Internet Mathematics Vol. 5, No. 4: 395–410

Incentive-Compatible Interdomain Routing with Linear Utilities

Alexander Hall, Evdokia Nikolova, and Christos Papadimitriou

Abstract. We revisit the problem of incentive-compatible interdomain routing, examining the quite realistic special case in which the utilities of autonomous systems (ASes) are linear functions of the traffic in the incident links and the traffic leaving each AS. We show that incentive-compatibility toward maximizing total welfare is achievable efficiently, and in the uncapacitated case, by an algorithm that can be easily implemented by the border gateway protocol (BGP), the standard protocol for interdomain routing.

1. Introduction

The Internet is in many ways a mysterious object, a complex wonder that we must approach with the same puzzled humility with which neuroscientists approach the brain and biologists the cell. Even at the most basic level of routing, for example, it is not clear at all how and why the approximately twenty thousand independent, and presumably selfish, autonomous systems (ASes) cooperate to provide connectivity between any two of them. The problem is quintessentially economic. ASes are known to have confidential financial agreements on how traffic between them is to be handled and paid for, and such agreements are reflected in the ways in which each AS routes traffic. We can think of the ASes as nodes of an undirected graph, with edges signifying the existence of such an agreement between the two endpoints (equivalently, the possibility of traffic routed directly, in either direction, between the two). In particular, ASes communicate in terms of the border gateway protocol (BGP), a flexible protocol allowing them

1542-7951/08 $0.50 per page

to implement routing decisions of arbitrary complexity, by "advertising" paths to adjacent (in the graph) ASes, and selecting among the paths advertised by their neighbors. Hence, the Internet is in essence an economy, a game, an arena where agents act selfishly and are affected by everybody's decisions; consequently, one can ask of it the questions we usually ask of such systems, for example the price of anarchy, or the possibility of incentive-compatible maximization of social welfare (questions typically studied by algorithmic mechanism design [Nisan and Ronen 99]); in this paper we address the latter.

Indeed, starting with [Feigenbaum et al. 02], BGP has been studied in the past under the lens of algorithmic mechanism design, and in particular in terms of the Vickrey–Clarke–Groves (VCG) mechanism (see, for example, [Mas-Colell et al. 95] for an introduction to mechanism design). It was noticed [Feigenbaum et al. 02] that social welfare can be optimized in routing if one assumes that each AS has a fixed per-packet cost via the VCG mechanism with payments, and in fact, that this can be achieved in a way that is very "BGP-friendly," i.e., can be implemented by minimal disruption of BGP's ordinary operation. Furthermore, it was observed that in the real Internet, VCG would result in relatively very small overpayments.

In a subsequent paper [Feigenbaum et al. 06b], the problem of more realistic BGP routing was addressed in the same spirit. Each AS was assumed to have a utility for each path to the destination (assumed in this literature to be a fixed node 0), and the goal was to maximize total utility. It was shown that the problem is too hard to solve in general even with no consideration to incentive compatibility, while a special case, in which the utility of a path depends only on the next hop, is easy to solve in an incentive-compatible way, but hard to implement in BGP. To establish this latter negative result, the authors of [Feigenbaum et al. 06b] formalize what it means for an algorithm to be "BGP-friendly": roughly speaking, a local distributed algorithm with quick convergence, small storage needs, and no rippling updates in case of small parameter changes. All said, the message of [Feigenbaum et al. 06b] was that if one models BGP routing a little more realistically, incentive compatibility becomes problematic. This negative message was ameliorated in [Feigenbaum et al. 06a], where it was pointed out that if one further restricts the special case of next-hop utilities so that paths are required to be of a particular kind mandated by the kinds of inter-AS agreements seen in practice, called *valley-free* in this paper, BGP-friendly incentive compatibility is restored.

There is an extensive literature on BGP (see, for example, [Feamster 04, Gao 01, Griffin and Wilfong 99, Rekhter and Li 95, Stewart 98, Subramanian et al. 02]). The protocol has also been examined within other game-theoretic contexts, such as with respect to network creation games, e.g., [Anshelevich et al. 06, Fabrikant et al. 03], cooperative game theory [Papadimitriou 01], and BGP oscillation prediction [Fabrikant and Papadimitriou 07].

In this paper we present an elementary model of BGP routing. The key feature of our model is that *path preferences are based exclusively on per-packet costs and per-packet agreed-upon compensation* between *adjacent* nodes. In other words, we look into the utilities of each path to each AS, taken as raw data in previous literature, and postulate that they are linear functions of the traffic, depending on two factors: objective per-packet costs to each AS for each incoming or outgoing link, and agreed per-packet payment, positive or negative, to the AS for this link and direction.

As a result, social welfare optimization becomes a min-cost flow problem, and incentive-compatibility can always be achieved (via VCG) in polynomial time. If there are no capacity constraints, we show (Theorem 4.1) that the resulting algorithm is BGP-friendly, essentially because the BGP-friendly version of the Bellman–Ford algorithm in [Feigenbaum et al. 02] can be extended to cover this case. When capacities are present, the algorithm becomes a more generic min-cost flow computation (Theorem 4.2), and, as we show by a counterexample, does not adhere to the criteria of BGP-friendliness (it may not converge fast enough), even though it is still a local, distributed algorithm with modest memory requirements and no need for global data.

