Mechanism Design for Network Virtualization

Technical Report CS-2009-34

Muntasir Raihan Rahman*

Abstract

Recently network virtualization has been proposed as a promising approach to thwart the current ossification of the Internet by allowing multiple heterogeneous virtual networks (VN) to coexist on a shared infrastructure which itself is controlled by self-interested infrastructure providers. A major challenge in this respect is the VN embedding problem that deals with efficient mapping of virtual nodes and virtual links onto the substrate network resources. Most previous research on this problem has focused on designing heuristic and approximation algorithms for the VN embedding problem. However a common aspect of these previous results is that they assume that the different stake-holders in the network virtualization environment do not act in strategic ways.

In this paper, we propose to utilize mechanism design to address this issue. Mechanism design is a branch of micro-economics that deals with protocols and algorithms for aligning the conflicting preferences of self interested agents with the global objective of a central designer. Specifically we show that the celebrated Vickrey Clarke Groves (VCG) mechanism can be used to find the optimal cost minimizing embedding of a virtual network on top of a substrate network, where different parts of the substrate network are owned by strategic agents. In the special case where each substrate link is owned by a separate agent, we show the exact formulation of the VCG payment function and develop simple algorithms for computing the payment functions based on shortest path algorithms. We also discuss extensions of the basic model in terms of more realistic economic and network models and also show how the VCG payment computation can be carried out in a distributed fashion.

1 Introduction

Network virtualization has been recently proposed as a very promising approach to overcome the current ossification of the Internet by allowing multiple heterogeneous virtual networks to coexist on a shared physical substrate infrastructure [29, 1, 5]. In a network virtualization environment, the traditional role of the Internet Service Provider (ISP) has been divided into two separate entities: the *infrastructure providers (InP)* who are responsible for deploying and maintaining physical network resources (routers, links etc.) and the *service providers (SP)* who will implement various network protocols and heterogenous network architectures on virtual networks composed from physical network resources leased from one or more infrastructure providers. Networking researchers believe that network virtualization should be a critical component of any next generation internet architecture[1]. Network virtualization support will allow seamless integration of new network services into the Internet without interrupting existing end users and systems. In addition to that it will open up the possibility of large scale experimentation with new network protocols and architectures in an isolated

*Cheriton School of Computer Science, mr2rahman@cs.uwaterloo.ca

[1]Also known as clean slate design approaches for the Internet [27].

and controlled manner [2]. A major challenge in this respect is the Virtual Network Embedding (VNE) problem, that deals with efficient mapping of virtual networks onto physical network resources. More specifically, for each virtual network creation request, the virtual network embedding is responsible for mapping the virtual nodes onto physical nodes and mapping virtual edges to physical paths. An example of virtual network embedding can be seen in figure 1. The basic graph theoretic problem underlying the VNE problem is actually an $\mathcal{NP}-$hard problem, so most research on this problem has focused on approximation algorithms or fast heuristics. In [35], the authors proposed simple greedy heuristic algorithms for the VNE problem and developed some additional heuristics to speed up their algorithms. A different approach was considered in [32], where the authors proposed to modify the physical substrate network to allow heterogeneous virtual networks to be easily accommodated, this approach in fact shifts the computational burden from the virtual networks to the substrate networks. Recently the authors in [3] have proposed to use mixed integer programming approaches to address the virtual network embedding problem. However a common feature of all these results, is the assumption of only one infrastructure provider who controls the management of all physical network resources. This assumption is not realistic in the context of the Internet, which is an artifact that was not designed by any one central authority, but rather emerged from complicated strategic interactions among several entities such as network operators, ISPs and end users, in varying degrees of coordination and competition.

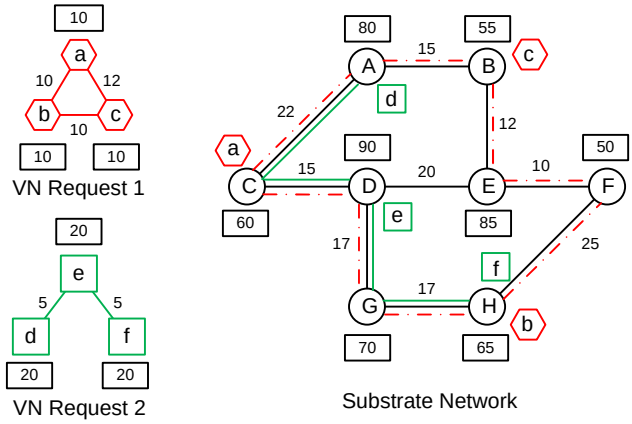

Figure 1: Mapping of VN requests onto a shared substrate network [3].

