Strategyproof Mechanisms towards Evolutionary Topology Formation in Autonomous Networks

Selwyn Yuen, Baochun Li
Department of Electrical and Computer Engineering
University of Toronto
$\{$*swsyuen,bli*$\}$*@eecg.toronto.edu*

***Abstract*— In autonomous networks, cooperations among nodes cannot be assumed, since each node is capable of making independent decisions based on their personal preferences. In particular, when a node needs the help of intermediate nodes to relay messages to other nodes, these intermediaries may be reluctant to contribute their network resource for the benefit of others. Ideally, the right amount of incentives should be provided to motivate cooperations among autonomous nodes so that a mutually beneficial network results. In this paper, we leverage the power of *mechanism design* in microeconomics to design a distributed incentive mechanism that motivates each node towards a more desirable network topology. Since network parameters and constraints change dynamically in reality, the desirable topology can evolve over time. Our solution presented in this paper has successfully encompassed this *evolutionary* nature of the network topology. In addition, we have transformed our solution to an easy-to-implement distributed algorithm, that converges towards the globally optimal topology. Although our presentation is specific to the evolutionary topologies problem, we believe that the design methodology employed here is sufficiently general to solve other problems in autonomous networks.**

I. Introduction

As communication networks become more distributed in nature, network functionalities are pushed towards the edge of these networks. Each end system has increased responsibilities and computational burden. In return, these end systems have the freedom to make decisions to their best interests. This leads to the recently thriving research towards *selfish* behaviour in various networks.

In this paper, we focus on the class of *autonomous networks*. The two defining characteristics of an autonomous network are that each network participant, termed *node*, has: (1) private information not known to others, and (2) the freedom to make individual decisions independent of others. Common networks that fall under this category include application overlay networks, wireless ad hoc networks, or peer-to-peer networks. Specifically, we wish to find a distributed algorithm to achieve a desirable topology in an autonomous network given that only local decisions are made by each node.

The term *network topology* is used quite differently in this paper from its traditional meaning in graph theory. In an autonomous network, when one node exchanges data messages with another node through several intermediate nodes, these intermediaries can either choose to cooperate and relay the message or refuse to cooperate. If the intermediate nodes do not cooperate, the communication channel between the source and destination nodes will be blocked. The set of source-destination pairs in which every intermediary is willing to relay messages forms a communication *topology*. Throughout the entire paper, we seek to find the most desirable network topology in this sense.

As in any real autonomous networks, the desirable topology is dynamic and changing over time. In other words, it *evolves* into a new topology once it discovers a better one. This can be due to a variety of reasons. For example, network nodes may join and leave a network dynamically over time. Further, miscellaneous network parameters such as message relaying cost, bandwidth, congestion level, individual node's preferences and communication pattern may all change and evolve dynamically over time. The distributed algorithm presented in this paper addresses this evolutionary nature of the topology. At any instant in time, the distributed algorithm will continuously converge towards the most desirable topology. We call this problem the *evolutionary topologies* problem.

Another aspect of our solution focuses on creating an *incentive* for autonomous nodes to cooperate to achieve the best system state. A central concern with autonomous networks, however, is that each node will naturally choose their own optimal decision based on their personal preferences. In economic terminologies, every node is *selfish*, and seeks to maximize its private utility. In the evolutionary topologies problem, for example, a node may be reluctant to contribute its resources such as bandwidth, energy, or processing cycles to relay messages for others. In most cases, this selfish behaviour results in an overall decision that may not be ideal at the system level. As a result, each node requires additional incentives to drive towards a globally optimal state.

An important objective of this paper is *to employ a systematic approach to design a mechanism that provides the right amount of incentive to each node so that the private-utility optima of every node coincides with the global optimum*. To our knowledge, we are the first to consider this evolutionary topologies problem in the context of mechanism design, and successfully solving it in a distributed manner.

The theoretical background of our solution is based on an important branch of game theory, called *mechanism design*, also known as implementation theory, or inverse game theory. In traditional game theory applications, a model is formulated based on the problem scenario. Various equilibriums are subsequently explored, according to the formulated model. This is the forward approach to applying game theory. The disadvantage of the forward approach is that if the equilibriums of the game are undesirable, nothing can be modified because these equilibriums are merely consequences of our model.

The inverse approach of mechanism design, however, is more appealing in light of our evolutionary topologies problem. Mechanism design requires us to first identify the globally optimal point, *i.e.*, the desirable topology in the context of our problem. Then, if a node's inherent preference is not sufficient to motivate itself to achieve the globally desirable topology, we alter its preference by paying it until it has the incentive to do so. In an autonomous network, this payment is not restricted to monetary payments, but can be interpreted as any virtual currency, as suggested by Buttyan *et al.* [1].

As in most literature in mechanism design [2], [3], [4], we assume that the utility function of each node is *quasi-linear* throughout this paper. This means that each node's utility function is the sum of: (1) the inherent valuation of a system state from that node's perspective, and (2) the payment received by that node. Since the valuation is intrinsic to a node's tastes and cannot be altered, we can only control a node's utility function by adjusting the payments. A well-known solution to the mechanism design problem is the *Vickrey-Clarke-Groves (VCG) mechanism*. It has been proven to be the only general solution to the mechanism design problem, and also possesses an important property known as *strategyproofness* [5]. This is useful in our evolutionary topologies problem because it means that every node in an autonomous network will have no reason to lie to the public about their private information, such as their message relay cost and other individual preferences.

Although the distributed algorithm presented in this paper is specific to achieving a desirable topology in an autonomous network, we feel that the current algorithmic mechanism design methodology is powerful enough to be applied to other problem scenarios in autonomous networks as well. A common theme in the solutions to these problems is that a node's private information will eventually be made public, if given sufficient incentives to honestly reveal their private information to their neighbours.

The remainder of this paper is organized as follows. Sec. II formally defines the problem scenario and justifies the need for an incentive mechanism. Sec. III introduces the idea of mechanism design and explains the VCG solution. Sec. IV presents how the VCG equations can be applied to our problem at hand in a centralized manner. We also prove several important properties regarding our solution. Sec. V logically extends the centralized solution to a distributed algorithm that is guaranteed to converge to the VCG solution. In Sec. VI, an evaluation of the distributed algorithm is presented based on whether it achieves the desirable topology and other convergence properties. We discuss some related work in Sec. VII, and conclude the paper in Sec. VIII.

II. Problem Formulation

One of the most fundamental values in forming a communication network is that it enables the exchange of data between two or more autonomous nodes. In conventional computer networks, a node reaches another node by relying on the intermediate nodes to relay data packets. The intermediate nodes are usually routing equipments maintained by network service providers, who charge the network end users for this service. These payments received by the network service providers give them incentives to relay data for end users.

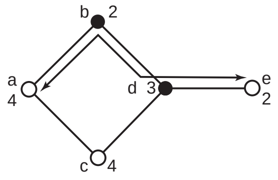


Fig. 1. Example: Data communications between nodes a and e require the cooperation of nodes b and d to relay the message. The nodes represented by a filled circle are intermediate nodes relaying messages. The nodes represented by an empty circle are not required to relay messages. The number beside each node is the relay cost.

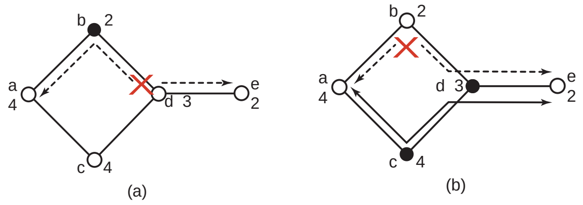


Fig. 2. Example: (a) The path from node a to e is broken if node d refuses to relay messages. (b) An alternate path $\langle a, c, d, e\rangle$ exists even if node b refuses to relay messages.