If, on top of this, we also require that the paths be of the "valley-free" kind suggested by the kinds of agreements between ASes one sees in practice (that is, the kind of restriction that led to tractability in [Feigenbaum et al. 06a]), the resulting algorithm solves a rather generic linear program (Theorem 4.3), and so local, distributed computation appears to be impossible.

2. Basic Model and the VCG Mechanism

We model interdomain routing as a symmetric directed network with node set $V = \{0, 1, \ldots, n\}$ and edges E, where node 0 is assumed to be a fixed *sink* (the destination of all packages, assumed unique, as is common in this literature). We postulate that the network is symmetric, in that if $(i, j) \in E$ then also $(j, i) \in E$. There are no self-loops. Each node i has a *demand* of k_i packets it wants to send to the sink. In addition, each node i has a *per-packet value* $v_{i,e}$ (sometimes also denoted by $v_i(e)$) for each of its incident edges e, and a value π_i for each of its packets that gets delivered. The *cost* of an edge $e = (i, j) \in E$ is the negative of the sum of values of i and j for it, $p_e = -(v_{i,e} + v_{j,e})$.

We denote by θ_i the *type* of node i, that is, the collection of values for its incident edges and its per-packet delivery value. Denote by θ the vector of all node types and by θ_{-i} the vector of all node types except that of node i.

If F is an integer-valued flow through this network, with sink 0 and sources at all other nodes with the given demands, then the welfare of each node $i \neq 0$ from this flow is

$$w_i(F, \theta_i) = \sum_{e:\ i \in e} v_i(e) F(e) + \pi_i F_i,$$

where by $F_i = \sum_j F(i,j) - \sum_j F(j,i)$ we denote the flow out of i, assumed to be at most k_i. The total welfare of all nodes under flow F is

$$W(F) = \sum_{i \in V \setminus \{0\}} \pi_i F_i - \sum_{e \in E} p_e F(e).$$

Let $F^*(\theta)$ be the welfare-maximizing flow for types θ, and let $F^*_{-i}(\theta^{-i})$ be the welfare of the optimum flow when node i is deleted from the network. We assume initially that all capacities are infinite, which implies that the optimum flow is the union of n or fewer flow-weighted source-to-sink shortest paths; this assumption is reconsidered in Section 2.2.

2.1. VCG Mechanism

Notice that in order to compute the optimum flow we need to know the types of all players; the difficulty is, of course, that the type of player $i > 0$ is known only to player i, who is not inclined to publicize it in the absence of appropriate incentives. The *VCG mechanism* for this problem incentivizes the players to reveal their true types, and thus participate in a socially optimum flow, by making payments to them. Consider in particular the following transfers for each node (negative for payments made by the node and positive for payments received by the node):

$$\begin{aligned} t_i(\theta) &= \Big[\sum_{j \neq i} w_j(F^*(\theta), \theta_j)\Big] - \Big[\sum_{j \neq i} w_j(F^*_{-i}(\theta_{-i}), \theta_j)\Big] \\ &= \Bigg[\sum_{\substack{j \neq i \\ \pi_j \geq P_j}} k_j \cdot (\pi_j - P_j^{-i})\Bigg] - \Bigg[\sum_{\substack{j \neq i \\ \pi_j \geq P_{j,-i}}} k_j \cdot (\pi_j - P_{j,-i})\Bigg], \qquad (2.1) \end{aligned}$$

where P_j is the cost of the cheapest path from j to 0; $P_{j,-i}$ is the cost of the cheapest path from j to 0 that does not go through node i; P_j^{-i} is the cost of the cheapest path $path_j$ from j to 0 without taking costs potentially incurred by i into account: if $i \notin path_j$, then $P_j^{-i} = P_j$; otherwise, $P_j^{-i} = P_j + (v_{i,e_1} + v_{i,e_2})$, where $e_1, e_2 \in path_j$ denote the edges incident to i.

Note that nodes send their own packets only if they have a nonnegative welfare for doing so, namely if the per-packet delivery value π_j is at least as big as the path cost P_j to the destination. Thus, they send either none of their packets or all k_j of them.

The proof that these transfers lead to truthful reporting is analogous to the corresponding proof about the Groves mechanism in [Mas-Colell et al. 95] specialized to the current situation. We include it here for completeness:

Theorem 2.1. *When the nodes are selfish agents and have values for adjacent edges and for the delivery of their own packets, there is a strategy-proof pricing mechanism under which the lowest-cost paths are chosen, and the payments are of the form given in* (2.1).