In this paper we attempt to initiate a systematic study of network virtualization from a game theoretic and mechanism design perspective. In an Internet scale network virtualization environment, it is not economically feasible to pool all network resources with one infrastructure provider that will be solely responsible for resource allocation. As a result the physical resources will be distributed across a number of infrastructure providers. Because of conflicting goals and economic interests, the InP's may act strategically to increase their own utility which may conflict with the global goal of social welfare maximization. If a virtual network request requires using network resources that are distributed across multiple InP's, then a participating InP might lease resources that locally optimizes its own resource usage. It can do that by lying about its local topology or choosing paths that are less costly locally but can result in higher global cost. In other words, since the internal topology and resources owned by an InP are local and private information, it can choose to misrepresent its resources and local topology in-order to maximize its utility. So assuming that the strategic InP's will act in their own interest, the resources (nodes and links) selected for mapping the virtual network

request might result in a sub-optimal resource allocation, which might not coincide with the virtual network embedding that would result if the InP's were truthful about their internal resources and local topology. This strategic setting suggests the use of mechanism design [15, 18, 17] to force the InP's to align their local interests with the global goal of social welfare maximization, where a social welfare maximizing outcome will be a virtual network embedding that minimizes the global cost of the selected resources. Here the system wide goal is to solve a global resource allocation problem, where each InP (agent) controls a certain part of the resource pool.

The rest of the paper is organized as follows. In section 2 we describe our network model. Section 3 consists of the central results of this paper. It includes our proposed economic model for strategic interaction in network virtualization environments and the components of the mechanism for virtual network embedding. It also includes our algorithm for computing the payment functions of the mechanism. Section 4 discusses some future research directions. Section 5 mentions some related research work and finally we conclude with section 6.

2 Network Model

We now mathematically define the virtual network embedding problem [35, 32, 3]. The components and assumptions of the network model are described below:

- The topology of the substrate network is modeled as a weighted graph $G_S(V_S, E_S, c(.))$, where V_S is the set of substrate nodes, E_S is the set of substrate links and $c : E_S \rightarrow \mathcal{R}$ is the edge cost function. Let $n_S = |V_S|$ and $m_S = |E_S|$. We assume that the substrate nodes have zero cost.
- The topology of the virtual network is modeled as another graph $G_V(V_V, E_V)$, where V_V, E_V, n_V, m_V are defined similarly.
- The goal of the virtual network embedding problem is to compute an embedding that maps each virtual node to a substrate node and maps each virtual link to a substrate path. Mathematically, we have to compute an embedding $\Gamma(\Gamma_V, \Gamma_E)$, where $\Gamma_V : V_V \rightarrow V_S$ and $\Gamma_E : E_V \rightarrow 2^{E_S}$.
- We assume that distinct virtual links will be mapped to edge-disjoint substrate paths, that is, for $e_V^1, e_V^2 \in E_V$ and $e_V^1 \neq e_V^2$, we have $\Gamma_E(e_V^1) \neq \Gamma_E(e_V^2)$. We call this the *edge-disjoint property.*
- We define the cost of an embedding as the sum of the edges selected for the embedding as

$$C(\Gamma) = \sum_{e_V \in E_V} \sum_{e_S \in \Gamma_E(e_V)} c(e_S) \tag{1}$$

Our main objective is to compute the embedding that minimizes the cost function defined above.

2.1 An Example

To clarify the network model, we refer to the scenario in figure 1. The substrate network is G_S with $V_S = \{A, B, C, D, E, F, G, H\}$ and $E_S = \{AB, AC, CD, DE, BE, EF, DG, GH, FH\}$. The virtual network VN Request 1 is G_V with $V_V = \{a, b, c\}$ and $E_V = \{ab, bc, ac\}$. We have the embedding of G_V on G_S as Γ with $\Gamma_V(a) = C, \Gamma_V(b) = H, \Gamma_V(c) = B$ and $\Gamma_E(ab) = \{CD, DG, GH\}, \Gamma_E(bc) = \{HF, FE, EB\}, \Gamma_E(ac) = \{CA, AB\}$.

For a quick summary of the mathematical notations used in this paper, we refer to table 1 at the end of the paper. We should mention a few comments regarding our adopted network model. First

of all, we have assumed that substrate nodes have zero cost. In reality, nodes will have non-zero cpu costs and router management costs which cannot be ignored. Secondly, the edge-disjoint property does not always hold. In practical scenarios, the bandwidth of a substrate link can be shared due to modern statistical multiplexing techniques. Last but not least, we have assumed that the virtual network embedding problem only consists of embedding a single virtual network on top of the substrate network. But, in realistic network virtualization environments, more than one virtual network will have to be embedded on the substrate network and it is also possible that the same substrate node and links have to be shared among multiple virtual networks.