In autonomous networks, however, the intermediate nodes are end users themselves, and therefore have no incentive to relay messages for other end users. In order for the intermediaries to cooperate and relay messages, a payment schedule must be established. The difficulty lies in determining the correct level of payment. If the payment is too large, it is unfair to the sender of the message. On the other hand, too small a payment may not sufficiently motivate cooperation from intermediaries.

An example is shown in Fig. 1, where node a tries to communicate with node e. This communication between nodes a and e is made possible by the cooperation of nodes b and d to relay other nodes' messages. In Fig. 1, nodes that are willing to relay messages for others are represented by a filled circle, whereas nodes that are not relaying any message are represented by an empty circle. Each node is also labeled with a number, which represents the cost to relay one message over that node. In this example, the minimum relay cost from node a to e is $2 + 3 = 5$. Note that any node can trivially reach itself or its one-hop neighbours without relay cost incurred.

Further, in Fig. 2a, if the intermediate node d refuses to relay messages, this lack of cooperation will make it impossible for node a and e to communicate. On the other hand, if the intermediate node b refuses to relay the message from node a to e, this message can still be relayed through the alternate path $\langle a, c, d, e\rangle$, provided that node c is willing to cooperate, as depicted in Fig. 2b. From this observation, we realize that the cooperation from node d is more important than that of node b. Therefore, a sensible payment scheme should pay more to node d than to node b.

The above example resembles a minimum-cost routing problem. However, our formulation of the problem is different in at least three ways. First, traditional routing problems usually minimize link costs. In our formulation of the problem, we are minimizing relay costs at each node. Second, an autonomous node can freely choose to refuse to forward other nodes' messages, whereas a link has no choice but to carry traffic. Third, and most importantly, we are trying to solve for a global optimum that will result in network connectivity as a whole. It is not our objective to minimize the cost of any single path, although this could be an appealing side effect. Throughout the entire paper, our focus is to maximize global welfare, instead of local welfare. In the following, we define several useful terminologies, which will eventually lead to our precise definition of a *desirable network topology*. Notice how one definition builds on top of the other.

Definition 1: A node is called a *relay node* in an autonomous network if it has been given sufficient incentives to relay messages for other nodes. A node that is not a relay node is a *non-relay node*.

For example, in Fig. 1, nodes b and d are relay nodes, whereas nodes a, c, and e are non-relay nodes.

Definition 2: A destination node k is *reachable* from a source node j if there exists a path from j to k such that every intermediate node is a relay node. Node k is called a *reachable node* from node j. By definition, all adjacent nodes are automatically reachable.

For example, in Fig. 2a, node e is not reachable from a. Node d is reachable from a since node b is a relay node. Finally, nodes b and c are also reachable from a since they are adjacent to a. Note that a reachable node does not have to be a relay node itself.

Definition 3: An autonomous network is *connected* if every node is reachable from every other node. A network that is not connected is *disconnected*.

Fig. 1 and Fig. 2b are both examples of connected autonomous networks, whereas the network shown in Fig. 2a is disconnected.

Definition 4: A node is called a *bottleneck* node if making it a non-relay node causes a connected autonomous network to become disconnected.

Note that node d is the only bottleneck node in Fig. 2a.

The above definitions are defined for the purpose of solving our problem, and may differ slightly from graph theory terminologies. Our goal in the remaining section is to present a comprehensive model to capture the value of data communication such that in the ideal case, every node's message can reach every other node. One trivial solution is to provide incentives for every single node to become relay nodes, so that a connected network is instantly obtained. What makes the problem challenging is that we would like to solve for the minimum set of relay nodes while still guaranteeing network connectedness.

Definition 5: A *desirable topology* of an autonomous network has the following properties:

1) the network is connected; and
2) average message relay cost should be minimized.

Fig. 1 is an example of a desirable topology, whereas Fig. 2a and Fig. 2b are both undesirable. This is because Fig. 2a does not meet the first criterion of a desirable topology, and Fig. 2b does not meet the second criterion. Note that the desirable topology may dynamically change over time. This happens when network nodes enter or leave the system over time, or when other network parameters change. Regardless of these changes, however, the mechanism should always converge back to the correct desirable topology in a finite number of steps. We now proceed to define the necessary mathematical notations used in our network model as well as our evolutionary topologies problem.

Parameter	Meaning of Parameter
$n, n \geq 0$	Number of nodes
N	The entire set of nodes
R_j	Set of reachable nodes from j
A_j	Set of nodes adjacent to j
$[m_{jk}], m_{jk} \geq 0$	Benefits to j of reaching k
$[\pi_{jk}], \pi_{jk} \geq 0$	Probability of j sending one message to k
$[c^*_{jk}], c^*_{jk} \geq 0$	Minimum relay cost from j to k
$c_l, c_l \geq 0$	Relay cost of node l
$\mathcal{P}_{jk}$	Minimum cost path from j to k

TABLE I
SUMMARY OF NOTATIONS IN NETWORK MODEL

A. Network Model

Consider an autonomous network modeled as an undirected graph (N, E), where N is the set of nodes and E is the set of edges. Let $n = |N|$ be the number of autonomous nodes in this network. We wish to identify all important parameters in our network model.

Consider a benefits matrix $[m_{jk}]$, where $m_{jk} \geq 0$ represents the benefits enjoyed by node i for being able to reach node k. Since any node can trivially reach itself, we set $m_{jj} = 0$ for convenience[1]. Note that the benefit m_{jk} will only be realized if $k \in R_j$, where R_j is the set of reachable nodes from j.

In addition, the communication between nodes j and k incurs a relay cost. Let $c_l \geq 0$ be the cost incurred to node l to relay one message. Also, let c^*_{jk} be the minimum relay cost from node j to k. Following our previous example in Fig. 1, $c^*_{ae} = 2 + 3 = 5$. Since a node does not have to relay any message to itself, $c_{jj} = 0$ for all j. Similarly, any node can reach its one-hop neighbour without the help of any intermediate nodes to relay messages, so we have $c_{jk} = 0$ for all $k \in A_j$, where A_j is the set of nodes adjacent to j. Let $\mathcal{P}_{jk}$ be the minimum cost path from j to k. Then, in general:

$$c^*_{jk} = \begin{cases} 0 & \text{if } k = j \text{ or } k \in A_j \\ \sum_{l \in \mathcal{P}_{jk}} c_l & \text{otherwise} \end{cases}$$

Finally, consider the probability matrix $[\pi_{jk}]$, where $0 \leq \pi_{jk} \leq 1$ is the probability of node j sending one message to node k in the next time period. For simplicity, we assume that we are able to pick a sufficiently small time period so that each source-destination pair (j, k) can at most initiate one message during this time period. The mathematical notations are summarized in Table I.

B. The Evolutionary Topologies Problem

Definition 6: The *evolutionary topologies problem* is to design a mechanism that will provide an incentive for every node to converge towards a desirable topology in a finite number of steps.

[1]This simplification assumption will not affect the validity of our future derivations presented in this paper.

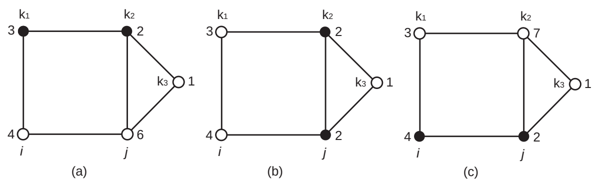


Fig. 3. The evolution of desirable topologies over time

Consider an example of an autonomous network in Fig. 3, where the desirable topologies are evolving over time. The three figures represent different snapshots of the network at different times, each of which has already converged to a stable desirable topology.