Proof. Suppose truth is not a dominant strategy for some node i, i.e., the node gets higher utility by reporting a collection of values $\hat{\theta}_i$ different from its true values θ_i when the other nodes report θ_{-i}. The utility of the node is its welfare plus the payment to the node by the mechanism:

$$w_i(F^*(\hat{\theta}_i, \theta_{-i}), \theta_i) + t_i(\hat{\theta}_i, \theta_{-i}) > w_i(F^*(\theta), \theta_i) + t_i(\theta_i, \theta_{-i}).$$

Substituting the form of the transfer on both sides and canceling identical terms, we get

$$\begin{aligned} & w_i(F^*(\hat{\theta}_i, \theta_{-i}), \theta_i) + \Big[\sum_{j \neq i} w_j(F^*(\hat{\theta}_i, \theta_{-i}), \theta_j)\Big] \\ & \quad > w_i(F^*(\theta), \theta_i) + \Big[\sum_{j \neq i} w_j(F^*(\theta), \theta_j)\Big] \Leftrightarrow W(F^*(\hat{\theta}_i, \theta_{-i}), \theta) > W(F^*(\theta), \theta). \end{aligned}$$

The last inequality contradicts the fact that $F^*(\theta)$ is the welfare-maximizing choice of paths (i.e., the least-cost paths) for node types θ. □

2.2. The Model with Capacities

In this subsection we consider the same basic model, with the addition that each edge e has a corresponding capacity c_e. We would like to find a min-cost (multicommodity) flow from all nodes to the sink 0, satisfying the demands of the nodes. We can transform the problem to an equivalent one by adding a new node—a supersource, which is connected to each node j via an edge of cost $-\pi_j$ and capacity equal to the demand k_j at node j (see Figure 1).

Denote by $F^*(\theta)$ the resulting min-cost flow with known types θ, and by $F^*_{-i}(\theta_{-i})$ the min-cost flow in the graph with node i removed. We can now get a VCG mechanism similar to the one in the basic model. As before, the total welfare is

$$W(F^*(\theta), \theta) = \sum_i w_i(F^*(\theta), \theta_i),$$

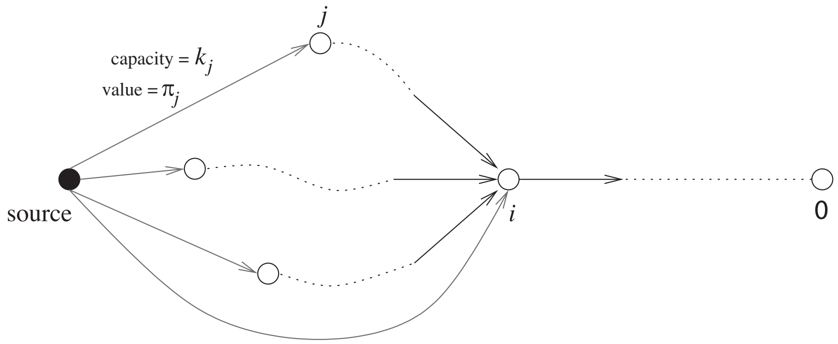

Figure 1. In a model with capacities, we add a supersource and edges from it to each node j, of capacity k_j and cost $-\pi_j$ (equivalently, value π_j). The socially optimal solution corresponds to the min-cost flow on the induced graph.

where $w_i(F^*(\theta), \theta_i)$ is the welfare of the flow from i (more precisely, from the supersource through i) to 0. Similarly, the VCG mechanism is specified by the transfers

$$t_i(\theta) = \Big[\sum_{j \neq i} w_j(F^*(\theta), \theta_j)\Big] - \Big[\sum_{j \neq i} w_j(F^*_{-i}(\theta_{-i}), \theta_j)\Big],$$

and a proof of correctness (truthfulness) of the mechanism follows as before.

3. Economic Relationships

The economic relationships between individual ASes in the Internet severely influence the paths that can be taken in the BGP graph. So far, we have assumed that all paths that are present in the underlying undirected graph (there is an edge between two ASes if they are connected by a physical link) are valid. In reality, this is not the case. Routing policies that are based on the economic relationships between ASes forbid many paths that theoretically exist. Inferring these economic relationships and investigating the resulting consequences for the connectivity of the BGP graph have attracted a large amount of scientific interest recently; see, e.g., [Achlioptas et al. 05, Barford et al. 01, Di Battista et al. 07, Erlebach et al. 04, Gao 01, Govindan and Reddy 97, Subramanian et al. 02].

Below we will give a detailed description of the valley-free path model that classifies the prohibited paths in the BGP graph.

The Valley-Free Path Model. In this model there are basically three different types of relationships in which a pair of connected ASes can find itself: *customer–provider*, in which the customer pays the provider to obtain access to the Internet; *peer–peer*, in which both peers agree on mutually routing traffic of their customers for each other free of charge; and *sibling*, in which both siblings agree on mutually routing any traffic for each other free of charge. Note that an individual AS may take several roles—as customer, provider, sibling, or peer—simultaneously; it can, for instance, be a customer of one AS and at the same time a provider for another AS.

In the following, for ease of exposition we will focus on customer–provider relationships only. The other types of relationships (peer–peer, sibling) can be incorporated easily, as we will note in Section 4.3.

