Resolve The Virtual Network Embedding Problem: A Column Generation Approach

Qian Hu, Yang Wang, Xiaojun Cao
Department of Computer Science
Georgia State University
Atlanta, Georgia 30303
Email:{qhu, ywang, cao}@cs.gsu.edu

Abstract—In this paper, we study the virtual network embedding (VNE) problem in the network virtualization context, which aims at mapping the virtual network requests of the service providers (SPs) to the substrate networks managed by the infrastructure providers (InPs). Given the NP-Completeness of the VNE problem, prior approaches primarily rely on solving/relaxing the link-based Integer Linear Programming (ILP) formulations, which lead to either extensive computational time, or non-optimal solutions. In this paper, for the first time, we present a path-based model for the VNE problem, namely P-VNE. By analyzing the dual formulation of the P-VNE model, we propose a column generation process, with which an optimal solution to the VNE problem can be found efficiently (when embedded into a branch-and-bound framework).

I. INTRODUCTION

Network virtualization is considered to be one promising solution to overcome the ossification of the current Internet [1]–[5]. With network virtualization, the traditional Internet Service Providers (ISPs) are decoupled into two independent parties or tiers: the infrastructure provider (InP), and the service provider (SP). To bridge the InP and SP, one of the most challenging problems raised in the network virtualization context is the Virtual Network Embedding problem, where the customized virtual network requests from the SPs are mapped to the substrate or physical networks managed by the InPs [5].

Virtual network embedding consists of two major components: the mapping of the virtual nodes (with computational capacity requirement) to the substrate nodes; and the mapping of the virtual links (with bandwidth capacity requirement) to the substrate path(s) [5]. Given the NP-Completeness of the network embedding problem [6], existing approaches can be broadly classified into three categories (see, for example, [7]–[13]): (i) optimal solutions based on solving the link-based Integer Linear Programming (ILP) formulations for the VNE problem; (ii) approaches based on the relaxation of the link-based VNE ILP formulations (e.g., relaxation and rounding in [9]); and (iii) heuristic algorithms (e.g., [7]). We note that the first suffers from the extensive computational time of the ILP solver in practice, while the latter two cannot provide an optimal solution or near-optimal solutions with guarantee on the performance.

Different from prior work, in this paper, we present a path-based ILP model for the VNE problem, namely P-VNE. Based on the dual formulations of the P-VNE model, a column generation process is presented, which can be embedded into a branch-and-bound framework to effectively resolve the VNE problem optimally in practice. To the best of our knowledge, this work is the first that presents the path-based VNE model, and proposes the column generation process to obtain an optimal solution to the VNE problem in practice.

The remaining of this paper is organized as follows. In Section II, we present the system model and problem definition of this work. In Section III, we present the path-based Integer Linear Programming model for the VNE problem. In Section IV, we elaborate the column generation process after presenting the dual formulations of the P-VNE model. We present the performance study in Section V, and conclude this paper in Section VI.

II. NETWORK MODEL AND PROBLEM DEFINITION

In this work, we model the virtual network as an undirected weighted graph \( G^V = (N^V, L^V) \), where \( N^V \) is the set of virtual nodes with computing requirements, and \( L^V \) is the set of virtual links with bandwidth demands (and/or QoS information such as the latency). The computing resource requirement of Node \( a \) is denoted as \( cr(a) \). The bandwidth demand of a virtual link (say \( a-b \)) reflects the bandwidth requirement between the two adjacent nodes (i.e., \( a \) and \( b \)) of the link, denoted by \( br(a-b) \). For example, Figure 1(a) shows a virtual network consisting of three virtual nodes and three virtual links. The numbers beside the nodes and links represent the computing and bandwidth.
requirements respectively. Figure 1(b) shows a virtual network with only two virtual nodes and one virtual link.