In Fig. 3a, nodes k_1 and k_2 are the only relay nodes. We can verify that this network is connected, since every node is reachable from every other node. For example, node i can reach k_3 via k_1 and k_2. Further, average relay cost has been minimized in the sense that the minimum cost path of every source-destination pair is achieved. For example, the minimum cost path from i to k_3 is through k_1 and k_2, and both of them are relay nodes.

In Fig. 3b, node j has decreased its relay cost from 6 to 2. All other relay costs remain unchanged. In order to maintain a desirable topology, the set of relay nodes evolve to nodes j and k_2. We can verify that the network topology is still desirable in this new solution.

Finally, in Fig. 3c, node k_2 has increased its relay cost from 2 to 7. All other relay costs remain unchanged. The new desirable topology is to make i and j the relay nodes. The network is once again connected. In fact, if we observe the transitions in these figures, we see that the network has shifted the relaying responsibilities from nodes $\{k_1, k_2\}$ to nodes $\{i, j\}$ over time. Node k_3, however, has never been a relay node. Indeed, if the nodes are not mobile, node k_3 will never become a relay node. The reason is that node k_3 will only be needed to relay messages between nodes j and k_2, but nodes j and k_2 are already adjacent to each other!

At this point, it would seem unfair to the relay nodes, since they have to contribute their private resources to the community. This is where mechanism design comes in and motivates them with the right amount of payments. These payments are transferred to the relay nodes from nodes who benefit from these relay services.

III. Mechanism Design and VCG Mechanism

In any game theory application, we first construct a model of the game and formulate the utility function of each node. Based on the set of utility functions, we then solve for certain equilibrium points, which represent the outcome of the game when stability is reached. Mechanism design, on the other hand, takes the inverse approach. Instead of finding the equilibrium outcomes from the model of the game, we have the freedom to choose what outcome we wish the game to stabilize at. Based on the desired outcome, we *manipulate* the utility function so that the desired outcome is achieved. In the following, we introduce the concepts and terminologies of mechanism design relevant to this paper. For a more in-depth treatment of the subject, refer to the excellent resources [2], [5].

Consider a game of n nodes. Each node can decide on a strategy $s_i \in S_i$ based on their utility function $u_i(s_i, s_{-i})$. S_i is called the strategy space of i, and s_{-i} simply means the set of strategies chosen by all nodes except i. The following definition formally defines dominant strategy equilibrium.

Definition 7: A *dominant strategy equilibrium* s^* satisfies the condition

$$u_i(s_i^*, s_{-i}) \geq u_i(s_i, s_{-i}) \qquad \forall i, (s_i, s_{-i}) \in S$$

In other words, a dominant strategy is one that maximizes the utility of each node, *no matter what strategies other nodes choose*. Since each node's decision is independent of other nodes' decisions, a node does not have to know others' equilibrium strategies. The only common knowledge known to all nodes is the strategy space of each node. This is an important advantage when designing distributed algorithms. When some private information of a node is announced to the public, it is possible that the node is lying about this information in order to obtain a higher utility. Fortunately, a well-studied class of mechanisms called *strategyproof mechanism* states that each node can be motivated to become honest, or at least does not have any incentive to lie. In economics, private information is usually called *type*, denoted by $\theta_i \in \Theta_i$. In our evolutionary topologies problem, $[\pi_{jk}]$, $[m_{jk}]$, and $[c^*_{jk}]$ are all private information to individual nodes, and thus are considered types. The following definition formally defines strategyproof mechanism.

Definition 8: A mechanism is *strategyproof* if for all node i:
(1) the strategy space is to declare their types, $S_i = \Theta_i$;
(2) declaring the true type is a dominant strategy, $s_i^* = \theta_i$.

Not all mechanisms are strategyproof, but a theorem in mechanism design called the *Revelation Principle* states that we can focus on strategyproof mechanism without loss of generality [5].

Up to this point, we have only discussed the importance of strategyproof mechanisms, but have not mentioned how to solve the mechanism design problem. In essence, a solution to the mechanism design problem should manipulate: (1) the outcome $o(\cdot)$ of the game; and (2) the utility $u_i(\cdot)$ of each node through payment $p_i(\cdot)$.

The good news is that a general solution to the mechanism design problem exists. It is known as the *VCG mechanism*. It has been proven to be the only general solution for utilitarian mechanism design problems [6]. Furthermore, it is a well-known fact that all VCG mechanisms are strategyproof mechanisms. This means that if we solve a mechanism design problem with VCG, then there would be no incentive for any node to lie about their private information.

Definition 9: A *Vickrey-Clarke-Groves (VCG) mechanism* is the family of mechanisms $M(\theta) = (o(\theta), p(\theta))$ such that:

$$o^*(\theta) \in \arg\max_{o} \sum_{j=1}^{n} v_j(\theta_j, o(\theta)) \tag{1}$$

$$p_i(\theta) = \sum_{j \neq i} v_j(\theta_j, o^*(\theta)) + h_i(\theta_{-i})$$

$h_i(\theta_{-i})$ is an arbitrary function of θ_{-i} that is often defined as $h_i(\theta_{-i}) = -\sum_{j\neq i} v_j(\theta_j, o^*_{-i}(\theta_{-i}))$. Therefore, the VCG payment function becomes:

$$p_i(\theta) = \sum_{j \neq i} v_j(\theta_j, o^*(\theta)) - \sum_{j \neq i} v_j(\theta_j, o^*_{-i}(\theta_{-i})) \quad (2)$$

Eq. (1) is the VCG outcome function, and it states that we are optimizing the total valuation of a network. Eq. (2) is the VCG payment function, which can be interpreted intuitively. The first term in the VCG payment function is the total value of the system excluding node i's valuation, given that the optimal outcome $o^*(\theta)$ is achieved. The second term is similar to the first, except we assume that *node i were to withdraw from the game when finding the optimal outcome $o^*_{-i}(\theta_{-i})$*. In all practical applications, the second term is always less than or equal to the first term, because the withdrawal of node i can only make the system worse off (or unchanged in the best case). As a result, *non-negative* payments are guaranteed. In summary, the VCG payment function claims that a node should be paid as much as its *best alternative*.

IV. Derivations of VCG Solution in the Evolutionary Topologies Problem

As previously mentioned, the utility function $u_i(\cdot)$ is the sum of the valuation function $v_i(\cdot)$ and the payment function $p_i(\cdot)$:

$$u_i(\theta_i, o(\theta)) = v_i(\theta_i, o(\theta)) + p_i(\theta_i, o(\theta))$$

For simplicity of notations, we sometimes drop the variables in the brackets and simply write:

$$u_i = v_i + p_i$$

In order to apply the VCG mechanism in the evolutionary topologies problem, we first need a reasonable valuation function $v_i(\cdot)$. This function can then be used with Eq. (1) and Eq. (2) to solve for the VCG mechanism. In the remaining section, we are first going to derive the valuation function $v_i(\cdot)$ for our evolutionary topologies problem. This is followed by a derivation of the VCG outcome function $o(\cdot)$ and the VCG payment function $p_i(\cdot)$. We end this section with an interpretation of the results, and explore some of the desirable properties of our solution.

