

# The Stratified Shortest-Paths Problem (Invited Paper)

Timothy G. Griffin

Computer Laboratory

University of Cambridge

Email: Timothy.Griffin@cl.cam.ac.uk

**Abstract**—In the last ten years it has become clear that some Internet routing protocols do not compute *globally* optimal paths, but only *locally* optimal ones. This represents something rather novel in the context of the vast literature on routing protocols for data networking. This paper introduces the *Stratified Shortest-Paths Problem* as a tool for exploring the borderline between local and global optimality problems. The paper contains a tutorial overview of the algebraic concepts used.

## I. A STEP BEYOND SHORTEST PATHS

In attempting to model inter-domain routing in the Internet it was discovered that *global optimality* is not an appropriate framework. Instead, *local optimality* seems more reasonable. With local optimality it may not be possible for nodes to achieve globally optimal paths, but only the best paths given all the paths available from neighbors. This was first formalized in a graph-theoretic model called the Stable Paths Problem [10], and later Sobrinho cast the analysis in a more algebraic framework [20], [21].

Section II reviews algebraic generalizations of the shortest paths problem. This research arose primarily from the Operations Research community — see [6] for an excellent summary of results from the last four decades. Since the algebraic approach to routing is not widely known in the data networking community, I have attempted to make this a short tutorial overview. The section ends by presenting some of the more exotic extensions that have evolved in recent years to model local optimality problems.

This paper introduces the *Stratified Shortest-Paths Problem* as an aid for exploring the borderline between local and global optimality problems. The basic idea (formalized in Section III), is to consider routing metrics that essentially have the form

$$(s, d),$$

where  $s$  is a *stratum level* in  $\{0, 1, 2, \dots, m-1\}$ , and  $d$  is a “shortest-paths” distance. Routing metrics are compared lexicographically, where

$$(s_1, d_1) < (s_2, d_2)$$

when  $s_1 < s_2$ , or when  $s_1 = s_2$  and  $d_1 < d_2$ . Policies on arcs have two components: one applied to the stratum and the other an integer weight added on to the accumulating distance. The policies on strata are restricted so that the resulting algebra always can be used to find local optima (Section II).

The full stratified shortest path problem for  $m$  strata comes with  $(m+1)!$  distinct policy functions over strata. Section IV considers some interesting subsets of this rich policy space which give rise to routing structures with special properties. Some of these sub-algebras can be used to find globally optimal paths, while others only locally optimal paths. Open problems are discussed in Section V.

## II. GLOBAL VS. LOCAL OPTIMA

Figure 1 presents a simple directed graph with arc weights in  $\mathbb{N}$  (in this example, the weight on arc  $(i, j)$  is the same as that on arc  $(j, i)$ , and so the orientation is not shown). We represent such a graph as  $G = (V, E)$ , where  $V$  is a set of vertices and  $E \subseteq V \times V$  is a set of arcs. Arc weights in this case are represented by a function  $E \rightarrow \mathbb{N}^\infty$ , where  $\mathbb{N}^\infty = \mathbb{N} \cup \{\infty\}$  and  $w(i, j) = \infty$  is taken to mean that there is no arc from  $i$  to  $j$ .

A weight function  $w$  can also be represented as an *adjacency matrix* [3]. For example, the weights on the graph of Figure 1 can be represented by the matrix

$$\mathbf{A} = \begin{matrix} & \begin{matrix} 1 & 2 & 3 & 4 & 5 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{matrix} & \begin{bmatrix} \infty & 2 & 1 & 6 & \infty \\ 2 & \infty & 5 & \infty & 4 \\ 1 & 5 & \infty & 4 & 3 \\ 6 & \infty & 4 & \infty & \infty \\ \infty & 4 & 3 & \infty & \infty \end{bmatrix} \end{matrix}$$

The value along the main diagonal is usually taken to be 0 or  $\infty$  depending on minor details of the presentation. The *solution* to the all-pairs shortest path problem is a matrix  $\mathbf{R}$  such that

$$\mathbf{R}(i, j) = \min_{p \in \mathcal{P}(i, j)} w(p),$$

where  $\mathcal{P}(i, j)$  represents the set of all paths from node  $i$  to node  $j$  in the graph  $G$  and  $w(p)$  is the summation of all arc weights along this path (the weight of the empty path is taken to be 0). Since  $p$  ranges over *all paths* from  $i$  to  $j$ , I will refer to this as a *globally optimal solution*.

For the example of Figure 1, the solution is

$$\mathbf{R} = \begin{matrix} & \begin{matrix} 1 & 2 & 3 & 4 & 5 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{matrix} & \begin{bmatrix} 0 & 2 & 1 & 5 & 4 \\ 2 & 0 & 3 & 7 & 4 \\ 1 & 3 & 0 & 4 & 3 \\ 5 & 7 & 4 & 0 & 7 \\ 4 & 4 & 3 & 7 & 0 \end{bmatrix} \end{matrix}$$

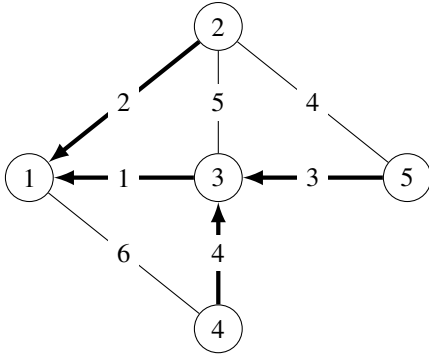


Fig. 1. Shortest paths example. Bold arrows indicate a shortest-path tree rooted at node 1.

There are many algorithms, centralized and distributed, for computing the solution from an adjacency matrix, see for example [3]. For simplicity, the only algorithm used in this paper is based on iteration of matrix multiplication.

Matrix multiplication is defined just as is the standard operation over matrices of reals, except that  $+$  is replaced by  $\min$  and  $\times$  is replaced by  $+$ . That is, if  $\mathbf{B}$  and  $\mathbf{C}$  are matrices, then

$$(\mathbf{BC})(i, j) = \min_{1 \leq q \leq n} \mathbf{B}(i, q) + \mathbf{C}(q, j). \quad (1)$$

Note that the identity matrix for multiplication, denoted  $\mathbf{I}$ , is defined as  $\mathbf{I}(i, i) = 0$  and  $\mathbf{I}(i, j) = \infty$ , for  $i \neq j$ . Matrix “addition” in this context is defined as

$$(\mathbf{B} \min \mathbf{C})(i, j) = \mathbf{B}(i, j) \min \mathbf{C}(i, j). \quad (2)$$

An iterative method for computing a solution  $\mathbf{R}$  is defined as

$$\begin{aligned} \mathbf{A}^{(0)} &= \mathbf{I} \\ \mathbf{A}^{(k+1)} &= \mathbf{AA}^{(k)} \min \mathbf{I}. \end{aligned}$$

It is then not too hard to show that for an  $n$ -node graph,  $\mathbf{A}^{(n)} = \mathbf{A}^{(n-1)}$ , so the iteration terminates after  $n - 1$  steps. In addition,  $\mathbf{A}^{(n-1)} = \mathbf{R}$ , the globally optimal solution.

Another way of looking at this is by noting that

$$\mathbf{A}^{(n-1)} = \mathbf{AA}^{(n-1)} \min \mathbf{I}.$$

Thus the matrix  $\mathbf{A}^{(n-1)}$  solves the matrix equation

$$\mathbf{X} = \mathbf{AX} \min \mathbf{I}. \quad (3)$$

We can read this equation as saying that a solution  $\mathbf{X}$  is a *locally optimal solution*. Suppose that  $i \neq j$  are two nodes. Then for any  $\mathbf{X}$  solving Equation 3 we have

$$\begin{aligned} \mathbf{X}(i, j) &= \min_{1 \leq q \leq n} \mathbf{A}(i, q) + \mathbf{X}(q, j) \\ &= \min_{q \in N(i)} w(i, q) + \mathbf{X}(q, j), \end{aligned}$$

where  $N(i) = \{q \mid (i, q) \in E\}$  are the neighbors of node  $i$ . That is,  $\mathbf{X}$  can be thought of as a *local solution* —  $\mathbf{X}(i, j)$  represents the best path weights that  $i$  can obtain given its immediate neighbors’ best paths. Happily, any globally optimal solution is a locally optimal solution! This will not be the case for the algebraic structures considered in Section II-C.

