# Routing in Equilibrium

João Luís Sobrinho
Instituto de Telecomunicações
Instituto Superior Técnico, Lisbon, Portugal
joao.sobrinho@lx.it.pt

Timothy G. Griffin
Computer Laboratory
University of Cambridge, UK
timothy.griffin@cl.cam.ac.uk

***Abstract*— Some path problems cannot be modeled using semirings because the associated algebraic structure is not distributive. Rather than attempting to compute *globally optimal* paths with such structures, it may be sufficient in some cases to find *locally optimal* paths—paths that represent a stable local equilibrium. For example, this is the type of routing system that has evolved to connect Internet Service Providers (ISPs) where link weights implement bilateral commercial relationships between them. Previous work has shown that routing equilibria can be computed for some non-distributive algebras using algorithms in the Bellman-Ford family. However, no polynomial time bound was known for such algorithms. In this paper, we show that routing equilibria can be computed using Dijkstra's algorithm for one class of non-distributive structures. This provides the first polynomial time algorithm for computing locally optimal solutions to path problems. We discuss possible applications to Internet routing.**

## I. Global vs. Local Optima

A great deal of research has followed from the observation—first made about 40 years ago—that the linear algebra structure $(\mathbb{R}, +, \times, 0, 1)$ and many of its associated algorithms can be generalized to a very large class of algebraic structures called *semirings* (see [6], [1] for modern surveys of this area). These structures have the form

$$(S, \oplus, \otimes, \overline{0}, \overline{1}),$$

where the crucial property is distributivity of $\otimes$ over $\oplus$. A particularly interesting sub-case occurs when the operation $\oplus$ is *selective*, as this relates to the kinds of problems one encounters in routing in communications networks. For instance, the structure $(\mathbb{R} \cup \{\infty\}, \min, +, \infty, 0)$ is associated with distances and shortest-paths in graphs.

A communication network $G = (V, E, a)$ is represented by an adjacency matrix $\mathbf{A}$ where $\mathbf{A}[i, j] = a(i, j)$ is the *weight* of link $(i, j) \in E$: $\mathbf{A}[i, j] = \overline{0}$ if there is no link from $i$ to $j$. The matrix $\mathbf{A}^*$ of optimal weights is defined as

$$\mathbf{A}^*[i, j] = \bigoplus_{P \in \mathcal{P}(i,j)} a(P), \tag{1}$$

where $\mathcal{P}(i, j)$ represents the set of all paths from node $i$ to node $j$ in network $G$ and $a(P)$ is the weight of path $P$, that is, the $\otimes$-multiplication of all link weights along path $P$. Since $P$ ranges over *all paths* from $i$ to $j$, we refer to this as a *globally optimal solution*.

With semirings, when $\mathbf{A}^*$ exists it is a solution for $\mathbf{L}$ in the left matrix equation

$$\mathbf{L} = \mathbf{A}\mathbf{L} \oplus \mathbf{I}, \tag{2}$$

with $\mathbf{I}$ the identity matrix, as well as the solution for $\mathbf{R}$ in the right matrix equation,

$$\mathbf{R} = \mathbf{R}\mathbf{A} \oplus \mathbf{I}. \tag{3}$$

Section III provides a very brief introduction to semirings and presents a sufficient condition for the existence of $\mathbf{A}^*$.

Removing distributivity from the semiring axioms leaves a truly impoverished algebra. Equations (2) and (3) may still have solutions, but they may be different from each other and from $\mathbf{A}^*$. Furthermore, solutions to Equations (2) and (3) do not correspond to globally optimal path weights. We now talk of *locally optimal paths*, and call solutions to (2) *left-local solutions* and solutions to (3) *right-local solutions*. These solutions represent two types of *routing in equilibria*.

Indeed, routing in the Internet today is not always based on finding globally optimal paths. One popular form of intra-domain routing relies on a composite metric which is not distributive [23], [7]. In the inter-domain context, routing policies reflect the bilateral commercial agreements between Internet Service Providers as well as local traffic engineering rules. In general, it is not possible to model such policies with distributive algebras [21]. Previous work has shown that routing equilibria can be computed for some non-distributive algebras using algorithms in the Bellman-Ford family. However, no polynomial time bound was known for such algorithms (this routing background is discussed further in Section II).

In this paper, we provide sufficient conditions which guarantee that local optima can be computed in polynomial time for structures we call *prebimonoids* and *bimonoids* (Section IV). The basic algorithm used is Dijkstra's algorithm, which can solve for one column of (2), or one row of (3), at a time. Put another way, this popular greedy algorithm can actually find *local optima* when distributivity does not hold.

However, there is a problem if we want to use Dijkstra's algorithm for Internet routing with prebimonoids and bimonoids. Suppose that for left- and right-local solutions we interpret $\mathbf{L}[i, j]$ and $\mathbf{R}[i, j]$ as the weights of the left- and right-locally optimal paths that are to carry traffic from node $i$ to node $j$. Then the left-local solution is entirely consistent with the destination-based, hop-by-hop forwarding paradigm used in most of the Internet today. However, right-local solutions can result in forwarding loops if paths are implemented with destination-based, hop-by-hop forwarding (an example is provided in Section IV).