A. The Valuation Function

Consider the *net value* V_{jk} of sending each message from node j to k, expressed in the form of benefits minus cost, *i.e.*, $V_{jk} = m_{jk} - c_{jk}$. Formally, the net value V_{jk} is a random variable that is realized only if a message is being transmitted in the next time period with probability π_{jk}, *i.e.*,

$$V_{jk} = \begin{cases} m_{jk} - c_{jk} & \text{with probability } \pi_{jk} \\ 0 & \text{with probability } (1 - \pi_{jk}) \end{cases} \quad (3)$$

The valuation function $v_j(\cdot)$ can be expressed succinctly as a summation of *expected net values*. The expected valuation to node j is:

$$\begin{aligned} v_j &= \sum_{k \in N} E\{V_{jk}\} && (4) \\ &= \sum_{k \in N \setminus R_j} E\{V_{jk}\} + \sum_{k \in R_j \setminus A_j} E\{V_{jk}\} \\ &\quad + \sum_{k \in A_j} E\{V_{jk}\} + E\{V_{jj}\} && (5) \\ &= \sum_{k \in A_j} E\{V_{jk}\} + \sum_{k \in R_j \setminus A_j} E\{V_{jk}\} && (6) \\ &= \sum_{k \in A_j} [\pi_{jk}(m_{jk} - 0) + (1 - \pi_{jk})(0)] + \\ &\quad \sum_{k \in R_j \setminus A_j} [\pi_{jk}(m_{jk} - c_{jk}) + (1 - \pi_{jk})(0)] && (7) \\ &= \sum_{k \in A_j} \pi_{jk} m_{jk} + \sum_{k \in R_j \setminus A_j} \pi_{jk}(m_{jk} - c_{jk}) && (8) \end{aligned}$$

Eq. (4) is the expected net value of a certain network topology to node j. The summation is decomposed into four terms in Eq. (5). Eq. (6) is obtained by realizing that $V_{jj} = m_{jj} - c_{jj} = 0$ as well as $V_{jk} = 0$ for node k that is unreachable from j. Applying Eq. (3) to expand the expectations, we obtain Eq. (7). After simplifying, we end up with Eq. (8), which will be used in our upcoming derivations of VCG outcome and payment functions.

B. The VCG Outcome

In the context of our evolutionary topologies problem, an outcome $o = (o_1, o_2, \ldots, o_n)$ is an indicator vector showing whether node i is a relay node or not, *i.e.*,

$$o_i = \begin{cases} 1 & \text{if } i \text{ is a relay node} \\ 0 & \text{otherwise} \end{cases} \quad (9)$$

Applying the valuation function in Eq. (8) to the VCG outcome function in Eq. (1), the VCG optimal outcome o^* reduces to solving an optimization problem:

$$\max \sum_{j=1}^{n} \left[\sum_{k \in A_j} \pi_{jk} m_{jk} + \sum_{k \in R_j \setminus A_j} \pi_{jk}(m_{jk} - c_{jk}) \right] \quad (10)$$

Recall that $[m_{jk}]$ is the benefits matrix, $[c_{jk}]$ is the cost matrix, and $[\pi_{jk}]$ is the probability matrix. Note that all these values are determined exogenously, for example, by the application running on each node. Since $\pi_{jk} \geq 0$ and $m_{jk} \geq 0$, the first term in Eq. (10) must be non-negative.

To maximize the second term, we should also find the minimum relay cost $c_{jk} = c^*_{jk}$ by solving for the minimum cost path $\mathcal{P}_{jk}$ from j to k. Note that the second term can be negative if $m_{jk} < c^*_{jk}$. In this case, to maximize Eq. (10), we would rather not transmit the message from node j to k. This consequence is reasonable because if the relay cost of a message outweighs its benefits, then it is only rational not to send the message. However, in most cases where $m_{jk} \geq c^*_{jk}$, node j not only prefers to send k the message, but also specifically along the minimum cost path. In other words, if a node is along any minimum cost path of any source-destination pair, it must a relay node in order to guarantee global optimality.

VCG Outcome $o^* = (o_1^*, o_2^*, \ldots, o_n^*)$:

$$o_i^* = \begin{cases} 0 & \text{if node } i \text{ is not on any} \\ & \text{minimum cost path} \\ 1 & \text{if node } i \text{ is on at least one} \\ & \text{minimum cost path with } m_{jk} \geq c_{jk}^* \end{cases} \quad (11)$$

C. The VCG Payment

Previously, we have only identified which node should become a relay node in order to achieve global optimality, but have left out the issue of *how* we motivate each node to become a relay node. In the following, we address this issue by solving for the VCG payment in the evolutionary topologies problem, which will provide the proper incentives to each node to achieve global optimality. We make the simplifying assumption that the flow of payments can only occur between adjacent nodes. Payments made to nodes further than one hop away are considered unrealistic, because we cannot be certain that payments will be relayed faithfully, when our incentive mechanism is not even in place yet.

Returning to the example in Fig. 1, suppose node a wants to send a message to e through the minimum cost path $\langle a, b, d, e \rangle$. In this case, node a pays b, b pays d, and d relays to e without further payment. Only nodes b and d receive a payment, since they are the only two relay nodes on this minimum cost path. Restricting payments among adjacent nodes has another advantage. In the previous example, instead of having one large global game that involves all nodes $\{a, b, c, d, e\}$, the problem has been reduced to several local games, as seen by rearranging Eq. (2):

$$\begin{aligned} p_i(\theta) &= \sum_{j \neq i} v_j(\theta_j, o^*(\theta)) - \sum_{j \neq i} v_j(\theta_j, o^*_{-i}(\theta_{-i})) \\ &= \sum_{j \neq i} (v_j - v_j^{-i}) \quad (12) \\ &= \sum_{j \in A_i} (v_j - v_j^{-i}) \quad (13) \\ &= \sum_{j \in A_i} p_{ji} \quad (14) \end{aligned}$$

Eq. (12) is merely a simplification of notations. v_j^{-i} can be interpreted as the valuation of node j *given that node i is not a relay node*. Eq. (13) is a consequence of the fact that we only consider a local game involving adjacent nodes. $p_{ji} = (v_j - v_j^{-i})$ is the payment from node j to node i. From Eq. (14), we observe that *the total payment received by a node is the sum of individual payments from each adjacent node*. Therefore, when calculating the payment p_b to node b in Fig. 1, we focus on a local game centered around node b, involving only nodes $\{a, b, d\}$. Similarly, when calculating p_d, we consider the local game with nodes $\{b, c, d, e\}$.

We emphasize that the payments p_{ji} are not made on a per-message basis, but rather, on an average basis per unit time. This alleviates a node's burden of having to make a separate decision each time it receives a message, and will increase the efficiency of our solution. For every constant time period, each node calculates how much *on average* it has to pay their one-hop neighbours. Their neighbours perform the same calculations based on their own perspectives. As a result, a node who pays less than the income they receive will be gaining capital over time. Intuitively, we expect the nodes located at the center of the network to have a net *inflow* of payments, since they are relaying for many other nodes. In contrast, nodes at the network edge do not need to relay messages, and are expected to have a net *outflow* of payments. Overall, the entire autonomous network functions as one *closed system* in the sense that the net inflow or outflow of virtual currency is zero. Virtual currencies are only *redistributed* within the network.

To solve for the VCG payment for the evolutionary topologies problem, we apply Eq. (8) to p_{ji}:

$$\begin{aligned} p_{ji} &= v_j - v_j^{-i} \\ &= \sum_{k \in A_j} \pi_{jk} m_{jk} + \sum_{k \in R_j \setminus A_j} \pi_{jk}(m_{jk} - c_{jk}^*) - \\ &\quad \sum_{k \in A_j} \pi_{jk} m_{jk}^{-i} - \sum_{k \in R_j \setminus A_j} \pi_{jk}(m_{jk}^{-i} - c_{jk}^{-i}) \quad (15) \end{aligned}$$

There are four different types of destination node k to be considered here, each of which yields a slightly different m_{jk}^{-i} and c_{jk}^{-i}.