### A. Semirings

The mathematical basis of shortest-path computations can be represented concisely as a tuple,

$$\text{sp} = (\mathbb{N}^\infty, \min, +, \infty, 0).$$

A great deal of research has followed from the observation — first made about 40 years ago — that this  $(\min, +)$  structure and many associated algorithms can be generalized to a very large class of algebraic structures call *semirings*. These structures have the form

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

where  $\oplus$  and  $\otimes$  are binary operations over set  $S$ ,  $\bar{0} \in S$  is the identity for  $\oplus$ , and  $\bar{1} \in S$  is the identity for  $\otimes$ . Figure 2 presents just a few semirings that might be useful in data networking.

The symbols normally used for semirings are derived from the familiar *ring*

$$(\mathbb{R}, +, \times, 0, 1).$$

The notation may be a bit confusing when speaking of the shortest path semiring where  $\oplus = \min$ ,  $\otimes = +$ , and  $\bar{1} = 0$ . The notation does have one thing going for it — the operations of matrix addition and multiplication now look more like their familiar counterparts,

$$\begin{aligned} (\mathbf{B} \oplus \mathbf{C})(i, j) &= \mathbf{B}(i, j) \oplus \mathbf{C}(i, j) \\ (\mathbf{BC})(i, j) &= \sum_{1 \leq q \leq n} \mathbf{B}(i, q) \otimes \mathbf{C}(q, j). \end{aligned}$$

It turns out that many of the matrix methods for solving several types of semiring equations are related to analogous methods in classical linear algebra (see [6] for an extensive account).

What properties of  $(S, \oplus, \otimes, \bar{0}, \bar{1})$  are required for it to be a semiring? First, both  $(S, \oplus)$  and  $(S, \otimes)$  must be *semigroups*, which is nothing more than saying their binary operations are associative. For example, it must be that  $a \oplus (b \oplus c) = (a \oplus b) \oplus c$  for every  $a, b, c \in S$ . The additive operator  $\oplus$  is required to be commutative. That is, for all  $a, b \in S$ ,  $a \oplus b = b \oplus a$ . Furthermore,  $\bar{0}$  must be the identity for  $\oplus$  and  $\bar{1}$  must be the identity for  $\otimes$ . That is, for all  $a \in S$ ,

$$\begin{aligned} a \oplus \bar{0} &= \bar{0} \oplus a = a \\ a \otimes \bar{1} &= \bar{1} \otimes a = a. \end{aligned}$$

In addition,  $\bar{0}$  must be an *annihilator* for  $\otimes$ . That is, for all  $a \in S$ ,

$$a \otimes \bar{0} = \bar{0} \otimes a = \bar{0}.$$

Finally, we need *distributivity* conditions to hold for multiplication on the right and left of a sum:

$$\begin{aligned} \text{L.DIST} &: a \otimes (b \oplus c) = (a \otimes b) \oplus (a \otimes c), \\ \text{R.DIST} &: (a \oplus b) \otimes c = (a \otimes c) \oplus (b \otimes c). \end{aligned}$$

A graph  $G = (V, E)$  can now be weighted over a semiring with a function  $w : E \rightarrow S$ . The weight of a path

| name            | $S$                 | $\oplus$ | $\otimes$ | $\bar{0}$ | $\bar{1}$ | possible use in routing                  |
|-----------------|---------------------|----------|-----------|-----------|-----------|--|
| sp              | $\mathbb{N}^\infty$ | min      | +         | $\infty$  | 0         | shortest paths                           |
| sp <sub>m</sub> | $[m]^\infty$        | min      | +         | $\infty$  | 0         | shortest paths (bounded distance)        |
| bw              | $\mathbb{N}^\infty$ | max      | min       | 0         | $\infty$  | widest paths (greatest capacity)         |
| bw <sub>m</sub> | $[m]^\infty$        | max      | min       | 0         | $\infty$  | widest paths (greatest bounded capacity) |
| rel             | $[0, 1]$            | max      | $\times$  | 0         | 1         | most-reliable paths                      |
| cup.cap( $W$ )  | $2^W$               | $\cup$   | $\cap$    | $\{\}$    | $W$       | shared link attributes?                  |
| cap.cup( $W$ )  | $2^W$               | $\cap$   | $\cup$    | $W$       | $\{\}$    | shared path attributes?                  |

Fig. 2. A few semiring examples together with possible interpretations in terms of network routing. For many more examples, see [1], [5], [6]. The following notations are used:  $\mathbb{N}^\infty = \mathbb{N} \cup \{\infty\}$  and  $[m]^\infty = \{0, 1, \dots, m-1, \infty\}$ .

$p = v_0, v_1, v_2 \dots v_{k-1}, v_k$  is defined as the *product* of the arc weights (remember,  $\otimes = +$  in shortest paths),

$$w(p) = w(v_0, v_1) \otimes w(v_1, v_2) \otimes \dots \otimes w(v_{k-1}, v_k).$$

The weight of the empty path is  $\bar{1}$  (which is 0 in shortest paths). The *globally optimal solution* to the all-pairs path problem is a matrix  $\mathbf{R}$  such that

$$\mathbf{R}(i, j) = \bigoplus_{p \in \mathcal{P}(i, j)} w(p).$$

Such an  $\mathbf{R}$  is also a solution to the matrix equation

$$\mathbf{X} = \mathbf{A}\mathbf{X} \oplus \mathbf{I}, \quad (4)$$

and again, globally optimal solutions (if they exist) are locally optimal solutions.

Figure 3 presents a *widest path* example using the same arc weights as the shortest path example of Figure 1, but now interpreted in the semiring bw (defined in Figure 2). For this example the solution matrix is

$$\mathbf{R} = \begin{matrix} & \begin{matrix} 1 & 2 & 3 & 4 & 5 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{matrix} & \left[ \begin{array}{ccccc} \infty & 4 & 4 & 6 & 4 \\ 4 & \infty & 5 & 4 & 4 \\ 4 & 5 & \infty & 4 & 4 \\ 6 & 4 & 4 & \infty & 4 \\ 4 & 4 & 4 & 4 & \infty \end{array} \right] \end{matrix}$$

Note that here global optimality is captured by

$$\mathbf{R}(i, j) = \max_{p \in \mathcal{P}(i, j)} w(p),$$

where  $w(p)$  represents the minimal weight along path  $p$ .

To find solutions, the iterative method for computing shortest paths can be generalized to semirings,

$$\begin{aligned} \mathbf{A}^{(0)} &= \mathbf{I} \\ \mathbf{A}^{(k+1)} &= \mathbf{A}\mathbf{A}^{(k)} \oplus \mathbf{I} \end{aligned}$$

If the following limit exists,

$$\mathbf{A}^* = \lim_{k \rightarrow \infty} \mathbf{A}^{(k)}$$

then  $\mathbf{A}^*$  is a solution to Equation 4 (in some sense the best solution, see [6]).

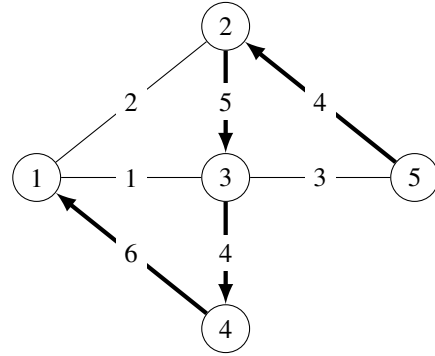


Fig. 3. Widest paths example, using semiring bw. Bold arrows indicate a widest-path tree rooted at 1.

There are various sufficient conditions that imply that  $\mathbf{A}^*$  exists. Here we assume that the multiplicative identity  $\bar{1}$  is also an additive annihilator,

$$a \oplus \bar{1} = \bar{1} \oplus a = \bar{1}.$$

This is the case with all of the semirings of Figure 2.

