failure region and then use resource sharing to reduce protection overheads. Note that sharing is possible here as it is assumed that stressor events are sufficiently rare so as to be mutually-exclusive. In particular, a MILP formulation is presented along with relaxation-based solutions using Lagrangian and decomposition-based rounding. Several heuristic strategies are also presented in [18]. Namely, the *separate optimization with unconstrained mapping* (SOUM) scheme computes separate working and backup VN mappings. Meanwhile the *incremental optimization with constrained mapping* (IOCM) scheme incrementally adds backup resources for each failure region. Nevertheless, these strategies are very resource intensive, especially if the number of failure regions is large. Hence an alternate solution in [20] pre-partitions failure regions into disjoint sets and computes two separate mappings. This scheme provides notable reduction in resource overheads with minimal impact on blocking (revenue) performance.

However, the above pre-provisioned VN disaster recovery schemes ignore the *probabilistic* nature of stressors. As a result, these methods can over-provision backup resources by treating all failures as equiprobable. Hence there is a further need to develop solutions that take into account the occurrence probabilities of regional failure events, especially if their relative frequencies are very low. Along these lines, two risk-aware VN mapping solutions are proposed.

III. NETWORK MODEL & NOTATION

To present the new risk-aware VN mappings, the requisite notation is introduced. First, the physical substrate network is defined as an undirected graph $G_s = (V_s, E_s)$, where V_s is the set of substrate nodes and E_s is the set of substrate links. Each substrate node $v_s \in V_s$ has $R(v_s)$ of generic computing/storage resources with a unit usage cost of $\mathbb{C}(v_s)$, i.e., modeling data-centers with access routers/switches. Meanwhile, each substrate link $e_s \in E_s$ also has $B(e_s)$ capacity and unit usage cost of $\mathbb{C}(e_s)$. Substrate links e_s are also denoted as (v_s, v_s'), where v_s and v_s' are the two end-points. Meanwhile a VN request is given by an undirected graph $G_v = (V_v, E_v)$, where V_v is the set of VN nodes and E_v is the set of VN links. Namely, each VN node $v_v \in V_v$ requests some node resources, $r(v_v)$, and each VN link $e_v \in E_v$ requests some bandwidth capacity, $b(e_v)$. Akin to physical substrate links, a VN link e_v is also denoted as (v_v, v_v').

Now consider large disaster-type events. It is safe to assume that there will be a finite number of such occurrences and that they will be mutually-exclusive, i.e., non-simultaneous, as per [1],[3],[18]-[20]. Now each potential disaster can be modeled as a "risk region" with certain probabilistic occurrence rates, i.e., as derived via off-line modeling of climatic, seismic, or geo-political conditions. Hence the respective network regions affected by all disasters can be grouped in a set U, where each failure region $u \in U$ has a "failure" subgraph $G_u = (V_u, E_u)$, $V_u \subseteq V_s$ and $E_u \subseteq E_s$. Given the probabilistic nature of regional failures, each stressor $u \in U$ can be assigned an occurrence probability $w_u \in [0, 1]$, where $\sum_{u \in U} w_u = 1$ due to mutually-exclusivity. These values can be further combined to yield a failure region *probability vector* for all events, $\vec{\boldsymbol{w}} = [w_{u^1}, w_{u^2}, ..., w_{u^{|U|}}]$. Finally, conditional *risk vectors* can be defined for each substrate node/link in a risk region u, i.e., $\vec{\boldsymbol{p}}_{v_s} = [p_{v_s}^{u^1}, p_{v_s}^{u^2}, ..., p_{v_s}^{u^{|U|}}]$ and $\vec{\boldsymbol{p}}_{e_s} = [p_{e_s}^{u^1}, p_{e_s}^{u^2}, ..., p_{e_s}^{u^{|U|}}]$. Specifically, $p_{v_s}^{u^i}$ ($p_{e_s}^{u^i}$) is the conditional node (link) failure probability to stressor u_i. Akin to the multi-failure regional models in [1],[3],[20], all conditional (node, link) failure probabilities are assumed to be independent.

IV. RISK-AWARE VN MAPPING DESIGN

As per Section II, there is a need to incorporate probabilistic risk state in the VN mapping process to improve reliability. Along these lines, two different solutions are proposed, one which only focuses on risk minimziation and another which also takes into account traffic engineering efficiency.

