Zhang and Sangaiah *EURASIP Journal on Wireless Communications and Networking* (2018) 2018:65
https://doi.org/10.1186/s13638-018-1075-8

EURASIP Journal on Wireless Communications and Networking

RESEARCH Open Access

Reliable design for virtual network requests with location constraints in edge-of-things computing

San-mei Zhang[1*] and Arun Kumar Sangaiah[2]

Abstract

How to efficiently map virtual networks (VNs) onto a shared physical network is a challenging issue in the field of network virtualization in edge-of-things computing. Since an efficient VN mapping approach can reduce network resource consumption, lower latency, and enhance service reliability, it is important for both customers and network service providers. In this paper, we study the problem of mapping multiple VNs with geographic location constraints onto a physical network while considering the survivability and reliability requirements of each VN request in edge-of-things based data centers. We present the model of this problem and propose a Geographic-Guided Survivable Multiple VN Mapping (GG-SMVNM) algorithm to efficiently solve this problem, which simultaneously considers resource sharing and mapping VN links and nodes in edge-of-things computing. Furthermore, we conduct a large amount of simulations to validate and evaluate our proposed approach. The simulation results show that the proposed method is superior to the existing solution.

Keywords: Network virtualization, Reliability, Mapping, Edge-of-things computing

1 Introduction

The emerging edge computing technologies [1–3], Internet of Things (IoT) [4, 5], and rich cloud services [6, 7] are used to create novel edge-of-things computing. In it, the data processing occurs in part at the network edge or between the cloud-to-end that can best meet customer necessities rather than entirely processing the data in a comparatively fewer number of massive clouds. Operators use the edge-of-things computing paradigm to provide network and computing services in a flexible and resource-efficient way [8–11]. Network virtualization is one of the main technologies and promoters of edge-of-things computing. Network virtualization allows multiple heterogeneous virtual networks (VNs) to share the same physical network in edge-of-things computing [12–16]. Due to the increasing popularity of edge-of-things computing, a great deal of research has been conducted on network virtualization and virtual network mapping technology [17–24].

* Correspondence: smzhang198@gmail.com
[1]China West Normal University, Nanchong, China
Full list of author information is available at the end of the article

A network virtualization environment (NVE) [25] is composed of shared resources (i.e., physical network with resource capacity) and virtual network (VN) requests in edge-of-things computing. A set of VN links and VN nodes makes up a VN request. Every VN node needs a fixed amount of nodal resources (i.e., storage resources, CPU and memory) to execute the edge-of-things computing services and applications, and each VN link that connects two VN nodes needs a great deal of communication bandwidth to exchange the data and information between the connected VN nodes. The progress of virtual network (VN) mapping is quite complicated because of the constraints of virtual links and nodes, despite knowing all VNs in advance. The VN mapping process is composed of two steps: the mapping of VN nodes and the mapping of VN links. Even if the mapping of all the virtual nodes is accomplished, the mapping of virtual links is still complicated. As a result, there are many VN mapping algorithms that can map as many VNs onto the physical network of edge-of-things computing as possible and minimize the VN mapping costs [26–30]. The VN mapping algorithm proposed in [26] maps the VNs under the guidance of minimizing mapping costs. However, it does not take the nodal survivability into consideration. The author comes up with the multiple VN mapping problems that consider survivability in [29], which introduces the VN mapping algorithm that considers the circumstance of physical network link failures. Research in [27] studies the VN mapping problem. It shares the backup resource among different VNs without considering the backup resource sharing of a single VN during the mapping process. The author in [28] researched the redeployment and migration problem of the dynamic VN. The authors in [29, 30] study the problem of VN mapping while considering the local failures of the physical network in edge-of-things computing.

Although many algorithms for VN mapping in edge-of-things computing have been designed, few of these algorithms take the effect of VN nodes' geographic constraints into account. Moreover, no contribution has been made on the backup resource sharing among multiple VNs arriving simultaneously when node survivability is taken into consideration. For example, the 1-*redundant* method and the *K-redundant* method introduced in [31] can realize working and backup link resource sharing when mapping backup nodes and links, although they only apply for a single VN. Meanwhile, the working and backup link resource sharing can be achieved among multiple VNs arriving spontaneously when at most one physical node fails at a certain time. Moreover, nodal resource sharing can be realized thanks to the geographic constraint, which is left to further exploration and research.

In this paper, we study the problem of resource efficiency and reliable VN mapping/deployment in edge-of-things computing. In our research, we take the geographic constraint of each VN node into consideration. We realize the resource sharing inside every VN and the resource sharing link among VNs while mapping multiple VNs arriving spontaneously on the premise that virtual nodes' survivability is guaranteed in edge-of-things computing applications. The main contributions of this paper are as follows:

- We study the problem of reliable mapping for VN requests with location constraints in edge-of-things computing.
- We propose the model and design an efficient algorithm for the studied problem.
- We conduct extensive simulations to evaluate the performance of our proposed algorithms.

The remainder of the paper is arranged as follows. Section 5 outlines the problem statements and is followed by the heuristic algorithms in Section 3. Section 4 gives the detailed simulation results. Finally, Section 5 gives the conclusion of this paper.

2 Problem statement and formulation

2.1 Virtual network request

In this work, we research the issue of mapping multiple VN requests that guarantees the nodes' survivability while at most only one node of physical network (PN) fails at any time in edge-of-things computing and while considering the geographic location constraints of VN nodes.