The implications for distributed routing algorithms are clear. We can use Dijkstra's efficient algorithm at each node $i$ to compute $\mathbf{R}(i, \_)$, at the cost of abandoning destination-based, hop-by-hop forwarding. Or we can maintain destination-based, hop-by-hop forwarding and construct all matrix $\mathbf{L}$ at each node $i$ at the cost of $|V|$ invocations of Dijkstra's algorithm. This and other applications to network routing are discussed in Section V.

## II. Related Work and Motivation

In [26] and in [7], Sobrinho, and Gouda and Schneider, respectively, introduced algebraic concepts to investigate the behavior of intra-domain routing protocols. Those works emphasize the role of algebraic distributivity in the distributed construction of Quality-of-Service (QoS) paths and in the subsequent hop-by-hop forwarding of packets along them, with [26] presenting a proof of correctness for link-state routing protocols.

The need for an ever broader understanding of routing in the Internet arose from inter-domain routing, which is today implemented with the Border Gateway Protocol (BGP) [27], [14]. The computational mechanism underlying BGP is called *path vectoring*—a variant of the distributed Bellman-Ford algorithm. However, in the inter-domain setting the gamut of routing policies that Internet Service Providers (ISPs) can apply and realize through BGP can lead to unwanted behaviors, such as protocol oscillations—as illustrated by Varadhan et al. [28]—and forwarding loops. Gao and Rexford [5] have shown that if ISPs use simple policies that reflect typical commercial relationships between ISPs [15], [16], then the BGP system is guaranteed to operate correctly. However, the model does not entirely capture the complexities of the Internet's evolving commercial relationships nor the complications associated with traffic engineering policies. In practice, many routing anomalies can arise due to unanticipated interactions of routing policies [10].

Griffin et al. [11] developed a generic model to understand and predict the effect of inter-domain routing policies on the behavior of BGP. The model is graph-theoretic, based on a path ranking function at each node and a notion of a stable solution. They called this the Stable Paths Problem [11], and they presented a sufficient condition for its solution. The concept of stable solution is a Nash equilibria, which can also be modeled as a left-local solution [9].

Sobrinho [21] extended his previous work on algebras for routing to path vectoring protocols, coming up with a sufficient condition on the cycles of the network that guarantees their correctness. The sufficient condition dispenses with distributivity implying that the resulting paths are only locally optimal. The proof of termination of vectoring protocols in [21] is grounded on concepts of temporal logic [17], [18].

These works gave rise to reverse-engineering of Internet routing protocols in an attempt to further uncover algebraic constructions that could be used to model existing protocols [13], [12]. In addition, efforts were made to connect the basic ideas used in the proofs of [11] and [21] with the matrix-oriented proofs of classical algebraic path problems [2], [9].

From an abstract point of view, much of this work involved showing that if non-distributive algebras are used in a constrained manner, then left-local solutions (2) can be found using algorithms in the Bellman-Ford family. However, for non-distributive algebras, no polynomial time bound was known for the convergence of such algorithms.

## III. Semirings: a Review

A semiring is a structure of the form

$$(S, \oplus, \otimes, \overline{0}, \overline{1}),$$

where $(S, \oplus, \overline{0})$ is a commutative monoid, $(S, \otimes, \overline{1})$ is a monoid, $\overline{0}$ is an annihilator for $\otimes$,

$$\forall_{a \in S} \quad a \otimes \overline{0} = \overline{0} \otimes a = \overline{0},$$

$\otimes$ left-distributes over $\oplus$[1],

$$\forall_{a,b,c \in S} \quad a \otimes (b \oplus c) = (a \otimes b) \oplus (a \otimes c),$$

and right-distributes over $\oplus$,

$$\forall_{a,b,c \in S} \quad (b \oplus c) \otimes a = (b \otimes a) \oplus (c \otimes a).$$

We will restrict $\oplus$ to be *idempotent*:

$$\forall_{a \in S} \ a = a \oplus a.$$

The algebra of square matrices whose elements are taken from an idempotent semiring, with the usual definitions of matrix $\oplus$-addition and matrix $\otimes$-multiplication [6], is and idempotent semiring as well.

In an idempotent semiring, the *canonical order* $\preceq$ is defined by $a \preceq b$ if $a \oplus b = b$. Since $\overline{0}$ is the additive identity, we have $\overline{0} \preceq a$ for all $a \in S$. Left-distributivity of $\otimes$ over $\oplus$ is equivalent to *left-isotonicity* of $\otimes$ for $\preceq$,

$$\forall_{a,b,c \in S} \quad b \preceq c \Rightarrow a \otimes b \preceq a \otimes c,$$

and right-distributivity of $\otimes$ over $\oplus$ is equivalent to *right-isotonicity* of $\otimes$ for $\preceq$,

$$\forall_{a,b,c \in S} \quad b \preceq c \Rightarrow b \otimes a \preceq c \otimes a.$$

A condition stronger than idempotency is *selectivity*,

$$\forall_{a,b \in S} \quad a \oplus b = a \vee a \oplus b = b,$$

which implies that the canonical order is total.