3 Virtual Network Embedding and Mechanism Design

3.1 Economic Model

The main objective of this paper is to study the virtual network embedding problem from a mechanism design perspective. To this end, we require a suitable economic model that captures the strategic interactions among self interested parties in a network virtualization environment. We choose to model this strategic interaction such that each substrate link on the substrate network topology is owned by a self interested strategic agent. There are a number of reasons for adopting this simple economic model. First of all, we believe that this model could be a reasonable starting point for more complicated economic models. And secondly, a similar model was adopted by Nisan and Ronen [16] in their seminal paper *Algorithmic Mechanism Design* to study the strategic aspects of Network Routing. Although in a realistic setting, various subnetworks of the Internet topology are owned by separate Internet Service Providers (ISP) or Infrastructure Providers (InP), we make the simplifying assumption to the make the analysis of the algorithms and the mechanism tractable, while maintaining practical viability. We hope our results where each agent only controls a single link will provide insights and guidelines for strategy proof pricing schemes for virtual network embedding where each InP or ISP owns a subnetwork instead of just a physical link of the global topology. So, our goal is to obtain a minimum cost embedding in the presence of self interested agents owning the substrate links. Mechanism design [15, 18, 17] is concerned with the aggregation of privately known preferences of self-interested rational agents into a social choice. The main goal of mechanism design is to discover protocols for dealing with strategic agents that achieve the stated goal of the designer. The protocol has to be designed in such a way that the actions of the selfish agents are properly aligned with the system wide goal of the central designer. In this paper, we assume the existence of a central authority responsible for implementing the mechanism for the embedding problem, the case where the computational task of the mechanism is distributed among the rational agents, called Distributed Algorithmic Mechanism Design (DAMD) [26, 7, 6], which is a more realistic model for an internet scale system, will be discussed later. The most famous result in the mechanism design literature is the Vickrey-Clarke-Groves (VCG) mechanism [30, 4, 9]. In a VCG mechanism, each agent submits its preferences directly in a single stage to a central mechanism design authority who then computes a payment function for each agent. The power of the VCG mechanism arises from the fact that it is a truthful mechanism, which means that agents do not have any incentive to lie about their preferences. So the VCG mechanism successfully aligns all the agent's private preferences with the system designers global design objective.

We are now ready to formally apply the VCG mechanism in the context of out network embedding problem. Since we are applying the VCG mechanism, we can assume that the agents will be truthful. In that case the bulk of the work shifts to designing appropriate payment functions in an efficient

manner. Our goal here is to compute the embedding that will minimize the cost function defined in 1. Since each substrate link (substrate edge) e is controlled by a strategic agent, we compute the VCG payment for each edge e as follows:

$$p^e = \begin{cases} C(\Gamma(\Gamma_V, \Gamma_E; G_S - e)) - C(\Gamma(\Gamma_V, \Gamma_E; G_S|_{e=0})) & \text{if } e \text{ belongs to the embedding} \\ 0 & \text{otherwise} \end{cases}$$

Here $C(\Gamma(\Gamma_V, \Gamma_E; G_S - e))$ corresponds to the cost of the embedding without taking the edge e into account, whereas the term $C(\Gamma(\Gamma_V, \Gamma_E; G_S|_{e=0}))$ is the cost of the embedding after subtracting the cost of the edge e. So we can see from the payment function, that if an edge e is not a part of the embedding, then it receives a zero payment. Otherwise its payment is the difference between the cost of the embedding without e and the cost of the embedding assuming e is free.

3.2 Components of the Proposed Mechanism

The main goal is to use VCG type mechanisms for the Virtual Network Embedding problem. Here we assume the existence of a central authority responsible for computing the mechanism. The components of the proposed mechanism are described as follows:

- **Players**: Each substrate link of the substrate graph is owned (controlled) by a strategic agent. So there are $n = |E_S|$ players in the mechanism.
- **Outcomes**: All possible embeddings of the virtual network on top of the substrate network form the set of outcomes $\mathcal{O}$. It should be noted that the number of possible outcomes will be exponential.
- **Bids**: Each agent bids the cost of its substrate link c_i. He may lie about the true cost in order to avoid being part of the selected embedding, that is c_i is not necessarily equal to the actual cost of the link $c(e_S)$, where $e_S \in E_S$.
- **Objective Function**: The objective function is to compute the minimum cost embedding from the set $\mathcal{O}$ of all possible embeddings. The output function is defined as $f : \mathcal{R}^n \to \mathcal{O}$.
- **Payment Functions**: We use VCG type payment functions, $p_i : \mathcal{R}^n \to \mathcal{R}$. The payment for an agent owning link e is as follows:

$$p^e = \begin{cases} C(\Gamma(\Gamma_V, \Gamma_E; G_S - e)) - C(\Gamma(\Gamma_V, \Gamma_E; G_S|_{e=0})) & \text{if } e \in \Gamma \\ 0 & \text{otherwise} \end{cases}$$