Figure 1 gives an example of three VNs simultaneously arriving. Note that, in practice, the number of simultaneously arriving VNs is random. Every VN can be described as $G_v = (N_v, L_v)$, where N_v represents the set of virtual nodes with resource requirements corresponding to each node (such as the CPU and the memory capacity of physical nodes) and L_v represents the set of virtual links. Every link has a bandwidth requirement for physical links to guarantee the communication between two VN nodes connected by a virtual link. Additionally, every node of the VN request has a geographic coordinate constraint. For example, the coordinate (x, y) beside the virtual node v_0 in the virtual network VN_0 means the geographic position of virtual node v_0, thus constraining the range of physical nodes that v_0 can be mapped onto. Every virtual node of each VN has its own geographic position constraint in Fig. 1, while only the coordinate of v_0 is labeled in Fig. 1.

2.2 Physical network in edge-of-things computing

The physical network of edge-of-things computing is composed of multiple data centers that are dispersed across multiple geographical locations interconnected by a network. Like the VN request, a physical network model can be represented as $G_S = (N_S, L_S)$, where N_S represents the set of physical nodes of the physical network (every node of which provides a physical resource such as CPU and memory capacity with corresponding geographic coordinates) and L_S represents the set of

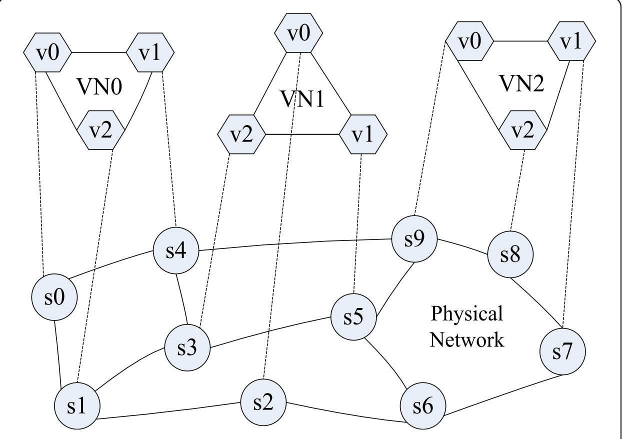

Fig. 1 An example of VN mapping

physical links of the physical network (with every physical link providing physical bandwidth resources to satisfy the communication demand between physical nodes). Furthermore, when a physical node fails, the corresponding physical links also fail. Hence, the virtual nodes mapped onto the failed physical node need to be migrated to another physical node that is not failed, and the corresponding virtual links also need to migrate. In this paper, there is at most only one physical node that fails at all times in the physical network of edge-of-things computing.

2.3 Reliable virtual network provisioning

2.3.1 Given

A substrate edge-of-things computing physical network PN is represented as $G_S = (N_S, L_S)$, and several virtual network requests arrive simultaneously.

2.3.2 Mapping constraints

They include the geographical position constraints of virtual nodes, the resource demands of virtual links, and the resource demands of virtual nodes.

2.3.3 Precondition

There is at most one physical node failure at any time, and if there is a virtual node mapped onto it, the virtual node and adjacent links need to be migrated and recovered.

2.3.4 Problem

Under the precondition above with the mapping constraints, the problem is how to design and realize a mapping algorithm that can map several VNs arriving simultaneously onto the physical network and guarantee the survivability of virtual nodes. It must realize the resource sharing in every VN and also the resource sharing among VNs to save the physical resources of edge-of-things computing, with the aim of minimizing the mapping costs and getting a generally comparatively good multiple VN mapping result.

2.4 Problem formulation

2.4.1 Residual resources

The residual resources include the residual bandwidth of every physical link and the residual node resource of every physical node in edge-of-things computing. The residual bandwidth capacity $R_l(l_s)$ of physical link l_s represents the total amount of bandwidth available on link l_s.

$$R_l(l_s) = \mathrm{Cap}(l_s) - \sum_{l_v \in S_{l_s}} \mathrm{req}(l_v) \tag{1}$$

where $\mathrm{Cap}(l_s)$ denotes the resource capacity of physical link l_s, $\mathrm{req}(l_v)$represents the amount of link resource requirements of VN link l_v, and S_{l_s} indicates the set of VN links mapped on the physical link l_s.

The residual node resource capacity $R_n(n_s)$ of physical node n_s can be computed as follows:

$$R_n(n_s) = \mathrm{Cap}(n_s) - \sum_{n_v \in S_{n_s}} \mathrm{req}(n_v) \tag{2}$$

where $\mathrm{Cap}(n_s)$ shows the resource capacity of physical node n_s, $\mathrm{req}(n_v)$ represents the amount of VN node n_v's requirement for nodal resources, and S_{n_s} indicates the set of VN nodes that have been mapped onto physical node n_s.

2.4.2 Objective

We define the costs $C(G_V)$ of the mapping VN request G_V as all of the costs of physical resources (i.e., physical link and nodal resources) in edge-of-things computing allocated to G_V.

$$C(Gv) = \sum_{l_v \in L_V} \sum_{l_s \in L_S} b_{l_v}^{l_s} p(l_s) + \sum_{n_v \in N_V} \sum_{n_s \in N_S} r_{n_v}^{n_s} p(n_s) \tag{3}$$

where $b_{l_v}^{l_s}$ denotes the amount of resources that physical link l_s assigned to virtual link l_v and $r_{n_v}^{n_s}$ denotes the amount of nodal resources that physical node n_s assigned to virtual node n_v. The symbols $p(l_s)$ and $p(n_s)$ respectively represent the costs of per unit of the physical link and nodal resources.

Our objective is to minimize the VN mapping costs:

$$\text{Minimize } C(Gv) \tag{4}$$

3 Algorithm design

To achieve the survivable VN mapping in a reasonable time, we design an effective heuristic to solve the problem that we researched in the paper. The Geographic-Guided Survivable Multiple VN Mapping (GG-SMVNM) algorithm is detailed in this section.