Perhaps the most familiar example is the selective semiring used for the classical shortest paths problem,

$$\mathrm{sp} = (\mathbb{R}^{\infty}, \min, +, \infty, 0),$$

where $\mathbb{R}^{\infty} = \mathbb{R} \cup \{\infty\}$. Another familiar semiring is used for finding widest paths,

$$\mathrm{bw} = (\mathbb{R}^{\infty}, \max, \min, 0, \infty),$$

which is often called a bottleneck algebra.

[1] Some authors would choose the term right when we chose left, and vice-versa.

Let $G = (V, E, a)$ be a network: $(V, E)$ is a directed graph and $a$ is a map $E \to S$. We represent a network $G$ by its adjacency matrix $\mathbf{A}$:

$$\mathbf{A}[i, j] = \begin{cases} a(i, j), & \text{if } (i, j) \in E; \\ \overline{0}, & \text{otherwise.} \end{cases}$$

A path $P = v_1 v_2 \cdots v_k v_{k+1}$ of length $k$ is a sequence of nodes such that $(v_m, v_{m+1}) \in E$ for each $m$, $1 \leq m \leq k$. The weight of path $P$ is

$$a(P) = a(v_1, v_2) \otimes a(v_2, v_3) \otimes \cdots \otimes a(v_k, v_{k+1}).$$

The empty path is given weight $\overline{1}$. Let $\mathcal{P}(i, j)$ be the set of all paths from $i$ to $j$ in $G$. An *optimal* path $P$ from $i$ to $j$, if it exists, is a path from $i$ to $j$ with $\preceq$-maximum weight:

$$a(P) = \bigoplus_{Q \in \mathcal{P}(i,j)} a(Q).$$

Denote by $\mathbf{A}^*$ the matrix the $(i, j)$ entry of which is the weight of an optimal path from $i$ to $j$.

A path is simple if it does not repeat a node. A circuit $C = v_1, v_2, \cdots v_k, v_1$ is a path that starts and ends at the same node and does not repeat any of the other nodes. If every circuit $C$ in the network is such that $a(C) \preceq \overline{1}$, then $\mathbf{A}^*$ exists and it is a solution for $\mathbf{L}$ in the left matrix equation

$$\mathbf{L} = \mathbf{A}\mathbf{L} \oplus \mathbf{I},$$

as well as the solution for $\mathbf{R}$ in the right matrix equation,

$$\mathbf{R} = \mathbf{R}\mathbf{A} \oplus \mathbf{I}.$$

Moreover, $\mathbf{A}^*$ can be computed with Bellman-Ford algorithm at polynomial time complexity. Under the stronger condition on the entries of $\mathbf{A}$,

$$\forall_{(i,j) \in E} \quad a(i, j) \preceq \overline{1}, \tag{4}$$

which implies that $a(C) \preceq \overline{1}$ for all circuits $C$, Dijkstra's algorithm also computes $\mathbf{A}^*$ [6].

## IV. Prebimonoids and Bimonoids: Primitive Structures

### A. Definition

A *prebimonoid* is an algebraic structure of the form

$$(S, \oplus, \otimes, \overline{0}, \overline{1}),$$

where $(S, \oplus, \overline{0})$ is a commutative monoid, $\overline{1}$ is an identity for $\otimes$, and $\overline{0}$ is an annihilator for $\otimes$. In a prebimonoid, we dispense with both the associativity of $\otimes$ and the distributivity of $\otimes$ over $\oplus$. Dispensing with associativity of $\otimes$ has two advantages. First, important routing problems can easily be modeled with prebimonoids (see Section IV-B). Second, the prebimonoids properties carry over from elements to matrices: the algebra of square matrices whose elements are taken from a prebimonoid is a prebimonoid as well. That would not be so if associativity of $\otimes$ were required in the definition of prebimonoid. (Indeed, associativity of matrix multiplication requires distributivity of multiplication over addition of the elements of the matrices.) A prebimonoid is a *bimonoid* if $\otimes$ is associative.

A prebimonoid is idempotent if $\oplus$ is idempotent and it is selective if $\oplus$ is selective. The canonical order is defined as is semirings. Since associativity of $\otimes$ is not required in prebimonoid, the sequence of successive $\otimes$-multiplications is relevant in the definition of weight of a path. The *left-weight* of path $P = v_1 v_2 \cdots v_k v_{k+1}$ is defined as

$$a_L(P) = a(v_1, v_2) \otimes (a(v_2, v_3) \otimes (\cdots \otimes a(v_k, v_{k+1}) \cdots)),$$

that is, the $\otimes$-multiplications are performed from right to left. A *left-optimal path* from $i$ to $j$, if it exists, is a path from $i$ to $j$ with $\preceq$-maximum left-weight,

$$\mathbf{A}_L^*[i, j] = \bigoplus_{P \in \mathcal{P}(i,j)} a_L(P). \tag{5}$$

Cognate definitions can be made for right-weights $a_L(P)$ and right-optimal paths $\mathbf{A}_R^*$. In a bimonoid, there is no distinction between left- and right-weights and left- and right-optimal paths. In this case, the matrix of optimal path weights is denoted simply by $\mathbf{A}^*$, as in semirings.