A. Risk Minimization Approach

A pure risk minimization mapping scheme is proposed to minimize VN failure probability given pre-defined failure regions, termed *risk minimization* (RM). This algorithm first defines risk values for each substrate node and link by computing the dot product of the respective risk vectors and the failure region probability vector $\vec{\boldsymbol{w}}$, i.e., $\xi_{v_s} = \vec{\boldsymbol{p}}_{v_s} \cdot \vec{\boldsymbol{w}}$ for substrate nodes and $\xi_{e_s} = \vec{\boldsymbol{p}}_{e_s} \cdot \vec{\boldsymbol{w}}$ for substrate links. Logarithmic values are then computed for use in graph-based heuristic routines, i.e., as additive weights for nodes and links, as follows:

$$r_{v_s} = log \frac{1}{1 - \xi_{v_s}} \tag{1}$$

$$e_{v_s} = log \frac{1}{1 - \xi_{e_s}} \tag{2}$$

Based upon the above, the failure risk of a mapped VN node is defined as the failure risk, r_{v_s}, of the underlying substrate node to which it is mapped. Similarly, the risk for a VN link

1: Set node/link vectors, $r_{v_s} = \vec{\boldsymbol{p}}_{v_s} \cdot \vec{\boldsymbol{w}}$, $r_{e_s} = \vec{\boldsymbol{p}}_{e_s} \cdot \vec{\boldsymbol{w}}$
2: Sort VN nodes in descending order of node degree
3: **for** each $v_v \in V_v$
4: v_s=Call FSN_RM(v_v), if success reserve node resources else return FAIL
5: $r_{v_s} = 0$
6: **for** each adjacent VN link (v_v, v'_v)
7: **if** virtual node v'_v is mapped
8: Compute min. cost path $\mathbb{P}_{(v_v, v'_v)}$ with Eqs. 1,2 if success reserve bandwidth on link routes else return FAIL
9: $r_{v_s} = 0, \forall v_s \in \mathbb{P}_{(v_v, v'_v)}$, $r_{e_s} = 0, \forall e_s \in \mathbb{P}_{(v_v, v'_v)}$

Fig. 1. Risk minimization (RM) mapping psuedocode

1: Input VN node v_v
2: **for** each $v_s \in V_s$
3: **if** v_s is not assigned
4: b_{max}=max. bandwidth of adjacent v_v VN links
5: B_{max}=max. bandwidth of adjacent v_s substrate links
6: b_{total}=total bandwidth of adjacent v_v VN links
7: B_{total}=total bandwidth of adjacent v_s substrate links
8: **if** $R(v_s) \geq r(v_v)$, $B_{max} \geq b_{max}$, $B_{total} \geq b_{total}$
9: Add v_s into candidate substrate node list $\mathbb{L}$
10: **for** each $v_s \in \mathbb{L}$
11: **if** v_v is the first VN node
12: $c_{v_s} = r_{v_s}$
13: **else if** v_v has no mapped neighbor VN nodes
14: hc=hop count from v_s to v'_s for all v'_s are mapped
15: $c_{v_s} = \theta * r_{v_s} + hc$
16: **else**
17: $mc = 0$
18: **for** each mapped neighboring VN node v'_v
19: Compute min. cost path $\mathbb{P}_{(v_v, v'_v)}$ using risk values, if success add cost to mc else $mc = \infty$
20: $c_{v_s} = \theta' * r_{v_s} + mc$
21: **if** v_v is the first VN node
22: Select $v_s \in \mathbb{L}$ w. prob. inversely proportional to c_{v_s}
23: **else**
24: Select $v_s \in \mathbb{L}$ w. min. c_{v_s} and if $c_{v_s} \neq \infty$
25: else return FAIL

Fig. 2. Subroutine FSN_RM psuedocode

is defined as the product (sum) of (logarithmic) failure risks ξ_{v_s} and ξ_{e_s} (r_{v_s} and r_{e_s}) of the substrate nodes (links) along the selected connection path. Note that most shortest-path algorithms do not incorporate node weights. However these values can be added by transforming the substrate network to a directed graph and adding the node weights to ingress links.