In spite of the fact that at most one physical node fails in the physical network of edge-of-things computing when multiple VNs simultaneously arrive, different from the *K*-redundant scheme, the GG-SMVNM algorithm first successively maps the spontaneously arriving VNs onto the physical network with the geographic location constraints with the objective of minimizing costs instead of enhancing each VN with additional backup nodes and links. During the successive mapping, virtual nodes from different VNs may be mapped to the same physical node. After mapping all these VNs, a new VN is generated based on the physical mapping topology of the physical network. Finally, similar to the *K*-redundant scheme, the GG-SMVNM algorithm enhances the new-generated VN with backup nodes and corresponding backup links. It maps the backup nodes and links to the physical network to ensure the survivability of virtual nodes while also realizing the nodal and link resource sharing among spontaneously arriving VNs and the resource sharing in VNs. The GG-SMVNM algorithm works with the following steps (Fig. 2).

3.1 Step 1: successively map all the simultaneously arriving VNs onto the physical network of edge-of-things computing

When mapping the VN requests, we aim to map the virtual nodes of different VNs to the same substrate node to make the newly generated VN more concise. The mapping results satisfying the geographic location constraints of VN_0, VN_1, and VN_2 are shown in Fig. 1. Figure 1 only depicts the mapping of virtual nodes, but the corresponding mappings of virtual links are not shown here. We map the virtual node v_1 from VN_0 and node v_0 from VN_1 to the same substrate node s_4. Therefore, the capacity demand of the new virtual node generated based on s_4 is the total amount of the resource demands of v_1 from VN_0 and v_0 from VN_1 when the new virtual network is generated from the physical mapping topology. Similarly, the resource demand of the new virtual node generated by physical node s_2 is the total demand of the resource demands of v_2 from VN_0 and v_2 from VN_1. The corresponding links also satisfy this demand if two or more virtual links are mapped onto the same physical link. Then, the resource requirement of the new virtual link generated by the physical link is the sum of the resource requirements of these virtual links. If there are several mapping results of the virtual nodes belonging to different VNs due to the geographic constraint, then we need to choose the best one with the minimum mapping costs. For example, in Fig. 1, v_1 from VN_0 and v_0 from VN_1 can both be mapped onto the physical node s_4, while node v_1 of VN_0 and v_2 of VN_1 can both be mapped onto s_4 as well. Then, we chose the mapping solution with minimum mapping costs.

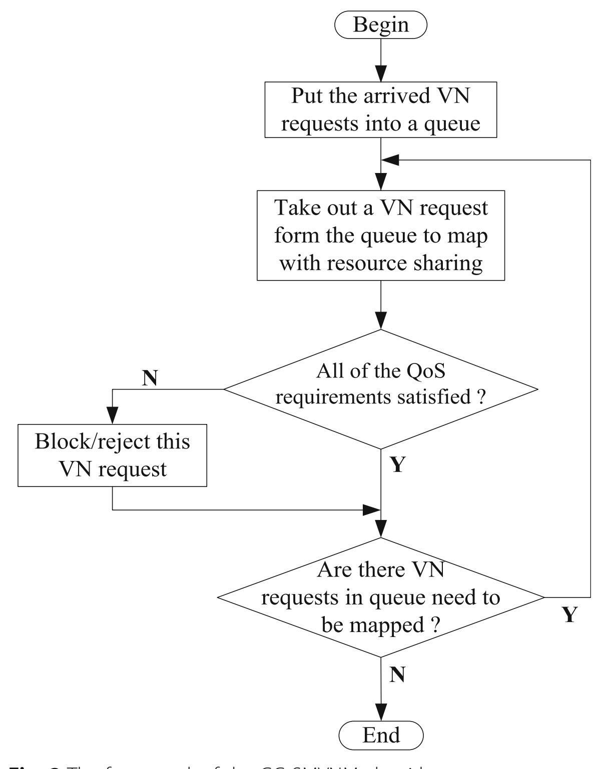
Begin
Put the arrived VN requests into a queue
Take out a VN request form the queue to map with resource sharing
All of the QoS requirements satisfied ?
N
Block/reject this VN request
Y
Are there VN requests in queue need to be mapped ?
Y
N
End

Fig. 2 The framework of the GG-SMVNM algorithm

3.2 Step 2: when all the virtual nodes are mapped, generate a new virtual network according to the mapping results

The new VN generated based on the mapping result of Fig. 1 is shown in Fig. 3. The virtual node v_0 in Fig. 3 is reversely generated due to the physical node s_0 in Fig. 1 with the same resource requirement as the resource requirement of v_0 from VN_0 mapped onto s_0. The virtual node v_1 in Fig. 3 is generated due to the physical node s_4 in Fig. 1 with the resource requirement equal to the sum of resource requirements of v_1 from the original VN_0 and v_0 from VN_1 mapped onto s_4. The other virtual nodes in Fig. 3 are generated in a similar way. The virtual links in Fig. 3 are also generated according to the mapped paths on the physical network such that if two or more virtual links are mapped onto the same physical path on the physical network, then the resource demand of the virtual link above is the sum of these virtual links' resource requirements.

3.3 Step 3: enhance the newly generated virtual network

This step is similar to the *K*-redundant scheme. The GG-SMVNM algorithm adds a backup node for each virtual node and adds the corresponding backup links as well. The enhanced newly generated virtual network with backup nodes and links is shown in Fig. 4. b_0, b_1, b_2, b_3, b_4, and b_5 are backup nodes, and the dashed lines represent the backup links.

