Estimating the Performance of VNE Algorithms with a Novel Loss Network Model

Qiao Lu
Department of Systems and Computer Engineering
Carleton University
Ottawa, Canada
qiaolu@sce.carleton.ca

Changcheng Huang
Department of Systems and Computer Engineering
Carleton University
Ottawa, Canada
huang@sce.carleton.ca

***Abstract*—A large number of virtual network embedding (VNE) algorithms have been proposed in the literature. Their performances have always been evaluated through simulations due to the formidable difficulty of developing analytical models for the performance of VNE algorithms. In this paper, we propose a novel loss network model with Dynamic Routing And Random Topology (DRART) that allows us to estimate the blocking probability of mapping a virtual link (VL). Furthermore, by integrating our new model with other existing models, we can estimate the blocking probability of VNE algorithms quite accurately. Our analytical model can be used as a benchmark for comparing different VNE algorithms as well as a tool to evaluate an operating virtual network service where substrate nodes and links can fail or be put into maintenance randomly.**

***Index Terms*—Loss network model, Virtual network embedding, Performance analysis, Random topology, Dynamic routing**

I. Introduction

Network virtualization plays an important role in software-defined networks (SDN) as well as network function virtualization (NFV). While the research on virtual network embedding(VNE) algorithms is extremely rich [1]–[5], performance estimation of these algorithms based on analytical models almost does not exist to our best knowledge. These algorithms have been evaluated based on simulation results, which are typically limited to simulation setups and specific network topology scenarios. The challenge of performance modeling lies in the complexity of VNE. For example, a blocking event can happen at a substrate link/node, a virtual link/node, or a virtual network level. We need models for all three levels to capture the performance of a VNE algorithm. Furthermore, a model developed under a specific substrate topology cannot be used to evaluate the performances of various embedding algorithms due to the lack of generality. By assuming a random substrate network (SN) topology, the difficulty of calculating blocking probability increases dramatically. A VL can have multiple candidate paths in an SN. This kind of dynamic routing and random topology requirements have not been studied so far to our best knowledge. Comprehensive coverage of various loss models can be found in [6]. The closest one is the so-called loss network with fixed routing, which is still quite far from our case here.

In this paper, we propose a systematic approach to evaluate the performance of embedding algorithms, which includes the following major components:

- We adopt the existing generalized Erlang loss model [7] for estimating substrate link (SL) blocking probability;
- We create a novel loss network model with Dynamic Routing And Random Topology (DRART) for estimating blocking probability at the VL level;
- We incorporate concepts from Jackson networks [8] for evaluating the performance of embedding at the virtual network(VN) level;
- We create a new recursive integration approach that forms a synergy among the three different levels.

The rest of the paper is organized as follows: Section II describes our approach in detail; Section III presents numerical results; Section IV concludes the paper.

II. System Models

In this section, we will start with the system environments and basic assumptions. Then, we will present our analytical results on the blocking probabilities of an SL, a VL, and a VN step-by-step with detailed proofs being omitted due to page limit. Interested readers can check [9] for more details.

A. System Environment and Basic Assumptions

To make our models general enough, we assume a generic environment used in previous VNE baseline solutions [2], [10].

An SN is represented as a random weighted undirected graph and denoted as $G_S(N_S, E_S)$, where N_S is a set of substrate nodes and $\widetilde{n}_s = |N_S|$ follows a probability distribution P_{N_S} with a maximum value $M_{S,N}$. $\bar{n}_s = \mathbb{E}(\widetilde{n}_s)$ is the average number of substrate nodes. E_S is a set of SLs and each pair of nodes have the probability $P_{S,L}$ to form a link $e_s \in E_S$. $\widetilde{e}_s = |E_S|$ is the number of SLs. $\bar{e}_s = \mathbb{E}(\widetilde{e}_s)$ is the average number of SLs. Each substrate node $n_s \in N_S$ is associated with a CPU capacity value c_{n_s} that follows a random distribution $P_{c_{n_s}}$ with a maximum CPU capacity C_{n_s} and a location (x_{n_s}, y_{n_s}) following some distribution $P_{L_{n_s}}(x, y)$ in a fixed area W. In this paper, we assume CPU capacity as the resource requirement for a node. We assume that each SL e_s between two substrate nodes has a random

bandwidth capacity b_{e_s} following the distribution $P_{b_{e_s}}$ with a maximum bandwidth capacity B_{e_s}.