![Virtual network 1 and 2](attachment:virtual_network.png)

**Fig. 1.** Embedding of Virtual Networks into a Substrate Network

Similarly, the substrate network is modeled as an undirected weighted graph \( G^S = (N^S, L^S) \), where \( N^S \) is the set of substrate nodes, and \( L^S \) is the set of substrate links. The computing capacity of a substrate node, say \( i \), is denoted as \( cc(i) \). The bandwidth capacity of a substrate link \( e \) is denoted as \( \mu_e \). The Virtual Network Embedding (VNE) problem can then be formally defined as a decision problem as shown in Definition 1. Aligned with the definition, Figure 1(c) shows one possible mapping scenario of the two virtual networks onto a substrate network which consists of five substrate nodes, connected by seven substrate links. The numbers beside the nodes and links in Fig. 1(c) represent the node and link capacities respectively.

**Definition 1: VNE Problem** Given the virtual network \( G^V = (N^V, L^V) \), and substrate network \( G^S = (N^S, L^S) \), can the virtual network be mapped to the substrate network while satisfying the follow requirements: (i) node mapping requirement: one virtual node is mapped onto one substrate node, with no two virtual nodes (from the same virtual network) share the same substrate node; (ii) link mapping requirement: one virtual link is mapped to path(s) between the two substrate nodes which hold the two virtual nodes of this virtual link; (iii) capacity constraint: for each virtual node/link, it is mapped to the substrate node/path(s) with sufficient computing/bandwidth capacity?

To facilitate the ILP modeling in the next section, we view the virtual network embedding process as a multi-commodity flow problem (i.e., one commodity per virtual link). This is achieved by constructing the auxiliary graph (AUG) that couples the virtual and the substrate networks as shown in Fig. 2. Specifically, for each virtual node (e.g., Node \( a \) in Fig. 1(a)), we connect it to a group of substrate network nodes (e.g., substrate Node 1, 2 for virtual node \( a \) in Fig. 2) which satisfies the node capacity constraint (i.e., \( cr(a) \leq cc(1) \), and \( cr(a) \leq cc(2) \)). The set of auxiliary links is denoted as \( AE \) (e.g., link \( a-1 \in AE \)). With this construction, a feasible flow between two virtual nodes in the auxiliary graph (e.g., Node \( a \) and \( b \) in Fig. 2) actually represents a feasible mapping of the corresponding virtual link (i.e., \( a-b \) of Fig. 1(a)).

![Auxiliary graph](attachment:auxiliary_graph.png)

**Fig. 2.** The Auxiliary Graph for the VNE Modeling

### III. MILP FORMULATIONS WITH PATH FLOWS FOR VIRTUAL NETWORK EMBEDDING

In this section, we present the path-based mixed integer linear programming (MILP) model for the virtual network embedding problem. We refer to this model as the P-VNE model, with notations described in the table below.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( f_p )</td>
<td>the flow on the path ( p ); ( f_p \geq 0; )</td>
</tr>
<tr>
<td>( c_p )</td>
<td>the per unit cost of the path ( p );</td>
</tr>
<tr>
<td>( x_{i,j} )</td>
<td>1 if virtual node ( i ) is mapped on to the physical node ( j ), 0 otherwise;</td>
</tr>
<tr>
<td>( \mu_e )</td>
<td>the bandwidth capacity of physical link ( e );</td>
</tr>
<tr>
<td>( P_k )</td>
<td>the set of path for virtual link (commodity) ( k );</td>
</tr>
<tr>
<td>( r_k )</td>
<td>the bandwidth requirement of each virtual link (commodity) ( k );</td>
</tr>
<tr>
<td>( \delta_{p,I,j} )</td>
<td>1 if link ( (I,j) ) is on the path ( p ), otherwise 0;</td>
</tr>
<tr>
<td>( M )</td>
<td>a big positive number;</td>
</tr>
<tr>
<td>( c_c )</td>
<td>the per unit cost of link ( e );</td>
</tr>
<tr>
<td>( P )</td>
<td>the set of all paths for all the commodities.</td>
</tr>
</tbody>
</table>

In this model, the objective is to minimize the overall cost of the flow, as shown in Eq. (1), where \( c_p \) is the per unit cost of path \( p \), and \( c_p = \sum_{e \in p} c_c \). Note that the node resources are excluded in the objective function since the consumed node resources are fixed for any mapping scheme.

\[
\min \left( \sum_{p \in P} c_p f_p \right) \tag{1}
\]

The constraint shown in Eq. (2) ensures that the allocated resources upon each physical link is limited within the capacity bound.