In many cases the additive operator  $\oplus$  is *idempotent*, meaning that  $a \oplus a = a$  for each  $a \in S$ . All of the examples of Figure 2 have an idempotent  $\oplus$ . In this case we can define a natural partial order on  $S$  as

$$a \leq b \iff a = a \oplus b.$$

For this order, the operation  $\oplus$  is the greatest lower bound (glb). The order is a total order when the operation  $\oplus$  is *selective*. That is, if  $a \oplus b \in \{a, b\}$  for every  $a, b \in S$ .

With the assumption that  $\bar{1}$  is an additive annihilator, it is easy to see that for every  $a \in S$ ,  $\bar{1} \leq a \leq \bar{0}$ . (For shortest paths, this says  $0 \leq a \leq \infty$ .) There is another interesting consequence of this assumption. Note that for any  $a, b \in S$ ,  $a = \bar{1} \otimes a = (\bar{1} \oplus b) \otimes a = a \oplus (b \otimes a)$ . Thus we have

$$a \leq b \otimes a, \quad (5)$$

which is related to the *inflationary* property required of some algebraic structures treated below.

Figure 4 illustrates the use of the semiring cup.cap( $\{a, b, c\}$ ), whose natural order is a *partial*

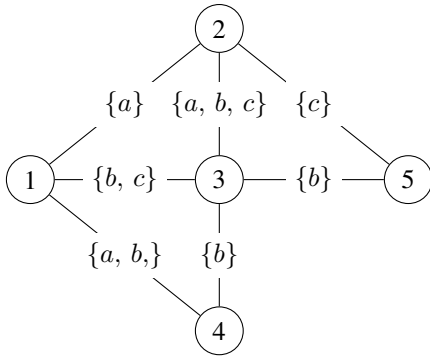


Fig. 4. An example using semiring  $\text{cup.cap}(\{a, b, c\})$ , which has a natural partial order.

order (in this case  $\supseteq$ ). This example is somewhat unusual from the network routing perspective, but it illustrates the generality of the algebraic approach. A solution is a matrix  $\mathbf{R}$  satisfying this global optimality condition:

$$\mathbf{R}(i, j) = \bigcup_{p \in P(i, j)} w(p),$$

where  $w(p)$  is now the intersection of all arc weights in  $p$ . The resulting solution for Figure 4 is

$$\mathbf{R} = \begin{matrix} & \begin{matrix} 1 & 2 & 3 & 4 & 5 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{matrix} & \left[ \begin{array}{ccccc} \{a \ b \ c\} & \{a \ b \ c\} & \{a \ b \ c\} & \{a \ b\} & \{b \ c\} \\ \{a \ b \ c\} & \{a \ b \ c\} & \{a \ b \ c\} & \{a \ b\} & \{b \ c\} \\ \{a \ b \ c\} & \{a \ b \ c\} & \{a \ b \ c\} & \{a \ b\} & \{b \ c\} \\ \{a \ b\} & \{a \ b\} & \{a \ b\} & \{a \ b \ c\} & \{b\} \\ \{b \ c\} & \{b \ c\} & \{b \ c\} & \{b\} & \{a \ b \ c\} \end{array} \right] \end{matrix}$$

which can be obtained by the iterative method. How can this be interpreted? For  $x \in \{a, b, c\}$ , interpret  $x \in \mathbf{R}(i, j)$  to mean that there is at least one path from  $i$  to  $j$  with  $x$  in every arc weight along the path. Perhaps this could be used for network management purposes, where  $\{a, b, c\}$  represent (desirable?) properties of links. The other partially ordered example from Figure 2 is  $\text{cap.cup}$ . Using that algebra,  $x \in \{a, b, c\}$  and  $x \in \mathbf{R}(i, j)$  says that every path from  $i$  to  $j$  contains an  $x$  in at least one arc weight.

Unlike the previous examples, Figure 4 does not depict a “best path tree” rooted at node 1. That is because we have a partial order and no such tree exists, in general. For example,  $\mathbf{R}(1, 2) = \{a, b, c\}$ , since path 1, 2 contributes  $\{a\}$ , path 1, 3, 2 contributes  $\{b, c\}$ , and path 1, 4, 3, 2 contributes  $\{b\}$ . In other words, no single path from node 1 to node 2 has weight  $\mathbf{R}(1, 2)$ . Not all path finding algorithms can be generalized to such semirings — for example, Dijkstra’s algorithm requires a total order.

## B. Beyond Semirings

Many problems (including the stratified shortest-paths problem) are best modeled by placing *functions* on arcs. That is, we sometimes need to replace the multiplicative structure  $(S, \otimes)$  of a semiring with a set of functions  $F \subseteq S \rightarrow S$ . Gondran

and Minoux call the resulting structures *algebras of monoid endomorphisms* (AME),

$$(S, \oplus, F, \bar{0}, \bar{1}).$$

As with semirings, we require that  $(S, \oplus)$  be a commutative semigroup with identity  $\bar{0}$ . Corresponding to the semiring requirement that  $\bar{0}$  is a multiplicative annihilator, we have here the requirement that  $f(\bar{0}) = \bar{0}$  for every  $f \in F$ . Finally, every function  $f \in F$  must be *distributive* with respect to  $\oplus$ ,

$$\text{DIST} : f(a \oplus b) = f(a) \oplus f(b).$$

A graph  $G$  is weighted with functions from  $F$  using a weight function  $w : E \rightarrow F$ . If  $p = v_0, v_1, v_2 \cdots v_{k-1}, v_k$  is a path in  $G$  of  $k$  arcs, then the weight of  $p$  is

$$w(p) = w(v_0, v_1)(w(v_1, v_2) \cdots (w(v_{k-1}, v_k)(\bar{1}) \cdots)).$$

Another way to write this is

$$w(p) = (w(v_0, v_1) \circ w(v_1, v_2) \circ \cdots \circ w(v_{k-1}, v_k))(\bar{1})$$

The empty path is given the weight  $\bar{1}$ .

Every semiring can be considered as an AME by defining functions  $f_b$ , where  $f_b(a) = b \otimes a$ , and letting  $F = \{f_b \mid b \in S\}$ . We can also easily restrict the values placed on arcs by taking only a subset of this  $F$ . For example, starting with the shortest-paths algebra  $\text{sp}$ , we can construct

$$\text{sp}^+ = (\mathbb{N}^\infty, \min, F_+, \infty, 0),$$

where  $F_+$  is defined so that no arc can be given weight 0. That is,  $F_+ = \{f_b \mid b \in \mathbb{N}^\infty \wedge b \neq 0 \wedge \forall a \in \mathbb{N}^\infty : f_b(a) = b + a\}$ . This algebra is important in Section III.

Matrix multiplication must be redefined. If  $\mathbf{H}$  is a matrix containing functions in  $F$  and  $\mathbf{C}$  is a matrix containing elements in  $S$ , then

$$(\mathbf{HC})(i, j) = \bigoplus_{1 \leq q \leq n} \mathbf{H}(i, q)(\mathbf{C}(q, j)).$$

The iterative method is redefined as

$$\begin{aligned} \mathbf{H}^{(0)} &= \mathbf{I} \\ \mathbf{H}^{(k+1)} &= \mathbf{HH}^{(k)} \oplus \mathbf{I}. \end{aligned}$$

Note that  $\mathbf{H}^{(k)}$  is a matrix over  $S$ .

Again, we seek sufficient conditions that ensure that  $\mathbf{H}^*$  exists. For semirings, we made the assumption that the multiplicative identity  $\bar{1}$  is an additive annihilator. Here we simply assume that there is an additive annihilator, and we denote it as  $\bar{1}$ . With semirings, Equation 5 was *derived* from the assumption concerning  $\bar{1}$ . Here we must *assume* that all functions are *inflationary* with respect to  $\leq$ :

$$\forall a \in S, f \in F : a \leq f(a).$$

Again, it is not too difficult to show that under these assumptions  $\mathbf{H}^{(n-1)}$  is a globally optimal solution for an  $n$ -node graph. In addition, globally optimal solutions are locally optimal ones, and satisfy the equation

$$\mathbf{X} = \mathbf{HX} \oplus \mathbf{I}. \quad (6)$$

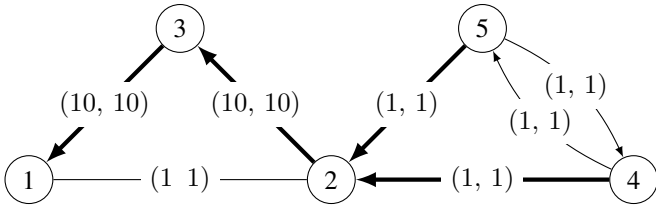


Fig. 6. A widest-shortest path tree rooted at node 1. The path 1, 3, 2, 4 is only locally optimal, since 1, 2, 4 has a lower weight but is not compatible with the best path from node 1 to node 2.

