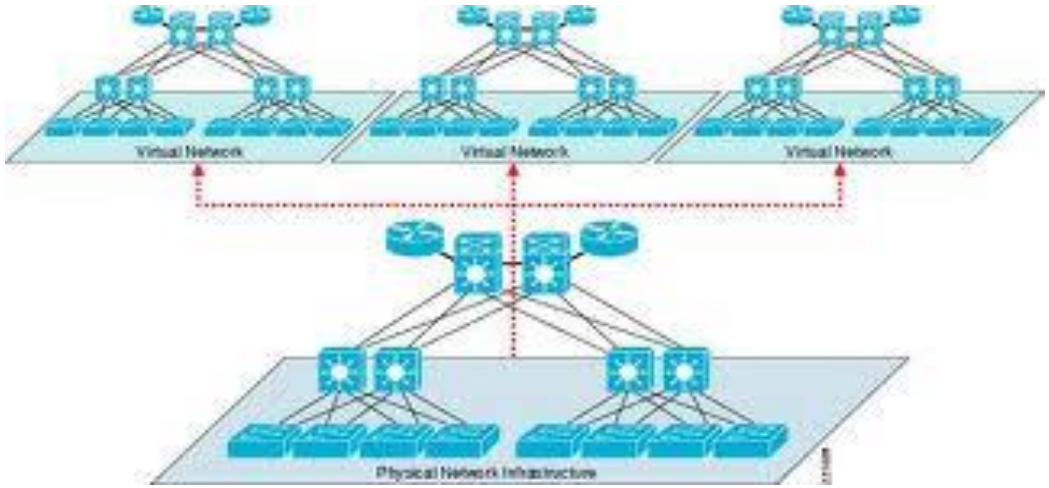
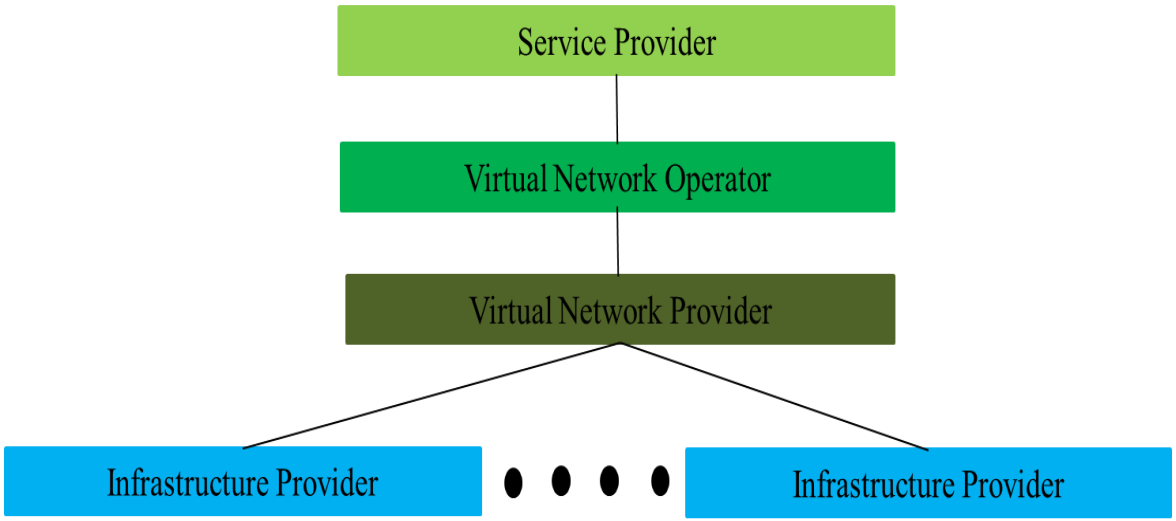
Introduction: Network Virtualization



- Aggregation of multiple virtual resources onto a single platform
- Completely isolated
- Can be programmed on any layer
- Network services same as those provided by non-virtualized network

Introduction : Entities in Network Virtualization Environment

- Virtual Network
- Substrate Network
- Infrastructure Provider
- Service Provider
- Virtual Network Provider
- Virtual Network Operator



Introduction : Important Features of Network Virtualization Environment

- Isolation
- Flexibility
- Scalability
- Manageability
- Evolvability

Introduction : Research Issues in Network Virtualization

- Virtual Network Mapping
- Resource Scheduling
- Admission Control
- Mobility and Dynamism in Network Virtualization Environment
- Interfacing
- Failure Handling
- Naming and Addressing
- Virtual Network Management
- Security and Privacy

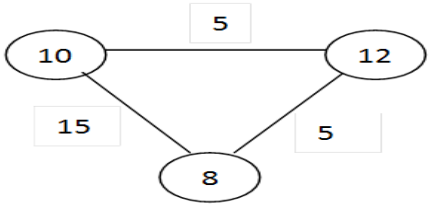
Problem Definition

- Virtual Network Mapping/Embedding(VNE)
- Efficient VNE algorithms for efficient utilization of the substrate network
- VNE problem : Mapping a virtual network with constraints on nodes and links, on to specific physical nodes and links in the substrate network, keeping capacity into account.
- Divided into 2 steps :Virtual Node Mapping
  : Virtual Link Mapping

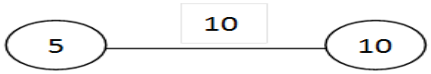
Problem Definition: Formal Description Of VNE Problem