Let matrix $\mathbf{L}$ satisfy the left matrix equation

$$\mathbf{L} = \mathbf{A}\mathbf{L} \oplus \mathbf{I}.$$

Such a matrix is called a *left-local solution*. Because a prebimonoid does not require left-distributivity, the $(i, j)$ entry of $\mathbf{L}$ is not necessarily the left-weight of a left-optimal path from $i$ to $j$—$\mathbf{L}$ is not necessarily equal to $\mathbf{L}^*$. We define a *left-locally optimal* path from $v_1$ to $v_{k+1}$ to be a path $P = v_1 v_2 \cdots v_k v_{k+1}$ such that

$$\mathbf{L}[v_m, v_{k+1}] = a(v_m, v_{m+1}) \otimes \mathbf{L}[v_{m+1}, v_{k+1}],$$

for all $m$, $1 \leq m \leq k$. Again, cognate definitions can be made for right-local solution and right-locally optimal paths.

### B. Customer-provider, peer-peer: no associativity

The Internet consists of a large number of ISPs with established commercial relationships between them. A highly simplified model classifies the relationships into either customer-provider or peer-peer. In a customer-provider relationship, the customer pays to the provider for access to the Internet whereas in a peer-peer relationship, the peers agree to exchange traffic between themselves and their customers free of charge. The routing-related rules governing these relationships were laid out in [5]: to a customer, an ISP exports all routes; to a provider or a peer, an ISP only exports routes learned from customers; routes learned from customers are preferred to routes learned from peers, and the latter are preferred to routes learned from providers. This routing paradigm was modeled algebraically in [21]. In the current formulation, the elements of the selective prebimonoid are $\{\overline{1}, c, r, p, \overline{0}\}$, where $c$, $r$, and $p$, stand, respectively, for customer route and customer link, peer route and peer link, and provider route and provider link. The prebimonoid operations $\oplus$ and $\otimes$ are given, respectively, in the next two charts (see [1]).

| $\oplus$ | $\overline{1}$ | $c$ | $r$ | $p$ | $\overline{0}$ |
|---|---|---|---|---|---|
| $\overline{1}$ | $\overline{1}$ | $\overline{1}$ | $\overline{1}$ | $\overline{1}$ | $\overline{1}$ |
| $c$ | $\overline{1}$ | $c$ | $c$ | $c$ | $c$ |
| $r$ | $\overline{1}$ | $c$ | $r$ | $r$ | $r$ |
| $p$ | $\overline{1}$ | $c$ | $r$ | $p$ | $p$ |
| $\overline{0}$ | $\overline{1}$ | $c$ | $r$ | $p$ | $\overline{0}$ |

| $\otimes$ | $\overline{1}$ | $c$ | $r$ | $p$ | $\overline{0}$ |
|---|---|---|---|---|---|
| $\overline{1}$ | $\overline{1}$ | $c$ | $r$ | $p$ | $\overline{0}$ |
| $c$ | $c$ | $c$ | $\overline{0}$ | $\overline{0}$ | $\overline{0}$ |
| $r$ | $r$ | $r$ | $\overline{0}$ | $\overline{0}$ | $\overline{0}$ |
| $p$ | $p$ | $p$ | $p$ | $p$ | $\overline{0}$ |
| $\overline{0}$ | $\overline{0}$ | $\overline{0}$ | $\overline{0}$ | $\overline{0}$ | $\overline{0}$ |

For example, $c \otimes p = \overline{0}$ means that an ISP does not export to a provider a route learned from another provider; $r \otimes c = r$ means that an ISP exports to a peer a route learned from a customer, there becoming a peer route.

The prebimonoid is not associative since, for example,

$$p \otimes (r \otimes p) = p \otimes \overline{0} = \overline{0} \neq p = p \otimes p = (p \otimes r) \otimes p.$$

The prebimonoid is also not right-distributive since, for example,

$$(c \oplus p) \otimes r = c \otimes r = \overline{0} \neq p = \overline{0} \oplus p = (c \otimes r) \oplus (p \otimes r).$$

However, the prebimonoid is left-distributive. From the results in [21], we can conclude that $\mathbf{A}_L^*$ exists and, because of left-distributivity, it is a left-local solution. In a similar way, if a prebimonoid is right-distributive, then $\mathbf{A}_R^*$ is a right-local solution.