### C. Beyond AMEs

We now enter a recently discovered and fairly wild territory — we are going to allow algebras that do not have distributivity! In order to show how this might in fact be meaningful and useful, we first construct one such wild and terrifying algebra using a lexicographic construction.

Let  $S = (S, \oplus_S, F_S, \bar{0}_S, \bar{1}_S)$  and  $T = (T, \oplus_T, F_T, \bar{0}_T, \bar{1}_T)$  be AMEs. Suppose that  $\oplus_S$  is commutative, idempotent, and selective. This means that the associated order  $\leq_S$  is total. The *lexicographic product* of semigroups  $(S, \oplus_S)$  and  $(T, \oplus_T)$  is the semigroup  $(S \times T, \oplus)$ , where

$$(s_1, t_1) \oplus (s_2, t_2) = \begin{cases} (s_1, t_1 \oplus_T t_2) & (\text{if } s_1 = s_2) \\ (s_1, t_1) & (\text{if } s_1 <_S s_2) \\ (s_2, t_2) & (\text{if } s_2 <_S s_1) \end{cases}$$

This construction will be used to construct the additive component of a new AME. The multiplicative component will be constructed using the *direct product*, where a pair  $(f, g) \in F_S \times F_T$  is taken to represent a new function  $h : (S \times T) \rightarrow (S \times T)$ ,

$$h(s, t) = (f(s), g(t)).$$

Now define the lexicographic product of  $S$  and  $T$ , denoted  $S \vec{\times} T$ , as

$$(S \times T, \oplus, F_S \times F_T, (\bar{0}_S, \bar{0}_T), (\bar{1}_S, \bar{1}_T)).$$

This construction does not always result in an AME, because it may not be distributive. It was shown in [11], [12] that

$$\text{DIST}(S \vec{\times} T) \iff \text{DIST}(S) \wedge \text{DIST}(T) \wedge (\mathbf{K}(S) \vee \mathbf{C}(T)),$$

where these properties are defined in Figure 5.

Note that the *shortest-widest paths* algebra,  $\text{sp} \vec{\times} \text{bw}$ , is distributive. On the other hand, reversing the product results in  $\text{bw} \vec{\times} \text{sp}$ , which might be used to find *widest shortest paths* in a graph. This structure is not distributive [19]. Why? Because  $\neg \mathbf{K}(\text{bw}) \wedge \neg \mathbf{C}(\text{sp})$ .

In spite of this lack of distributivity, we can find widest-shortest path trees as illustrated in Figure 6. Note that the tree illustrated in that figure does not represent a *globally* optimal tree, only a *locally* optimal tree. To see this, note that path 4, 2, 1 has weight  $(1, 2)$  while path 4, 2, 3, 1 has weight  $(1, 21)$ . Although  $(1, 2) < (1, 21)$ , the best path from node 4 to node

1 is 4, 2, 3, 1 of weight  $(1, 21)$ . Thus, 4, 2, 3, 1 represents a *locally optimal* path.

When can we find local optima when algebraic structures lack distributivity? Today we know a few sufficient conditions (for various flavors of proof, see [2], [8], [10], [11], [20], [21]). Here we use the condition that  $(S, \oplus, F, \bar{0}, \bar{1})$  be *strictly inflationary*, property S.INFL of Figure 5. The cited proofs all restrict consideration to elementary paths in a graph — those that do not contain any loops.

To see why this last restriction might be needed, consider applying the iterative method to the widest-shortest example of Figure 6:

$$\begin{array}{c} \mathbf{A}^{(2)}(i, 1) \\ 1 \begin{bmatrix} (0, \infty) \\ (10, 20) \\ (10, 10) \\ (1, 2) \\ (1, 2) \end{bmatrix} \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \rightarrow \begin{array}{c} \mathbf{A}^{(3)}(i, 1) \\ 1 \begin{bmatrix} (0, \infty) \\ (10, 20) \\ (10, 10) \\ (1, 3) \\ (1, 3) \end{bmatrix} \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \rightarrow \dots \rightarrow \begin{array}{c} \mathbf{A}^{(4)}(i, 1) \\ 1 \begin{bmatrix} (0, \infty) \\ (10, 20) \\ (10, 10) \\ (1, 4) \\ (1, 4) \end{bmatrix} \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \rightarrow \dots \rightarrow \begin{array}{c} \mathbf{A}^{(21)}(i, 1) \\ 1 \begin{bmatrix} (0, \infty) \\ (10, 20) \\ (10, 10) \\ (1, 21) \\ (1, 21) \end{bmatrix} \\ 2 \\ 3 \\ 4 \\ 5 \end{array}$$

Happily, we only *count to convergence* in this example. In general, the iterative method can diverge when using non-distributive, strictly inflationary algebras due to (self-induced) *counting-to-infinity*.

If the iterative method is to be salvaged, then we need some way of restricting weight calculations to elementary paths. For this we introduce the *absorbing product* construction. First we select a new “infinity”  $\bar{0}$  that is not in  $S$  or  $T$  and form the set

$$S \times_{\bar{0}} T = ((S - \{\bar{0}_S\}) \times (T - \{\bar{0}_T\})) \cup \{\bar{0}\}.$$

Now, rather than pairs of the form  $(s, t)$ , we use pairs of the form  $\langle s, t \rangle$ , which denote elements of  $S \times_{\bar{0}} T$ ,

$$\langle s, t \rangle = \begin{cases} \bar{0} & (\text{if } s = \bar{0}_S \vee t = \bar{0}_T) \\ (s, t) & (\text{otherwise}) \end{cases}$$

The direct product is now re-defined so that every pair  $(f, g) \in F_S \times F_T$  is taken to represent the function  $h : (S \times_{\bar{0}} T) \rightarrow (S \times_{\bar{0}} T)$ ,

$$\begin{aligned} h(s, t) &= \langle f(s), g(t) \rangle \\ h(\bar{0}) &= \bar{0}. \end{aligned}$$

The resulting lexicographic product is denoted  $S \vec{\times}_{\bar{0}} T$ . It can be shown that

$$(\text{DIST}(S) \wedge \text{DIST}(T) \wedge \mathbf{K}_{\bar{0}}(S)) \implies \text{DIST}(S \vec{\times}_{\bar{0}} T),$$

where these properties are defined in Figure 5.

We then define an algebra, called *paths*, as follows. Let  $\mathcal{E}_V$  represent the set of all sequences of vertices in graph  $G =$

| property      | definition   | description           |
|---------------|--|-----------------------|
| DIST          | $\forall a, b \in S, f \in F : f(a \oplus b) = f(a) \oplus f(b)$                             | distributive          |
| INFL          | $\forall a \in S, f \in F : a \leq f(a)$   | inflationary          |
| S.INFL        | $\forall a \in S, F \in F : a \neq \bar{0} \implies a < f(a)$                                | strictly inflationary |
| K             | $\forall a, b \in S, f \in F : f(a) = f(b) \implies a = b$                                   | cancellative          |
| $K_{\bar{0}}$ | $\forall a, b \in S, f \in F : f(a) = f(b) \implies (a = b \vee f(a) = \bar{0})$             | almost cancellative   |
| C             | $\forall a, b \in S, f \in F : f(a) = f(b)$  | constant              |
| $C_{\bar{0}}$ | $\forall a, b \in S, f \in F : f(a) \neq f(b) \implies (f(a) = \bar{0} \vee f(b) = \bar{0})$ | almost constant       |

Fig. 5. Some properties for algebraic structures of the form  $(S, \oplus, F, \bar{0}, \bar{1})$ . Note that in this paper we consider only lexicographic products whose first component is totally ordered. In [11], [12] the properties K and C are more complex since a more general form of the lexicographic product is studied. The property  $C_{\bar{0}}$  is used in Section IV-C. The property *inflationary* is called *monotonicity* in [20], [21] and *increasing* in [8], [12].