The RM scheme uses a joint VN mapping strategy [5] and is shown in Fig. 1. Here, all VN nodes are first sorted in descending order of node degree and then processed sequentially. Namely, a risk-aware strategy is used to first place the node within a candidate pool of nodes, i.e., subroutine FSM_RM. If a valid mapping is found, then all associated VN links whose end point VN nodes have also been mapped (earlier) are mapped to underlying connections in a risk-averse manner, i.e., by using shortest-path computation with link and node weights set as per Eqs. 1 and 2 (step 8, Fig. 1). In addition, the respective risk values for all assigned/traversed substrate nodes/links on the path are also set to zero, i.e., since re-using them will not introduce any further risk. Next consider the details of the node mapping procedure.

The risk-aware node mapping scheme is shown in Fig. 2. The algorithm starts by building a candidate substrate node list, $\mathbb{L}$, for a VN node, v_v, subject to availability of resources, i.e., feasible substrate node must be able to support VN node and emanating VN link capacity demands, steps 4-9, Fig. 2. This list is then checked sequentially to determine the final VN mapping choice. Now there are several cases to consider here. Namely, if v_v is the first VN node to be mapped in the request, then the costs for all nodes $v_s \in \mathbb{L}$, c_{v_s} are set to their risk values, r_{v_s}. The selection is then made by randomly choosing a substrate node with the selection probabilities inversely-proportional to c_{v_s} (step 22, Fig. 2). Now if v_v is not the first node but still has no mapped neighbors, then the node costs for $v_s \in \mathbb{L}$, c_{v_s} are set to weighted sums of the risk values, r_{v_s}, and total static hop count from v_s to all other substrate nodes v'_s assigned to already mapped VN nodes. The substrate node with the minimum c_{v_s} value is then chosen as the final mapping. This strategy tries to map v_v closer to other mapped (non-neighboring) nodes (note that a weighting fraction θ is introduced to balance between risk and hop counts, Fig. 2). Finally, if v_v already has some mapped VN neighbor(s) then the cost value for a node $v_s \in \mathbb{L}$, c_{v_s} is set to a weighted sum of its risk value, r_{v_s}, and total path cost to all other nodes v'_s representing the mapped neighboring VN nodes of v_v. Again, a fractional value θ' is used here to balance between risk and path cost (step 20 in Fig. 2). The candidate substrate node with the minimum c_{v_s} value is selected.

The overall complexity of the node mapping algorithm (FSM_RM, Fig.2) is $O(|V_s||E_s|log|V_s|)$. Since this procedure is called from the main RM scheme (Fig. 1), the full complexity is $O(|E_v||V_s||E_s|log|V_s|)+O(|V_v||log|V_v|)$, where the latter term bounds VN node sorting overheads.

B. Joint Traffic Engineering & Risk Minimization Approach

In general, mapping connections (VN links) to strictly minimize failure risks can yield notably higher resource consumption and lower operator revenues [3]. To address this concern, a modified mapping scheme is proposed to jointly account for both risk minimization and resource efficiency concerns, i.e., *traffic engineering and risk minimization* (TERM) scheme, Fig. 3. Akin to the RM scheme, this algorithm sorts the VN nodes in descending order of node degree and then maps them in a sequential manner. If a valid mapping is found here, the respective VN link connections are routed for all adjacently-mapped VN nodes. However, unlike the RM scheme, the routes are selected using a *joint* risk minimization and traffic

engineering approach. Namely, the *k*-shortest paths between the respective substrate nodes are first resolved, $r_{\mathbb{P}}$, and the path with the minimum end-to-end risk is selected to route the VN link. Now in order to compute these path risks, a set $\mathbb{S}$ is also defined to keep track of all substrate nodes and links that have already been used in the VN mapping, i.e., $v_s \in \mathbb{S}$ if v_s is assigned to a VN node (or is an intermediate node of a VN link) and $e_s \in \mathbb{S}$ if e_s is assigned to a VN link (steps 12-13, Fig. 3). Based upon this, the end-to-end risk for path $\mathbb{P}$ is:

$$r_{\mathbb{P}} = \sum_{i=1}^{|U|} w_{u^i}(1 - \prod_{v_s \in \mathbb{P}, v_s \notin \mathbb{S}} (1 - p_{v_s}^{u^i}) \prod_{e_s \in \mathbb{P}, e_s \notin \mathbb{S}} (1 - p_{e_s}^{u^i})) \quad (3)$$

$$r_{v_s} = \begin{cases} \vec{\boldsymbol{p}}_{v_s} \cdot \vec{\boldsymbol{w}} & \text{if } v_s \notin \mathbb{S} \\ 0 & \text{if } v_s \in \mathbb{S} \end{cases} \quad (4)$$

Overall, Eq. 3 represents the failure probability of $\mathbb{P}$ under all stressors. Carefully note that nodes (links) already in $\mathbb{S}$ are assigned zero failure probability in Eq. 3 since re-using them to map additional elements of the same VN does not introduce additional risk. Consider the node mapping now.