- SN = (N,L) be a substrate network
- $VNR^i = (N^i, L^i)$ be a set of i = 1, ..., n Virtual Network Requests
- $R = \prod_{j=1}^{m} R_j$ be a vector space of resource vectors over resource sets $R_{1, \ldots..}, R_m$
- *cap* : $N \cup L \rightarrow R$,a function that assigns available resources to elements of the substrate network
- For each VNR^i, let $dem_i : N^i \cup L^i \rightarrow R$ be a function that assigns demands to elements of all Virtual Network Requests
- Thus, a virtual network embedding consists of two functions $f_i : N^i \rightarrow N$ and $g_i : L^i \rightarrow SN' \subseteq SN$ for each VNR^i such that $\forall n^i \in N^i : dem_i(n^i) \leq cap(f_i(n^i))$ and $\forall l^i \in L^i : \forall l \in g_i(l^i) : dem_i(l^i) \leq cap(l)$.
- f_i is then called a node mapping function (VNoM) and g_i is called a link mapping function (VLiM).
- Together, they form an embedding for VNR^i

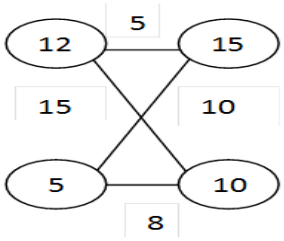
VN Requests Embedded on Substrate Networks of Multiple Infrastructure Providers