\[
\sum_{p \in P} f_p \leq \mu_e, \forall e \tag{2}
\]
To ensure that all the commodities (i.e., the virtual links) is satisfied, the aggregated flow over all the paths of the given commodity should be equal to the demand size of the commodity, which is reflected in Eq. (3).

$$\sum_{p \in P_k} f_p = r_k, \forall k$$  \hspace{1cm} (3)

The constraints shown in Eq. (4) and Eq. (5) achieve the node mapping. Specifically, the constraints in Eq. (4) guarantees that no more than one virtual node resides in the same substrate node, while Eq. (5) forces that each virtual node is mapped to exactly one substrate node.

$$\sum_{I,(I,i) \in AE} x_{I,i} \leq 1, \forall i \in N^S$$  \hspace{1cm} (4)

$$\sum_{I,(I,i) \in AE} x_{I,i} = 1, \forall I \in N^V$$  \hspace{1cm} (5)

Finally, a non-zero path flow passes through the auxiliary link \((I,i)\) only when virtual node \(I\) is mapped onto physical node \(i\), which is ensured by Eq. (6). In Eq. (6), \(M\) is a big number. Specifically, when \(x_{I,i} = 1\), the dominant \(M\) ensures that the right term is greater than the left. When \(x_{I,i} = 0\), we ensure that no path flow can pass the auxiliary link \((I,i)\).

$$\sum_{p \in P^d(I,i)} f_p \leq x_{I,i}M, \forall (I,i) \in AE, I \in N^V, i \in N^S$$  \hspace{1cm} (6)

It worth noting that: first, the size of the set \(P (= \cup_k P_k)\) can be exponential; second, not all the paths over the auxiliary graph can be included in the set \(P\). For instance, for the commodity between node-pair \((a, b)\) in Fig. 2, one cannot adopt the path going through other virtual nodes (e.g., the path \(a-1-5-c-4-3-b\) passing virtual node \(c\) is not legitimate). We name the path for a given node-pair that only goes through substrate nodes (except the source and the sink node) as a legitimate path. In the formulation, we hence have to ensure that all the paths in \(P\) are legitimate paths.

To deal with the first issue, we have two approaches in practice. In both ways, we limit the number of chosen paths \(P^c (\subset P)\). The resulted formulation is referred to as \(\text{P-VNE}'\). One way is to directly solve the smaller \(\text{P-VNE}'\) problem at the expense of the optimality. The other way, is to apply the column generation approach and the branch-and-bound framework, as elaborated in the next sections. To tackle the second issue, we will show in the next section how to guarantee the chosen paths are always legitimate.

IV. COLUMN GENERATION FOR THE LP-VNE

In this section, we present a column generation based scheme to solve the linear relaxation of the P-VNE problem, namely LP-VNE. The idea of column generation is to start with a limited set of paths \(P' (i.e., the \text{LP-VNE}'\) problem), and incrementally obtain the optimal set of paths via a column generation process that gradually incorporates new paths into the solution base.

To facilitate the presentation of the column generation scheme, we first present the dual of the LP-VNE problem below (Eq. (7), Eq. (8), Eq. (9), Eq. (10)), which is referred to as \(\text{D-LP-VNE}'\). The variables \(y_e, \lambda_k, y_I, y_{I,i}\) and \(\pi_{I,i}\) are the dual variables for the constraints shown in Eq. (2), Eq. (3), Eq. (4), Eq. (5) and Eq. (6) respectively. The path reduced cost is reflected in Eq. (11) after a transformation of Eq. (8).

$$\max \left( \sum_k r_k \lambda_k - \sum_e \mu_e y_e - \sum_I y_I + \sum_i y_{I,i} \right)$$  \hspace{1cm} (7)

$$- \sum_{e \in P^c} y_e + \lambda_k - \sum_{I,(I,i) \in P^c} \pi_{I,i} \leq c_p, \forall p \in P_k, \forall k$$  \hspace{1cm} (8)

$$y_I - y_{I,i} + M \pi_{I,i} \leq 0, \forall (I, i) \in AE$$  \hspace{1cm} (9)

$$y_e, y_I, \pi_{I,i} \geq 0, \lambda_k, y_{I,i}, \text{ unrestricted}$$  \hspace{1cm} (10)

$$\sum_{e \in P^c} (y_e + c_e) + \sum_{I,(I,i) \in P^c} \pi_{I,i} - \lambda_k \geq 0, \forall p \in P_k, \forall k$$  \hspace{1cm} (11)