VN node mapping is also done using a joint strategy, as shown in Fig. 4. Akin to the procedure in the RM algorithm (Fig. 2) this algorithm also builds a candidate substrate node list, $\mathbb{L}$. Again, consider several cases here. If v_v is the first VN node being mapped, its cost value $c^1_{v_s}$ is set to $\mathbb{C}(v_s)r(v_v)$ for each $v_s \in \mathbb{L}$. In addition, another cost value, $c^2_{v_s}$, is also computed for this node based upon the substrate node risk, i.e., r_{v_s} in Eq. 4. Using these values, a sub-list of candidates nodes, $\mathbb{L}'$, is generated by randomly selecting $\lceil \frac{|\mathbb{L}|}{k'} \rceil$ substrate nodes from $\mathbb{L}$ with selection probabilities inversely-proportional to the node $c^1_{v_s}$ values. The final mapping is then done by choosing a node v_s from $\mathbb{L}'$ with the selection probabilities set as inversely-proportional to the $c^2_{v_s}$ values (steps 26-27, Fig. 4). Overall, this strategy attempts to evenly-distribute the first VN node over the substrate and also incorporate failure risks to minimize its vulnerability.

Next consider the case of a subsequent VN node v_v with no mapped neighboring VN nodes. Again, the mapping cost for a substrate node v_s, $c^1_{v_s}$, is set to a weighted sum of $\mathbb{C}(v_s)r(v_v)$ and static hop counts to all other substrate nodes v'_s assigned to already mapped VN nodes (as per RM algorithm). In addition, $c^2_{v_s}$ is also set to r_{v_s}, via Eq. 4. Based upon the above, substrate nodes in $\mathbb{L}$ are sorted by ascending $c^1_{v_s}$ values and a further sub-list $\mathbb{L}'$ is generated by selecting the first $\lceil \frac{|\mathbb{L}|}{k'} \rceil$ substrate nodes from $\mathbb{L}$. From this reduced list, the substrate node v_s with the smallest $c^2_{v_s}$ value is selected as the mapping for v_v (step 31, Fig. 4). Finally, if v_v already has some mapped neighboring VN node(s), then the cost values for nodes in the candidate list $\mathbb{L}$, $c^1_{v_s}$, are computed as weighted sums of $\mathbb{C}(v_s)r(v_v)$ and the total path costs from v_s to all other nodes v'_s corresponding to mapped VN neighboring nodes of v_v. Meanwhile, the $c^2_{v_s}$ values are set to a weighted sum of r_{v_s} and the total path risks, as computed using Eq. 3. Using these values, the VN node mapping is computed in exactly the same manner as for VN nodes without any mapped neighbors, i.e., generating and selecting from a further sub-list $\mathbb{L}'$.

1: Clear $\mathbb{S} = \emptyset$ (set of used substrate nodes, links)
2: Sort VN nodes in descending order of node degree
3: **for** each $v_v \in V_v$
4: v_s=Call FSN_TERM(v_v), if success reserve node resources else return FAIL
5: Add v_s to $\mathbb{S}$
6: **for** each adjacent VN link (v_v, v'_v)
7: **if** virtual node v'_v is mapped
8: Compute k-shortest path for (v_v, v'_v) using link costs as link weights, if none return FAIL
9: **for** each path $\mathbb{P}$ of k paths
10: Compute risk value $r_{\mathbb{P}}$ for $\mathbb{P}$ using Eq. 3
11: Select $\mathbb{P}$ with minimum $r_{\mathbb{P}}$, reserve bandwidth
12: Add $v_s, \forall v_s \in \mathbb{P}$ to $\mathbb{S}$
13: Add $e_s, \forall e_s \in \mathbb{P}$ to $\mathbb{S}$

Fig. 3. Joint traffic engineering and risk minimization (TERM) psuedocode

Overall, the complexity of the above VN node mapping scheme is $O(|V_s||E_s|log|V_s|))$. From this, the complexity of the TERM algorithm is given by $O(|E_v|(|V_s|+k)|E_s|log|V_s|)$.