$(V, E)$  that have no repeated vertices. For each arc  $(i, j) \in E$  define the function  $f_{(i, j)}$  as

$$f_{(i, j)}(\sigma) = \begin{cases} \infty & (\text{if } \sigma = \infty \vee i \in j\sigma) \\ j\sigma & (\text{otherwise}) \end{cases}$$

Here  $j\sigma$  denotes the concatenation of  $j$  onto the sequence  $\sigma$ . Let  $F_E$  denote the set of all such functions. We abuse notation a little and extend the application of such functions to sets  $U \subseteq \mathcal{E}_V$ ,

$$f_{(i, j)}(U) = \{\sigma' \mid \sigma' = f_{(i, j)}(\sigma) \wedge \sigma \in U \wedge \sigma' \neq \infty\}$$

Finally, the algebra paths is defined as

$$\text{paths} = (2^{\mathcal{E}_V} \cup \{\infty\}, \cup, F_E, \mathcal{E}_V, \{\}).$$

This is essentially an algebraic encoding of the ASPATH from BGP (but not meant to be in any way efficient, since here we carry along a *set* of paths, that may grow exponentially large with respect to the size of the graph).

We can then “fix” the widest-shortest path algebra by adding paths using an absorbing product,

$$(\text{bw } \vec{\times} \text{ sp}) \vec{\times}_{\infty} \text{ paths},$$

and the iterative method will always converge. Note that this is not the best way to implement the iterative method, since it could lead to path sets that are exponentially large (with respect to the size of the graph). However, this approach allows us to make connections with extensive literature on algebraic routing. It also allows us to make a clear distinction between the problem being solved and the algorithms used to solve it.

### III. STRATIFIED SHORTEST-PATHS

We now construct the *stratified shortest-paths algebra* as

$$\text{strsp}_m = \mathcal{S}_m \vec{\times}_{\infty} \text{sp}^+,$$

where the *strata algebra*  $\mathcal{S}_m$  is defined below. We want  $\text{strsp}_m$  to be strictly inflationary so that it can be used to solve local optimality problems. With a minor change to the proofs of [11], [12] to accommodate the absorbing product, it is easy to show that

$$(\text{s.INFL}(S) \vee (\text{INFL}(S) \wedge \text{s.INFL}(T))) \implies \text{s.INFL}(S \vec{\times}_{\bar{0}} T).$$

Since  $\text{s.INFL}(\text{sp}^+)$  this means that  $\mathcal{S}_m$  need only be inflationary. Therefore, let  $\text{INF}_m$  denote the set of all inflationary policy functions over  $[m]^\infty$ ,

$$\text{INF}_m = \{f \in [m]^\infty \rightarrow [m]^\infty \mid \forall a \in S, a \leq f(a)\}.$$

The (inflationary) strata algebra (an AME) is then defined as

$$\mathcal{S}_m = ([m]^\infty, \min, \text{INF}_m, \infty, 0).$$

Note that there are  $(m+1)!$  policies in  $\text{INF}_m$ .

Let's make sure we understand the definition of  $\text{strsp}_m$ . Metrics are of the form  $\infty$ , representing lack of reachability, or of the form  $(s, d)$ , where  $s \in [m]^\infty$  is a stratum and  $d \in \mathbb{N}$  is a distance. Such metrics are compared lexicographically with the stratum being most significant. Policy functions (arc weights) are represented as pairs,  $(f, w)$ , where  $f \in \text{INF}_m$  and  $w \in \mathbb{N}$  and  $0 < w$ . The policy application is defined as

$$\begin{aligned} (f, w)(s, d) &= \langle f(s), w + d \rangle, \\ (f, w)(\infty) &= \infty. \end{aligned}$$

Note the use of the absorbing product (defined in Section II-C). If either element of the pair becomes  $\infty$ , then the pair is replaced by  $\infty$ .

Figure 7 presents the sets  $\text{INF}_m$ , for  $m \in \{1, 2, 3\}$ . Functions  $f \in \text{INF}_m$  are represented as a vector

$$f = (x_0, x_1, \dots, x_{m-1})$$

where each  $x_i$  is in  $[m]^\infty$ , and  $f(s) = x_s$ . Since  $f(\infty) = \infty$  for all inflationary functions, this is left implicit. For example, for function  $\mathbf{q}$  of Figure 7 (c) we have

$$\begin{aligned} \mathbf{q}(0) &= 2 \\ \mathbf{q}(1) &= \infty \\ \mathbf{q}(2) &= 2 \\ \mathbf{q}(\infty) &= \infty. \end{aligned}$$

Figure 7 also contain information concerning some properties of each policy that are discussed in the next section.

Figure 8 presents two examples over  $\text{strsp}_3$ . The configurations of the two graphs are nearly the same — they differ only in the weight on the first link (2, 1). Yet this change results in very different best-path trees rooted at node 1.

|          | 0        | DIST | $K_\infty$ | $C_\infty$ |
|----------|----------|------|------------|------------|
| <b>a</b> | 0        | *    | *          | *          |
| <b>b</b> | $\infty$ | *    | *          | *          |

(a) Policy functions for one stratum,  $\text{INF}_1$ .

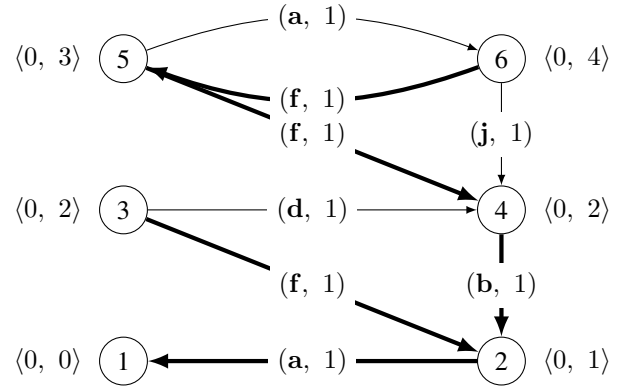
|          | 0        | 1        | DIST | $K_\infty$ | $C_\infty$ |
|----------|----------|----------|------|------------|------------|
| <b>a</b> | 0        | 1        | *    | *          |            |
| <b>b</b> | 0        | $\infty$ | *    | *          | *          |
| <b>c</b> | 1        | 1        | *    |            | *          |
| <b>d</b> | 1        | $\infty$ | *    | *          | *          |
| <b>e</b> | $\infty$ | 1        |      | *          | *          |
| <b>f</b> | $\infty$ | $\infty$ | *    | *          | *          |