3.4 Step 4: map the extra backup links and nodes

We apply the strategy similar with [31] to map backup links and nodes in order to decrease the costs of backup

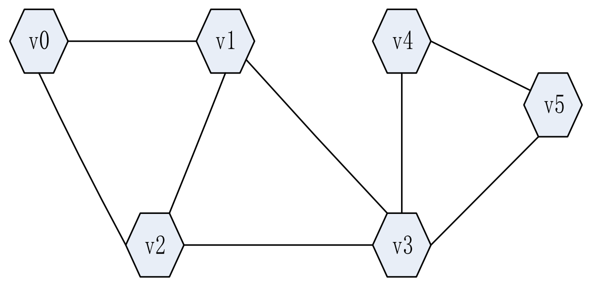

Fig. 3 New virtual network

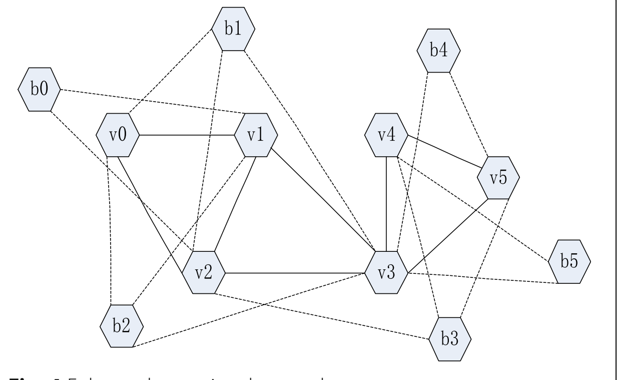

Fig. 4 Enhanced new virtual network

nodal mapping. Moreover, resource sharing of working links and backup links during the backup link and node mapping process leads to the resource sharing of links and nodes among original VNs, which will reduce the mapping costs as much as possible.

The detailed pseudo code of the GG-SMVNM algorithm is described in Table 1.

Table 1 Pseudo code of the GG-SMVNM algorithm

```
GG-SMVNM Algorithm
Input: A physical network $G_S = (N_S, L_S)$; the arrived VN requests $VN_0$, $VN_1 \ldots VN_X$.
Output: A survivable mapping solution considering the location constraints of virtual nodes.
Step 1: Map the VNs that are arriving simultaneously.
The geographic locations constraints of the virtual node must be considered while mapping the VNs, and we should map the virtual nodes that belong to different VNs to the same substrate node.
Step 2: Update the resources of the physical network.
Compute the remaining recourses of physical nodes and links and calculate the costs of mapping multiple VNs.
Step 3: Generate a new virtual network based on the mapping results.
    3.1) Traverse the nodes of the physical network and record the amount of resources allocated to virtual nodes. Add the virtual nodes to the set of virtual nodes from the new VN, and the capacity demand of these virtual nodes are the amount of recourses on corresponding physical nodes. Take the geographic locations of the corresponding physical nodes as the location constraints of the newly generated virtual nodes.
    3.2) Traverse the routing table of the physical network. If the IDs of the corresponding substrate nodes of the virtual nodes in the set mentioned in 3.1 match the routing table, then there is a virtual link between these two virtual nodes.
Step 4: Map the backup nodes and links of the enhanced newly generated VN.
Traverse every backup node that needs to be mapped in descending order of resource demands and map the backup nodes and corresponding links onto the physical network.
    4.1) Traverse the physical network nodes and compute the mapping costs of backup nodes and corresponding links.
    4.2) Select a physical node with the minimum costs. Map the backup node onto it and map the corresponding backup links.
    4.3) Update the resources and record the remaining capacities of the physical nodes and links.
Step 5: Compute the mapping cost of backup nodes and links.
```

Next, we provide the GG-SVNM algorithm that does not consider the resource sharing among different VNs. The detailed pseudo code of the GG-SVNM is shown in Table 2.

In this, CMS_i means the set of virtual nodes which physical node s_i can host.

4 Simulations and results

We first introduce the environment of our simulation in the section, and then we give the results. Last, we analyze the results of simulation.

4.1 Simulation environment

4.1.1 Physical network

In the simulations, we use two different networks as the physical network of edge-of-things computing with 46 nodes and 55 nodes, respectively. Each node has a real geographic coordinate presented by a longitude and latitude. In the two different networks, we assume that the bandwidth capacities of each link follow a uniform distribution from 500 to 1000 and the resource capacities of each node are uniformly distributed from 200 and 400, respectively. The topologies of the physical networks used in our simulations are shown in Fig. 5.

4.1.2 Virtual network configuration

In the case of multiple VNs mapping, the number of VNs simultaneously arriving is random in real applications. In our simulations, without the loss of generality, the number of arriving VNs is randomly distributed from

Table 2 Pseudo code of GG-SVNM algorithm

```
GG-SVNM Algorithm
Input: A physical network $G_S = (N_S, L_S)$, a virtual network request $G_V = (N_V, L_V)$.
Output: A survivable mapping solution considering the location constraints of virtual nodes.
Step 1: Calculate $CMS_i$ for each physical node.
Step 2: Map the original virtual network.
The location constraints must be satisfied while mapping the virtual nodes.
Step 3: Update the resources of the physical network.
Calculate the remaining capacities of the physical nodes and links after mapping the original virtual network. Compute the total mapping costs of all virtual links and nodes from the original virtual network.
Step 4: Map the backup links and nodes.
    4.1) Traverse all physical nodes. Then compute the mapping costs of the backup virtual nodes and relevant backup links.
        a) Compute the mapping costs of backup links corresponding to backup nodes.
        b) Compute the mapping costs of backup nodes.
        c) Calculate the reduced costs brought on by the other virtual nodes in $CMS_i$.
    4.2) Select the physical node with minimum mapping costs. Then map the backup node onto it and map the corresponding backup links.
    4.3) Update the resources. Calculate the available resources for the links and nodes on the physical network.
Step 5: Calculate the mapping costs of backup nodes and backup links.
```