V. PERFORMANCE EVALUATION

The performance of the risk-aware VN mapping schemes is tested using custom-developed $OPNETModeler^{TM}$ models. Simulations are done using a 24-node network topology with 5 pre-defined failure regions, as shown in Fig. 5 (related failure probabilities $\vec{\boldsymbol{w}}$ noted along with a sample risk vector). Here all network nodes have 100 units of capacity and all substrate links have 10,000 units of bandwidth. Meanwhile, incoming VN requests vary between 4-7 nodes with an average node degree of 2.6, i.e., ratio of VN links to VN nodes. The average requested VN node capacity varies between 1-10 units (uniform) and the average requested VN link capacity ranges between 50-1,000 units (uniform). All incoming VN requests have average exponential holding times of μ=600 seconds and inter-arrival times are also exponential with means varied according to the desired input load, i.e., λ requests/second.

To ensure adequate observation, simulations are done with 100,000 random user VN requests and multiple stressor events. Namely, a stressor region is randomly selected after approximately every 1,000 requests (based upon the pre-defined occurrence probabilities in $\vec{\boldsymbol{w}}$), and its respective nodes and links failed independently as per their pre-defined conditional failure probabilities. For comparison purposes, a "non-risk" VN mapping scheme is also tested here, i.e., *non-survivable virtual infrastructure mapping* (NSVIM) algorithm in [18],[19] (as it gives very good results versus some other well-established working-mode VN embedding schemes).

Now several metrics are also used to gauge performance. First, the revenue associated with a VN request is given by:

$$REV(G_v) = \sum_{e_v \in E_v} b(e_v) * \mathbb{I}(e_v) + \rho \sum_{v_v \in V_v} r(v_v) * \mathbb{I}(v_v) \quad (5)$$

where ρ is the fraction of node resource revenue, $\mathbb{I}(e_v)$ is the revenue per unit bandwidth, and $\mathbb{I}(v_v)$ is the revenue per unit

Input VN node v_v
for each $v_s \in V_s$
  if v_s is not assigned
    b_{max}=max. bandwidth of adjacent v_v VN links
    B_{max}=max. bandwidth of adjacent v_s substrate links
    b_{total}=total bandwidth of adjacent v_v VN links
    B_{total}=total bandwidth of adjacent v_s substrate links
    if $R(v_s) \geq r(v_v)$, $B_{max} \geq b_{max}$, $B_{total} \geq b_{total}$
      Add v_s into candidate substrate node list $\mathbb{L}$
for each $v_s \in \mathbb{L}$
  if v_v is the first VN node
    $c^1_{v_s} = \mathbb{C}(v_s)r(v_v)$
    $c^2_{v_s} = r_{v_s}$ using Eq. 4
  else if v_v has no mapped neighboring VN nodes
    hc=hop count from v_s to mapped virtual nodes, v'_s
    $c^1_{v_s} = \theta * \mathbb{C}(v_s)r(v_v) + hc$
    $c^2_{v_s} = r_{v_s}$ using Eq. 4
  else
    $mc = 0$, $rc = 0$
    for each mapped neighboring VN node v'_v
      Compute min.cost path $\mathbb{P}$ for (v_v, v'_v) w. link cost weights, if success add cost to mc else mc=∞
      Compute risk value $r_{\mathbb{P}}$ for $\mathbb{P}$ by 3 and add to rc
    $c^1_{v_s} = \theta' * \mathbb{C}(v_s)r(v_v) + mc$
    $c^2_{v_s} = \theta' * r_{v_s} + rc$, where r_{v_s} is in Eq. 4
if v_v is the first VN node
  Build $\mathbb{L}'$ by randomly selecting $\lceil \frac{|\mathbb{L}|}{k'} \rceil$ substrate nodes from $\mathbb{L}$ w. probability inversely proportional to $c^1_{v_s}$
  Select $v_s \in \mathbb{L}'$ w. prob. inversely proportional to $c^2_{v_s}$
else
  If $c^1_{v_s} = \infty \ \forall v_s \in \mathbb{L}$ return FAIL, else sort $\mathbb{L}$ by increasing $c^1_{v_s}$
  Select first $\lceil \frac{|\mathbb{L}|}{k'} \rceil$ substrate nodes in $\mathbb{L}$ to build $\mathbb{L}'$
  Select $v_s \in \mathbb{L}'$ w. minimum $c^2_{v_s}$ value