VN1

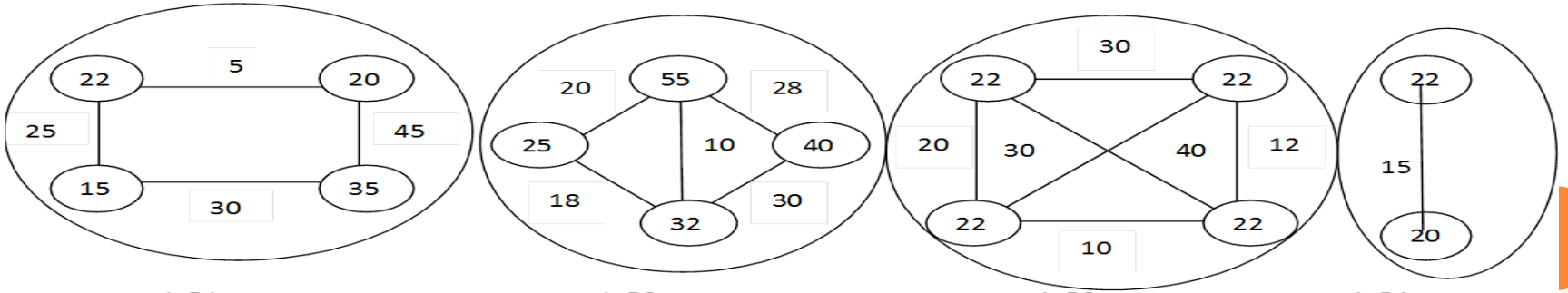


VN2



VN3

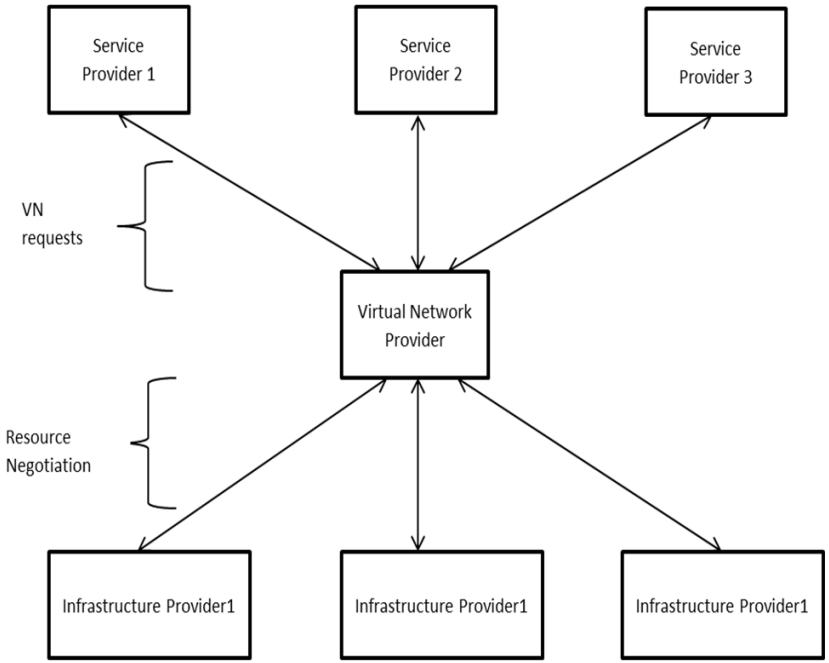
Virtual Network Requests



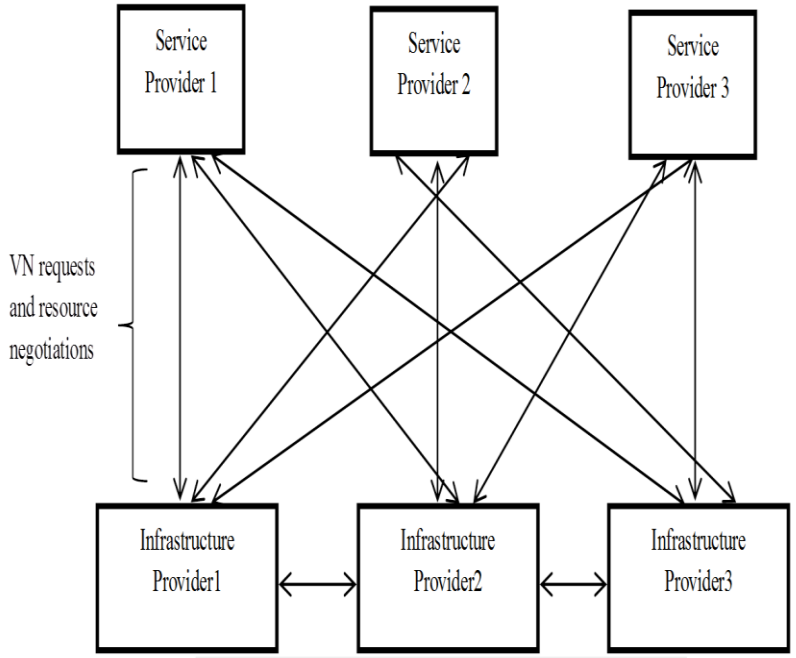
InP1 InP2 InP3 InP4

Categorization of Virtual Network Embedding Approaches

- Static vs. Dynamic
- Centralized vs. Distributed
- One Stage vs. Two Stage



Inter Infrastructure Provider Cooperation and Negotiation



Inter Infrastructure Provider Cooperation and Negotiation

Virtual Network Embedding Metrics

- Residual Saved SN Resources
- Runtime of Embedding Algorithm
- Revenue
- Cost
- Acceptance Ratio
- Utilization
- Number of Communicated Messages

The virtual network embedding strategies to optimize the above metrics are categorized as:

- Exact Methods
- Heuristic Methods
- Meta-Heuristic Methods

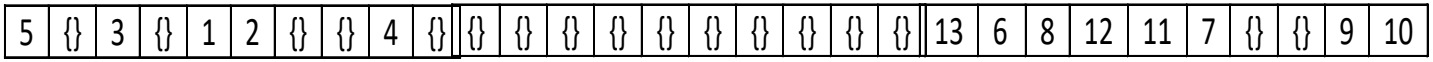
VN Embedding Across Multiple Infrastructure Providers using Genetic Algorithm

THE PROBLEM STRUCTURE

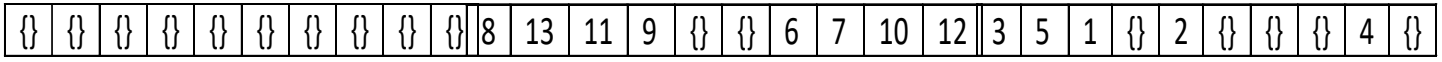
- The proposed model is based on GA, so various modules related with GA based VNE problem are discussed in this section.

Chromosome Structure: In the proposed VNE problem, the solution is encoded using integer encoding as with binary encoding it would become too large to incorporate the required combined information of the virtual networks and substrate networks. The chromosome string representation is described through an example

Parent 1:



Parent 2:

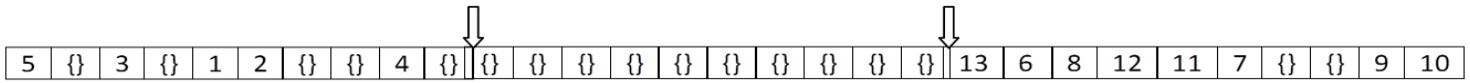


The Problem Structure

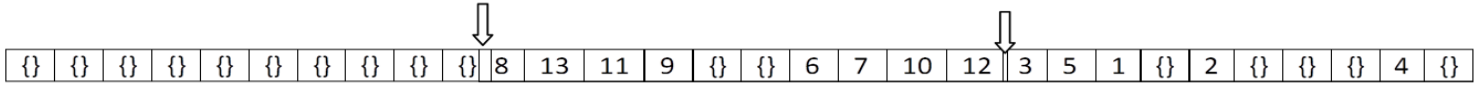
Crossover: After the random initial population generation, crossover is performed on the parent chromosomes. Multipoint crossover is used for the purpose and the crossover points are chosen such that it represents the beginning of the new allele of the substrate network. These multiple points are denoted by arrows in figure where the crossover operation is demonstrated.

Before Crossover:

Parent 1:

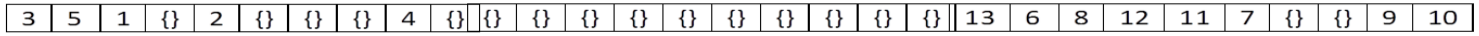


Parent 2:

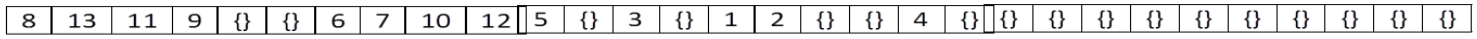


After Crossover:

Child 1:

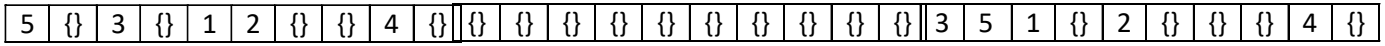


Child 2:



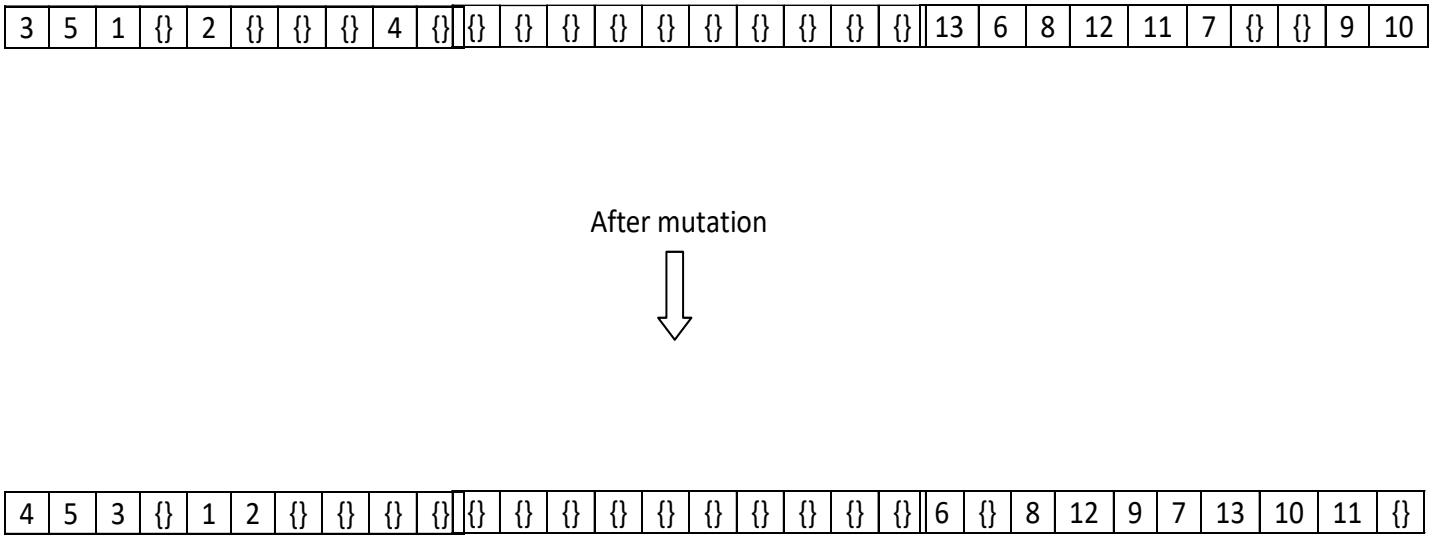
The Problem Structure

- **Feasibility Check:** It has been observed that after crossover, few of the solutions (children) generated are invalid. Therefore, a feasibility test is performed on the offspring. The solutions where one virtual network is seen to be mapped on two substrate networks are infeasible solutions hence discarded. The crossover operation is repeated again to get feasible solutions. Child 1 and child 2 are examples of feasible solutions. Figure below shows an infeasible chromosome that is to be discarded. This is because the same virtual network with 5 nodes is mapped on substrate network 1 as well as substrate network 3 making it an infeasible solution.



The Problem Structure

- **Mutation:** In the mutation phase the individual elements of every allele of a child are shuffled with a mutation probability discussed further in the performance evaluation section.



The Problem Structure

- **Fitness Function**
- SN = {SN_1, SN_2, SN_3.....}: set of substrate networks owned by multiple InPs
- VN = {VN_1, VN2, VN3.....}: set of virtual network requests to be mapped on multiple InPs SN
- VN_i = {$V_{i1,}$ V_{i2}, V_{i3}...}: set of vertices in the i^{th} VN
- A_{ij} : j^{th} vertex of the i^{th} VN
- P_{ij} : Position of A_{ij} i.e. SN node to which A_{ij} is mapped
- $U[A_{ij}]$: weight of A_{ij} i.e. resource (CPU) requested by j^{th} vertex of the i^{th} VN
- $V[P_{ij}]$: weight of P_{ij} i.e. available resource (CPU) on P_{ij}
- E_{Aijk} : Edge connecting A_j vertex to A_k vertex of i^{th} VN
- E_{Pijk} : Smallest link (path with minimum weight) connecting P_{ij} node to P_{ik} node
- $w(E_{Aijk})$: weight on the edge E_{Aijk} i.e. bandwidth requested by virtual network edge (E_{Aijk})
- $w(E_{Pijk})$: weight on the link E_{Pijk} i.e. bandwidth available on substrate network link (E_{Pijk})
- Weight difference W_{ij} of VN vertices (A_{ij}) and the substrate network nodes (P_{ij}) is calculated as in equation 1.
- $W_{ij} = V[P_{ij}] - U[A_{ij}]$

THE PROBLEM STRUCTURE :FITNESS FUNCTION

- Node weight difference (Node_Weigh_diff) for all the corresponding virtual networks and substrate network pairs represented in an individual chromosome is given as in equation 10.

$$\text{Node\_Weigh\_diff} = \sum_i \sum_j W_{ij} \qquad (10)$$

- Define a set, $S^i = \{(A_{ij}, P_{ij}) \mid \forall j \in VN_i\}$ (obtaining VN vertices and the corresponding SN node pairs for every VN request mapped on SN)
- (Edge_Weigh_diff) is obtained by summing up all the corresponding edge/link weight differences of the virtual networks mapped on substrate networks for an individual chromosome as given in equation 11.

$$\text{Edge\_Weigh\_diff} = \sum_i \sum_{j,k} \{(w(E_{Pijk}) - w (E_{Aijk}))\} \text{ (for each unique } <(A_{ij},P_{ij}), (A_{ik},P_{ik})> \in \text{ ordered pair of } S^i) \qquad (11)$$

- Fitness (Z) for an individual chromosome is calculated by adding the node and edge weight differences as in equation 12..

$$Z = \text{Node\_Weigh\_diff} + \text{Edge\_Weigh\_diff} \qquad (12)$$

- After the fitness for all the chromosomes is calculated, the chromosome that has the minimum fitness value is selected as given by equation 13.

$$\text{Weights\_diff} = min_{k=1}^{k=popsize} (Z) \qquad (13)$$

- Here, k varies from 1 to *popsize*. *Popsize* is the total population size.

The Problem Structure

- **Selection :** Selection is carried out to select the parent chromosomes for the next generation. The selected chromosomes are then assigned opportunities to reproduce. Several types of parent selection methods exist such as roulette wheel selection, random selection, rank selection, random selection, elitism selection, tournament selection etc. In this work, the selection is done through sorting mechanism wherein the best half chromosomes with respect to the fitness score are selected as the parent chromosomes for the reproduction of the offspring.

The Algorithm

- The pseudo-code of the algorithm, for the proposed model, is as follows.