Although we do not explore the details here, it is fairly straightforward to extend this analysis to capture arc weights that are a function of path weights, which is a common feature of Internet routing. For example, assume $\Lambda$ be a set of indices for a set of function $F = \{f_\lambda \in S \to S \mid \lambda \in \Lambda\}$. We then define the left-application function $\rhd \in \Lambda \to (S \to S)$ as

$$\lambda \rhd s = f_\lambda(s), \tag{6}$$

and the left-weight of a path as

$$\begin{aligned} & a_L(P) \\ = \quad & a(v_1, v_2) \rhd (a(v_2, v_3) \rhd \cdots \rhd (a(v_k, v_{k+1}) \rhd \overline{1} \cdots)) \\ = \quad & f_{a(v_1,v_2)}(f_{a(v_2,v_3)}(\cdots f_{a(v_k,v_{k+1})}(\overline{1}) \cdots)). \end{aligned}$$

It is then easy to extend Dijkstra's algorithm to handle such algebras, which have been used to model BGP-like routing [21], [8].

### *C. Widest-shortest paths: no distributivity*

We now present an example of a bimonoid where $\otimes$ distributes neither to the left nor to the right over $\oplus$. From the shortest-path algebra and the widest-path algebra, we can compose two product algebras: the shortest-widest path algebra and the widest-shortest path algebra. Whereas the first is distributive the second is not [26]. The widest-shortest path algebra is the algebraic structure

$$\text{bw} \vec{\times} \text{sp} = (\mathbb{R}^\infty \times \mathbb{R}^\infty, \oplus, \otimes, (0, \infty), (\infty, 0)),$$

where

$$(b_1, d_1) \oplus (b_2, d_2) = \begin{cases} (b_1, \min\{d_1, d_2\}), & \text{if } b_1 = b_2, \\ (b_1, d_1), & \text{if } b_1 > b_2, \\ (b_2, d_2), & \text{if } b_1 < b_2, \end{cases}$$

and

$$(b_1, d_1) \otimes (b_2, d_2) = (\min\{b_1, b_2\}, d_1 + d_2).$$

Figure 1 (a) presents an example network over bw $\vec{\times}$ sp. Its adjacency matrix $\mathbf{A}$ is

$$\mathbf{A} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \left[ \begin{array}{ccccc} (0,\infty) & (5,1) & (0,\infty) & (0,\infty) & (0,\infty) \\ (0,\infty) & (0,\infty) & (0,\infty) & (0,\infty) & (0,\infty) \\ (0,\infty) & (5,4) & (0,\infty) & (5,1) & (0,\infty) \\ (5,1) & (0,\infty) & (0,\infty) & (0,\infty) & (10,1) \\ (10,5) & (0,\infty) & (5,1) & (0,\infty) & (0,\infty) \end{array} \right] \end{array}$$

The matrix of optimal weights, the left-local solution, and the right-local solution are, respectively,

$$\mathbf{A}^* = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \left[ \begin{array}{ccccc} (\infty,0) & (5,1) & (0,\infty) & (0,\infty) & (0,\infty) \\ (0,\infty) & (\infty,0) & (0,\infty) & (0,\infty) & (0,\infty) \\ (5,2) & (5,3) & (\infty,\ 0) & (5,1) & (5,2) \\ (10,6) & (5,2) & (5,2) & (\infty,0) & (10,1) \\ (10,5) & (5,4) & (5,1) & (5,2) & (\infty,0) \end{array} \right] \end{array},$$

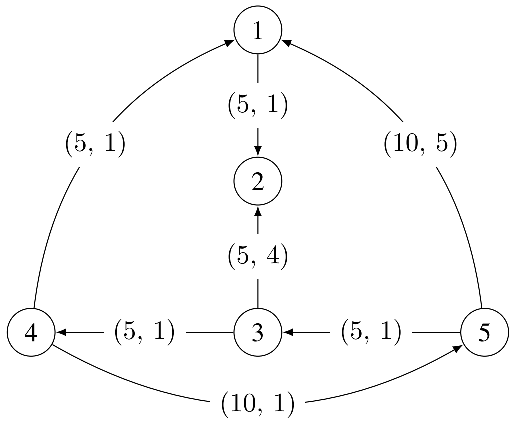
$$\mathbf{L} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \left[ \begin{array}{ccccc} (\infty,0) & (5,1) & (0,\infty) & (0,\infty) & (0,\infty) \\ (0,\infty) & (\infty,0) & (0,\infty) & (0,\infty) & (0,\infty) \\ (\mathbf{5},\mathbf{7}) & (5,3) & (\infty,0) & (5,1) & (5,2) \\ (10,6) & (5,2) & (5,2) & (\infty,0) & (10,1) \\ (10,5) & (5,4) & (5,1) & (5,2) & (\infty,0) \end{array} \right] \end{array},$$