Fig. 4. Subroutine FSN_TERM psuedocode

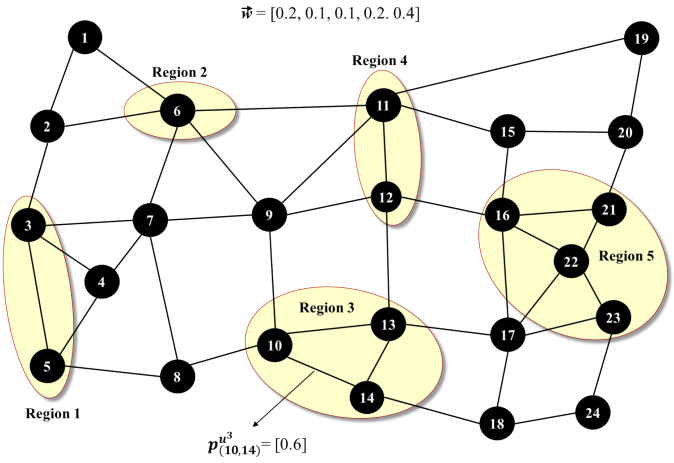

Fig. 5. 24-node test network with $|U|$=5 a-priori failure regions

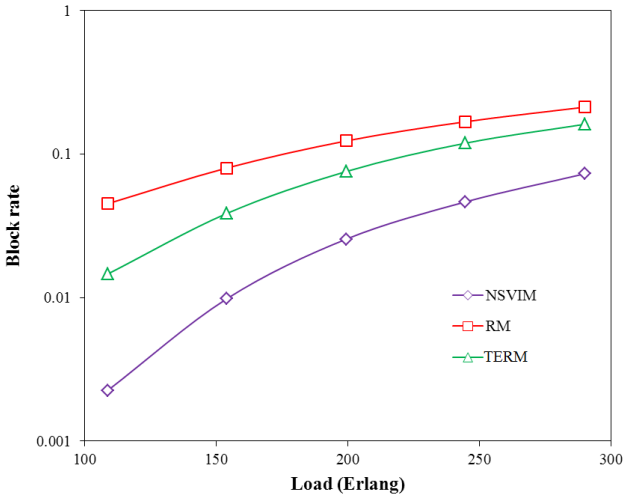

Fig. 6. VN request blocking rate

node resource. Meanwhile the cost of accepting a VN is:

$$COST(G_v) = \sum_{e_s \in E_s} \mathcal{F}^{G_v}_{e_s} * \mathbb{C}(e_s) + \pi \sum_{v_s \in V_s} \mathcal{N}^{G_v}_{v_s} * \mathbb{C}(v_s) \quad (6)$$

where π is the fraction of node resource cost, $\mathcal{F}^{G_v}_{e_s}$ is the total bandwidth allocated on substrate link e_s, $\mathcal{N}^{G_v}_{v_s}$ is the total node resources allocated on the substrate node v_s, and $\mathbb{C}(e_s)$ and $\mathbb{C}(v_s)$ are defined in Section III. In addition, a penalty is also defined for disrupting a VN request, G_v, as:

$$PE(G_v) = \sum_{e_v \in E_v} b(e_v) * \mathbb{PE}(e_v) + \psi \sum_{v_v \in V_v} r(v_v) * \mathbb{PE}(v_v) \quad (7)$$

where $\mathbb{PE}(e_v)$ is the unit link penalty, $\mathbb{PE}(v_v)$ is the unit node penalty, and ψ is the fraction of node penalty. Using the above, the net revenue is computed as:

$$\frac{\sum_i (REV(G^i_v) - COST(G^i_v)) - \sum_j PE(G^j_v)}{T}, \forall G^i_v \in \mathbb{A} \quad (8)$$

Initial tests measure VN request blocking rates for the various schemes, see Fig. 6. Overall, these findings show that the baseline NSVIM scheme yields the lowest blocking, whereas the RM scheme gives the highest blocking. This is expected since the NSVIM scheme tries to minimize VN resource usage. By contrast, the RM scheme does not incorporate any resource usage concerns and chooses longer and more detoured connection routes for the VN links. Furthermore, the joint TERM heuristic also achieves a very good tradeoff between the above two alternatives. For example this scheme closely tracks the NSVIM scheme at medium-high loads (with 1.2 times higher blocking) and notably outperforms the RM scheme (almost an order of magnitude lower blocking).