We call a directed graph $G = (V, E)$ a *ToR graph* if it contains no self-loops and the edge directions describe the economic relationships. If the AS i is a customer of a provider AS j, we direct the edge between i and j toward j. This follows the terminology of [Subramanian et al. 02].

In practice, routing is done in the following way. If AS j is a provider of i (i.e., $(i, j) \in E$), it announces all its routes to i, but AS i on the other hand announces its own routes and the routes of its customers only to j. The idea behind this is that i pays j for the connection and thus is not inclined to take over "work" for j. This would happen if i also announced the routes it has from other providers. Then it would potentially have to donate bandwidth to packets that arrive from the provider j, only to proceed to another provider. This clearly is of no benefit to i itself or to any of its customers. From its point of view, j should find a different route for the corresponding packets.

This leads to the model proposed in [Subramanian et al. 02] that a path is *valid* if and only if it consists of a sequence of customer–provider edges (●⟶●) followed by a sequence of provider–customer edges (●⟵●). The first part, containing only customer–provider edges, is called the *forward part* of the path. The last part, containing only provider–customer edges, is called the *backward part* of the path. It is easy to see that the following is an equivalent definition of the validity of a path:

> A path with edges $\{e_1, e_2, \ldots, e_r\}$ in the ToR graph G is a *valid path* in G if and only if there is no inner node in the path for which its incident edges e_{i-1} and e_i are both outgoing from the node.

If such an inner node—one that has this property—exists, it is called a *valley*. The intuition behind this name is that outgoing edges point "upward," out of the valley. In the literature the situation that a path contains a valley is also called an *anomaly*. We shall call a flow that uses only valley-free paths a *valid* or *valley-free* flow.

The VCG mechanism. The transfers can be specified as in Section 2.2 for the model with capacities. The only difference is that all flows (i.e., F^* and F^*_{-i} for all i) must be valley-free (and may be fractional).

4. Distributed Computation of VCG Payments

It is of great interest to determine to what extent the payments $t_i(\theta)$ can be computed not only efficiently, but in a distributed manner that is "BGP-friendly," that is, compatible with current usage of the BGP protocol. In [Feigenbaum et al. 06b] this concept of "BGP-friendliness" was formalized as three requirements:

(1) The algorithm should converge in a number of rounds that is proportional to the diameter of the graph, and not its size.

(2) Only local data should be needed.

(3) No rippling updates should be needed as data changes.

Here we relax requirement (1) to a number of rounds that is proportional to the diameter times R, where R is the ratio between the largest and smallest edge costs. This is necessary (also in [Feigenbaum et al. 06b], where the stricter version is used by oversight) because the computed shortest path, whose length is the upper bound on convergence time, may be longer than the diameter. We do not bother to formalize here the second and third requirements (the reader is referred to [Feigenbaum et al. 06b]) because our algorithms either trivially satisfy any conceivable version, or fail to satisfy (1). As it turns out, this important aspect of BGP-friendliness sets the basic model apart from the model with capacities. In both cases the implementation is quite simple and makes only modest use of local resources. But only in the former case are the strict conditions on the convergence time fulfilled.

4.1. Basic Model

For the basic model it is easy to adapt the approach presented in [Feigenbaum et al. 02]. BGP is a *path-vector* protocol that computes the lowest-cost paths (LCPs) in a sequence of stages. In a stage each node in the network sends all the LCPs it knows of to its neighbors. It also receives LCPs from its neighbors. If these contain shorter paths than the ones it has currently stored, it updates the list of its own LCPs. This basically corresponds to a distributed computation of all shortest paths via the Bellman–Ford algorithm. The computation terminates after d stages and involves $O(nd)$ communication on any edge, where d denotes the maximum number of edges on an LCP.

Let $\operatorname{diam}'(G)$ denote the maximum diameter of $G \setminus \{i\}$ over all nodes i for which $G \setminus \{i\}$ is still connected.

Theorem 4.1. *In the basic per-packet utility model without capacity constraints (described in Section 2), the Vickrey–Clarke–Groves allocation and payments can be computed in a distributed, BGP-friendly manner. The computation converges in $O(\operatorname{diam}'(G) \cdot R)$ rounds of communication, where R is the ratio between the largest and smallest edge costs.*

Proof. Our proof follows analogously to that of [Feigenbaum et al. 02, Theorem 2]. The authors there propose an extension of the path-vector protocol that computes not only the lowest-cost paths, but at the same time the lowest-cost paths that do not traverse a given node i. These two quantities are then used to compute the payments for node i. The computation of paths avoiding node i increases the number of stages and communication needed to d' and $O(nd')$, respectively. Here d' denotes the maximum number of edges on an LCP avoiding node i, over all nodes i for which $G \setminus \{i\}$ is still connected. Feigenbaum et al. argue that this is still an acceptable convergence time.