GA Based VNE ( )

{

Generate initial population P1 randomly // random VN mapping chromosomes are generated

Calculate the fitness of each chromosome using the fitness function (3)

for (i=1; i<=number_of_generations; i++)

{

Sort the population of P1

Select the best half of P1and store it in a new population P2

Randomly select pairs of parents from the population P2

Perform crossover based on the crossover probability on the parents to produce offspring

Apply feasibility check on each new individual

Mutate offspring based on the mutation probability

Store this newly generated population in P1

Evaluate the fitness of each new individual in P1

}

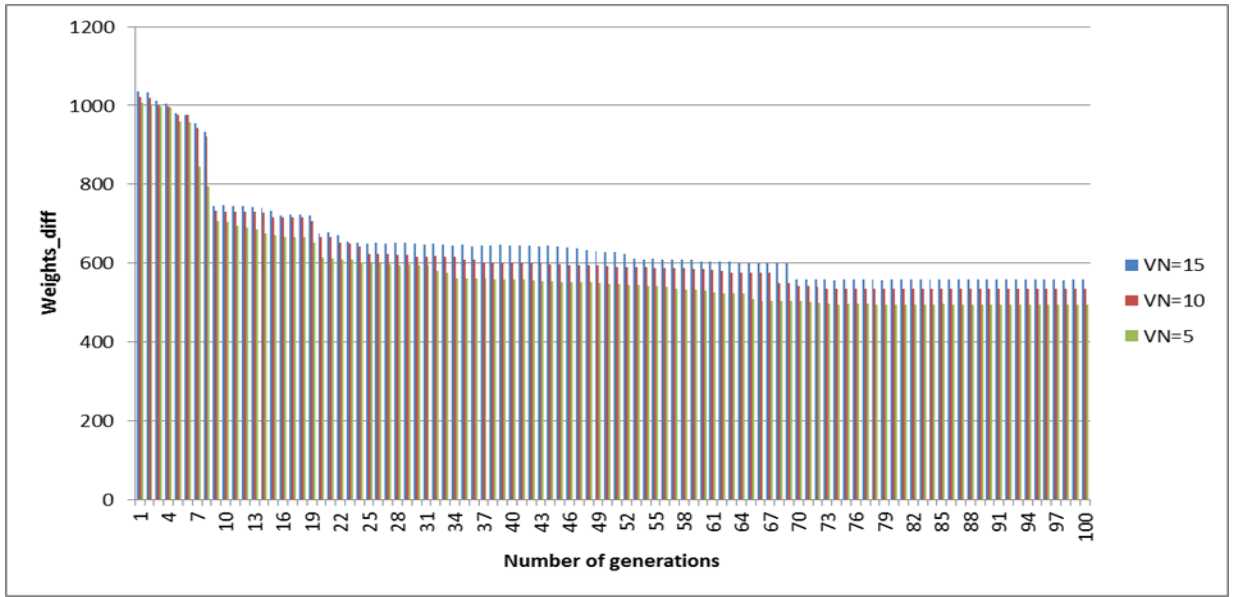
}

Performance Evaluation

- Experimentation is done ten times on various parameters and the average is shown in various results. Input parameters for the experiments are as follows:
- Population size is 50 (generated randomly).
- Size of substrate network varies uniformly in a range of 50 to 100.
- All pairs of substrate nodes and VN vertices are randomly connected with a probability of 0.5
- Weights on the nodes and links of the SN are uniformly distributed between 50 and 100.
- Weights on the vertices of the VN requests are uniformly distributed within a range of 0 to 20
- Weights on the VN's edges are uniformly distributed between 0 and 50.
- The arrival of VN requests is modeled by a Poisson process with rate λ_A= 4 VN requests per 100 time unit
- VN lifetime is modeled by exponential distribution with mean μL= 100 time units.
- Crossover probability is 0.7 and the mutation probability is 0.03.

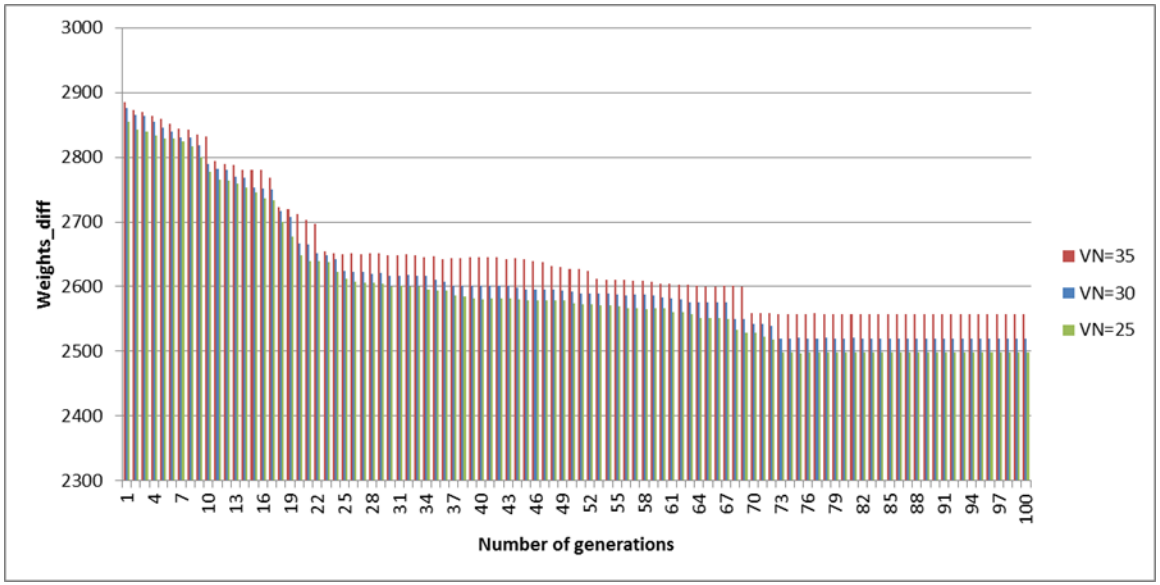
Observation on Weight Difference

- **Experiment 1: Small Sized VNs**
- Small virtual network sets comprise of VN vertices varying from 2 to 15. The number of substrate networks serving these VN requests varies in a range of 20 to 30. Other input parameters are as given above. GA is iterated for 100 generations.
- The result for the fitness function is evaluated for three random instances of the varying VN size i.e. 15, 10 and 5 and is shown in figure