Next, net revenues are also plotted in Fig. 7 and results indicate that the NSVIM (RM) scheme gives the highest (lowest) revenues. As expected, the TERM design achieves a good tradeoff here, yielding about 25% lower net revenue than the NSVIM scheme at high loads, but 22% higher revenue than the RM scheme. Overall, this improved performance with NSVIM and TERM schemes is due to the fact they are more

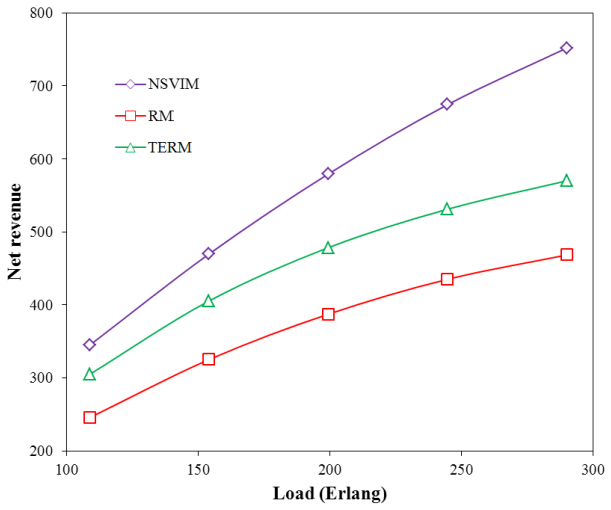

Fig. 7. Net operator revenues

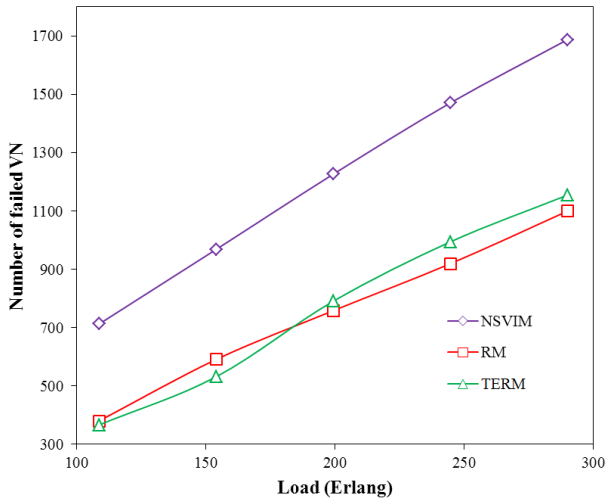

Fig. 8. Number of failed (affected) VN requests

efficient in terms of resource utilization.

Finally, the number of failed VNs is plotted in Fig. 8, i.e., those experiencing at least one or more VN node or link failures. Tracking the number of VN failures is important because these events can interrupt services, even for transient outages. Here the base NSVIM scheme gives the worst reliability, averaging 53% more failures than the risk-aware RM scheme at high loads. However, the joint TERM scheme is very effective, and closely tracks the "risk-only" RM approach, tracking to within 8% of failed VN requests at most input loads. Also, at low loads the TERM scheme emphasizes risk minimization over TE concerns as there little network congestion. Hence the corresponding number of failed VNs is lower than the RM scheme. However, TE concerns dominate at higher loads, and hence the number of failed VNs increases more quickly, i.e., crossover in Fig. 8.

VI. CONCLUSIONS & FUTURE WORK

This paper studies VN mapping for cloud services under probabilistic regional failures. Namely, two risk-aware schemes are proposed, with one focusing purely on risk minimization and the other jointly incorporating resource efficiency and risk concerns. Simulation analysis shows that risk minimization gives the highest reliability but lowest revenues. Meanwhile the joint strategy achieves a much better balance here, closely tracking the revenue of an advanced traffic engineering scheme. Future efforts will look at developing more advanced optimization formulations to bound the performance of VN mapping under multiple probabilistic failures.