The only difference of the approach in [Feigenbaum et al. 02] to ours is that the per-packet values in our model are given individually for each edge and node, i.e., as $v_{i,e}$, and not only as one total value per node.[1] Hence, it is easy to adapt their method to compute the values P_j and $P_{j,-i}$, for $j, i \in \{1, \ldots, n\}$, which is all we need to compute the VCG payments

$$t_i(\theta) = \Bigg[\sum_{\substack{j \neq i \\ \pi_j \geq P_j}} k_j \cdot (\pi_j - P_j^{-i}) \Bigg] - \Bigg[\sum_{\substack{j \neq i \\ \pi_j \geq P_{j,-i}}} k_j \cdot (\pi_j - P_{j,-i}) \Bigg].$$

Note that the partial path cost P_j^{-i} can be easily derived from the cost P_j of the cheapest path from j to the sink.

Since $d' \leq \operatorname{diam}'(G) \cdot R$, we get the desired running time. □

4.2. Model with Capacities

Instead of lowest-cost paths and lowest-cost paths avoiding node i, we now need to know a min-cost flow $F(\theta)$ and a min-cost flow $F_{-i}(\theta)$ avoiding node i for each of the payments $t_i(\theta)$, $i \in \{1, \ldots, n\}$. In the following we will explain how to

[1] This allows for more fine-grained and thus more realistic modeling.

compute $F(\theta)$ in a distributed fashion. The flow $F_{-i}(\theta)$ can be computed correspondingly by blocking node i. Therefore, altogether $(n+1)$ flow computations are performed, one for $F(\theta)$ and n for the $F_{-i}(\theta)$, $i \in V \setminus \{0\}$.

We assume that the sink 0 controls all the computations: it chooses which node is blocked (in the $F_{-i}(\theta)$ case), it selects paths along which to send flow together with the corresponding amounts, and it recognizes when a min-cost flow computation is finished. These all are computationally simple tasks. The only intensive computations needed will be those to obtain the shortest paths with respect to certain costs and those in which certain edges may be blocked. These will be done in a distributed manner applying the standard distributed Bellman–Ford algorithm, which is used by BGP as mentioned above.

Distributed Computation of $F(\theta)$. We start with a description of a simple Ford–Fulkerson approach [Ford and Fulkerson 58] of computing a min-cost flow from the supersource to the sink via augmenting shortest paths. Then we explain how to modify it to use the Edmonds–Karp scaling technique [Edmonds and Karp 72].

A virtual residual graph is overlayed on the given network. The residual edge capacities and costs are derived from the original graph. The residual capacities depend on the flow present on the corresponding residual edges and thus may change during the computation of the flow. Each node keeps track of flow values on residual edges incident to it.

Consider an original pair of directed edges (i,j) and (j,i) with costs $p_{(i,j)}$ and $p_{(j,i)}$. We assume the costs to be greater than or equal to 0. Let $f_{(i,j)}$ and $f_{(j,i)}$ denote the flow amounts on these edges, only one of which may be greater than 0. Otherwise, a circular flow of $\min(f_{(i,j)}, f_{(j,i)})$ is subtracted from both without increasing the costs. The residual capacities are set to $c'_{(i,j)} = c_{(i,j)} - f_{(i,j)}$ and $c'_{(j,i)} = c_{(j,i)} - f_{(j,i)}$. Additionally, we add the virtual edges $\overline{(i,j)}$ and $\overline{(j,i)}$ with capacities $c_{\overline{(i,j)}} = f_{(j,i)}$ and $c_{\overline{(j,i)}} = f_{(i,j)}$ and costs $p_{\overline{(i,j)}} = -p_{(i,j)}$ and $p_{\overline{(j,i)}} = -p_{(j,i)}$. Flow sent onto these edges is subtracted from the corresponding flow on the edge in the opposite direction. Finally, for each $i \in V \setminus \{0\}$ a virtual edge is added from the supersource to node i with cost $-\pi_i$.

The algorithm now proceeds as follows; steps 2–4 constitute a phase.

1. For each node $i \in V$ initialize the flow values $f_{(i,j)} = f_{(j,i)} = 0$ of all incident residual edges. Update the local capacities as described above.

2. Compute the shortest paths in the current residual graph considering only edges with capacities greater than 0. Do this with the distributed Bellman–Ford algorithm, adapting the BGP implementation. Modify the algorithm so that it also forwards the bottleneck capacity of each path.

3. The sink checks the min-cost path to the supersource. If the cost is greater than or equal to 0, we are done. Otherwise, send a flow corresponding to the bottleneck capacity along the path. This is done by sending a message along the path that notifies the contained nodes to update their local flow values (and thus capacities).

4. Continue at step 2 with the updated residual graph.

Theorem 4.2. *In the per-packet utility model with capacity constraints (described in Section 2.2), the Vickrey–Clarke–Groves allocation and payments can be computed in a distributed manner. The computation converges in $O(n^3 \cdot \log C)$ rounds of communication, where $C = \max\{c_e | e \in E\}$ is the maximum edge capacity.*

Proof. Each phase consists in a (re)computation of the shortest paths in step 2. Note that in the capacitated case, the rounds of communication for a shortest-paths computation is no longer bounded by d or d'. It may actually take up to n rounds of communication, as the example in the paragraph below shows.