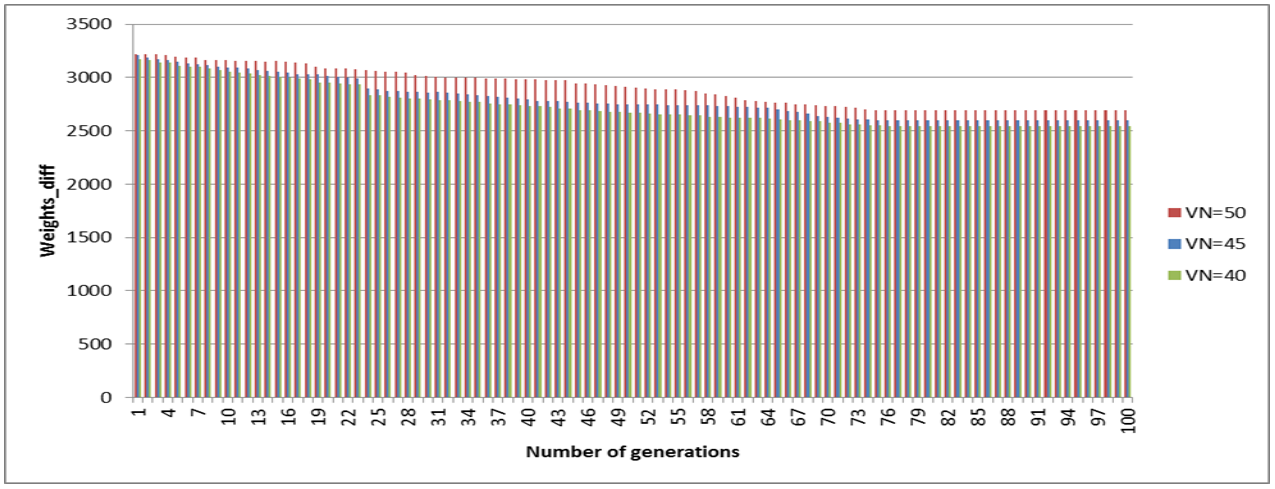
Observation on Weight Difference

- **Experiment 2: Medium Sized VNs**
- Medium virtual network sets comprise of VN vertices varying from 15 to 35. The number of substrate networks serving these VN requests varies in a range of 30 to 40. Other input parameters are same as given above and GA is iterated for 100 generations.
- The result for the fitness function is evaluated for three random instances of the varying VN size i.e. 35, 30 and 25 and is shown in figure



Observation on Weight Difference

- **Large Sized VNs**
- Large virtual network sets comprise of VN vertices varying from 35 to 50. The number of substrate networks serving these VN requests varies in a range of 40 to 50. Other input parameters remain same as given above and GA is iterated for 100 generations.
- The result for the fitness function is evaluated for three random instances of the varying VN size i.e. 50, 45 and 40 and the result is shown in figure



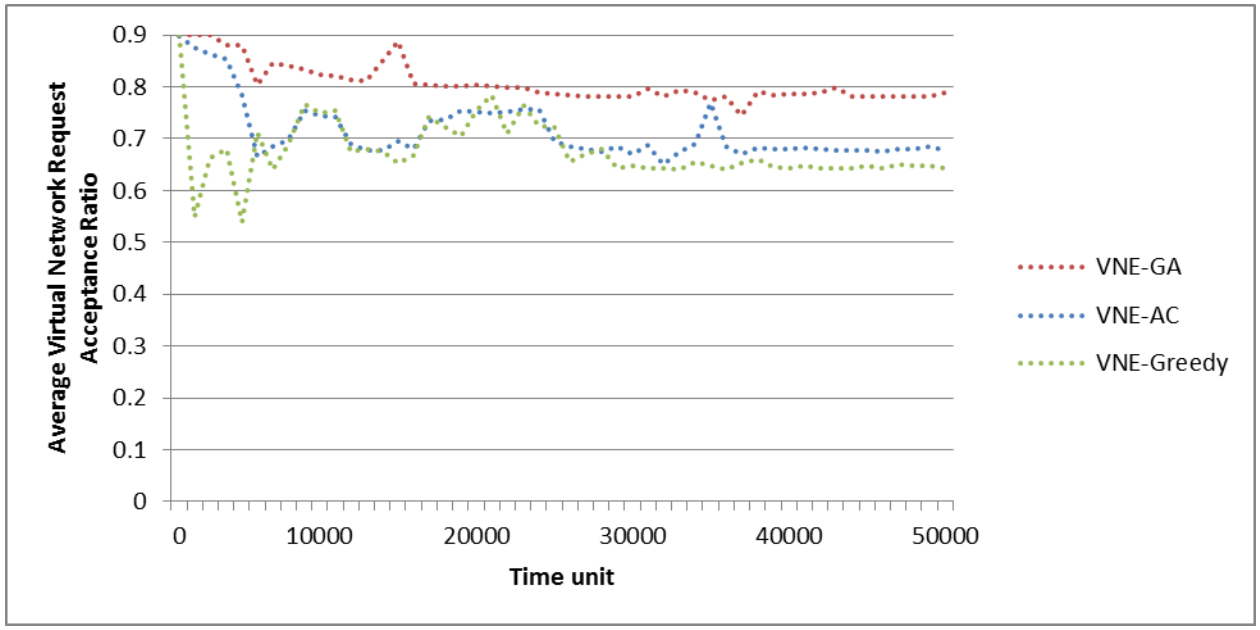
Conclusion On Observations derived for all sized VN embeddings