3.3 Efficient Computation of the Payment Functions

We now shift our focus towards the algorithmic aspects of the problem. Without VCG payments the algorithmic problem only consists of computing the cost minimizing embedding only once. However in the presence of VCG payments, we need to compute the embedding $n+1$ times (we assume n denotes the number of edges in the selected embedding): once for the original network G_S and n times for the substrate network with each edge e removed, that is for each $G-e$. We will start with simple brute force algorithms and then gradually tweak the algorithm to reduce its time complexity. The idea of our first algorithm is very simple, however we believe it is a good starting point for developing strategy proof embedding algorithms for network virtualization. The virtual network embedding problem actually

asks for a minimum cost subgraph of the substrate network onto which the virtual network will be mapped. An optimal way to do it will be to check all possible ways to map the virtual nodes on the physical nodes, and then for each possible node mapping, find the shortest path between the physical nodes corresponding to the end point virtual nodes of each virtual link. This computation must be performed $n + 1$ times. The pseudo-code of the algorithm is given below.

Algorithm 1 VCG-ViNE: VCG Computation for Virtual Network Embedding

procedure VCG-ViNE(G_S, G_V)
  Let C_{min} be the cost of the lowest cost embedding
  $C_{min} \leftarrow \infty$
  Let p_{vcg} denote the VCG payment vector corresponding to the lowest cost embedding
  for all $\binom{n_S}{n_V}$ possible node embeddings **do**
    Let Γ_V be the currently selected node embedding
    Let p_c be the current VCG payment vector
    $C \leftarrow 0$
    for all $e_V = (n_V^1, n_V^2) \in E_V$ **do**
      Let $n_S^1 = \Gamma_V(n_V^1)$ and $n_S^2 = \Gamma_V(n_V^2)$
      Find the shortest path P between n_S^1 and n_S^2 on G_S
      Let $d(P)$ be the cost of P
      $C \leftarrow C + d(P)$
      for all $e \in P$ **do**
        Compute shortest path $d(P - e)$ when e is removed from G_S
        $p_c[e] \leftarrow (d(P - e)) - (d(P)|_{e=0})$
      end for
    end for
    if $C < C_{min}$ **then**
      $C_{min} \leftarrow C$
      $p_{vcg}[e] \leftarrow p_c[e], \forall e \in E_S$
    end if
  end for
  Return C_{min} and p_{vcg}
end procedure

The correctness of the algorithm depends on the following lemma. The lemma holds because of the edge-disjoint property that we imposed on our network model (section 2).

Lemma 1 *Let $G_S(V_S, E_S, c(.))$ be the substrate network and $G_V(V_V, E_V)$ be the virtual network. Let the lowest cost embedding of G_V onto G_S be $\Gamma(\Gamma_V, \Gamma_E)$. Consider a substrate link $e \in E_S$ that has been selected by the embedding Γ. Let p_e^Γ denote the VCG price for e with respect to the embedding Γ. There exists a path $\mathcal{P} \in \Gamma_E(E_V)$ such that[2] $e \in \mathcal{P}$. Then $p_e^\Gamma = p_e^\mathcal{P}$, where $p_e^\mathcal{P}$ denotes the VCG price for e with respect to the path $\mathcal{P}$.*

[2] $\Gamma_E(E_V)$ is the set of all paths in G_S that correspond to the virtual links in G_V.

3.3.1 Description of the Algorithm

We now describe some of the finer details of the algorithm. The for loop in line 5 runs over all possible node embeddings. The second for loop in line 9 goes through each virtual link with respect to a fixed node embedding. For each virtual link, we compute the shortest path between the substrate nodes corresponding to the end point virtual nodes of that virtual link. Once a shortest path has been computed, we compute the payment for each substrate link on that shortest path by removing that substrate link from the topology and recomputing the shortest path. We keep track of the node embedding that corresponds to the lowest cost embedding and finally return the cost of the minimum embedding and the VCG payment vector.

3.3.2 Computational Complexity

We now discuss the time complexity of the proposed algorithm. There are $\binom{n_S}{n_V}$ ways to map the virtual nodes to physical nodes. For each possible node mapping, we have to compute $O(n_S)$ shortest paths[3] for each of the $O(n_V^2)$ virtual links. One shortest path computation can be done in $O(n_S \log n_S + n_S^2)$ time using standard Dijkstra's algorithm. So the total running time of the proposed algorithm is $O(\binom{n_S}{n_V}.n_S.n_V^2.(n_S \log n_S + n_S^2))$.