(b) Policy functions for two strata,  $\text{INF}_2$

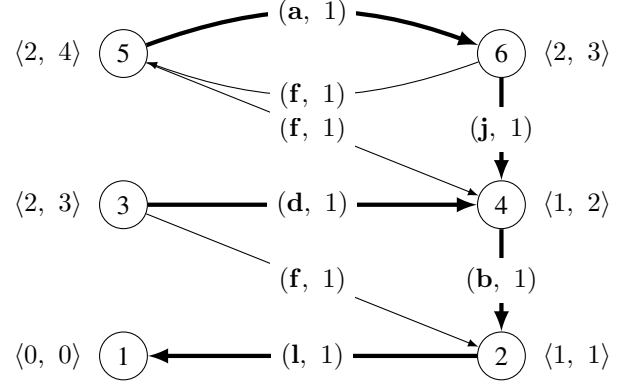
|          | 0        | 1        | 2        | DIST | $K_\infty$ | $C_\infty$ |
|----------|----------|----------|----------|------|------------|------------|
| <b>a</b> | 0        | 1        | 2        | *    | *          |            |
| <b>b</b> | 0        | 1        | $\infty$ | *    | *          |            |
| <b>c</b> | 0        | 2        | 2        | *    |            |            |
| <b>d</b> | 0        | 2        | $\infty$ | *    | *          |            |
| <b>e</b> | 0        | $\infty$ | 2        |      | *          |            |
| <b>f</b> | 0        | $\infty$ | $\infty$ | *    | *          | *          |
| <b>g</b> | 1        | 1        | 2        | *    |            |            |
| <b>h</b> | 1        | 1        | $\infty$ | *    |            | *          |
| <b>i</b> | 1        | 2        | 2        | *    |            |            |
| <b>j</b> | 1        | 2        | $\infty$ | *    | *          |            |
| <b>k</b> | 1        | $\infty$ | 2        |      | *          |            |
| <b>l</b> | 1        | $\infty$ | $\infty$ | *    | *          | *          |
| <b>m</b> | 2        | 1        | 2        |      |            |            |
| <b>n</b> | 2        | 1        | $\infty$ |      | *          |            |
| <b>o</b> | 2        | 2        | 2        | *    |            | *          |
| <b>p</b> | 2        | 2        | $\infty$ | *    |            | *          |
| <b>q</b> | 2        | $\infty$ | 2        |      |            | *          |
| <b>r</b> | 2        | $\infty$ | $\infty$ | *    | *          | *          |
| <b>s</b> | $\infty$ | 1        | 2        |      | *          |            |
| <b>t</b> | $\infty$ | 1        | $\infty$ |      | *          | *          |
| <b>u</b> | $\infty$ | 2        | 2        |      |            | *          |
| <b>v</b> | $\infty$ | 2        | $\infty$ |      | *          | *          |
| <b>w</b> | $\infty$ | $\infty$ | 2        |      | *          | *          |
| <b>x</b> | $\infty$ | $\infty$ | $\infty$ | *    | *          | *          |

(c) Policy functions for three strata,  $\text{INF}_3$

Fig. 7. The sets  $\text{INF}_m$  of increasing policy functions, for  $m \in \{1, 2, 3\}$ . The last entry of each policy is not shown since it is always  $\infty$ . For example, in table (c), **n** represents the policy vector  $(2, 1, \infty, \infty)$ . The set  $\text{INF}_m$  can be treated as a subset of  $\text{INF}_{m+1}$ . However, there is no relationship between these tables with respect to the (arbitrary) names assigned to policy functions.



(a) A weighted graph over  $\text{strsp}_3$ .



(b) Like (a), but the weight on arc  $(2,1)$  is changed from  $(a, 1)$  to  $(1, 1)$ .

Fig. 8. A best path trees rooted at node 1 are shown in bold. Note that  $(a, 1)(0, 0) = (0, 1)$ , whereas  $(1, 1)(0, 0) = (1, 1)$ . This makes a large impact on the best path tree rooted at 1.

#### IV. SUB-ALGEBRAS

Given that  $|\text{INF}_m| = (m+1)!$ , the policy space on  $m$  strata is quite large. In this section we investigate restricting this space to subsets of policies  $W \subseteq \text{INF}_m$ . The associated sub-algebras are denoted by

$$\mathcal{S}_m(W) = ([m]^\infty, \min, W, \infty, 0),$$

$$\text{strsp}_m(W) = \mathcal{S}_m(W) \vec{\times}_\infty \text{sp}^+.$$

Of course this gives rise to an *enormous* space of sub-algebras! For example, with  $m = 3$  there are  $2^{24}$  sub-algebras. Our challenge then is to identify subsets  $W$  so that the resulting sub-algebras are somehow useful or interesting. Some may be useful for solving particular routing problems, while others may prove useful in solving problems that have nothing whatsoever to do with routing!

If  $P$  is a property (such as DIST), then  $\mathbf{E}_m(P)$  represents the set of all policies in  $\text{INF}_m$  satisfying property  $P$ . Of course the algebra  $\mathcal{S}_m(\mathbf{E}_m(P))$  has property  $P$  by definition. However, note well that since the algebra  $\text{strsp}_m(\mathbf{E}_m(P))$  is constructed with the lexicographic product, it may fail to

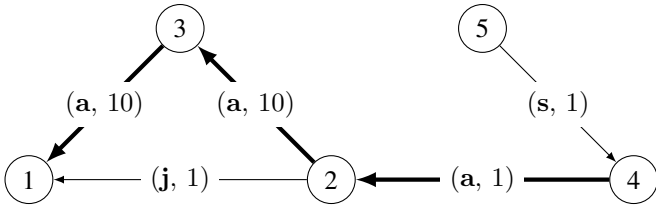


Fig. 9. Shortest paths with filters, over  $\mathbf{INF}_3$ . Note that the path 5, 4, 2, 1 with weight (1, 3) would be the globally best path from node 5 to node 1. But in this case, poor node 5 is left with no path! The locally optimal solution has  $\mathbf{R}(5, 1) = \infty$ .

have property P. But remember, by construction  $\text{strsp}_m(W)$  is always strictly inflationary, so locally optimal solutions to path problems can always be found (Section II-C).

### A. Almost shortest paths

Some policy functions in  $f_k \in \mathbf{INF}_m$  can be represented as  $f_k = (k, k+1, \dots, m-1, \infty, \infty, \dots, \infty)$ , and interpreted as the  $+k$  function. Since  $f_0, f_1, \dots, f_m \in \mathbf{INF}_m$ , we see that  $\text{sp}_m$  (defined in Figure 2) is contained in  $\mathcal{S}_m$  as a sub-algebra. For example,

|          | 0        | 1        | 2        | DIST | $K_\infty$ | interpretation |
|----------|----------|----------|----------|------|------------|----------------|
| <b>a</b> | 0        | 1        | 2        | *    | *          | +0             |
| <b>j</b> | 1        | 2        | $\infty$ | *    | *          | +1             |
| <b>r</b> | 2        | $\infty$ | $\infty$ | *    | *          | +2             |
| <b>x</b> | $\infty$ | $\infty$ | $\infty$ | *    | *          | +3             |

are the “additive” policy functions of  $\mathcal{S}_3$ . Note that this set results in a stratified shortest path algebra that is distributive.

We can extend this set to construct an algebra that might be interpreted as *shortest paths with filtering*. In  $\mathbf{INF}_3$ , these functions are

|          | 0        | 1        | 2        | DIST | $K_\infty$ | interpretation |
|----------|----------|----------|----------|------|------------|----------------|
| <b>a</b> | 0        | 1        | 2        | *    | *          | +0             |
| <b>j</b> | 1        | 2        | $\infty$ | *    | *          | +1             |
| <b>r</b> | 2        | $\infty$ | $\infty$ | *    | *          | +2             |
| <b>x</b> | $\infty$ | $\infty$ | $\infty$ | *    | *          | +3             |
| <b>b</b> | 0        | 1        | $\infty$ | *    | *          | filter 2       |
| <b>e</b> | 0        | $\infty$ | 2        | *    | *          | filter 1       |
| <b>f</b> | 0        | $\infty$ | $\infty$ | *    | *          | filter 1, 2    |
| <b>s</b> | $\infty$ | 1        | 2        | *    | *          | filter 0       |
| <b>t</b> | $\infty$ | 1        | $\infty$ | *    | *          | filter 0, 2    |
| <b>w</b> | $\infty$ | $\infty$ | 2        | *    | *          | filter 0, 1    |

Note that the associated stratified shortest paths problem admits only locally optimal solutions in general. Figure 9 presents an example over  $\mathbf{INF}_3$ . This example illustrates another important point concerning non-distributive path problems — a shortest path tree may not be a spanning tree even in a completely connected graph.