Similarly, a virtual network request (VNR) is represented as a random weighted graph, denoted by $G_V\left(N_V,\ E_V, t_a, t_d,\ D\right)$. N_V is a set of virtual nodes. $\widetilde{n}_v = |N_V|$ follows a probability distribution P_{N_V} with a maximum value $M_{V,N}$. $\bar{n}_v = \mathbb{E}(\widetilde{n}_v)$ is the average number of virtual nodes. E_V is a set of VLs and each pair of nodes has the probability $P_{V,L}$ to form a VL $e_v \in E_V$. $\widetilde{e}_v = |E_V|$ is the number of VLs in the VN. $\bar{n}_v = \mathbb{E}(\widetilde{e}_v)$ is the average number of VLs. t_a is the arrival time of a VNR, which follows a Poisson process with an arrival rate λ_V. t_d is the duration of the VNR, which follows an exponential distribution with a mean holding time τ_V. Each virtual node $n_v \in N_V$ is associated with a CPU capacity requirement c_{n_v} that follows a random distribution $P_{c_{n_v}}$ with a maximum CPU capacity requirement C_{n_v} and a location (x_{n_v}, y_{n_v}) following some distribution $P_{L_{n_v}}(x, y)$ in the same fixed area W. A substrate node is called a candidate node for the virtual node if the substrate node is located within the distance D to the virtual node. We assume e_v between two virtual nodes has a random bandwidth requirement b_{e_v} following the distribution $P_{b_{e_v}}$ with a maximum bandwidth requirement B_{e_v}.

B. Substrate Node/Link Blocking Probability

In this subsection, we focus on SL blocking probability. Substrate node blocking probability can be calculated exactly in the same way by replacing bandwidth with CPU capacity.

We first assume VNRs arrive at an SL following Poisson distribution with a mean rate $\lambda_{S,L}$. Each VNR at an SL is associated with a VL of a VNR. The holding time of a request received at an SL follows the same exponential distribution with the same mean holding time τ_V as the VN holding time. The bandwidth requirement $b_{e_s} \in \mathbb{R}$ also follows the same distribution $P_{b_{e_v}}$ as the VL.

There is no existing way to calculate the blocking probability when b_{e_v} is a random real number. The generalized Erlang loss model [7] is the closest one that can be used to approximate this blocking probability. However, the Erlang model requires the requested bandwidth and link capacity to be discrete. In order to use Erlang model, we have to quantize bandwidth requests into a fixed number of intervals denoted as R. We approximate each interval r: $(b_{e_v,r}, b_{e_v,r+1}], 1 \le r \le R$ with one bandwidth request $b_r = \int_{b_{e_v,r}}^{b_{e_v,r+1}} b_{e_v} dP_{b_{e_v}}$, which is the average bandwidth of the interval. The probability that a request has bandwidth b_r will be $P_r = \int_{b_{e_v,r}}^{b_{e_v,r+1}} dP_{b_{e_v}}$. The arrival rate within each interval is $\lambda_{S,L,r} = \lambda_{S,L} P_r$. The load of each interval is $\rho_{S,L,r} = \lambda_{S,L,r}\tau_V$. We also quantize the bandwidth capacity of SL b_{e_s} into U intervals with each interval $u : (b_{e_s,u}, b_{e_s,u+1}], 1 \le u \le U$. We approximate each interval with one bandwidth capacity $b_u = \int_{b_{e_s,u}}^{b_{e_s,u+1}} b_{e_s} dP_{b_{e_s}}$. The probability that an SL has the capacity b_u is $P_u = \int_{b_{e_s,u}}^{b_{e_s,u+1}} dP_{b_{e_s}}$.

Assume at a specific time the link is hosting h_r requests with bandwidth b_r for each interval r. Let $\vec{h} = (h_1, \cdots h_R)$, then the set of all feasible combinations that can be carried by a specific SL is shown in (1).

$$\mathcal{F}(b_u) = \left\{\vec{h} \ge 0 :\ \textstyle\sum_{r\in R} b_r h_r \le b_u\right\} \tag{1}$$

Following the generalized Erlang loss model [7], the average blocking probability of the SL is:

$$P_{S,L}^{(B)} = \textstyle\sum_{u\in U} P_u \sum_{r\in R} \left(1 - \frac{G(b_u - b_r)}{G(b_u)}\right) P_r, \tag{2}$$

where

$$G(b_u) = \textstyle\sum_{\vec{h}\in\mathcal{F}(b_u)} \prod_{r\in R} \frac{\rho_{S,L,r}^{h_r}}{h_r!}. \tag{3}$$

Similarly, we can find the substrate node blocking probability $P_{S,N}^{(B)}$ by replacing bandwidth with CPU capacity.