3.3.3 Improving the Time Complexity

Hershberger and Suri [11] showed that VCG prices for each edge on a shortest path can be computed in time bounded by the time complexity of just one shortest path computation. Using this result, the brute force approach can be improved to $O(\binom{n_S}{n_V}.n_V^2.(n_S \log n_S + n_S^2))$ time.

4 Road to Reality: Three Directions for Future Research

In this section, we discuss some future research directions. Specifically we mention some ways to extend the basic network and economic model adopted here in order to be closer in spirit to real network virtualization scenarios.

4.1 More Realistic Network Models

In section 2, we mentioned that our adopted network model had some limitations. For example, we have been concerned with the embedding of a single virtual network. A natural extension is to consider more than one virtual network request arriving over time. In that case, we can apply the VCG mechanism and the algorithm VCG-ViNE separately for each arriving virtual network request. Since it is possible to share the bandwidth of one substrate link among multiple virtual networks, we would have to keep track of the remaining available bandwidth for each substrate link. However, since the remaining available bandwidth for each substrate link is also private information only known to the agent owning that link, this might lead to another level of strategic behavior on behalf of the agents. Moreover, it is also possible to hypothesize strategic interaction at a different granularity in the presence of incoming online virtual network requests. We can assume that each separate virtual network request is performed by strategic agents (the service providers) who want to maximize their utility from using the substrate network resources as much as possible. Normally a virtual network request will also contain a *time duration* parameter indicating how long the virtual network wants

[3]The length of any path in a graph containing n nodes is atmost $n-1$

to utilize the substrate resources. An agent in this case might lie about the duration in-order to maximize utility. As an example, consider a service provider that requires a virtual network to serve end users every alternate day for one month. So it requires substrate network resources for 15 days, but in-order to avoid the virtual network creation costs and release costs, the service provider agent might register the required time duration parameter as one month, which is not its true time duration preferences. Also a service provider might want to keep the virtual network for more time than required in anticipation of future resource needs. This will block other service providers from using the substrate network resources and will decrease network resource utilization in the long run. So we can see that service provider agents can not only lie about their preferences for substrate network resources, but they can also misrepresent their time duration by reporting false start times and end times. This scenario is very close in nature to the *online mechanism design* setting [8, 21, 22, 19], where a dynamically changing set of agents interact with the mechanism over an extended period of time. An online mechanism is an extension of the classical mechanism design framework, which introduces the notion of time dependency and the agents can arrive and depart at discrete points of time. As a result the mechanism must make decisions at each time step. One of our future research directions would be to investigate the online mechanism design aspect of dynamic network virtualization environments.

4.2 More Realistic Economic Models

In our basic economic model, we assumed that each substrate link was owned and controlled by a strategic agent. This is not very realistic and does not correspond to the way the real internet topology is organized. If we consider the Internet to be a massive scale graph, then different disjoint subgraphs would be owned by separate *autonomous systems* (AS). The AS's themselves are connected by inter-domain links. An example of this is shown in figure 2.

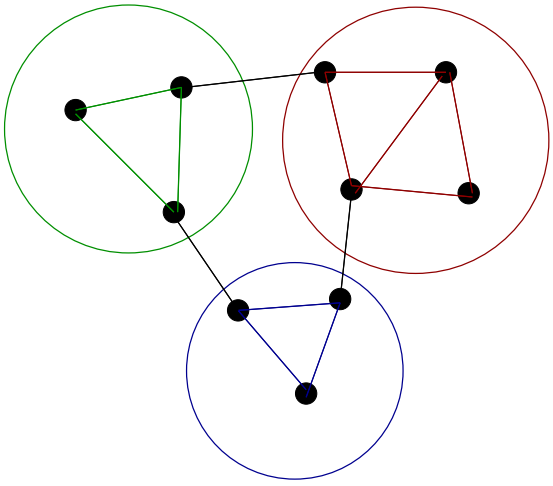

Figure 2: More Realistic Economic Model. The colored links belong to separate agents or AS's. The black links are inter-domain links

An easy way to extend our results would be allow each AS to bid separately for each of its substrate links. However this works only if we ignore substitutability and complementarity effects[4]. If

[4]Complementarity: valuation for a set of substrate links can be more than sum of valuations for the individual substrate links, Substitutability: valuation for a set of substrate links can be less than sum of valuations for the individual substrate links.

such effects cannot be ignored, then we can resort to the known results in case of *single minded bidders* [13]. A single minded bidder only cares about one particular set of items. In our case, the agent will only be concerned with the set of substrate links that fall within its designated AS. For example in figure 2, the agent owning the green subgraph will bid a value for the entire subset of green links. We can then apply the greedy LOS payment scheme from [13] which preserves truthfulness. Notice that this payment function is a non VCG payment scheme and only works for a restricted subclass of single minded bidders. It should also be noted that the inter-domain links (black links in the figure 2) are not controlled by any agent. It might be possible that these links are controlled by the central mechanism designer or jointly by the AS's that are interconnected by a particular inter-domain link.