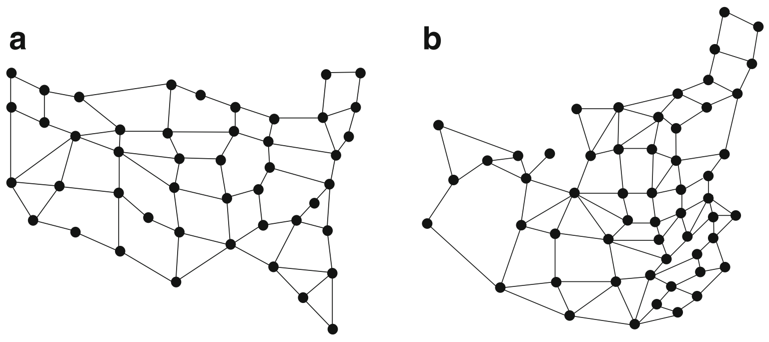

Fig. 5 Physical networks used in our simulations. **a** Net-1: a network with 46 nodes. **b** Net-2: a network with 55 nodes

1 to 4, and each VN randomly consists of 3 or 4 nodes. The resource demand of each node is a variable that is randomly generated between 20 and 30, and its geographic coordinate is also generated randomly with the longitude and latitude whose values fall into a specific range. The possibility that there exists a virtual link between two virtual nodes is 50%. The bandwidth demand of the virtual link is randomly generated between 50 and 80.

Parameter g represents the ratio of the unit node resource overhead to the unit bandwidth overhead. Different values of g can compare the effects that different ratios of the unit node resource costs to unit bandwidth costs on VN mapping costs. We set the parameter g to 5 in the simulations.

In our simulations, we presume that zero or one substrate node fails at any time and several VNs arrive simultaneously. For evaluating our proposed algorithm, we compare the mapping performances of our GG-SMVNM and GG-SVNM algorithms with the EVPF approach proposed in [32] and consider the geographic location constraints of virtual nodes. Furthermore, we vary the number of simultaneously arriving VNs (the number of nodes on each VN is 3 or 4) on the premise that the physical resources are abundant and compare the total mapping costs, backup node mapping costs, and backup link mapping costs of these two algorithms.

We used Microsoft Visual Studio 2008 and C++ programming language to implement the compared algorithms.

4.2 Performance metrics

We define some metrics for evaluating the performance of our proposed algorithm in the simulation.

1. The total VN mapping cost: the total expenses of using physical network resources to provide all VN requests. It can be calculated as follows:

$$M_{\text{cost}}^{\text{total}} = \sum_{i=1}^{|\text{ArrivedVN}|} M_c^i, \tag{5}$$

where M_c^i represents the mapping cost of i-th VN demand and ArrivedVN denotes the set of arrived VN demand.

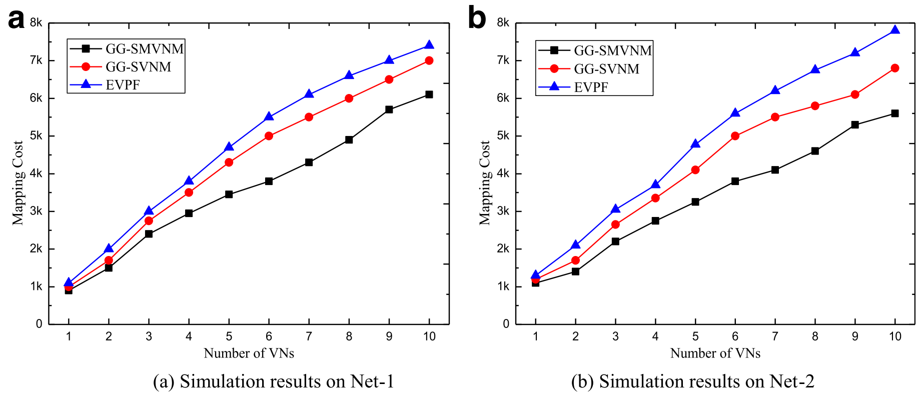

Fig. 6 Mapping costs vs. number of VNs. **a** Simulation results on Net-1. **b** Simulation results on Net-2

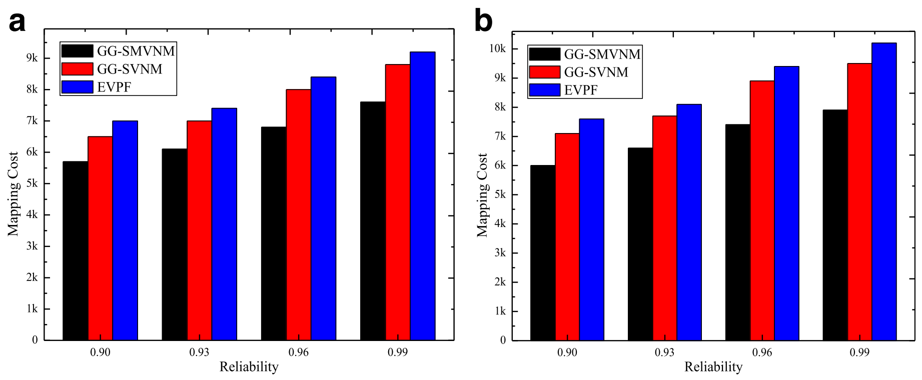

Fig. 7 Mapping costs vs. different reliability requirements. **a** Simulation results on Net-1. **b** Simulation results on Net-2

2. The backup node mapping cost: the total expenses of using physical node resources to host the backup virtual nodes. It can be calculated as follows:

$$M_{\text{cost}}^{\text{bakNode}} = \sum_{i=1}^{|\text{bakNode}|} M_{\text{node}}^{i}, \tag{6}$$

where M_{node}^{i} denotes the mapping cost of the i-th backup virtual node and bakNode represents the set of backup virtual nodes.