ACKNOWLEDGMENT

This work was made possible by the NPRP 5-137-2-045 grant from the Qatar National Research Fund (a member of Qatar Foundation). The statements made herein are solely the responsibility of the authors.

REFERENCES

[1] H. Lee, K. Lee, E. Modiano,"Diverse Routing in Networks with Probabilistic Failures, *IEEE/ACM Transactions on Networking*, Vol. 18, No. 6, December 2010, pp. 1895-1907.
[2] M. Rahnamay-Naeini, *et al*, "Modeling Stochastic Correlated Failures and Their Effects on Network Reliability," *IEEE ICCCN 2011*, Maui, Hawaii, August 2011.
[3] O. Diaz, *et al*, "Network Survivability for Multiple Probabilistic Failures", *IEEE Comm. Letters*, Vol. 16, No. 8, 2012, pp. 1320-1323.
[4] A. Fischer, J. Botero, *et al*, "Virtual Network Embedding: A Survey"', *IEEE Comm. Surveys & Tutorials*, Vol. 15, No. 4, 4th Quater 2013.
[5] M. Chowdhury, R. Boutaba, "A Survey of Network Virtualization", *Computer Networks*, Vol. 54, No. 5, April 2010, pp. 862-876.
[6] N. Chowdhury, *et al*, "Virtual Network Embedding with Coordinated Node and Link Mapping," *IEEE INFOCOM 2009*, Brazil, April 2009.
[7] Y. Zhu, *et al*, "Algorithms for Assigning Substrate Network Resources to Virtual Network Components," *IEEE INFOCOM 2006*, April 2006.
[8] M. Yu, *et al*, "Rethinking Virtual Network Embedding: Substrate Support for Path Splitting and Migration," *ACM SIGCOMM Computer Communication Review*, Vol. 38, No. 2, April 2008, pp. 17-29.
[9] J. Lischka, *et al*, "A Virtual Network Mapping Algorithm Based on Subgraph Isomorphism Detection," *ACM SIGCOMM Workshop on Virtualized Infrastructure Systems & Architectures 2009*, August 2009.
[10] X. Cheng, *et al*, "Virtual Network Embedding Through Topology-Aware Node Ranking," *ACM SIGCOMM Computer Communication Review*, Vol. 41, No. 2, April 2011.
[11] X. Zhang, *et al*, "An Overlay Mapping Model for Achieving Enhanced QoS and Resilience Performance," *International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT) 2011*, Hungary, October, 2011.
[12] Y. Chen, *et al*, "Resilient Virtual Network Service Provision in Network Virtualization Environments," *IEEE International Conference on Parallel and Distributed Systems (ICPADS) 2010*, China, Dec. 2010.
[13] T. Guo, *et al*, "Shared Backup Network Provision for Virtual Network Embedding," *IEEE ICC 2011*, Japan, June 2011.
[14] M. Rahman, *et al*, "Survivable virtual network embedding," *IFIP NETWORKING 2010*, India, May 2010.
[15] H. Yu, *et al*, "Cost Efficient Design of Survivable Virtual Infrastructure to Recover from Facility Node Failures," *IEEE ICC 2011*, June 2011.
[16] H. Yu, *et al*, "Enhancing Virtual Infrastructure to Survive Facility Node Failures," *OFC 2011*, Los Angeles, CA, March 2011.
[17] C. Qiao, *et al*, "A Novel Two-Step Approach to Surviving Facility Failures," *OFC 2011*, Los Angeles, CA, March 2011.
[18] H. Yu, *et al*, "Survivable Virtual Infrastructure Mapping in a Federated Computing and Networking System Under Single Regional Failures," *IEEE GLOBECOM 2010*, Miami, USA, December 2010.
[19] G. Sun, *et al*, "Efficient Algorithms for Survivable Virtual Network Embedding," *Asia Communications and Photonics Conference and Exhibition (ACP) 2010*, China, December 2010.
[20] F. Gu, H. Alazemi, A. Rayes, N. Ghani, "Survivable Cloud Networking Services," *IEEE ICNC 2013*, San Diego, January 2013.