The algorithm finishes in $O(|E| \cdot C)$ phases. This can be improved to $O(|E| \cdot \log C)$ by applying the well-known scaling technique. To this end, a variable Δ is introduced and initialized to $2^{\lceil \log C \rceil - 1}$ in step 1. In step 2, only edges with capacity at least Δ are considered. In step 3, Δ is updated to $\Delta/2$ if no more negative-cost paths are found (unless $\Delta = 1$, in which case we are done). The updated Δ is broadcast to all nodes.

As mentioned, with $(n+1)$ such flow computations we can compute all node payments $t_i(\theta)$, which yields the desired number of rounds $O(n^3 \cdot \log C)$. □

Shortest-Paths Computation. With capacities, the number of rounds of communication to compute the shortest paths can no longer be bounded by d (or d'). Figure 2 shows a counterexample in which the shortest path in the residual graph has

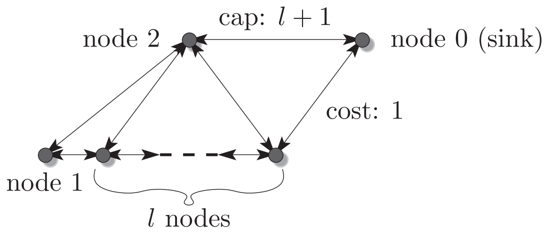

Figure 2. All edges have capacity 1, except the top edge with capacity $l + 1$. The edge costs are all 0, except the rightmost edge with cost 1. All nodes have a demand of 1 to be sent to node 0.

length $n - 2$, whereas the number of hops in the corresponding LCP in the original graph is 2. Assume that each node has already (virtually) sent its flow through the residual graph except node 1, which is selected last. Since the nodes are indistinguishable, we may assume this. The only path remaining in the residual graph is the one at the bottom of length $n-2$, since the capacities of all other edges (except $(1, 2)$) are fully saturated by flow sent to the sink via node 2. This compares to the LCP with only two edges from node 1 over node 2 directly to node 0.

4.3. Model with Economic Relationships

In the following we will explain the two-layer graph, a helpful notion that was originally suggested in [Erlebach et al. 04]. With the help of the two-layer graph it will be easy to see that one can compute min-cost valley-free flows as needed in our model with capacities introduced in Section 2.2.

The Two-Layer Model. From a ToR graph $G = (V, E)$ with source and sink $s, t \in V$ we construct a directed *two-layer graph* H in the following way (see Figure 3 for an example): Two copies of the graph G are made, called the *lower* and the *upper layers*. In the upper layer, all edge-directions are reversed. In addition, every node i in the lower layer is connected by an edge to the corresponding copy of i, denoted by i', in the upper layer. The edge is directed from i to i'. Finally, we obtain the two-layer graph H by merging the two s-nodes (of the lower and upper layers) and also the two t-nodes, and by removing the incoming edges of s and the outgoing edges of t.

A valid path $path_G = \{i_1 \dots i_r\}$ in G with $i_1 = s$ and $i_r = t$ is equivalent to a directed path in H in the following way. The forward part of $path_G$, which is the part containing all edges $(i_q, i_{q+1}) \in path_G$, is routed in the lower layer. Then there is a possible switch to the upper layer with a (i, i')-type edge (there can be at most one such switch for each path). The backward part of $path_G$ is routed in the upper layer. In other words, for each original edge $(i_{q+1}, i_q) \in path_G$, the corresponding edge (i'_q, i'_{q+1}) of the upper layer is traversed. If there is only a forward (respectively backward) part of $path_G$, then the corresponding path in H is in only the lower (respectively upper) layer.

This definition of the two-layer graph can easily be extended to the case of multiple sources. Note that a peer–peer relationship between two nodes $i, j \in V$ can be incorporated by adding the edges (i, j') and (j, i') from the lower to the upper layer (reflecting that at most one peer–peer edge is allowed between the forward and the backward parts of a path). Similarly, a sibling relationship between two nodes $i, j \in V$ can be incorporated by adding the symmetric edges

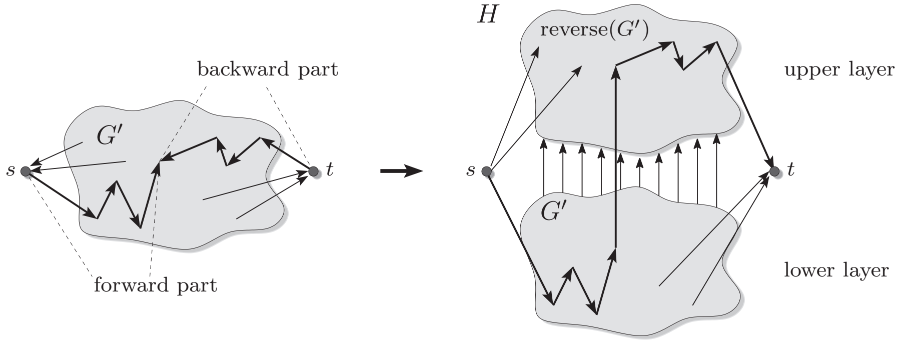

Figure 3. A path in the ToR graph G and the corresponding path in the two-layer graph H. (G' is G, excluding s and t.)