We note that a feasible solution \(< f_p^*, \geq p \in P^c > \text{LP-VNE}'\) yields a feasible solution \(< f_p \geq p \in P \text{LP-VNE}\) by setting \(f_p = f_p^*\) for \(p \in P^c\) and \(f_p = 0\) for \(p \in P - P^c\). Since \(P^c \subset P\), the optimal solution value \(\text{OPT} (\text{LP-VNE}')\) is no less than the optimal solution of LP-VNE (i.e., \(\text{OPT} (\text{LP-VNE}') \geq \text{OPT} (\text{LP-VNE})\)). In addition, due to the LP duality, we have the equivalent objective value when solving the primal and the dual problem optimally. Consequently, we have the relations among the objectives of above problems as summarized in Eq. (12).

$$\text{OPT (D-LP-VNE')} = \text{OPT (LP-VNE') } \geq \text{OPT (LP-VNE)} = \text{OPT (D-LP-VNE)}$$  \hspace{1cm} (12)

Next, we show that the LP-VNE problem can be solved in polynomial time, as shown in Theorem 1, although potentially there exist an exponential number of paths. This fact can be established by showing that the dual of the LP-VNE, i.e., D-LP-VNE has a polynomial time separation oracle, which can take a given candidate solution and either confirm the
feasibility or find a violated constraint in polynomial time. This feature in turn indicates that the LP-relaxation can be solved in polynomial time using the ellipsoid algorithm [14]. The detailed proof of Theorem 1 is omitted due to the space limitation, and will be presented in an extended version of this work.

**Theorem 1:** The LP-VNE can be solved in polynomial time.

The idea of our column generation process, is to gradually add paths to D-LP-VNE’ that cause the dual D-LP-VNE infeasible (i.e., with negative reduced cost). When no paths with negative reduced cost can be added to D-LP-VNE’, we have \( \text{OPT(D-LP-VNE’)} = \text{OPT(D-LP-VNE)} \), which further guarantees that \( \text{OPT(LP-VNE’)} = \text{OPT(LP-VNE)} \) based on Eq. (12), and leads to the optimal solution. To ensure that all the paths are legitimate paths, we construct the induced AUG for each commodity where all the other virtual nodes are isolated. For instance, when computing the shortest path for the node-pair \((a, b)\), the virtual node \(c\) is disconnected in the corresponding subgraph shown in Fig. 3. In this way, one can guarantee the shortest path found on the subgraph is always legitimate for the respective commodity. The detailed column generation process is presented in Algorithm 1. In Algorithm 1, a path with negative reduced cost is found by running the shortest path algorithm on the induced AUG between the source and the destination of each commodity, say Commodity \(k\), where the obtained shortest path cost is compared to the dual variable associated with the commodity, i.e., \(\lambda_k\).

The column generation process can be embedded into the branch-and-bound framework [15] to prune the solution space and efficiently obtain the optimal solution for the VNE problem. In the performance study presented below, the results are based on our implementation of the branch-and-bound framework.

**V. PERFORMANCE EVALUATION AND ANALYSIS**

In this section, we evaluate the proposed scheme and compare the performance with other approaches. We randomly generate the virtual network requests with the number of nodes ranging in [2, 10]. The size of the substrate network falls in the range of [10, 50]. Similar to [9], both the virtual and substrate network average connectivity is 50%. The bandwidth/computational requirement of the virtual network link/node is randomly generated within [1, 20], while the link/node capacity of the substrate network is randomly generated within [1, 50]. Thousands of the instances are simulated and the average performance is reported below.