Case 1 $[k \in A_j]$: Node k is adjacent to node j. In this case, the benefit m_{jk}^{-i} is guaranteed, since adjacent nodes must be reachable. Also, as mentioned before, relay cost is zero for adjacent node k.

$$\begin{aligned} m_{jk}^{-i} &= m_{jk}, \quad \text{and} \\ c_{jk}^{-i} &= 0 \end{aligned}$$

Case 2 $[k \in R_j \setminus A_j$ and $i \notin \mathcal{P}_{jk}]$: The minimum cost path from j to k does not pass through i. In this case, both the benefit m_{jk} and the minimum cost c_{jk}^* are achieved independent of whether i is a relay node.

$$\begin{aligned} m_{jk}^{-i} &= m_{jk}, \quad \text{and} \\ c_{jk}^{-i} &= c_{jk}^* \end{aligned}$$

Case 3 $[k \in R_j \setminus A_j$ and $i \in \mathcal{P}_{jk}$ and $\mathcal{P}_{jk}^{-i} \exists]$: An alternate path $\mathcal{P}_{jk}^{-i}$ exists from node j to k without passing through i. In this case,

$$\begin{aligned} m_{jk}^{-i} &= m_{jk}, \quad \text{and} \quad (16) \\ c_{jk}^{-i} &= \sum_{l \in \mathcal{P}_{jk}^{-i}} c_l \quad (17) \end{aligned}$$

Eq. (16) simply states that the benefit to node j of reaching node k is the same, regardless of what path is taken to get there. Eq. (17) calculates the minimum cost of the path from node j to k, *without going through node i*. $\mathcal{P}_{jk}^{-i}$ is interpreted as the *best alternative* to the minimum cost path $\mathcal{P}_{jk}$.

Case 4 $[k \in R_j \setminus A_j$ and $i \in \mathcal{P}_{jk}$ and $\mathcal{P}_{jk}^{-i} \nexists]$: No alternate path exists from node j to node k. In this case,

$$\begin{aligned} m_{jk}^{-i} &= 0, \quad \text{and} \quad (18) \\ c_{jk}^{-i} &= 0 \quad (19) \end{aligned}$$

Eq. (18) and (19) state that there is neither benefit or cost to node j, if node i were to withdraw from the network. Note that node i is a bottleneck node on the path from j to k.

We apply the results of Case 1 to 4 to Eq. (15):

$$\begin{aligned}
p_{ji} &= v_j - v_j^{-i} \\
&= \sum_{k\in A_j} \pi_{jk} m_{jk} + \sum_{k\in R_j \setminus A_j} \pi_{jk}(m_{jk} - c^*_{jk}) - \\
&\quad \sum_{k\in A_j} \pi_{jk} m^{-i}_{jk} - \sum_{k\in R_j \setminus A_j} \pi_{jk}(m^{-i}_{jk} - c^{-i}_{jk}) \qquad (20) \\
&= \sum_{k\in\{R_j \setminus A_j \cap i \in \mathcal{P}_{jk} \cap \mathcal{P}^{-i}_{jk} \exists\}} \pi_{jk}(c^{-i}_{jk} - c^*_{jk}) \\
&\quad + \sum_{k\in\{R_j \setminus A_j \cap i \in \mathcal{P}_{jk} \cap \mathcal{P}^{-i}_{jk} \nexists\}} \pi_{jk}(m_{jk} - c^*_{jk}) \qquad (21)
\end{aligned}$$

The first and third term of Eq. (20) is canceled according to Case 1, whereas the second and fourth term is simplified according to Case 2 to 4. Eq. (21) is the final VCG payment function.

VCG Payment p_{ji}: In summary, in an autonomous network, node k will contribute to the payment p_{ji} an amount equal to $\pi_{jk}(c^{-i}_{jk} - c^*_{jk})$, if:

1) k is a reachable node;
2) k is not adjacent to j;
3) i is on the minimum cost path from j to k;
4) i is not a bottleneck node from j to k.

On the other hand, node k will contribute to the payment an amount equal to $\pi_{jk}(m_{jk} - c^*_{jk})$, if:

1) k is a reachable node;
2) k is not adjacent to j;
3) i is on the minimum cost path from j to k;
4) i is a bottleneck node from j to k.

D. Properties of Our Solution

To intuitively understand our VCG outcome and payment, several properties of our solution are given below. These properties are valuable in two ways. First, they serve as necessary conditions that every well-designed mechanism of the evolutionary topologies problem should meet. Second, they are good verifications against common sense.

Property 1: A non-relay node i receives no payment, $p_i = 0$.
Proof: Eq. (11) demands that a non-relay node i must not be on any minimum cost path, *i.e.*, $i \notin \mathcal{P}_{jk}$, $\forall\ j \in A_i, k \in N$. If node i is not relaying messages for *any* adjacent node at all, then from Eq. (21), $p_{ji} = 0$, $\forall\ j \in A_i$. As a result, $p_i = \sum_{j\in A_i} p_{ji} = 0$. □

Property 2: A relay node i has non-negative utility, $u_i \geq 0$.
Proof: Since c^*_{jk} is the minimum relay cost from j to k, we have $c^{-i}_{jk} \geq c^*_{jk}$. Furthermore, Eq. (11) demands that $m_{jk} \geq c^*_{jk}$ for relay nodes. Applying these inequalities in Eq. (8) and Eq. (21) gives $v_i \geq 0$ and $p_i \geq 0$ respectively for all i. Therefore, $u_i = v_i + p_i \geq 0$. □

Property 3: (a) A node can obtain at most equal, but never higher, utility than declaring the truthful private information. (b) A node will derive strictly worse utility if it declares untruthful private information that results in sub-optimal global outcome.
Proof: The proofs are almost identical to the general proof of VCG being a strategyproof mechanism, and therefore are

```
$j \leftarrow Id$
msgIn $\leftarrow$ receiveMsg()
$i \leftarrow$ msgIn.$Id$
for each $k \in N \setminus \{$msgIn.$A_i\}$
    challenger $\leftarrow$ msgIn.$c^*_{ik}$ + msgIn.$c_i$
    challenger2 $\leftarrow$ msgIn.$c2_{ik}$ + msgIn.$c_i$
    if challenger $< c^*_{jk}$
        $h_{jk} \leftarrow i$
    end if
    $c^*_{jk} \leftarrow \min($challenger$, c^*_{jk})$
    $tmp \leftarrow \{c^*_{jk}, c2_{jk},$ challenger, challenger2$\}$
    $c2_{jk} \leftarrow$ secondMin($tmp$) with next hop $\neq h_{jk}$
    $h2_{jk} \leftarrow$ update next hop
end for

for each $i \in A_j$
    $p_{ji} \leftarrow 0$
    for each $k \in \{R_j | h_{jk} = i, k \neq i\}$
        if $c2_{jk} = \infty$ and $c^*_{jk} \neq \infty$
            $p_{ji} \leftarrow p_{ji} + \pi_{jk}(m_{jk} - c^*_{jk})$
        elseif $c2_{jk} \neq \infty$ and $c^*_{jk} \neq \infty$
            $p_{ji} \leftarrow p_{ji} + \pi_{jk}(c2_{jk} - c^*_{jk})$
        end if
    end for
end for
```

TABLE II
THE DISTRIBUTED ALGORITHM RUNNING AT EACH NODE

omitted due to space constraints. The interested reader is referred to page 877-880 in [5].