(i, j), (j, i), (i', j'), and (j', i') in both layers (reflecting that sibling edges are allowed at arbitrary points in a path).

Min-Cost Valley-Free Flows. By simply computing a min-cost flow in the two-layer graph it is easy to derive a valley-free flow that will have at most the cost of an optimum min-cost valley-free flow. The edge capacities may be violated by at most a factor of two, though, since each edge may be used twice: once in the upper and once in the lower layer. Note that such a min-cost flow could be computed in a distributed fashion by slightly modifying the approach described in Section 4.2. Unfortunately, this approximate solution cannot be used to compute the VCG allocation and payments, because the truthfulness of the mechanism holds only for the exact form of payments from the optimal solution. Thus, we need to compute the optimal solution.

Theorem 4.3. *In the per-packet utility model with capacity constraints and economic relationships (see Section 3), the Vickrey–Clarke–Groves allocation and payments can be computed in polynomial time with an approach based on linear programming.*

Proof. For a given graph, consider its corresponding two-layer graph described above. Then the exact allocation and payments in the original graph can be computed with the help of a standard linear programming flow formulation on the two-layer graph, with added constraints to bound the joint flow on the edges of the upper and lower layers. In particular, for each edge $(i, j) \in E$ in the original ToR graph, we add a joint capacity constraint for (i, j) and (j', i') in the two-layer graph. □

Note that the existence of an exact algorithm based on augmenting paths seems unlikely. Usually, for integral capacities such algorithms aim at computing an optimal integral solution, i.e., for unit capacities a solution would consist of edge-disjoint paths. However, computing the maximum number of disjoint valley-free paths between two nodes s and t is inapproximable within a factor of $(2 - \varepsilon)$, unless P = NP [Erlebach et al. 04].

5. Conclusions and Open Problems

Despite the fact that incentive compatibility for BGP routing has been known to be problematic in general, as well as for several apparently realistic special cases, we have identified one important special case of practical importance, namely the one in which path utilities depend on local per-packet costs as well as delivery values. In this case, incentive compatibility is achievable through payments that can be computed efficiently and in a BGP-compatible way; adding capacities and the "valley-free" constraint for paths makes incentives harder to compute in a BGP-compatible way, but still tractable.

Regarding the latter point, in this work we have simply pointed out that the algorithms we devised for VCG incentive computation are not implementable in a BGP-compatible way; it would be interesting to actually prove that this is inherent to the problem, that is, to prove a lower bound on the convergence time of any algorithm solving the min-cost flow problem and its valley-free constrained case.

Our model for path utilities is suggestive of a more general project for understanding BGP routing: We postulate that each directed edge in and out of every node has a value for this node, depending on the cost to this node, as well as agreed-upon payments to or from its neighbors, for each packet sent or received along this edge. Suppose that the graph, as well as the demand and per-packet cost and delivery value of each node, is given. A game is thus defined in which strategies are payment agreements between neighbors, and the utility to each node is that obtained by our model of BGP min-cost routing. This game is thus a very realistic network-creation game, with special emphasis on BGP routing. The quality of equilibria compared to the social optimum (i.e., the price of anarchy and its variants) for this game would be a most interesting research direction. The social optimum is, of course, the min-cost flow with only costs and delivery values taken into account. Further, such a model would allow one to study how inter-AS agreements can depend on the underlying fundamentals of each AS, such as costs, delivery value, demand, and position in the network.

References

[Achlioptas et al. 05] D. Achlioptas, A. Clauset, D. Kempe, and C. Moore. "On the Bias of Traceroute Sampling; or, Power-Law Degree Distributions in Regular Graphs." In *Proceedings of the Thirty-Seventh Annual ACM Symposium on Theory of Computing*, pp. 694–703. New York: ACM Press, 2005.

[Anshelevich et al. 06] E. Anshelevich, B. Shepherd, and G. Wilfong. "Strategic Network Formation through Peering and Service Agreements." In *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science*, pp. 77–86. Washington, DC: IEEE Computer Society, 2006.

[Barford et al. 01] P. Barford, A. Bestavros, J. Byers, and M. Crovella. "On the Marginal Utility of Deploying Measurement Infrastructure." In *Proceedngs of the 1st ACM SIGCOMM Workshop on Internet Measurement*, pp. 5–17. New York: ACM Press, 2001.

[Di Battista et al. 07] G. Di Battista, T. Erlebach, A. Hall, M. Patrignani, M. Pizzonia, and T. Schank. "Computing the Types of the Relationships between Autonomous Systems." *IEEE/ACM Transactions on Networking* 15:2 (2007), 267–280.