**A. Resource Consumption**

The comparison of the total consumed resources is achieved by setting \(c_e = 1, \forall e\), as shown in Fig. 4. The link-based ILP is the formulations adapted from [9]. ILP P-VNE refers to the optimal solution obtained with the approach proposed in this study. ILP P-VNE’\((k=n)\) refers to the path-based formulation with limited path variables, where \(k\) specifies the number of shortest paths for each commodity provided as the input. In Fig. 4, the X-axle represents the size of the virtual network, while the Y-axle denotes the objective value (i.e., total consumed resources) after solving the VNE problem using different approaches. In all the cases, we set the size of the substrate network to be 50. Clearly, both the Link-based ILP and ILP P-VNE can obtain the optimal value in all the cases. However, in our experiments, the proposed column generation approach can obtain the optimal solution much faster. For the ILP P-VNE’ with various \(k\), one can see that increasing \(k\) clearly improves

---

**Algorithm 1** Column Generation for Virtual Network Embedding

1: \(\text{repeat}\)
2: \(\text{Solve the LP-VNE’ with the current } P'\)
3: \(\text{Obtain the dual variable } \pi, \lambda, \text{ and } y\)
4: \(\text{For each substrate network edge, say } e, \text{ assign the } y_e + c_e \text{ as the edge weight}\)
5: \(\text{Assign } \pi \text{ as the weight of the auxiliary edges between the virtual node and the substrate node}\)
6: \(\text{for each commodity } k \text{ corresponding to the virtual edge}\)
7: \(\text{In the induced AUG of Commodity } k, \text{ find a shortest } s_k - d_k \text{ path, assuming the weight of the selected path}\)
8: \(\text{if } c_p < \lambda_k \text{ then}\)
9: \(P' \leftarrow P' \cup p\)
10: \(\text{end if}\)
11: \(\text{end for}\)
12: \(\text{until No path can be added in the current iteration}\)
the performance, but the increase from \( k = 2 \) to \( k = 3 \) is more obvious than that of from \( k = 1 \) to \( k = 2 \).

\[\text{Fig. 4. The Comparison of the Consumed Resources}\]

\[\text{B. QoS Comparison}\]

The proposed model can also incorporate the QoS metrics into consideration. In our evaluation, we compare above approaches when setting the \( c_e = \text{lat}_e \), i.e., the latency of the link \( e \). The link latency is randomly generated within \([0, 1, 1] (\text{second})\). The overall latency comparison is shown in Fig. 5. Again, we can see that the ILP P-VNE and the link-based ILP can obtain the least latency in the embedding. Similarly, increasing \( k \) in ILP P-VNE’ reduces latency due to the increased path search space, which is at the expense of the computational time.

\[\text{Fig. 5. The Comparison of the Overall Latency}\]

VI. CONCLUSION

In this paper, for the first time, we have presented a path-based Integer Linear Programming (ILP) model for the virtual network embedding (VNE) problem. Based on the dual analysis of the proposed model, we have designed a column generation process, which can be embedded into a branch-and-bound framework to efficiently obtain an optimal solution for the VNE problem. When the branch-and-bound framework is well designed, the overall approach can also achieve a near-optimal solution with per-instance guarantee on the closeness to the optimality in a timely manner, which will be investigated in our future work. In addition, we plan to extend the proposed scheme to a survivable virtualization context.

Acknowledgement

This work is supported in part by the National Science Foundation (NSF) under CNS-1117229, and GSU Ph.D Dissertation Grant to Yang Wang.

References

Internet impasse through virtualization,” in Proceedings of the 3rd ACM
Workshop on Hot Topics in Networks (HotNets-III), 2004, pp. 34–41.
Internet impasse through virtualization,” IEEE Computer, vol. 38, no. 4,
spare time,” ACM SIGCOMM Computer Communication Review, vol. 37,
art and research challenges,” IEEE Communications Magazine, vol. 47,
unpublished Manuscript.
on subgraph isomorphism detection,” in Proceedings of the 1st ACM
workshop on Virtualized infrastructure systems and architectures, 2009,
pp. 81–88.
5634–5640.
embedding with coordinated node and link mapping,” in Proceedings of
IEEE INFOCOM’09, April 2009, pp. 783–791.
resources to virtual network components,” in Proceedings of IEEE
INFOCOM’06, April 2006, pp. 1–12.
embedding: Substrate support for path splitting and migration,” ACM
29, April 2008.
optimization mechanism for virtual network embedding,” in Proceedings
“Virtual network embedding through topology-aware node ranking,” ACM
its consequences in combinatorial optimization,” Combinatorica, vol. 4,