### B. Distributive sub-algebras

A interesting subset of  $\mathbf{INF}_m$  are those functions that have property DIST. Figure 10 presents  $\mathbf{E}_3(\text{DIST})$ . Observe that the set  $\mathbf{E}_m(\text{DIST})$  itself can be treated as a semiring! If

|          | 0        | 1        | 2        | $K_\infty$ | $C_\infty$ |
|----------|----------|----------|----------|------------|------------|
| <b>a</b> | 0        | 1        | 2        | *          |            |
| <b>b</b> | 0        | 1        | $\infty$ | *          |            |
| <b>c</b> | 0        | 2        | 2        |            |            |
| <b>d</b> | 0        | 2        | $\infty$ | *          |            |
| <b>f</b> | 0        | $\infty$ | $\infty$ | *          | *          |
| <b>g</b> | 1        | 1        | 2        |            |            |
| <b>h</b> | 1        | 1        | $\infty$ |            | *          |
| <b>i</b> | 1        | 2        | 2        |            |            |
| <b>j</b> | 1        | 2        | $\infty$ | *          |            |
| <b>l</b> | 1        | $\infty$ | $\infty$ | *          | *          |
| <b>o</b> | 2        | 2        | 2        |            | *          |
| <b>p</b> | 2        | 2        | $\infty$ |            | *          |
| <b>r</b> | 2        | $\infty$ | $\infty$ | *          | *          |
| <b>x</b> | $\infty$ | $\infty$ | $\infty$ | *          | *          |

Fig. 10. The set  $\mathbf{E}_3(\text{DIST})$  containing all distributive policies functions of Figure 7 (c).

we define  $f \min g$  to be the function such that  $(f \min g)(s) = f(s) \min g(s)$ , then

$$(\mathbf{E}_m(\text{DIST}), \min, \circ, (\infty, \infty, \dots, \infty), (0, 1, 2, \dots, m, \infty))$$

is a semiring (proof left to the reader).

Figure 11 presents an example a graph using this semiring. As with Figure 4, this semiring is associated with a partial order, and so no best-paths tree is shown. The values (functions in this case!) next to each node are the globally optimal values associated with paths to node 1. For example, the value shown at node 5 results from

$$(0, 2, \infty) = (0, \infty, \infty) \min(1, 2, \infty),$$

where  $(0, \infty, \infty) = w(5, 4, 2, 1)$  and  $(1, 2, \infty) = w(5, 6, 4, 2, 1)$ . How can we interpret the function  $(0, 2, \infty)$  at node 5? One possibility is to say that if node 1 originated a 0-value, node 5 would see a 0-value, if node 1 originated a 1-value, node 5 would see a 2-value, and if node 1 originated a 2-value, node 5 would see a  $\infty$  (no path). (Perhaps such *what-if* functions could actually be exploited in data networking.)

But the set  $\mathbf{E}_m(\text{DIST})$  does not give rise to a stratified shortest path sub-algebra that is distributive! From the rules of Section II-C we see this set of policies must be further restricted to those that are almost cancellative. Figure 12 presents these policy functions. Note that both examples of Figure 8 use this algebra. This may seem strange if our intuition is entirely based on shortest paths, where sub-paths of shortest paths are themselves shortest paths. In other words, distributivity alone is not enough to prevent this kind of behavior (an algebra needs to be cancellative).

### C. Autonomous routing — BGP revisited

BGP is the interdomain routing protocol used to maintain connectivity between autonomous networks in the global Internet [13], [18], [22]. BGP does not have any guarantees of convergence [16] — some routing policies can cause the BGP speakers to exchange routes forever without establishing



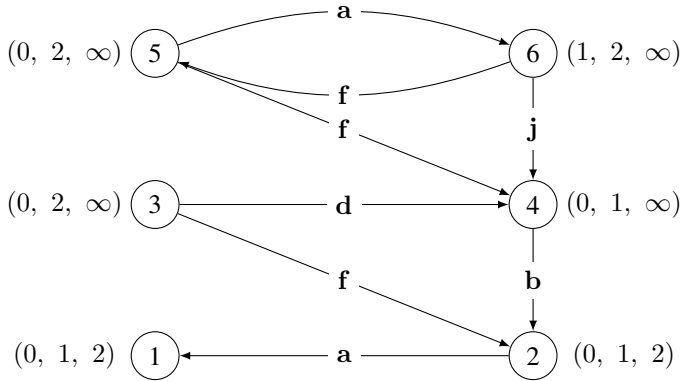


Fig. 11. An example using the semiring of distributive policy functions. This is essentially the configuration of Figure 8 (a), except that here policy functions are used as weights *and* values.

|          | 0        | 1        | 2        |
|----------|----------|----------|----------|
| <b>a</b> | 0        | 1        | 2        |
| <b>b</b> | 0        | 1        | $\infty$ |
| <b>d</b> | 0        | 2        | $\infty$ |
| <b>f</b> | 0        | $\infty$ | $\infty$ |
| <b>j</b> | 1        | 2        | $\infty$ |
| <b>l</b> | 1        | $\infty$ | $\infty$ |
| <b>r</b> | 2        | $\infty$ | $\infty$ |
| <b>x</b> | $\infty$ | $\infty$ | $\infty$ |

Fig. 12. Policy functions of Figure 7 (c) having both  $\text{DIST}$  and  $\text{K}_{\vec{\sigma}}$ . It follows from Section III that the stratified shortest paths sub-algebra  $\text{strsp}_m(\mathbf{E}_m(\text{DIST} \wedge \text{K}_{\vec{\sigma}}))$  is distributive.

a stable routing. However, the natural economic constraints of interdomain routing [14], [15] do seem to go a long way toward ensuring stable routing policies [4].

Sobrinho first modeled this algebraically [20], [21]. Suppose we have three strata  $\{0, 1, 2\}$ , interpreted as follows: 0 is the type of a *downstream* route, 1 is the type of a *peer* route, and 2 is the type of an *upstream* route. The policy functions for customer-peer-provider links can be taken from Figure 7 (c),

|          | 0 | 1        | 2        |
|----------|---|----------|----------|
| <b>f</b> | 0 | $\infty$ | $\infty$ |
| <b>l</b> | 1 | $\infty$ | $\infty$ |
| <b>o</b> | 2 | 2        | 2        |

Here **f** would be used on a *downstream* arc, **l** on a *peer* arc, and **o** on an *upstream* arc.

These simple policies are naturally seen as a subset of all *autonomous policy functions* — those functions having property  $\text{C}_{\infty}$ . Figure 13 lists these functions from  $\text{INF}_3$ . Why are these called autonomous<sup>1</sup> policy functions? Because in a distributed environment, these policies can be implemented without passing a stratum number “across the wire”. For

<sup>1</sup>While I am on the topic of autonomy, let me apologize for the confusing way in which this topic was treated in [9]. Happily, in the same session of that SIGCOMM, João Luís Sobrinho first mapped the way forward by sketching out an algebraic approach to modeling BGP [20].

|          | 0        | 1        | 2        | $\text{DIST}$ | $\text{K}_{\infty}$ |
|----------|----------|----------|----------|---------------|---------------------|
| <b>f</b> | 0        | $\infty$ | $\infty$ | *             | *                   |
| <b>h</b> | 1        | 1        | $\infty$ | *             |                     |
| <b>l</b> | 1        | $\infty$ | $\infty$ | *             | *                   |
| <b>o</b> | 2        | 2        | 2        | *             |                     |
| <b>p</b> | 2        | 2        | $\infty$ | *             |                     |
| <b>q</b> | 2        | $\infty$ | 2        |               |                     |
| <b>r</b> | 2        | $\infty$ | $\infty$ | *             | *                   |
| <b>t</b> | $\infty$ | 1        | $\infty$ |               | *                   |
| <b>u</b> | $\infty$ | 2        | 2        |               |                     |
| <b>v</b> | $\infty$ | 2        | $\infty$ |               | *                   |
| <b>w</b> | $\infty$ | $\infty$ | 2        |               | *                   |
| <b>x</b> | $\infty$ | $\infty$ | $\infty$ | *             | *                   |

Fig. 13. The policy functions of Figure 7 (c) having property  $\text{C}_{\infty}$ .

example, in  $\mathbf{E}_3(\text{C}_{\infty})$  we have

$$\mathbf{p} = (2, 2, \infty) = (2, 2, 2) \circ (1, 2, \infty),$$

which means that the one router can use  $(1, 2, \infty)$  as an export policy, while the other end of the link uses  $(2, 2, 2)$  as an import policy — the composition giving rise to the actual arc’s policy  $(2, 2, \infty)$ . This is exactly how pure BGP works (without communities) since the  $\text{LOCAL\_PREF}$  attribute is not sent over the wire and the abstract “policy on the arc” is implemented as the composition of export and import policies.