[Edmonds and Karp 72] J. Edmonds and R. M. Karp. "Theoretical Improvements in Algorithmic Efficiency for Network Flow Problems." *J. ACM* 19:2 (1972), 248–264.

[Erlebach et al. 04] T. Erlebach, A. Hall, A. Panconesi, and D. Vukadinovic. "Cuts and Disjoint Paths in the Valley-Free Path Model of Internet BGP Routing." In *Combinatorial and Algorithmic Aspects of Networking: First Workshop on Combinatorial and Algorithmic Aspects of Networking, CAAN 2004, Banff, Alberta, Canada, August 5–7, 2004, Revised Selected Papers*, Lecture Notes in Computer Science 3405, pp. 49–62. Berlin: Springer, 2004.

[Fabrikant and Papadimitriou 07] A. Fabrikant and C. H. Papadimitriou. "The Search for Equilibria: Sink Equilibria, Unit Recall Games, and BGP Oscillations." Preprint, 2007.

[Fabrikant et al. 03] A. Fabrikant, A. Luthra, E. Maneva, C. H. Papadimitriou, and S. Shenker. "On a Network Creation Game." In *Proceedings of the Twenty-Second Annual Symposium on Principles of Distributed Computing*, pp. 347–351. New York: ACM Press, 2003.

[Feamster 04] N. Feamster, J. Winick, and J. Rexford. "A Model of BGP Routing for Network Engineering." In *Proceedings of the Joint International Conference on Measurement and Modeling of Computer Systems*, pp. 331–342. New York: ACM Press, 2004.

[Feigenbaum et al. 02] J. Feigenbaum, C. H. Papadimitriou, R. Sami, and S. Shenker. "A BGP-Based Mechanism for Lowest-Cost Routing." In *Proceedings of the Twenty-First Annual Symposium on Principles of Distributed Computing*, pp. 173–182. New York: ACM Press, 2002.

[Feigenbaum et al. 06a] J. Feigenbaum, V. Ramachandran, and M. Schapira. "Incentive-Compatible Interdomain Routing." In *Proceedings of the 7th ACM Conference on Electronic Commerce*, pp. 130–139. New York: ACM Press, 2006.

[Feigenbaum et al. 06b] J. Feigenbaum, R. Sami, and S. Shenker. "Mechanism Design for Policy Routing." *Distrib. Comput.* 18:4 (2006), 293–305.

[Ford and Fulkerson 58] L. R. Ford and D. R. Fulkerson. "Constructing Maximal Dynamic Flows from Static Flows." *Operations Research* 6 (1958), 419–433.

[Gao 01] L. Gao. "On Inferring Autonomous System Relationships in the Internet." *IEEE/ACM Trans. Networking* 9:6 (2001), 733–745.

[Govindan and Reddy 97] R. Govindan and A. Reddy. "An Analysis of Internet Interdomain Topology and Route Stability." In *Proceedings of the Sixteenth Annual Joint Conference of the IEEE Computer and Communications Societies*, pp. 850–857. Los Alamitos, CA: IEEE Press, 1997.

[Griffin and Wilfong 99] T. G. Griffin and G. Wilfong. "An Analysis of BGP Convergence Properties." In *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pp. 277–288. New York: ACM Press, 1999.

[Mas-Colell et al. 95] A. Mas-Colell, M. D. Whinston, and J. R. Green. *Microeconomic Theory.* New York: Oxford University Press, 1995.

[Nisan and Ronen 99] N. Nisan and A. Ronen. "Algorithmic Mechanism Design." In *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing*, pp. 129–140. New York: ACM Press, 1999.

[Papadimitriou 01] C. H. Papadimitriou. "Algorithms, Games, and the Internet." In *Proceedings of the Thirty-Third Annual ACM Symposium on Theory of Computing*, pp. 749–753. New York: ACM Press, 2001.

[Rekhter and Li 95] Y. Rekhter and T. Li. "A Border Gateway Protocol." RFC 1771, March 1995. Available at http://www.ietf.org/rfc/rfc1771.txt.

[Stewart 98] J. W. Stewart. *BGP4: Inter-Domain Routing in the Internet*. Reading, MA: Addison-Wesley, 1998.

[Subramanian et al. 02] L. Subramanian, S. Agarwal, J. Rexford, and R. Katz. "Characterizing the Internet Hierarchy from Multiple Vantage Points." In *Proceedings of the Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies*, Vol. 2, pp. 618–627. Los Alamitos, CA: IEEE Press, 2002.

Alexander Hall, Google Switzerland GmbH, Brandschenkestrasse 110, CH-8002 Zurich, Switzerland (alex.hall@gmail.com)

Evdokia Nikolova, Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 32 Vassar Street, Cambridge, MA 02139
(nikolova@mit.edu)

Christos Papadimitriou, Department of Electrical Engineering and Computer Science, University of California, Berkeley, Soda Hall 689, Berkeley, CA 94720
(christos@cs.berkeley.edu)

Received March 30, 2008; accepted January 30, 2009.