Property 4: Nodes with degree 1 does not receive any payment, and will never be relay nodes.
Proof: For a node i that has only one adjacent node j, the set of $k \in \{R_j \setminus A_j \cap i \in \mathcal{P}_{jk}\}$ is empty, and so $p_{ji} = 0$. This implies that $p_i = p_{ji} = 0$. Finally, nodes that receive no payments are not relay nodes. □

Property 5: If $m_{jk} > c^*_{jk}$ for all source-destination pairs (j, k), the autonomous network will be connected.
Proof: For a message to be relayed all the way from any node j to any other node k in an autonomous network, every intermediate node must be a relay node. Consider the minimum cost path $\mathcal{P}_{jk} = \{i_1, i_2, \ldots, i_L\}$ from j to k that passes through L intermediate nodes. Given that $m_{i_1 i_2} > c^*_{i_1 i_2}$, $m_{i_2 i_3} > c^*_{i_2 i_3}$, $m_{i_3 i_4} > c^*_{i_3 i_4}$, $\cdots$, we have:

$$o_i = 1, \qquad \forall\ i = i_1, i_2, \cdots, i_L$$

As a result, node k is reachable from j. Generalizing this result for all source-destination pairs (j, k) in the network, connectedness will be guaranteed. □

V. The Distributed Algorithm

We have so far derived the VCG outcome and payment functions, as applied to the evolutionary topologies problem. We have also stated some observations and properties of our solution. We will design the distributed algorithm based on our centralized solution in the previous section. Convergence and efficiency issues will also be discussed.

The first step towards solving for the correct optimal outcome as well as calculating the accurate payment is to identify all private information that has to be made public. In the context of our evolutionary topologies problem, the private information includes the probability matrix $[\pi_{jk}]$, the benefits matrix $[m_{jk}]$, as well as the relay cost matrices $[c^*_{jk}]$ and $[c^{-i}_{jk}]$. From Eq. (21), payment p_{ji} is a function of π_{jk}, m_{jk}, c^*_{jk}, and

c_{jk}^{-i}. Both π_{jk} and m_{jk} are private information to node j, and therefore are known. The relay costs c_{jk}^* and c_{jk}^{-i}, however, have to be passed to node j from other nodes. This can be achieved by any distributed shortest-path routing algorithm such as the distance-vector or the link-state algorithm. In this paper, we have chosen the distance-vector algorithm to demonstrate the distributed computation of VCG payments, although there is no apparent constraint preventing us from using any other algorithm.

Since our proposed algorithm is distributed, it runs on every single node in the network. Similar to the distance-vector algorithm, local information at each node such as relay cost is disseminated throughout the network by message passing between adjacent nodes. We assume that each node in the network possesses a unique identifier, such as an IP address. Suppose the algorithm is running on node j (the payer), and has just received an incoming message from its adjacent node i (the payee). For every node k that is reachable but not adjacent to j, we would like to keep track of four pieces of information:

1) $c_{jk}^* =$ Minimum cost from node j to k;
2) $h_{jk} =$ Next hop on the minimum cost path $\mathcal{P}_{jk}$;
3) $c2_{jk} =$ Second-minimum cost from node j to k, where the first hop is not h_{jk};
4) $h2_{jk} =$ Next hop on $\mathcal{P}_{jk}^{-i}$.

The minimum cost from node i to k is easily obtained by the well-known distance-vector equation:

$$c_{jk}^* = \min(c_{jk}^*, c_{ik}^* + c_i) \tag{22}$$

c_{jk}^* is the minimum cost from j to k, as far as node j is concerned. $c_{ik}^* + c_i$ can be considered a *challenger* to the current minimum cost. If $c_{ik}^* + c_i < c_{jk}^*$, then a better path has just been discovered. In this case, the minimum cost c_{jk} will be updated to the new minimum $c_{ik}^* + c_i$, and h_{jk} will also be updated to i. We adopt the convention that no path updates are made when there is a tie, *i.e.*, $c_{ik}^* + c_i = c_{jk}^*$. Finally, when $c_{ik}^* + c_i > c_{jk}^*$, the challenger cannot beat the current minimum cost, so no path updates are necessary. Note that even if the challenger $c_{ik}^* + c_i$ cannot beat the current minimum cost, it can still be our *best alternative to the current minimum cost path*, $c2_{jk}$.

Calculating $c2_{jk}$ and $h2_{jk}$ are slightly more challenging. Finding $c2_{jk}$ can be interpreted as finding the second-minimum. In addition to the previous challenger $c_{ik}^* + c_i$, we also have a second challenger $c2_{ik} + c_i$. Our end goal is to find the second-minimum from the set $\{c_{jk}^*, c2_{jk}, c_{ik}^* + c_i, c2_{ik} + c_i\}$. Although detailed comparisons are not presented here, it can be seen that the second-minimum out of 4 elements can be found in constant time $O(1)$.

The second part of our distributed algorithm calculates the payments according to Eq. (21). For each node i (the payee) that is adjacent to node j (the payer), we find all destination node k such that:

1) k is a reachable node;
2) k is not adjacent to j;
3) i is on the minimum cost path from j to k.

And if c_{jk}^* is finite but $c2_{jk}$ is infinite, then there is no alternate path from node j to k, except going through i. Based on Eq. (21), the payment is updated as follows:

$$p_{ji} \leftarrow p_{ji} + \pi_{jk}(m_{jk} - c_{jk}^*) \tag{23}$$

On the other hand, if c_{jk}^* and $c2_{jk}$ are both finite, then there is an alternate path from node j to k that does not pass by i. Based on Eq. (21), the payment is updated as follows:

$$p_{ji} \leftarrow p_{ji} + \pi_{jk}(c2_{jk} - c_{jk}^*) \tag{24}$$

Table II summarizes the distributed algorithm that calculates the VCG payments.

Correctness: The correctness of our distributed algorithm depends on two parts: (1) whether this algorithm finds the minimum cost and second-minimum cost correctly; and (2) whether the payment equations are correct. The correctness of the first part relies on the correctness of distance-vector algorithms, the proof of which is readily available from references such as [7]. The correctness of the second part relies on our derivations of Eq. (21) in Sec. IV-C.