C. The DRART Model for VL Blocking Probability

Each VL can be mapped to a substrate path that has enough residual bandwidth capacity in each SL the path traverses to support the VL bandwidth requirement. We first want to know the maximum number of potential paths there may be for a source-destination pair in an SN.

There are many paths that may exist between a source-destination pair, shorter paths are more favored for VNE due to the fact that the shorter paths use fewer link resources. However, shorter paths between a source-destination tend to have many joint links. Without taking any extra measures, if one short path is blocked, other short paths will likely be blocked too. This will make blocking probability higher as justified with simulation results in Section III. In addition, paths with joint links make the calculation of blocking probability more complex due to the complex correlation structure. we consider selecting candidate paths from link-disjoint paths only.

We first need to know how many link-disjoint paths exist for a given source-destination pair. This is quite challenging due to the random topology assumption.

Theorem 2.1. *For any source-destination pair in any SN with $\widetilde{n}_s$ nodes, the maximum number of potential link-disjoint paths that have e links is $K_{I,e} = \widetilde{n}_s - 2$, where $2 \le e < \frac{\widetilde{n}_s}{2} + 1$.*

We now try to estimate the probability of the existence of certain number of link-disjoint paths with various path lengths.

Theorem 2.2. *For any source-destination pair in any SN with $\widetilde{n}_s$ nodes, the probability that there exist at least k_e link-disjoint paths with length e can be approximately calculated by* (4).

$$P_I\left(k \ge k_e |\ K_{I,e}\right) \approx \widetilde{P}_I\left(k_e | K_{I,e}\right)^2 \check{P}_I\left(k_e |\ K_{I,e}\right), \tag{4}$$

where

$$\widetilde{P}_I\left(k_e |\ K_{I,e}\right) = \textstyle\sum_{i=k_e}^{K_{I,e}} C_i^{K_{I,e}} {P_{S,L}}^i \left(1 - P_{S,L}\right)^{K_{I,e}-i},$$

and

$$\check{P}_I\left(k_e |\ K_{I,e}\right) = \sum_{i=k_e(e-2)}^{L(\widetilde{n}_s-2)} C_i^{L(\widetilde{n}_s-2)} {P_{S,L}}^i \left(1 - P_{S,L}\right)^{L(\widetilde{n}_s-2)-i}.$$

By using subtraction operations shown in Corollary 2.1, we can obtain the probability of the existence of certain number of link-disjoint paths with a fixed path length e.

Corollary 2.1. *For any source-destination pair in any SN with $\widetilde{n}_s$ nodes, the probability that there exist exactly k_e link-disjoint paths with length e can be estimated as* (5)*.*

$$P_I(k_e|K_{I,e}) = P_I\left(k \geq k_e|\ K_{I,e}\right) - P_I\left(k \geq k_e + 1|\ K_{I,e}\right) \quad (5)$$

Now we move on to look at the probability that there exist multiple link-disjoint paths with different path lengths.

Corollary 2.2. *For any source-destination pair in any SN with $\widetilde{n}_s$ nodes, the joint probability of $\vec{k} = \{k_e,\ e = 1, \cdots, E_m\}$ link-disjoint paths are estimated as:*

$$P_I\left(\vec{k}|K_{I,e}\right) = P_I\left(k_1, \cdots, k_e|\widetilde{n}_s\right) \approx P_I(k_1) \prod_{j=2}^{e} P_I\left(k_j|\ k_{I,j}\right), \quad (6)$$

where

$$P_I(k_1) = \begin{cases} P_{S,L}, & \text{if } k_1 = 1 \\ 1 - P_{S,L}, & \text{if } k_1 = 0 \end{cases},$$

and

$$k_{I,\ j} = K_{I,j} - k_1 - k_2 - \cdots - k_{j-1}.$$

Using the chain rule [11], we know the distribution of available paths. We assume the blocking events at each SL are independent. Given a path with $e-1$ ordered substrate nodes ($\vec{n}_{s,e-1} = \{n_{s,1}, \cdots, n_{s,e-1}\}$) the blocking probability of the path $P^{(B)}_{p,\ \vec{n}_{s,e-1}}$ can be calculated by (7).

$$P^{(B)}_{p,\ \vec{n}_{s,e-1}} = 1 - \left(1 - P^{(B)}_{S,L}\right)^e \quad (7)$$

The blocking probability for a VL depends on the embedding algorithms used. We consider a typical embedding algorithm that selects a path as short as possible from maximum K existing link-disjoint shortest paths with a maximum path length E_m. Our approach can be extended to other embedding algorithms if the algorithms are known.