and

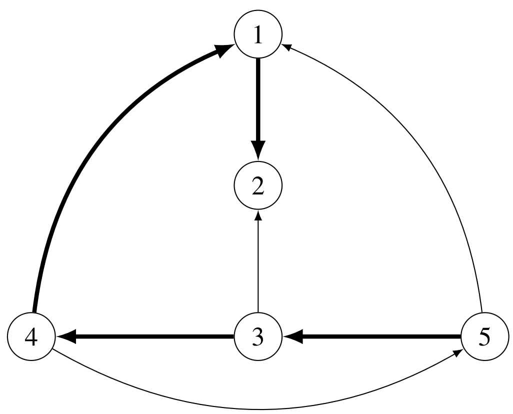
$$\mathbf{R} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \left[ \begin{array}{ccccc} (\infty,0) & (5,1) & (0,\infty) & (0,\infty) & (0,\infty) \\ (0,\infty) & (\infty,0) & (0,\infty) & (0,\infty) & (0,\infty) \\ (5,2) & (5,3) & (\infty,\ 0) & (5,\ 1) & (5,2) \\ (10,6) & (\mathbf{5},\mathbf{7}) & (5,2) & (\infty,0) & (10,1) \\ (10,5) & (\mathbf{5},\mathbf{5}) & (5,1) & (5,2) & (\infty,0) \end{array} \right] \end{array},$$

where the entries marked in bold indicate those values which are not optimal weights.

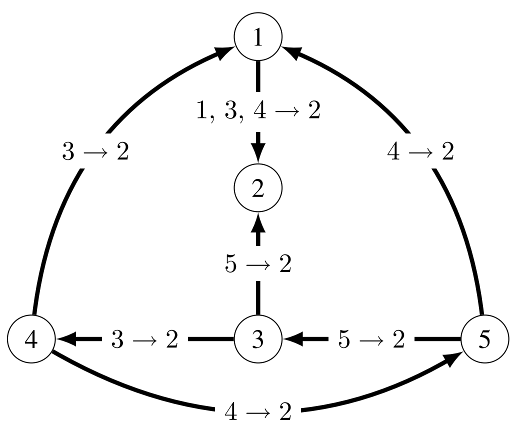
Figure 1 (b) illustrates the left-locally optimal paths to node 2. Note that the left-local solution is compatible with the kind of destination-based, hop-by-hop forwarding used in the Internet. On the other hand, Figure 1 (c) shows right-locally optimal paths to node 2. (The annotations on arcs indicate which arcs are used by which source nodes to reach destination 2.) Observe that the union of all right-locally optimal paths to node 2 is not a tree rooted at 2, and not even an acyclic graph rooted at 2. This means that right-local solutions are not compatible with destination-based, hop-by-hop forwarding: a link-state routing protocol does not operate correctly in this situation. Column 2 of matrix $\mathbf{R}$ gives the width-length of the (right-locally optimal) paths computed by the various nodes to reach destination 2. Node




(a) A weighted graph.




(b) The left-locally optimal paths to node 2.




(c) The right-locally optimal paths to node 2.

Fig. 1. (a) A network over the algebraic structure of widest-shortest paths. Paths to node 2 are indicated with thicker links in the left-local solution (b) and the right-local solution (c). Note that (c) cannot be implemented with the standard destination-based, hop-by-hop forwarding mechanism typically used in the Internet. Packets destined to node 2 would get trapped in the loop 3 4 5 3.

**Algorithm 1** Generalized Dijkstra's algorithm.

```
    for all nodes do
2:      est[u] := 0̄
    est[t] := 1̄
4:  set := nodes
    repeat
6:      find v ∈ set such that est[v] = ⨁_{x∈set} est[x]
        set := set − {v}
8:      for each in-neighbor u of v do
            if est[u] ≺ a(u, v) ⊗ est[v] then
10:             est[u] := a(u, v) ⊗ est[v]
    until set = ∅
```

3 finds the path 3 4 1 2 with width-length $(5, 3)$, meaning that it forwards packets addressed at 2 to node 4; node 4 finds the path 4 5 3 2 with width-length $(5, 7)$, meaning that it forwards packets addressed at 2 to node 5; and node 5 finds the path 5 3 2 with width-length $(5, 5)$, meaning that it forwards packets addressed at 2 to node 3. Hence, packets addressed at 2 are trapped in the forwarding loop 3 4 5 3.

### D. Dijkstra's algorithm

Given that the Left Matrix Equation (2) and the Right Matrix Equation (3) describe routing equilibria, we explore algorithms that solve them. We focus on finding a left-local solution to the left matrix equation. (A right-local solution is found by transposition.) It turns out that under an appropriate condition on the links of the network, Dijkstra's algorithm does compute one column of a left-local solution. The condition is called *left-absorption*:

$$\forall_{(i,j)\in E, b\in S} \quad a(i,j) \otimes b \preceq b. \tag{7}$$

Left-absorption generalizes to prebimonoids Condition (4)—needed for the use of Dijkstra's algorithm in selective semirings—since (7) follows from (4) and right-isotonicity.

*Theorem 4.1:* If the network is left-absorptive, then Algorithm 1 computes the $t$-th column of a left-local solution.

*Proof:* Clearly, Algorithm 1 terminates after $|V|$ iterations of the **repeat** loop. Let $l(u,t)$ be the value of $est[u]$ upon protocol termination. Once a node is extracted from $set$ in line 7 it is never put back there again. Thus, $l(u,t)$ is the value of $est[u]$ after the extraction of $u$ from $set$. The value of variable $est[u]$ can only $\preceq$-increase during execution of the algorithm. Hence,

$$est[u] \preceq l(u,t), \tag{8}$$

at any time.