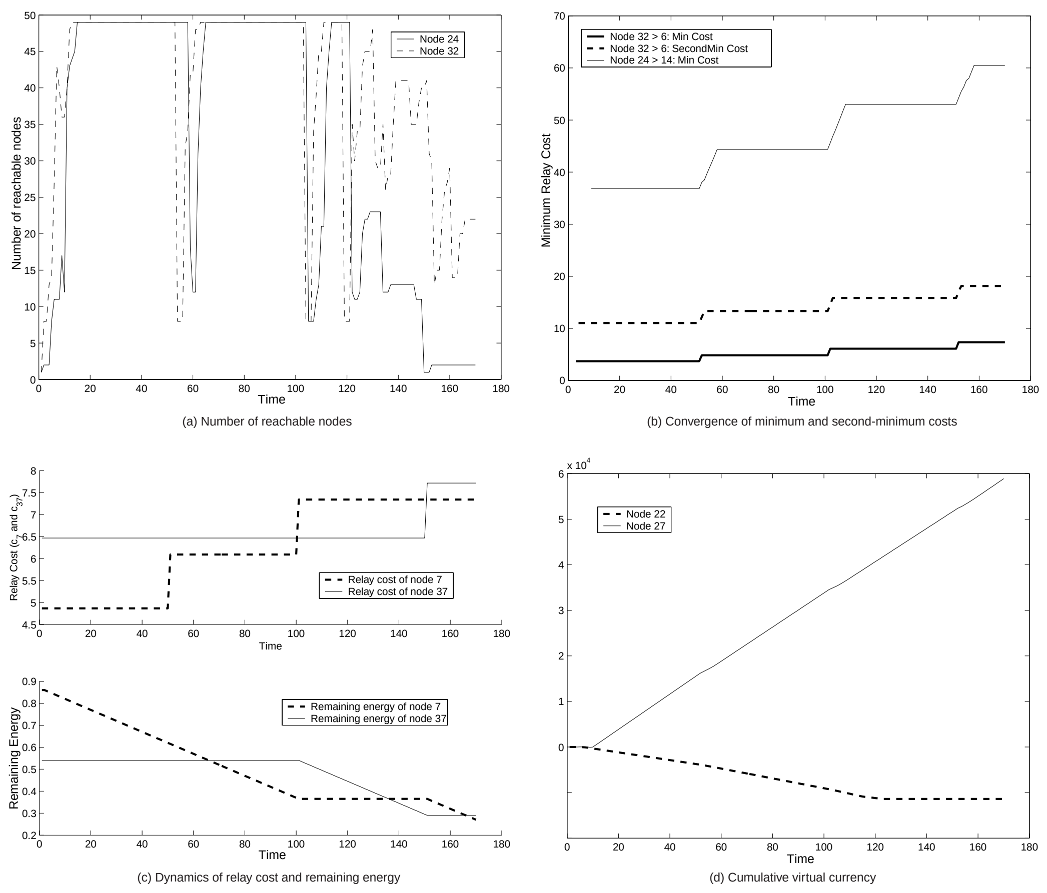
Convergence: If messages are exchanged on a periodic basis, our distributed algorithm is guaranteed to converge in $O(d)$ time, where d is the diameter of the network graph. The graph's diameter is the largest number of nodes which must be traversed from one node to another in a loop-free path. Note that messages exchanged between adjacent nodes do not have to be synchronous. Each time a node receives a new message, the algorithm in Table II will be invoked once, and running the algorithm will ensure that the minimum and second-minimum costs are found, *based on the available information so far*. Therefore, the only requirement for the algorithm to converge is that if some private information is revealed by a destination node k at a certain time, then it must eventually reach every other node on the network in a finite amount of time. Finally, we note that the number of control messages of this distributed algorithm is $O(n^2)$, as in any distance-vector algorithm.

In addition to calculating the VCG payments, the algorithm must also determine when to relay messages for another node. According to our analysis in Sec. IV-B, a node j should relay messages for another node only when it is part of at least one minimum cost path $\mathcal{P}_{jk}$. However, we have shown previously that the next hop of any minimum cost path will receive a payment p_{ji}. As a result, a node should set $o_j = 1$ only when it receives a payment from adjacent nodes. Since this payment is derived from the VCG mechanism and has been accurately calculated by our distributed algorithm, each node will have just the right amount of incentive to relay messages.

VI. Evaluations

In the following, we evaluate our distributed algorithm through simulations in Matlab. These simulations explore some of the intuitive results discussed in previous sections.

We begin by generating $n = 50$ nodes uniformly distributed on a 10-by-10 grid. Two nodes are joined by a link if the Euclidean distance between them is less than 2 units. The resulting network is shown in Fig. 4. In the simulation, we have assumed that messages are passed synchronously between adjacent nodes at each time period, even though our distributed algorithm does not rely on this assumption to converge. We simulated a total of 170 time periods. During



(a) Number of reachable nodes

(b) Convergence of minimum and second-minimum costs

(c) Dynamics of relay cost and remaining energy

(d) Cumulative virtual currency

Fig. 5. Simulation results

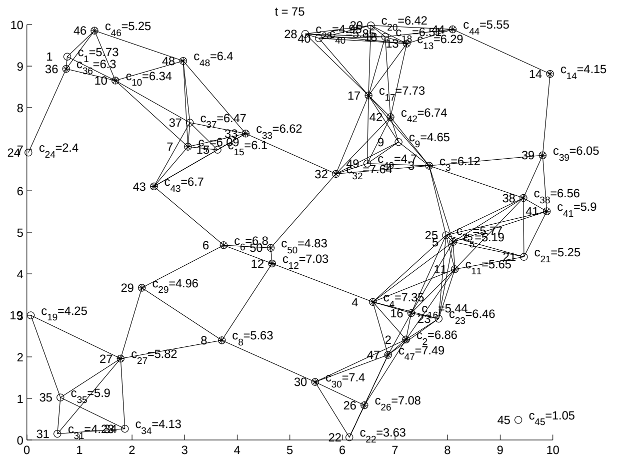


Fig. 4. Network topology in our simulation

Parameters	Range of Parameters
p_{jk}	$0 \leq \pi_{jk} \leq 1$
m_{jk}	$0 \leq m_{jk} \leq 50$
c_i	$0 \leq c_i \leq 10$
$E_0(i)$	$0.5 \leq E_0(i) \leq 1$
$x_e(i,t)$	$0 \leq x_e(i,t) \leq 1$
$x_{bw}(i)$	$0 \leq x_{bw}(i) \leq 1$

TABLE III
SIMULATION PARAMETERS AND THEIR RANGES

each period, every node j receives and interprets messages sent by its adjacent node $i \in A_j$.

The probability matrix $[p_{jk}]$ and the benefits matrix $[m_{jk}]$ are generated randomly in this simulation. The relay cost c_i is modeled as a function of remaining energy (as in a wireless ad hoc network) and bandwidth. For example, it is reasonable to expect that the cost of relaying messages for another node is larger when a node has less energy. Similarly, a low residual bandwidth usually implies a higher cost. The relay cost function is:

$$c_i = 10 - 5x_e(i,t) - 5x_{bw}(i) \tag{25}$$

where:

$$x_e(i,t) = \max\{E_0(i) - 0.01t, 0\}$$

$$x_{bw}(i) = \begin{cases} 1/|A_i| & \text{if } |A_i| \neq 0 \\ 1 & \text{if } |A_i| = 0 \end{cases}$$

$E_0(i)$ is the initial energy of node i, and is randomly generated. We further define x_e as the fraction of remaining energy, so $0 \leq x_e \leq 1$, and define x_{bw} as the fraction of remaining bandwidth, so $0 \leq x_{bw} \leq 1$. Note that we have assumed $x_{bw}(i)$ is simply inversely proportional to the number of adjacent nodes. Therefore, it is not time-varying, whereas $x_e(i,t)$ is decreasing linearly in time. Overall, cost $c_i(t)$ is increasing in time. The range of values of each simulation parameter is summarized in Table III.

The above relay cost, modeled as a function of energy and bandwidth, is only chosen for the purpose of our simulation. In general, our theoretical derivations in Sec. IV allow us to use virtually any cost model, as long as the model reflects the true preferences of each node. In this simulation, we wish to have some dynamics in the relay costs of each node, so that the evolution of desirable topologies becomes apparent in the simulation results.

Although the cost is continuously changing in time according to our cost model in Eq. (25), we only update the new relay costs to every node at $t = 50$, 100, and 150. This is because our algorithm takes $O(d)$ time periods to converge, where d is the diameter of network graph. If the cost is continuously changing for the entire duration of the simulation, the convergence will always be delayed by $O(d)$ time steps. Using discrete cost updates, we should expect to observe clear convergence behaviour shortly after $t = 50$, 100, and 150.