Lemma 2.1. *For any source-destination pair in any SN with $\widetilde{n}_s$ nodes, under K shortest paths algorithm with maximum path length E_m, the paths can be identified as:*

$$\tilde{k}_e = \begin{cases} k_e, & \text{if } k_e \leq K - \sum_{i<e} k_i \\ K - \sum_{i<e} k_i, & \text{if } k_e > K - \sum_{i<e} k_i > 0 \\ 0, & \text{if } K - \sum_{i<e} k_i \leq 0 \end{cases}, \quad (8)$$

where

$$e \leq E_m.$$

Combining the conditions of embedding algorithms (e.g., E_m, K), we can obtain the VL blocking probability by (9).

Theorem 2.3. *For any source-destination pair, under the K shortest paths algorithm with maximum path length E_m, the average blocking probability of a VL can be calculated by:*

$$P^{(B)}_{V,L}\left(E_m, K\right) = \sum_{\widetilde{n}_s=2}^{M_{S,N}} \sum_{\vec{k}} [\prod_{e=1}^{E_m} \left(P^{(B)}_{p,\ \vec{n}_{s,e-1}}\right)^{\tilde{k}_e}] P_I\left(\vec{k}|K_{I,e}\right) P_{N_S}(\widetilde{n}_s). \quad (9)$$

We have already obtained the VL blocking probability. There are some statistics that are worth discussing further.

Corollary 2.3. *For any source-destination pair, under the K shortest paths algorithm with maximum path length E_m, the acceptance probability of a VL at a given path length e, when all paths with shorter lengths than e are blocked, can be calculated by* (10)*.*

$$P^{(A)}_{V,L}\left(e,\ K\right) = P^{(B)}_{V,L}\left(e-1, K\right) - P^{(B)}_{V,L}\left(e, K\right) \quad (10)$$

The average accepted path length of a VL mapping is another statistic that we are keen to know. This can be estimated by $P^{(A)}_{V,L}\left(e,\ K\right)$.

Corollary 2.4. *For any source-destination pair, under the K shortest paths algorithm with maximum path length E_m, the average path length for accepted VL requests can be obtained from* (11)*.*

$$\bar{l}_{V,L}\left(K, E_m\right) = \sum_{e=1}^{E_m} e P^{(A)}_{V,L}\left(e,\ K\right) \quad (11)$$

D. Virtual Node Blocking Probability

In this subsection, we start to find the acceptance probability of a virtual node.

Lemma 2.2. *The acceptance probability for a virtual node mapping can be calculated as:*

$$P^{(A)}_{V,N} = \sum_{\widetilde{n}_s=2}^{M_{S,N}} \sum_{\widetilde{n}_{c,s}=2}^{\widetilde{n}_{c,s}} \left(1 - (P^{(B)}_{S,N})^{\widetilde{n}_{c,s}}\right) P\left(\widetilde{n}_{c,s}|\widetilde{n}_s\right) P_{N_S}(\widetilde{n}_s), \quad (12)$$

where $P\left(\widetilde{n}_{c,s}|\widetilde{n}_s\right)$ is the conditional probability that there are $\widetilde{n}_{c,s}$ candidate nodes for a virtual node given that there are $\widetilde{n}_s$ substrate nodes in the SN. $P^{(B)}_{S,N}$ is the blocking probability of a substrate node as defined earlier.

E. Blocking Probability for VNs

We now try to calculate the blocking probability for a specific VN. When we deal with a VN with multiple VLs, these VLs may be mapped onto substrate paths that are partially overlapped. To make our analysis tractable, we assume blocking events across all virtual nodes and link mappings are independent. This assumption allows us to have a product-form solution like Jackson networks.

Theorem 2.4. *Assume a VN with $\widetilde{n}_v = |N_V|$ nodes and $\widetilde{e}_v = |E_V|$ VLs. Assume all virtual nodes and VLs are independent. Then the acceptance probability of the VN can be calculated as* (13)*.*

$$P^{(A)}_N(\widetilde{n}_v, \widetilde{e}_v) = \left(1 - P^{(B)}_{V,L}\right)^{\widetilde{e}_v} \left(P^{(A)}_{V,N}\right)^{\widetilde{n}_v} \quad (13)$$

Equation (13) assumes a fixed number of virtual nodes and a fixed number of virtual links in a VNR. Now, we consider the conditional probability of the number of virtual nodes and the conditional probability of the number of virtual links. Let $L\left(\widetilde{n}_v\right) = \frac{\widetilde{n}_v(\widetilde{n}_v-1)}{2}$ denote the maximum number of VLs that may exist given $\widetilde{n}_v$ virtual nodes. The average blocking probability can be calculated.