4.3 Distributed Computation of VCG Payments

In the traditional mechanism design (MD) and algorithmic mechanism design (AMD) setting [17], it is assumed that a trusted center exists that implements the required economic mechanisms. However in an Internet scale network virtualization environment, it is no longer feasible to transmit all relevant private information to a central trusted authority. There are a number of reasons for this. First of all, the InP's will most likely form a Peer-to-Peer (P2P) network instead of a traditional client-server system. As a result no one InP might agree to take on the role of the trusted center. Secondly, the communication overhead of transmitting information to a center (if one exists) might be prohibitive at such a large scale. This calls for a decentralized approach that not only distributes incentives but also the computation of mechanism among the InP's and leads us to the realm of Distributed algorithmic mechanism design (DAMD) [26, 7, 6]. DAMD differs from AMD in a number of different ways, for example the complexity measures for DAMD include *network complexity* which measures the number of messages passed over the network, message size, local computation and local storage required for the messages used to implement the mechanism. Another important difference is that the task of computing the mechanism is now distributed across the strategic agents. As a result in DAMD agents can not only manipulate their private valuations, but also the results of the local computations required for mechanism. These issues suggest that DAMD is the most suitable economic model for studying problems in the domain of networking. The DAMD approach has been recently used to study networking problems like web-caching, P2P file sharing, and overlay network construction.

We now discuss some informal approaches towards distributed computation of VCG payments [20] for our proposed mechanism. A distributed algorithm will delegate the task of computing the outcome and payment functions of the mechanism among the agents themselves, in the absence of any central authority. It can be easily seen that distribution leads to further scope of manipulation since the agents can change the computational results that they have control over. As an example, consider a set of agents organized in a *ring topology* and the agents are performing a distributed algorithm for the second price auction by passing around the two top most bids (tokens) along the ring. As a result an agent can manipulate the second highest bid when the token comes to that agent and if no other agent further changes the token, then that manipulative agent can get the item at a lower price. One approach to prevent computational manipulations is to use *replication* [20]. In this technique, the set of agents are divided into two groups and each group computes their own version of the outcome and payment functions. If the results from the two groups match, then the outcome and payments are enforced. If they don't match, then at least one agent in one group cheated and a severe penalty is enforced on all agents[5]. It should be noted that in these distributed settings, truthfulness can no longer be achieved in dominant strategies, rather we have to resort to the slightly weaker solution concept of

[5]In this case we need a central authority to enforce penalties, so the approach is not completely distributed.

ex-post Nash equilibrium [28]. Another issue that should be addressed is that the payment function for an agent i should be computed by any subset of agents excluding i to avoid manipulation. An approach that can be adapted to our economic model is to utilize a *distributed hash table* (DHT) type system, where the payment for each substrate link agent will be performed by a subset of neighboring[6] substrate link agents. This type of scheme has been applied in the context of incentive mechanisms for promoting cooperation in P2P networks [10, 31].

5 Related Work

Traditionally computer networks has been a very rich application area for game theory and mechanism design. However to the best of our knowledge our work is the first to formally apply mechanism design in the context of the virtual network embedding problem in network virtualization. There has been a number of research projects that studied the stability and co-existence of mutliple overlay networks on top of a native IP network which is quite similar to the network virtualization setting [24, 12, 14]. However these results are concerned with post mapping phases, that is after the overlay networks have been constructed, where as our results are more concerned with the strategic aspects of the virtual network formation phase. Also these results are more related to the classic results on *selfish routing* by Roughgarden and Tardos [25]. Recently Yuen and Li applied mechanism design to study the dynamic multicast tree formation problem in overlay networks [33]. The same authors also investigated applications of mechanism design for dynamic topology formation in autonomous networks [34].