- The large sized VNs have higher weights_diff as compared to small and medium sized VNs.
- The virtual network requests with large number of vertices and favorably highly weighted vertices and edges have greater weights_diff than those VN requests which have less number of vertices and low weights on the vertices and edges.
- GA takes 100 generations to converge in all the experiments

Observation on Acceptance Ratio of VN Requests

- Experiment compares the acceptance ratio of VNs of the proposed GA based VNE algorithm with some existing embedding strategies VNE_AC and VNE_Greedy.
- The acceptance ratio is calculated as

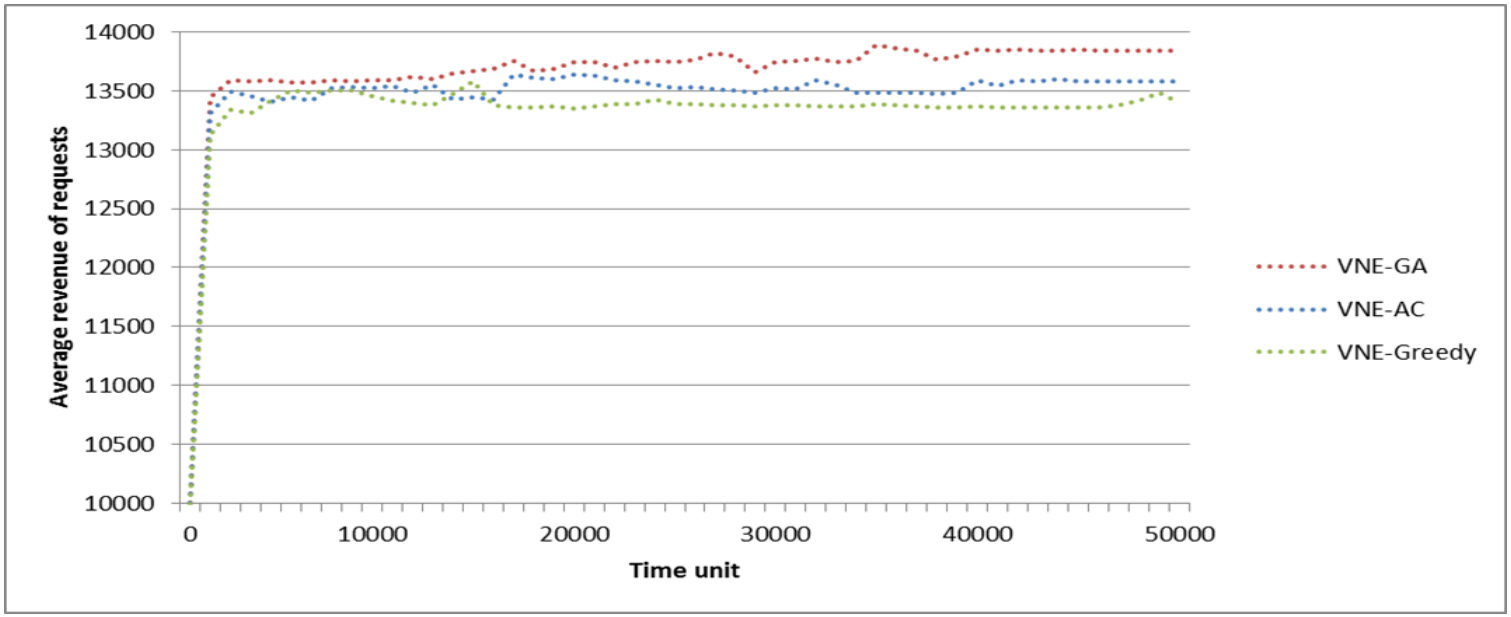
$$\text{Accepted\_VN} = \frac{n(VN')}{n(VN)} = \frac{number\ of\ VN\ requests\ accepted}{number\ of\ VN\ requests} \quad (14)$$



Observation on Average Revenue

- While minimizing the fitness function, the average revenue earned through the proposed VNE-GA algorithm is compared with VNE-AC and VNE-Greedy approach.
- Experiment is performed for 50,000 time units and the result is depicted by a graph in figure