Corollary 2.5. *The average acceptance probability for an arbitrary VN can be calculated as* (14)*:*

$$P^{(A)}_N = \sum_{\widetilde{n}_v=2}^{M_{V,N}} \sum_{\widetilde{e}_v=1}^{L(\widetilde{n}_v)} P^{(A)}_N\left(\widetilde{n}_v, \widetilde{e}_v\right) P\left(\widetilde{e}_v|\widetilde{n}_v\right) P_{N_V}(\widetilde{n}_v), \quad (14)$$

where $P(\widetilde{e}_v|\widetilde{n}_v)$ is the conditional probability that there exist $\widetilde{e}_v$VLs given $\widetilde{n}_v$ virtual nodes in a VNR; and $L\left(\widetilde{n}_v\right)$ is the maximum number of different links.

F. Estimating the Offered Load $\lambda_{S,L}$ for An SL

At the beginning of Section II-B, we mentioned that $\lambda_{S,L}$, the offered load for an SL, is not the same as λ_V, the VNR rate. This is also true for the offered load of a substrate node $\lambda_{S,N}$. We now discuss how to estimate $\lambda_{S,L}$, $\lambda_{S,N}$.

The offered loads $\lambda_{S,L}$, $\lambda_{S,N}$ should include two parts, the part that is blocked by the substrate node or link and the part that is accepted. We call the part that is accepted and carried by an SL as effective loads denoted as $\lambda_{E,L}$, $\lambda_{E,N}$ respectively. A VNR is blocked if any VL or node is blocked. It is important to note that the offered loads $\lambda_{S,L}$, $\lambda_{S,N}$ should not include the loads blocked by other links or nodes. Because even a substrate node/link accepts this load, it still does not carry it. It does not constitute the effective load of the substrate node/link. Therefore, we assume only the VNs that have been accepted become the offered loads for the SL/node. It should be noted that not all offered loads are effective loads for a specific node/link due to the fact it may be rejected by this node/link while being accepted by other links/nodes. Following the above discussion, we have:

$$\lambda_{S,L} \approx \lambda_V P_N^{(A)}\left(\lambda_{S,L}, \lambda_{S,N}\right) \bar{l}_{V,L}\left(E_m, K, \lambda_{S,L}\right) \bar{e}_v / \bar{e}_s, \quad (15)$$

and

$$\lambda_{S,N} \approx \lambda_V P_N^{(A)}\left(\lambda_{S,L}, \lambda_{S,N}\right) \bar{n}_v / \bar{n}_s, \quad (16)$$

where we emphasized the network acceptance probability and average path length for accepted VLs depending on the offered loads. This is a fixed-point problem. It is not difficult to see that $P_N^{(A)}\left(\lambda_{S,L}, \lambda_{S,N}\right)$ and $\bar{e}_v\left(E_m, K, \lambda_{S,L}\right)$ go down monotonically with increasing offered loads. Therefore, (15) and (16) will converge through a recursive process.

III. Numerical Results

In this section, we will compare the results of our analytical model with the simulation results obtained with some representative VNE algorithms. The purposes of these comparisons are twofolds: 1) We made some assumptions about the independence of certain random variables and some approximation in deriving (5) during our modeling process. 2) We want to show the performance gaps between our analytical model and existing VNE algorithms.

A. Environment Setups

Our proposed performance analytical model aims to estimate the performance of a VNE algorithm under very general environments. To get numerical results, We use the same setup as the previous baseline solution in [2].

The random VNRs are generated by a Poisson process with λ_V ranging from 4 to 8 requests per 100 time units. Each request required a holding time, which was exponentially distributed with an average of $\tau_V = 1000$ time units. The distribution of the number of virtual nodes in a VNR P_{N_V} followed a piece-wise uniform distribution the same as [2].