From Table III, note that the values of m_{jk} is on average larger than the values of c_{jk} to ensure that the network topology is at least mostly connected when our distributed algorithm is executed. Strictly speaking, according to Property 5 in Sec. IV-D, connectedness is not guaranteed in our network because we have not imposed the condition $m_{jk} < c^*_{jk}$ for all source-destination pairs (j, k). However, the current simulation indicates that a looser condition on m_{jk} and c^*_{jk} can also result in a connected network.

In Fig. 4, the relay nodes, as determined by our algorithm, are marked with an asterisk $\circledast$. Property 4 in Sec. IV-D stated that a node with only one adjacent node is never a relay node. This property is verified by observing that node 24 is not a relay node in Fig. 4. We further plots the number of reachable nodes from node 24 and 32 respectively in Fig. 5a. We observe that our algorithm stabilizes at around $t = 15$ and $t = 63$, and that both nodes 24 and 32 are able to reach 49 nodes upon convergence. (The unreachable node is the singleton node 45 at the right-bottom corner of the Fig. 4.) Beyond $t = 100$, we begin to observe a more random behaviour, and this is due to the fact that some nodes started to go out of energy, and therefore, the number of reachable nodes drop. In essence, we have achieved a desirable network topology where every node can reach every other node in the network during the network's lifetime.

Fig. 5b plots the minimum and second-minimum costs c^*_{jk} and $c2_{jk}$ from node 32 to 6 and from 24 to 14 as they evolve over time. Node 24 at the network edge has no alternate path to 14. Its second-minimum cost is therefore infinity, and has been left out of the figure. Recall that we update the relay costs only every other 50 time periods. The convergence of minimum cost can thus be observed shortly after time $t = 50$, 100, and 150, where minimum costs are held constant before the next update.

Fig. 5c is an excellent example of the evolutionary nature of our problem. Consider the top graph, which plots the relay cost of nodes 7 and 37 over time. Note the proximity of these two nodes in the network. Node 7 starts at a lower cost than node 37, and is therefore chosen to be the relay node. After two cost updates according to Eq. (25), the relay cost of node 7 has increased beyond the cost of node 37 at $t = 100$. Consequently, the responsibility of relaying messages has shifted from node 7 to node 37. This transfer of relay responsibility is especially obvious from the bottom graph, which plots the remaining energy of these nodes over time. Note that energy decreases only when a node is a relay node. After the third cost update at $t = 150$, the relay cost of node 37 is once again greater than the cost of node 7, so node 7 once again becomes a relay node. If the energy of these nodes were unlimited, these two nodes will continue to switch their roles at every subsequent cost update. This demonstrates an interesting side-effect of our solution — an even distribution of network resource consumption over time, thereby lengthening network lifetime.

Finally, we would like to explore the flow of payments between adjacent nodes. In Fig. 5d, the cumulative virtual currency of node 22 and 27 are plotted as a function of time. Node 22 experiences a net outflow of payments, which is to be expected for non-relay nodes. In contrast, the bottleneck node 27 has a net inflow of payments, which is also expected for bottleneck nodes.

VII. Related Work

Although many previous work applied game theory to solve network problems [8], the application of *mechanism design* to networking was pioneered by Nisan *et al.* [2] and Feigenbaum *et al.* [3], [4], and have later been discussed in other representative work such as Roughgarden [9]. A common theme in their work is that VCG is applied to solve the shortest-path routing problem.

Though our work also applies the VCG mechanism from microeconomics to solve networking problems, our network model and problem formulation are different. *First*, while previous papers considers the computation at the message level per sender-receiver pair, our work focuses on the handling of computation at a time-interval level: given a traffic distribution and benefit matrix, we attempt to form the network that maximizes total utility of all the nodes. This *adaptive* nature of our distributed algorithm would ensure that our topology remains optimal, despite varying network capacity, traffic patterns, and applications. *Second*, our formulation of the problem introduces the notion of benefits to communicate between two nodes. This addition not only promotes a more accurate model of each node's preferences and payments, but also enables us to achieve desirable global properties. The most important focus of this paper is network connectedness, which was proven to be achievable in Sec. IV-D when benefits are sufficiently large. Another desirable global property is that our algorithm will spread out the consumption of network resources, thus maximizing network utilization. Both of these properties have been demonstrated to work in our evaluation in Sec. VI. *Finally*, our derivations are probabilistic, rather than deterministic. In microeconomic terminology, we have designed a *Baysian mechanism*.

To our knowledge, Srinivasan *et al.* [10] is the only work to have explored the selfish behaviour in a scenario similar to our evolutionary topologies problem. Our work is different from, and in some cases, improves upon their work in at least three different ways. First, their work assumed the primary goal of meeting a system lifetime constraint, whereas our main objective here is to achieve a connected network. Moreover, Nash equilibrium was used throughout their work. However, since *solving for the Nash equilibrium requires knowledge of every other node's Nash equilibrium strategy*, we feel that this is too strong an assumption for an autonomous network. In contrast, we implemented the *dominant-strategy* equilibrium, which requires no knowledge of other nodes' equilibrium strategies except the strategy space. Third, and most importantly, their work has not considered a mechanism design approach, which uses payments to motivate nodes to cooperate. Therefore, if the Nash equilibrium found from their analysis is undesirable, there is no way to modify the structure of the game so that a more desirable state is reached.

VIII. Conclusion

In this paper, we have solved the evolutionary topologies problem by applying the VCG mechanism. The resulting topology is guaranteed to maximize the global valuation of the network, provided that the right amount of incentives is given to each relay node in the form of payments. Under the designed payment scheme, we showed that a relay node always receives a non-negative utility, and so truthful private information will always be revealed. Finally, we transformed our centralized solution to a fully distributed algorithm. This algorithm, which has been evaluated by a direct implementation in Matlab, dynamically adapts to changes in network parameters and conditions.

References

[1] L. Buttyan and J.P. Hubaux, "Enforcing Service Availability in Mobile Ad-Hoc WANs," in *IEEE/ACM MobiHoc*, 2000.

[2] N. Nisan and A. Ronen, "Algorithmic Mechanism Design," *Games and Economic Behavior,* vol. 35, pp. 166–196, 2001.

[3] J. Feigenbaum and S. Shenker, "Distributed Algorithmic Mechanism Design: Recent Results and Future Directions," in *Proceedings of ACM Dial-M,* Atlanta, Georgia, September 2002.

[4] J. Feigenbaum, C. Papadimitriou, R. Sami, and S. Shenker, "A BGP-based Mechanism for Lowest-Cost Routing," in *Proceedings of ACM PODC,* New York, 2002, pp. 173–182, ACM Press.

[5] A. Mas-Colell, M. Whinston, and J. Green, *Microeconomic Theory*, chapter 23, Oxford University Press, New York, 1995.

[6] J. Green and J.J. Laffont, "Characterization of satisfactory mechanisms for the revelation of preferences for public goods," *Econometrica*, pp. 427–438, 1977.

[7] T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein, *Introduction to Algorithms*, chapter 24, The MIT Press, Cambridge, Massachusetts, 2001.

[8] C.H. Papadimitriou, "Algorithms, Games, and the Internet," in *Proceedings of STOC*, Crete, Greece, July 2001.

[9] T. Roughgarden, "Selfish Routing," *PhD thesis, Cornell University, http://www.cs.berkeley.edu/ timr/papers/thesis.pdf*, 2002.

[10] V. Srinivasan, P. Nuggehalli, C. Chiasserini, and R. Rao, "Cooperation in Wireless Ad Hoc Networks," in *Proceedings of INFOCOM*, 2003.