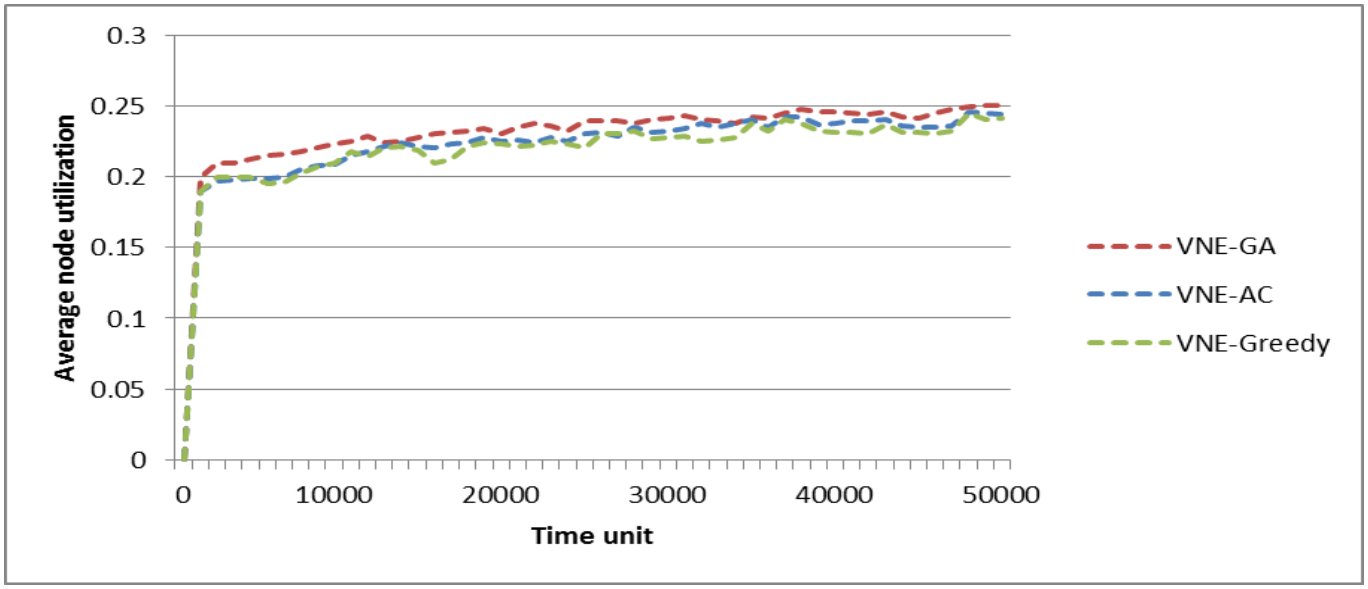
$$R(G_v(t)) = \sum_{i=0}^{i=All\_VN} \alpha_i \left(\sum_{ev \in E} \mu * E_r(ev) + \sum_{nv \in V} \lambda * V_r(nv) \right) \quad (15)$$



Observation on Node Utilization

- The average node utilization of the substrate network is measured by averaging the stress of all the substrate nodes of the substrate networks.
- The substrate nodes stress is calculated using following equation and is defined as the total amount of CPU capacity allocated to different virtual nodes hosted on the substrate node ns ϵ N.

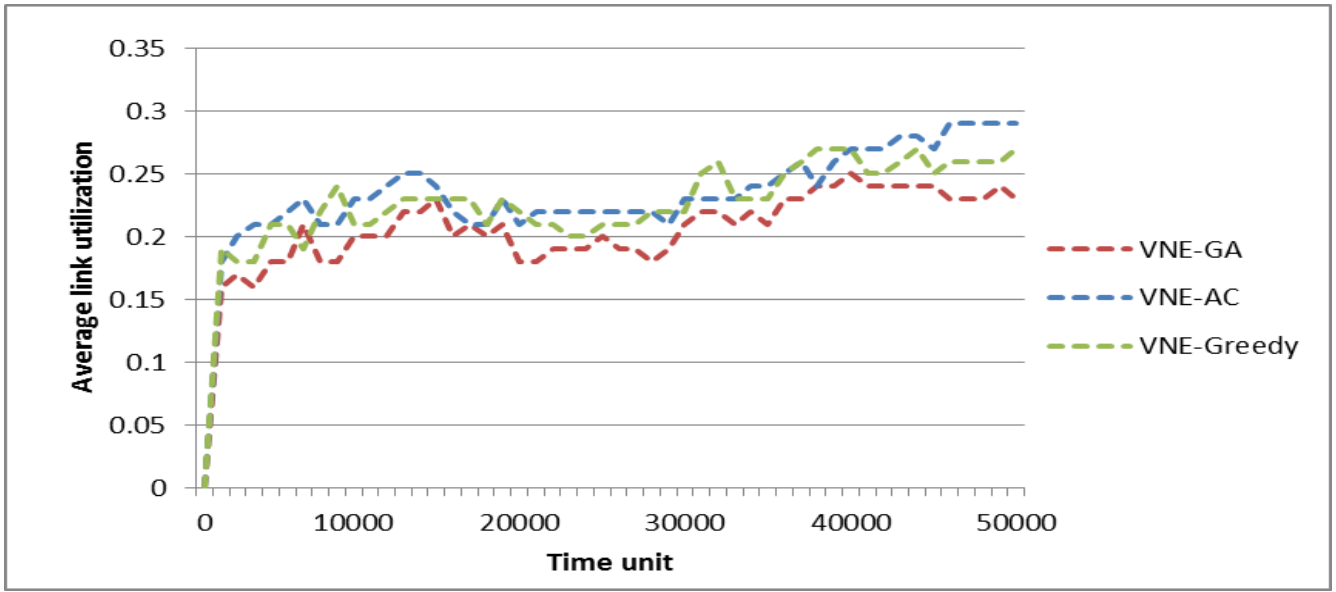
$$\text{Stress(ns)} = \sum_{nv \to ns} V_r(nv) \quad (16)$$



Observation on Link utilization

- The average link utilization of the substrate network is measured by averaging the stress of all the substrate links of the substrate networks.
- The substrate links stress is calculated using following equation and is defined as the total amount of bandwidth reserved for the virtual links whose substrate paths pass through the substrate link $es \in L$.

$$\text{Stress(es)} = \sum_{ev \to es} E_r(ev) \quad (17)$$



Conclusion of Model

- The work addresses the problem of optimal provisioning of multiple virtual network requests among multiple infrastructure providers.
- A GA based virtual network embedding algorithm is proposed to address this NP-class problem.
- The model has been simulated and performance study is done based on various parameters such as revenue generation for InPs, acceptance ratio and efficient node and link utilization.
- Experimental results show that the model performs very well.
- The comparative study of the proposed models with few other contemporary models exhibit that on the mentioned characteristic parameters the proposed model performs consistently.


Thank You