6 Conclusion

The presence of multiple types of stake-holders at different levels of granularity (end-users, service providers and infrastructure providers) in a network virtualization environment naturally lead to large scale strategic interactions among the different parties. We believe that mechanism design is an appropriate tool for smoothing out the conflicting interests and preferences of this diverse set of agents. In this paper we have attempted to initiate a systematic study of network virtualization from a mechanism design perspective. We have specifically focused on the virtual network embedding problem in the presence of multiple infrastructure providers controlling disjoint parts of the shared substrate infrastructure. Our main contribution in this paper is the application of the classic VCG mechanism to solve the virtual network embedding problem in the presence of strategic agents. We have developed a simple algorithm for computing the VCG payments and discussed ways to extend the results to more realistic scenarios. In the future we would attempt to perform experiments using simulation tools and PlanetLab test-beds [23] to test the scalability and performance of our initial results. We would also try to apply tools from game theory and mechanism design to address some other important problems in network virtualization. Especially we would look into efficient market mechanisms for dynamic network virtualization environments where service providers and infrastructure providers can buy and sell substrate network resources in the presence of brokers who monitor the market and determine market clearing prices. We would also investigate recursive network virtualization environments where the service providers themselves can resell the network resources they purchased from the infrastructure providers and add infinitum. These hierarchical and recursive environments and interactions would ultimately lead to a very complex and multi-level *network virtualization marketplace* (NVM) that can

[6] Two substrate links are neighbors if they share a substrate node.

be analyzed and controlled using tools from micro-economics (game theory, mechanism design) and probably also some techniques from macro-economics.

References

[1] T. Anderson, L. Peterson, S. Shenker, and J. Turner. Overcoming the internet impasse through virtualization. *Computer*, 38(4):34–41, April 2005.

[2] Andy Bavier, Nick Feamster, Mark Huang, Larry Peterson, and Jennifer Rexford. In vini veritas: realistic and controlled network experimentation. *SIGCOMM Comput. Commun. Rev.*, 36(4):3–14, 2006.

[3] N. M. Mosharaf Kabir Chowdhury, Muntasir Raihan Rahman, and Raouf Boutaba. Virtual network embedding with coordinated node and link mapping. *submitted to INFOCOM 2009*.

[4] Edward H. Clarke. Multipart pricing of public goods. *Public Choice*, 11(1), September 1971.

[5] Nick Feamster, Lixin Gao, and Jennifer Rexford. How to lease the internet in your spare time. *SIGCOMM Comput. Commun. Rev.*, 37(1):61–64, 2007.

[6] Joan Feigenbaum, Christos Papadimitriou, Rahul Sami, and Scott Shenker. A bgp-based mechanism for lowest-cost routing. *Distrib. Comput.*, 18(1):61–72, 2005.

[7] Joan Feigenbaum and Scott Shenker. Distributed algorithmic mechanism design: recent results and future directions. In *DIALM '02: Proceedings of the 6th international workshop on Discrete algorithms and methods for mobile computing and communications*, pages 1–13, New York, NY, USA, 2002. ACM.

[8] Eric Friedman and David C. Parkes. Pricing WiFi at Starbucks– Issues in online mechanism design. In *Fourth ACM Conf. on Electronic Commerce (EC'03)*, pages 240–241, 2003.

[9] T. Groves. Incentives in teams. *Econometrica*, 41:617–631, 1973.

[10] David Hausheer. *PeerMart: Secure Decentralized Pricing and Accounting for Peer-to-Peer Systems*. PhD thesis, ETH Zurich, Aachen, Germany, March 2006.

[11] J. Hershberger and S. Suri. Vickrey prices and shortest paths: What is an edge worth? In *FOCS '01: Proceedings of the 42nd IEEE symposium on Foundations of Computer Science*, page 252, Washington, DC, USA, 2001. IEEE Computer Society.

[12] Wenjie Jiang, Dah-Ming Chiu, and John C. S. Lui. On the interaction of multiple overlay routing. *Perform. Eval.*, 62(1-4):229–246, 2005.

[13] Daniel Lehmann, Liadan Ita Oćallaghan, and Yoav Shoham. Truth revelation in approximately efficient combinatorial auctions. *J. ACM*, 49(5):577–602, 2002.

[14] Y. Liu, H. Zhang, W. Gong, and D. Towsley. On the interaction between overlay routing and underlay routing. *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, 4:2543–2553 vol. 4, March 2005.

[15] A. Mas-Colell, M.D. Whinston, J.R. Green, and Universitat Pompeu Fabra Facultat de Ciències Econòmiques i Empresarials. *Microeconomic theory*. Oxford University Press New York, 1995.

[16] Noam Nisan and Amir Ronen. Algorithmic mechanism design. In *Games and Economic Behavior*, pages 129–140, 1999.

[17] Noam Nisan, Tim Roughgarden, Eva Tardos, and Vijay V. Vazirani. *Algorithmic Game Theory*. Cambridge University Press, New York, NY, USA, 2007.