3. The backup link mapping cost: the total expenses of using physical link resources to host the backup links. It can be defined as follows:

$$M_{\text{cost}}^{\text{bakLink}} = \sum_{i=1}^{|\text{bakLink}|} M_{\text{link}}^{i}, \tag{7}$$

where M_{link}^{i} denotes the mapping costs of the i-th backup virtual link and bakLink represents the set of backup virtual links.

4.3 Simulation results and analysis

We can see from Fig. 6 that the total mapping costs of our proposed GG-SMVNM and GG-SVNM algorithms are lower than that of the existing EVPF approach [32]. Furthermore, the total mapping costs of multiple VNs of the GG-SMVNM is less than the GG-SVNM and that the advantage of the GG-SMVNM on mapping costs gets more obvious with the increase in the number of simultaneously arriving VNs. This is because when there are several VNs simultaneously arriving, one physical node may host more than one virtual node because of the geographic location constraints of virtual nodes. Therefore, while using the GG-SMVNM for mapping the multiple arrived VNs, the new VN generated according to the mapping solutions for mapping multiple VNs onto the physical network is simpler than the multiple original VNs. Furthermore, the resource sharing in every VN and the node and link resource sharing across VNs can be realized when the GG-SMVNM performs the mapping of backup nodes and links while at most one node fails in the physical network, whereas the node and link resource sharing only occurs in each VN in the GG-SVNM algorithm. Therefore, the mapping costs of

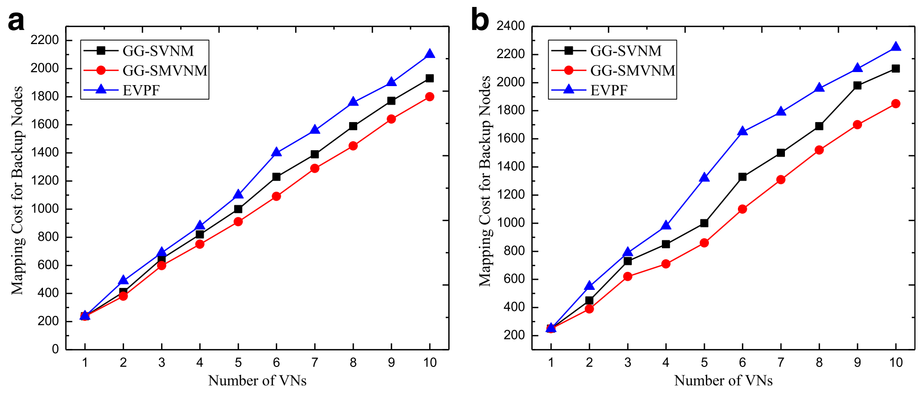

Fig. 8 Mapping costs for backup nodes. **a** Simulation results on Net-1. **b** Simulation results on Net-2

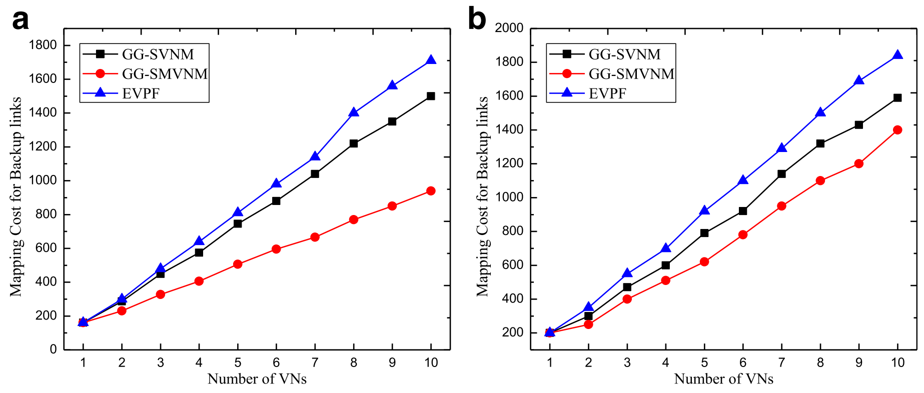

Fig. 9 Mapping costs for backup links. **a** Simulation results on Net-1. **b** Simulation results on Net-2

multiple VNs achieved by using the GG-SMVNM algorithm are less than that of the GG-SVNM algorithm.

Furthermore, Fig. 7 shows the simulation results on total mapping costs under various reliability requirements. For a specific reliability requirement, our proposed algorithms have lower total mapping costs than the existing approach since our approaches can efficiently deploy the arrived virtual network requests and thus consume less physical network resources. Moreover, the mapping costs of the compared algorithms increase with the increasing reliability requirements since it is necessary to allocate greater and more expensive resources for a VN request with higher reliability demand to guarantee the reliability.

Figure 8 depicts the backup nodes' mapping costs in the GG-SVNM, GG-SMVNM, and EVPF algorithms for multiple VNs. Figure 9 shows the physical resource costs for mapping the backup links of VNs using the EVPF, GG-SVNM, and GG-SMVNM algorithms, respectively. We can see from Figs. 8 and 9 that the GG-SMVNM algorithm achieves the lowest costs as the number of VNs increases. Furthermore, compared with the advantage in backup node mapping costs, the advantage in backup link mapping costs is more obvious. By analyzing the reason for this simulation result, as we said before, the resource sharing is in each VN, and the node and link resource sharing occurs across VNs while using the GG-SMVNM algorithm. Since the geographic location constraints and virtual nodes from different VNs may be abstracted to a new virtual node, the backup node resource sharing in VNs can be realized. In comparison, the backup link mapping is much more complicated and the probability of sharing resources among backup links of different VNs is much higher. Therefore, more resource sharing opportunities exist in the GG-SMVNM algorithm than in the GG-SVNM algorithm, which leads to lower mapping costs.