We first show by induction on the number of extractions from $set$ that

$$est[x] \preceq l(u,t) \tag{9}$$

for all $u$ outside $set$ and all $x$ still in $set$. Consider the moment when $u$ is extracted from $set$. By the choice of $u$ in line 6, we have that inequality (9) is valid after the extraction. Any node $x$ that subsequently witnesses a change in variable $est[x]$ must do so in line 10, yielding

$a(x, u) \otimes l(u, t) = est[x]$. Because of left-absorption, we have $est[x] = a(x, u) \otimes l(u, t) \preceq l(u, t)$, as we wanted to show. Inequality (9) implies

$$l(v, t) \preceq l(u, t), \tag{10}$$

if $u$ is extracted from $set$ before $v$.

We can now prove that the $l(u, t)$, $u \in V$, satisfy the $t$-column of a left-local solution. Trivially, $l(t, t) = \overline{1}$. In addition, if $\overline{0} \prec l(u, t)$, then there is an out-neighbor $v$ of $u$ such that $l(u, t) = a(u, v) \otimes l(v, t)$. We are left to show that $a(u, v) \otimes l(v, t) \preceq l(u, t)$ for all links $uv$. Consider an arbitrary link $uv$. If $v$ is extracted from $set$ before $u$, we get $a(u, v) \otimes l(v, t) \preceq est[u]$ after the extraction of $v$ and execution of lines 9 and 10. From (8), we conclude that

$$a(u, v) \otimes l(v, t) \preceq est[u] \preceq l(u, t).$$

Otherwise, if $v$ is extracted from $set$ after $u$, from left-absorption and (10), we write

$$a(u, v) \otimes l(v, t) \preceq l(v, t) \preceq l(u, t).$$

Thus, in both cases, $a(u, v) \otimes l(v, t) \preceq l(u, t)$. ■

Assuming $\oplus$ and $\otimes$ are constant-time operations, the algorithmic complexity of Dijkstra's algorithm remains the same as traditional accounts [3]. That is, a worst-case running time of $O(V^2)$. Since left- or right-local solution requires $|V|$ calls to this algorithm, the worst-case running time to solve is $O(V^3)$. In sparse graphs this could be improved since the running time for Dijkstra's algorithm can be improved to $O(E \log V)$ using binary heaps or to $O(E + V \log V)$ using Fibonacci heaps.

## V. Discussion and Open Problems

We have made a clear distinction between left- and right-local solutions to path problems. Vectoring protocols, such as the Border Gateway Protocol (BGP) [27], [14], the Interior Gateway Protocol (IGRP), and the Enhanced IGRP (EIGRP) [23], [7], settle for the left-local solutions, thus yielding locally optimal paths rather than globally optimal paths. We have also shown that for some prebimonoids both left- and right-local solutions can be computed with multiple applications of Dijkstra's algorithm. This provides the first polynomial time algorithm for arriving at locally optimal solutions over such algebras.

We suspect that the results presented here may find applications in some *intra-domain* routing settings that require the use of routing policies going beyond what can be expressed within a simple shortest paths model. For example, distributivity can be lost easily when routes are discarded due to policy. If locally optimal solutions are sufficient in these cases, then we have shown that Dijkstra's algorithm can in fact be used to compute solutions in some policy-rich settings. The most efficient approach would be to compute right-local solutions using Dijkstra's algorithm at each router in the network as is currently done with protocols such as OSPF and IS-IS. However, with non-distributive algebras this is not compatible with destination-based, hop-by-hop forwarding and so would require some kind of *tunneling* mechanism such as MPLS [4]. (Note that tunneling would also be required if we attempted to implement globally optimal paths for non-distributive routing metrics.)

On the other hand, left-local solutions could be computed using Dijkstra's algorithm, but the entire all-pairs shortest path problem would have to be solved *at each router* in order to achieve loop-free hop-by-hop forwarding. The additional expense could be traded off against the fact the tunneling is not required and slow path exploration techniques avoided. This might prove to be a reasonable approach in some networks.

Distributed Bellman-Ford algorithms have better scaling properties since memory requirements at each router scale with the number of network *destinations*, while link-state routing protocols must store records for *all links* in the network. Inter-domain link-state routing would have enormous space requirements since the link-state domain would include the global Internet. But we suspect that there is another intrinsic obstacle to using Dijkstra's algorithm in the inter-domain setting. The problem is that link-state flooding *reveals* policy, which is today considered *proprietary*. However, the link-state approach may be applicable to Internal BGP (an important mode of BGP used *within* a network) since link state announcements would remain private within an ISP. Another possible application might be to improve convergence time for a popular method of implementing Virtual Private Networks [24] that currently employs BGP.

Several interesting theoretical problems remain open. First, for a large class of prebimonoids it is known that the Bellman-Ford algorithm (or iterative matrix methods) will eventually terminate when paths are restricted to simple paths. Yet no polynomial bound on the number of iterations required is known [9].