[18] David C. Parkes. Classic mechanism design. In *Iterative Combinatorial Auctions: Achieving Economic and Computational Efficiency*, chapter 2. PhD Thesis, University of Pennsylvania, 2001.

[19] David C. Parkes. Online mechanisms. In Noam Nisan, Tim Roughgarden, Eva Tardos, and Vijay Vazirani, editors, *Algorithmic Game Theory*, chapter 16. Cambridge University Press, 2007.

[20] David C. Parkes and Jeffrey Shneidman. Distributed implementations of Vickrey-Clarke-Groves mechanisms. In *Proc. 3rd Int. Joint Conf. on Autonomous Agents and Multi Agent Systems*, pages 261–268, 2004.

[21] David C. Parkes and Satinder Singh. An MDP-Based approach to Online Mechanism Design. In *Proc. 17th Annual Conf. on Neural Information Processing Systems (NIPS'03)*, 2003.

[22] David C. Parkes, Satinder Singh, and Dimah Yanovsky. Approximately efficient online mechanism design. In *Proc. 18th Annual Conf. on Neural Information Processing Systems (NIPS'04)*, 2004.

[23] Larry Peterson, Tom Anderson, David Culler, and Timothy Roscoe. A Blueprint for Introducing Disruptive Technology into the Internet. In *Proceedings of HotNets–I*, Princeton, New Jersey, October 2002.

[24] Lili Qiu, Yang Richard Yang, Yin Zhang, and Scott Shenker. On selfish routing in internet-like environments. *IEEE/ACM Trans. Netw.*, 14(4):725–738, 2006.

[25] Tim Roughgarden and Éva Tardos. How bad is selfish routing? *J. ACM*, 49(2):236–259, 2002.

[26] Rahul Sami. *Distributed algorithmic mechanism design*. PhD thesis, New Haven, CT, USA, 2003. Director-Joan Feigenbaum.

[27] Scott Shenker. Fundamental design issues for the future internet (invited paper). *IEEE Journal on Selected Areas in Communications*, 13(7):1176–1188, 1995.

[28] Jeffrey Shneidman and David C. Parkes. Specification faithfulness in networks with rational nodes. In *Proc. 23rd ACM Symp. on Principles of Distributed Computing (PODC'04)*, pages 88–97, St. John's, Canada, 2004.

[29] J.S. Turner and D.E. Taylor. Diversifying the internet. *Global Telecommunications Conference, 2005. GLOBECOM '05. IEEE*, 2:6 pp.–, Nov.-2 Dec. 2005.

[30] William Vickrey. Counterspeculation, auctions, and competitive sealed tenders. *The Journal of Finance*, 16(1):8–37, 1961.

[31] Vivek Vishnumurthy, Sangeeth Chandrakumar, and Emin Gun Sirer. Karma: A secure economic framework for p2p resource sharing. In *Workshop on the Economics of Peer-to-Peer Systems*, Berkeley, California, 2003.

[32] Minlan Yu, Yung Yi, Jennifer Rexford, and Mung Chiang. Rethinking virtual network embedding: substrate support for path splitting and migration. *SIGCOMM Comput. Commun. Rev.*, 38(2):17–29, 2008.

[33] Selwyn Yuen and Baochun Li. Strategyproof mechanisms for dynamic tree formation in overlay networks. In *INFOCOM*, pages 2135–2146, 2005.

[34] Selwyn Yuen and Baochun Li. Strategyproof mechanisms towards dynamic topology formation in autonomous networks. *MONET*, 10(6):961–970, 2005.

[35] Y. Zhu and M. Ammar. Algorithms for assigning substrate network resources to virtual network components. *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, pages 1–12, April 2006.

Table 1: Mathematical Notations

Notation	**Description**
$G_S(V_S, E_S, c(.))$	Topology of the Substrate Network
V_S	Set of Substrate Nodes
$n_S = \|V_S\|$	Number of Substrate Nodes
E_S	Set of Substrate Links
$m_S = \|E_S\|$	Number of Substrate Links
$c : E_S \to \mathcal{R}$	Edge Cost Function
$G_V(V_V, E_V)$	Topology of the Virtual Network
V_V	Set of Virtual Nodes
$n_V = \|V_V\|$	Number of Virtual Nodes
E_V	Set of Virtual Links
$m_V = \|E_V\|$	Number of Virtual Links
$\Gamma(\Gamma_V, \Gamma_E)$	Embedding of G_V on Top of G_S
$\Gamma_V : V_V \to V_S$	Node Embedding Function
$\Gamma_E : E_V \to 2^{E_S}$	Link Embedding Function
$C(\Gamma)$	Cost of Embedding Γ
p_e	VCG Price Enforced on Agent Owning Substrate Link $e \in E_S$