Figure 10 shows the simulation results on the average deployment time of the GG-SVNM, GG-SMVNM, and EVPF algorithms for multiple VN requests. In this set of simulations, we evaluate the performance of the compared algorithms under different numbers of VN requests and calculate the average value to eliminate the randomness and output it as the result. From the figure, we can see that our proposed GG-SMVNM algorithm has the lowest time complexity, whereas the EVPF has

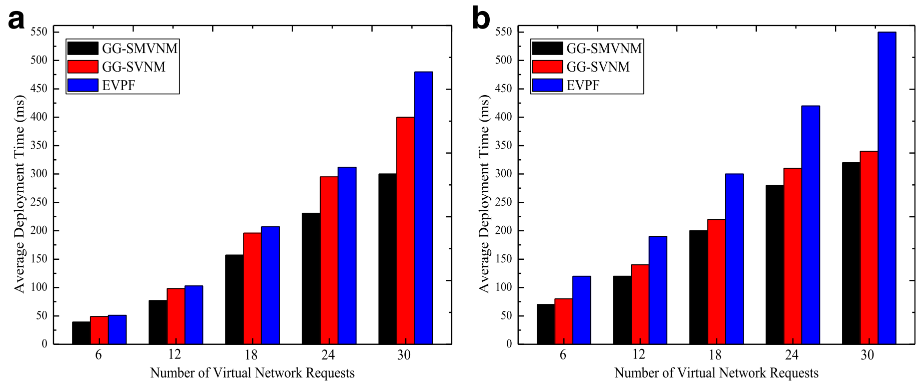

Fig. 10 Simulation results for deployment time. **a** Simulation results on Net-1. **b** Simulation results on Net-2

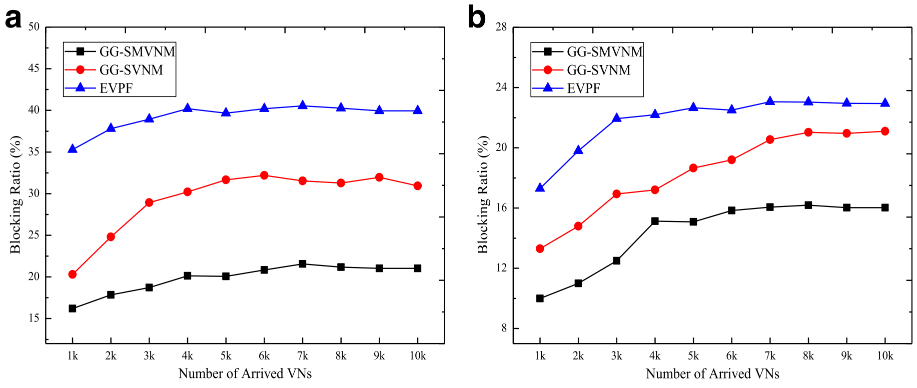

Fig. 11 Simulation results on blocking ratio. **a** Simulation results on Net-1. **b** Simulation results on Net-2

the worst time efficiency. Since an efficient routing strategy is used in our proposed algorithm, it can be used to quickly map a VN link onto a feasible physical path.

The simulation results shown in Fig. 11 evaluate the blocking ratios of the compared algorithms in different scenarios. The blocking ratio is defined as the number of blocked/rejected virtual network requests to the number of total arrived VN requests. The blocking ratios increase with the growth of the number of arrived VNs, since more VN requests means more resource consumption. However, the blocking ratios remain stable when the number of arrived VN requests is more than 4000. The blocking ratio of our proposed algorithm is lower than that of the existing approach EVPF, since our approaches consume less physical network resources while guaranteeing the same level of reliability of VN requests, thus lowering the blocking ratio.

5 Conclusions

In this paper, we propose a survivable VN mapping algorithm (GG-SMVNM) that considers the geographic location constraints of virtual nodes for efficiently mapping multiple VNs in edge-of-things computing. We introduce the resource sharing strategy in VNs and also across multiple VNs to save the physical resources of edge-of-things computing and reduce the mapping costs. Furthermore, the geographic location constraints are considered in both original virtual nodes mapping and backup node mapping, which makes significant sense in real edge-of-things computing applications. We conduct extensive simulations on different networks to evaluate our proposed algorithms. The simulation results show that our proposed algorithm has better performance than existing approaches.

In this research, we mainly focus on the problem of provisioning VN requests with location constraints in an autonomous domain network in edge-of-things computing. However, in practical applications, there are some VNs need to be deployed onto multiple autonomous domain networks. Therefore, in our future research, we are going to study and solve the problem of reliable VN mapping in multiple domains while considering the quality of service (QoS) requirements.

Acknowledgements
Not applicable.

Funding
This research was partially supported by the Fundamental Research Funds of China West Normal University (No.17D075).

Availability of data and materials
Not applicable.

Authors' contributions
SZ is in charge of the major theoretical analysis, algorithm design, and numerical simulations. AKS is in charge of part of the theoretical analysis and algorithm design. Both authors read and approved the final manuscript.

Competing interests
Both authors declare that they have no competing interests.

Publisher's Note

Author details
[1]China West Normal University, Nanchong, China. [2]School of Computing Science and Engineering, Vellore Institute of Technology (VIT), Vellore, Tamil Nadu 632014, India.