We compare the final VN acceptance probabilities. Fig. 1 is our analytical results in comparison with simulation results. We first compare the simulation results for non link-disjoint paths (GAOne [1] non link-disjoint) and link-disjoint paths (GAOne link-disjoint). The results for link-disjoint paths are

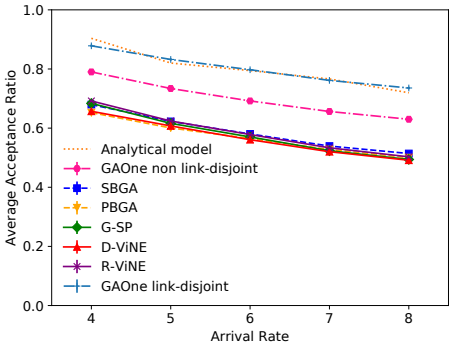

Fig. 1: VN acceptance ratio over arrival rates.

consistently better than the non link-disjoint paths. This justifies finding link-disjoint paths are reasonable and acceptable as discussed in Section II-C. We observe that our analytical results are very close to the GAOne simulation results with link-disjoint paths. This confirms that the independence assumptions we made in Section II-C are fair and rational. We also compare our analytical results with other algorithms. SBGA and PBGA are two-stage genetic algorithms to solve VNE [1]. D-ViNE and R-ViNE [2] are considered as baseline solutions for VNE two-stage mapping. SP [10] is a greedy two-stage mapping solution with small computing complexity, which is widely used for online scenarios to provide fast solutions. As we discussed above, the two-stage approaches may be trapped into the local optimum easily. The results justify that our analytical model can provide a guideline for general VNE approaches to predict near-optimal solutions.

Finally, in Fig. 2, we show the results of the analytical model getting converged after several iterations. Specifically, We observe that when the arrival rates get higher, more iterations are required before the system gets steady. Fig. 2 verifies the correctness of our recursive procedure discussed in II-F.

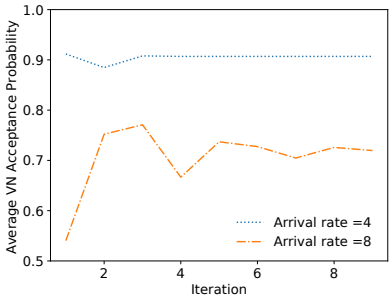

Fig. 2: The VN acceptance ratio converging process.

IV. Conclusion and Future Work

There are very few analytical results on estimating the performance of VNE algorithms. In this paper, we proposed an integrated and systematic approach for estimating the performance of VNE algorithms. Our numerical results have justified that the models we created are accurate and can serve as a benchmark for the performance prediction of VNE algorithms.

REFERENCES

[1] Q. Lu, K. Nguyen, and C. Huang. Distributed parallel algorithms for online virtual network embedding applications. *International Journal of Communication Systems*, page e4325, 2020.

[2] M. Chowdhury, M. Rahman, and R. Boutaba. Vineyard: Virtual network embedding algorithms with coordinated node and link mapping. *IEEE/ACM Transactions on Networking*, 20(1):206–219, 2012.

[3] Christian Aguilar-Fuster and Javier Rubio-Loyola. A novel evaluation function for higher acceptance rates and more profitable metaheuristic-based online virtual network embedding. *Computer Networks*, page 108191, 2021.

[4] Z. Yan, J. Ge, Y. Wu, L. Li, and T. Li. Automatic virtual network embedding: A deep reinforcement learning approach with graph convolutional networks. *IEEE Journal on Selected Areas in Communications*, 38(6):1040–1057, 2020.

[5] C. Wang, R. Batth, P. Zhang, G. Aujla, Y. Duan, and L. Ren. Vne solution for network differentiated qos and security requirements: from the perspective of deep reinforcement learning. *Computing*, 103(6):1061–1083, 2021.

[6] M. Harchol-Balter. *Performance modeling and design of computer systems: queueing theory in action*. Cambridge University Press, 2013.

[7] H. Kobayashi and B. Mark. Generalized loss models and queueing-loss networks. *International Transactions in Operational Research*, 2002.

[8] J. Jackson. Networks of waiting lines. *Operations research*, 1957.

[9] Qiao Lu. *Dynamic and Parallel Resource Allocation with Analytical Performance Estimation for Virtualized Environment*. PhD thesis, Carleton University, 2022.

[10] M. Yu, Y. Yi, J. Rexford, and M. Chiang. Rethinking virtual network embedding: Substrate support for path splitting and migration. *ACM SIGCOMM Computer Communication Review*, 38(2):17–29, 2008.

[11] David A Schum. *The evidential foundations of probabilistic reasoning*. Northwestern University Press, 2001.