Second, the problem of finding *globally optimal* paths for prebimonoids does not seem to have a general solution at this time. Some authors have demonstrated techniques for finding globally optimal paths for a modified algebra which is distributive, and then translating solutions back to the non-distributive metrics (see [20], [19], [25] and Chapter 5 of [1]). Perhaps these techniques can be generalized to a large class of prebimonoids.


## Acknowledgements

We have benefited from discussions with Fred Baker, John S. Baras, Bruce Davie, Alexander Gurney, Vilius Naudžiūnas, Iulian Nitescu, Cristel Pelsser, Philip Taylor, and Gordon Wilfong. Timothy G. Griffin would like to thank the following for their financial support: Cisco Systems, Boeing, and EPSRC (grant EP/F002718/1).


## References


[1] J. S. Baras and G. Theodorakopoulos. *Path problems in networks*. Morgan & Claypool, 2010.

[2] C. Chau, R. Gibbens, and T. G.Griffin. Towards a unified theory of policy-based routing. In *Proc. IEEE INFOCOM*, April 2006.

[3] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press, Cambridge, MA, second edition, 2001.

[4] B. S. Davie and Y. Rekhter. *MPLS: Technology and Applications*. Morgan Kaufmann, 2000.

[5] L. Gao and J. Rexford. Stable internet routing without global coordination. *IEEE/ACM Transactions on Networking*, pages 681–692, December 2001.

[6] M. Gondran and M. Minoux. *Graphs, Dioids, and Semirings : New Models and Algorithms*. Springer, 2008.

[7] M. G. Gouda and M. Schneider. Maximizable routing metrics. *IEEE/ACM Trans. Netw.*, 11(4):663–675, 2003.

[8] T. G. Griffin. The stratified shortest-paths problem. In *The third International Conference on COMmunication Systems and NETworkS (COMSNETS)*, January 2010.

[9] T. G. Griffin and A. J. T. Gurney. Increasing bisemigroups and algebraic routing. In *10th International Conference on Relational Methods in Computer Science (RelMiCS10)*, April 2008.

[10] T. G. Griffin and G. Huston. RFC 4264: BGP Wedgies, November 2005. IETF.

[11] T. G. Griffin, F. B. Shepherd, and G. Wilfong. The stable paths problem and interdomain routing. *IEEE/ACM Transactions on Networking*, 10(2):232–243, April 2002.

[12] A. Gurney. *Construction and verification of routing algebras*. PhD thesis, University of Cambridge, 2009.

[13] A. J. T. Gurney and T. G. Griffin. Lexicographic products in metarouting. In *Proc. Inter. Conf. on Network Protocols*, October 2007.

[14] S. Halabi and D. McPherson. *Internet Routing Architectures*. Cisco Press, second edition, 2001.

[15] G. Huston. Interconnection, peering and settlements: Part I. *Internet Protocol Journal*, 2(1), June 1999.

[16] G. Huston. Interconnection, peering and settlements: Part II. *Internet Protocol Journal*, 2(2), June 1999.

[17] L. Lamport. An assertional correctness proof of a distributed algorithm. *Science of Computer Programming*, 2(3):175–206, December 1982.

[18] L. Lamport. The temporal logic of actions. *ACM Trans. on Programming Languages and Systems*, 16(3):872–923, April 1994.

[19] T. Lengauer and D. Theune. Efficient algorithms for path problems with general cost criteria. *Lecture Notes in Computer Science*, 510:314–326, 1991.

[20] T. Lengauer and D. Theune. Unstructured path problems and the making of semirings. *Lecture Notes in Computer Science*, 519:189–200, 1991.

[21] J. L.Sobrinho. An algebraic theory of dynamic network routing. *IEEE/ACM Transactions on Networking*, 13(5):1160–1173, October 2005.

[22] R. Perlman. *Interconnections — Bridges, Routers, Switches, and Internetworking Protocols*. Addison-Wesley, second edition, 2000.

[23] A. Retana, R. White, and D. Slice. *EIGRP: Basic Operation and Configuration*. Addison Wesley, Reading, MA, 2000. ISBN 0201657732.

[24] E. Rosen and Y. Rekhter. BGP/MPLS VPNs. RFC 4364, 2006.

[25] A. Sen, K. S. Candan, A. Ferreira, B. Beauquier, and S. Perennes. On shortest path problems with "non-markovian" link contribution to path lengths. In *NETWORKING '00: Proceedings of the IFIP-TC6 / European Commission International Conference on Broadband Communications, High Performance Networking, and Performance of Communication Networks*, pages 859–870, London, UK, 2000. Springer-Verlag.

[26] J. L. Sobrinho. Algebra and algorithms for QoS path computation and hop-by-hop routing in the Internet. *IEEE/ACM Transactions on Networking*, 10(4):541–550, August 2002.

[27] J. W. Stewart III. *BGP4: Inter-Domain Routing in the Internet*. Addison Wesley, Reading, MA, 1999. ISBN 0201379511.

[28] K. Varadhan, R. Govindan, and D. Estrin. Persistent route oscillations in inter-domain routing. *Computer Networks*, 32:1–16, 2000. An early version was a widely circulated ISI technical report from 1996.