However, there is no reason to suppose that the full range of autonomous policies are not actually being used somewhere in the wide Internet. Each of them can be given a perfectly reasonable economic interpretation. For example,  $\mathbf{p} = (2, 2, \infty)$  could be used for some kind of discount provider that only gives me routes from its peers and customers, but none of its own upstream routes. And  $\mathbf{q} = (2, \infty, 2)$  could be a provider that does not provide me with its peer routes.

Note that there may be many interesting subsets of autonomous policies having interesting and distinct interpretations. For example, consider the following properties,

|          | 0        | 1        | 2        | $\text{DIST}$ | $\text{K}_{\infty}$ | $\text{C}_{\infty}$ |
|----------|----------|----------|----------|---------------|---------------------|---------------------|
| <b>f</b> | 0        | $\infty$ | $\infty$ | *             | *                   | *                   |
| <b>t</b> | $\infty$ | 1        | $\infty$ |               | *                   | *                   |
| <b>w</b> | $\infty$ | $\infty$ | 2        |               | *                   | *                   |
| <b>l</b> | 1        | $\infty$ | $\infty$ | *             | *                   | *                   |
| <b>r</b> | 2        | $\infty$ | $\infty$ | *             | *                   | *                   |
| <b>v</b> | $\infty$ | 2        | $\infty$ |               | *                   | *                   |

This might be interpreted as *an algebra of independent paths*.

#### D. Sub-strata

We could replace use of the strata algebra  $\mathcal{S}_m$  with the lexicographic product of two strata algebra

$$\mathcal{S}_m \vec{\times}_{\infty} \mathcal{S}_k$$

in order to model *strata with sub-strata*. Note that we are not really introducing anything new here since this product can be encoded as a sub-algebra of  $\mathcal{S}_{m \times k}$ . But it may be easier to

comprehend the rules when they are written explicitly using sub-strata. For example, the safe backup policies of [7] can be modeled as a sub-algebra of of  $\mathcal{S}_m \vec{\times}_\infty \mathcal{S}_3$ .

|          |                        |                            |                            |
|----------|------------------------|----------------------------|----------------------------|
|          | $\langle s, 0 \rangle$ | $\langle s, 1 \rangle$     | $\langle s, 2 \rangle$     |
| <b>D</b> | $\langle s, 0 \rangle$ | $\langle s + 1, 0 \rangle$ | $\langle s + 1, 0 \rangle$ |
| <b>P</b> | $\langle s, 1 \rangle$ | $\langle s + 1, 1 \rangle$ | $\langle s + 1, 1 \rangle$ |
| <b>U</b> | $\langle s, 2 \rangle$ | $\langle s, 2 \rangle$     | $\langle s, 2 \rangle$     |

The first primary strata acts as a counter for how many times the customer-provider-peer rules have been “violated”. Of course we could augment this collection of policies with various types of filters.

## V. OPEN PROBLEMS

This paper presents only a preliminary peek at the stratified shortest-paths problem. My hope is that at least a few readers will be inspired to look deeper. I’ll wrap up by presenting some open problems that seem worth pursuing.

**Complexity.** In general, we have very little understanding of the computational complexity of finding solutions for the locally optimal paths in strictly inflationary algebras. Perhaps the more restricted nature of the stratified shortest-paths problem will allow some progress to be made on this question. It is not clear that worst-case intractability would be entirely fatal since the global Internet routing system seems to be chugging along fairly well.

**New algorithms.** I have made every effort here to make a clear distinction between the specification of the problem being solved and algorithms used to solve a problem. In fact, for strictly inflationary, non-distributive algebras we know only algorithms derived from path-vectoring. Perhaps there exist many others for  $\text{strsp}_m$  or for special case  $\text{strsp}_m(W)$ , where  $W$  is a subset of the full set of  $(m + 1)!$  policy functions. Again, it is hoped that the concrete nature of the stratified shortest path problem might make it easier to make progress along new algorithmic avenues.

**Applications.** The stratified shortest-paths problem arose from models of inter-domain routing, but I suspect that there may be many applications far removed from networking that can be modeled with such algebraic structures. There may be many interesting sub-languages lurking within  $\text{INF}_n$ .

**Interdomain Stratigraphy.** Some researchers have attempted to infer AS-path relationships from collected data, (for example [17]). This work seems to assume that all the ASes are using a very simple set of policies. But how can we tell if this is true? The LOCAL\_PREF attribute is a 32 bit quantity, so in the worst case we have to consider  $\text{INF}_{2^{32}}$ , which contains  $2^{32}!$  policies — and with the use of communities, each of these policies is actually implementable in BGP.

It would be interesting to know the number of distinct LOCAL\_PREF values used by each AS in the public Internet. Call the maximum value among these  $k$ . If we were omniscient and could peek inside of every router configuration in the world, then we would probably do a good job of modeling

BGP using a subset of  $\text{INF}_k$  (ignoring MED and other traffic engineering attributes). But how is the inference problem impacted by  $k$ ? What is the difference between inference for  $\text{INF}_k$  and  $\text{INF}_{k+1}$ ?

## ACKNOWLEDGEMENTS

I would first like to thank the organizers of COMSNETS 2010 for inviting me to present this paper. Some of the ideas in the paper arose first while I was presenting a tutorial at the University of Carlos III, in Madrid. Discussions there with Iljitsch van Beijnum were especially helpful. Thanks also to M. Abdul Alim, Arthur Azevedo de Amorim, John Billings, Jon Crowcroft, Marcelo Fiore, Alexander Gurney, Vilius Naudžiūnas, João Sobrinho, and Philip Taylor for their helpful discussions and feedback. Finally, I am grateful for financial support from EPSRC (grant EP/F002718/1), Boeing, and Cisco Systems.

## REFERENCES

- [1] B. Carré. *Graphs and Networks*. Oxford University Press, 1979.
- [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] L. Gao and J. Rexford. Stable internet routing without global coordination. *IEEE/ACM Transactions on Networking*, pages 681–692, December 2001.
- [5] M. Gondran and M. Minoux. *Graphs and Algorithms*. Wiley, 1984.
- [6] M. Gondran and M. Minoux. *Graphs, Dioids, and Semirings : New Models and Algorithms*. Springer, 2008.
- [7] T. G. Griffin, L. Gao, and J. Rexford. Inherently safe backup routing with BGP. In *Proc. IEEE INFOCOM*, April 2001.
- [8] 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.
- [9] T. G. Griffin, A. D. Jaggard, and V. Ramachandran. Design principles of policy languages for path vector protocols. In *Proc. ACM SIGCOMM*, 2003.
- [10] 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.
- [11] A. Gurney. Construction and verification of routing algebras. PhD dissertation, 2009.
- [12] A. J. T. Gurney and T. G. Griffin. Lexicographic products in metarouting. In *Proc. Inter. Conf. on Network Protocols*, October 2007.
- [13] S. Halabi and D. McPherson. *Internet Routing Architectures*. Cisco Press, second edition, 2001.
- [14] G. Huston. Interconnection, peering and settlements: Part I. *Internet Protocol Journal*, 2(1), June 1999.
- [15] G. Huston. Interconnection, peering and settlements: Part II. *Internet Protocol Journal*, 2(2), June 1999.
- [16] K. Varadhan, R. Govindan, and D. Estrin. Persistent route oscillations in inter-domain routing. *Computer Networks*, 32:1–16, 2000.
- [17] Z. M. Mao, L. Qiu, J. Wang, and Y. Zhang. On as-level path inference. In *Proc. ACM SIGMETRICS*, 2005.
- [18] Y. Rekhter, T. Li, and S. Hares. A Border Gateway Protocol 4 (BGP-4). RFC 4271, January 2006.
- [19] 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.
- [20] J. L. Sobrinho. Network routing with path vector protocols: Theory and applications. In *Proc. ACM SIGCOMM*, September 2003.
- [21] J. L. Sobrinho. An algebraic theory of dynamic network routing. *IEEE/ACM Transactions on Networking*, 13(5):1160–1173, October 2005.
- [22] I. van Beijnum. *BGP : Building Reliable Networks with the Border Gateway Protocol*. O’Reilly, 2002.