Received: 24 January 2018 Accepted: 9 March 2018
Published online: 16 March 2018

References
1. W Shi, J Cao, Q Zhang, et al., Edge computing: vision and challenges. IEEE Internet Things J **3**(5), 637–646 (2016)
2. TX Tran, A Hajisami, P Pandey, et al., Collaborative mobile edge computing in 5G networks: new paradigms, scenarios, and challenges. IEEE Commun. Mag. **55**(4), 54–61 (2017)
3. S Sardellitti, G Scutari, S Barbarossa, Joint optimization of radio and computational resources for multicell mobile-edge computing. IEEE Trans Signal Inf Proc Over Netw **1**(2), 89–103 (2015)
4. G Sun, V Chang, M Ramachandran, et al., Efficient location privacy algorithm for internet of things (IoT) services and applications. J. Netw. Comput. Appl. **89**, 3–13 (2017)

5. X Sun, N Ansari, EdgeIoT: mobile edge computing for the Internet of Things. IEEE Commun. Mag. **54**(12), 22–29 (2016)
6. J Li, X Huang, J Li, et al., Securely outsourcing attribute-based encryption with checkability. IEEE Trans Parallel Syst **25**(8), 2201–2210 (2014)
7. X Chen, X Huang, J Li, et al., New algorithms for secure outsourcing of large-scale systems of linear equations. IEEE Trans Inf Forensics Secur **10**(1), 69–78 (2015)
8. J Li, J Li, X Chen, et al., Identity-based encryption with outsourced revocation in cloud computing. IEEE Trans. Comput. **64**(2), 425–437 (2015)
9. G Sun, D Liao, D Zhao, et al., Live migration for multiple correlated virtual Machines in Cloud-based Data Centers. IEEE Trans. Serv. Comput., 1–14 (2016)
10. J Li, J Li, D Xie, et al., Secure auditing and deduplicating data in cloud. IEEE Trans. Comput. **65**(8), 2386–2396 (2016)
11. P Li, J Li, Z Huang, et al., Privacy-preserving outsourced classification in cloud computing. Clust. Comput., 1–10 (2017)
12. J Li, Y Zhang, X Chen, et al., Secure attribute-based data sharing for resource-limited users in cloud computing. Comput Secur **72**, 1–12 (2018)
13. G Sun, D Liao, D Zhao, et al., Towards provisioning hybrid virtual networks in federated cloud data centers. Futur. Gener. Comput. Syst. (2017) Available online 18 October
14. Y Zhang, X Chen, J Li, et al., Ensuring attribute privacy protection and fast decryption for outsourced data security in mobile cloud computing. Inf. Sci. **379**, 42–61 (2017)
15. G Sun, V Anand, D Liao, et al., Power-efficient provisioning for online virtual network requests in cloud-based data centers. IEEE Syst. J. **9**(2), 427–441 (2015)
16. P Li, J Li, Z Huang, et al., Multi-key privacy-preserving deep learning in cloud computing. Futur. Gener. Comput. Syst. **74**, 76–85 (2017)
17. S Su, Z Zhang, A Liu, et al., Energy-aware virtual network embedding. IEEE/ACM Trans. Networking **22**(5), 1607–1620 (2014)
18. R Mijumbi, JL Gorricho, J Serrat, et al., A neuro-fuzzy approach to self-management of virtual network resources. Expert Syst. Appl. **42**(3), 1376–1390 (2015)
19. J Li, X Chen, X Huang, et al., Secure distributed deduplication systems with improved reliability. IEEE Trans. Comput. **64**(12), 3569–3579 (2015)
20. G Sun, V Chang, G Yang, et al., The cost-efficient deployment of replica servers in virtual content distribution networks for data fusion. Inf. Sci. **432**, 495-515 (2017) Available online 10 August
21. J Li, Y Li, X Chen, et al., A hybrid cloud approach for secure authorized deduplication. IEEE Trans Parallel Distrib Syst **26**(5), 1206–1216 (2015)
22. G Sun, D Liao, V Anand, et al., A new technique for efficient live migration of multiple virtual machines. Futur. Gener. Comput. Syst. **55**, 74–86 (2016)
23. H Yu, T Wen, H Di, et al., Cost efficient virtual network mapping across multiple domains with joint intra-domain and inter-domain mapping. Opt. Switch. Netw. **14**, 233–240 (2014)
24. J Li, Z Liu, X Chen, et al., L-EncDB: a lightweight framework for privacy-preserving data queries in cloud computing. Knowl.-Based Syst. **79**, 18–26 (2015)
25. G Sun, H Yu, V Anand, et al., A cost efficient framework and algorithm for embedding dynamic virtual network requests. Futur. Gener. Comput. Syst. **29**(5), 1265–1277 (2013)
26. NMK Chowdhury, MR Rahman, R Boutaba, Virtual network embedding with coordinated node and link mapping. IEEE Infocom, 783–791 (2009)
27. WL Yeow, C Westphal, UC Kozat, Designing and embedding reliable virtual infrastructures. ACM Sigcomm Comput Commun Rev **41**(2), 57–64 (2011)
28. Z Cai, F Liu, N Xiao, et al., Virtual network embedding for evolving networks. IEEE Globecom, 1–5 (2010)
29. H Yu, C Qiao, V Anand, et al., Survivable virtual infrastructure mapping in a federated computing and networking system under single regional failures. IEEE Globecom, 1–6 (2010)
30. G Sun, H Yu, L Li, et al., The framework and algorithms for the survivable mapping of virtual network onto a substrate network. IETE Tech. Rev. **28**(5), 381–391 (2011)
31. H Yu, V Anand, C Qiao, et al., Cost efficient design of survivable virtual infrastructure to recover from facility node failures. IEEE ICC, 1–6 (2011)
32. G Sun, D Liao, S Bu, et al., The efficient framework and algorithm for provisioning evolving VDC in federated data centers. Futur. Gener. Comput. Syst. **73**, 79–89 (2017)