On the Interaction of Multiple Routing Algorithms

M. Abdul Alim
Lancaster University
a.alim@lancaster.ac.uk

Timothy G. Griffin
University of Cambridge
timothy.griffin@cl.cam.ac.uk

ABSTRACT

Internet routers can obtain distinct routes to the same destination from multiple routing protocols. A mechanism called *administrative distance* (AD) has evolved to implement route selection in such cases. AD simply imposes a preference ranking among routing protocols. A related mechanism called *route redistribution* (RR) is used to increase connectivity by injecting routes from one protocol into another. Recent research has shown that the current design of RR with AD mechanisms can give rise to routing and forwarding anomalies that are difficult to debug. We present a new algebraic model that captures not only RR with AD, but also sub-protocol interactions that exist within protocols such as OSPF and IS-IS. Compared with previous work, our model provides a clearer view of the issues and trade-offs involved.

Protocol	Cisco	Juniper
Connected interface	0	0
Static route	1	1
EIGRP summary route	5	
External BGP	20	20
Internal EIGRP	90	
IGRP	100	
OSPF	110	110
IS-IS	115	115
RIP	120	120
EGP	140	
ODR	160	
External EIGRP	170	
Internal BGP	200	200

Figure 1: Common administrative distances.

1. MOTIVATION

If a router calculates more than one route to the same destination using distinct routing protocols, then some mechanism is needed to determine which routes are actually used for forwarding. The most common mechanism used today is called *administrative distance* (AD). This means that each protocol (or sub-protocol) is associated with a weight, and routes from the protocol with the lowest weight are selected. The default configurations of AD values for two vendors are shown in Figure 1. So for example, both Cisco and Juniper routers will by default prefer routes from OSPF over routes from RIP.

Each routing protocol receives routing information from adjacent routers (details vary between protocols) and constructs a Routing Information Base (RIB). Then each protocol exports its best routes to a Forwarding Information Base (FIB) manager. The FIB manager implements route selection based on AD values and installs forwarding entries into the operational forwarding table.

The FIB manager also implements the inter-protocol mechanism of *route redistribution* (RR). That is, routes can be exported from one protocol into other protocols as implemented by router configuration commands. The complexity of configuration and management in a network using RR/AD has been the subject of recent research [10, 11]. This work shows how today's RR/AD mechanisms can give rise to routing and forwarding anomalies that are difficult to debug, including forwarding loops, protocol oscillation, and unexpected loss of connectivity. Motivated by these problems, a recent paper [12] proposed New Routing Primitives (NRP) that might be used to improve the reliability and robustness of RR/AD.

In the current paper we attempt to generalize the NRP scheme as well as approach the problem in a more top-down manner. Broadly speaking there are two approaches to designing a system for RR/AD:

- **Tightly coupled approach**: Use protocol-specific

ACM CoNEXT 2011, December 6–9 2011, Tokyo, Japan.
Copyright 2011 ACM 978-1-4503-1041-3/11/0012 ...$10.00.

details to formulate a model.

- **Loosely coupled approach**: Construct a model that works for *any* routing protocols.

One possible advantage of a tightly coupled approach is that it may be able provide more guarantees. However, in an environment like the Internet this would place a tremendous burden on protocol designers. For example, the introduction of a new protocol might require modifications to all existing protocols to ensure compatibility with RR/AD. Of course this would be avoided with a loosely coupled approach. The downside of a loosely coupled design is that the entire routing system could be corrupted by one misbehaving protocol. Having said that, it seems that this is the only scalable approach for an environment such as the Internet. We will argue that NRP represents a tightly coupled approach.

Section 2 reviews some of the algebraic preliminaries required in the paper. An advantage of a top-down algebraic approach is that it allows a complete separation between *what* set of equations are being solved and *how* they are solved using specific routing algorithms (link state, path vector, FIB mechanisms, and so on).

Section 3 reviews the algebraic distinction between *path metrics* and *routing metrics* recently presented in [4]. This approach echoes the idea of having distinct name spaces for infrastructure (routers) and destinations (a range of IP addresses) [5] which has recently been embodied in the Locator/ID split work [14]. Put simply, path metrics are used to select paths between routers, while *mapping* attaches destinations to create routes, which may have a metric distinct from the path metric. Routes then results from a combination of path finding and mapping. The distinction between path and route metrics plays a key role in our algebraic model of RR/AD.

Section 4 reviews the distinction between global and local optimality in Internet routing protocols. It is well known that protocols such as EIGRP [8] BGP [18] find only *locally optimal* routes, not globally optimal routes. This is related to the fact that the metrics of these protocols violate some of the algebraic rules described in Section 2. We show that this can complicate the interaction of protocols in a manner not discussed in the NRP paper.

Section 5 shows how similar problems arise *within* some of today's routing protocols — the interaction of intra-area and inter-area routing in OSPF and Level 1 and Level 2 routing in IS-IS are shown to replicate many aspects of RR with AD in miniature. However, the internal design of each of these protocols is by its very nature tightly coupled.

Section 6 extends the routing and forwarding model of [4] to capture today's RR *with* AD. The conclusions are both positive and negative. On the positive side we argue that many of the design decisions made in today's RR/AD framework are reasonable. On the negative side, it seems difficult to avoid the fact that a stable forwarding solutions may depend on the details of the algorithms used (path-vector vs. link-state) by the participating protocols.

name	S	$\oplus$	$\otimes$	$\overline{0}$	$\overline{1}$	in network
sp	$\mathbb{N}^\infty$	min	$+$	∞	0	shortest paths
bw	$\mathbb{N}^\infty$	max	min	0	∞	greatest capacity
rel	[0, 1]	max	$\times$	0	1	greatest reliability

Figure 2: Three simple semirings. Here $\mathbb{N}^\infty$ represents $\mathbb{N} \cup \{\infty\}$, where ∞ is used to denote the absence on arc or a path.

In Section 7 we examine the NRP scheme. The NRP paper seems to argue that only a tightly coupled design can address the issues raised by [10, 11]. We argue for a loosely coupled approach.

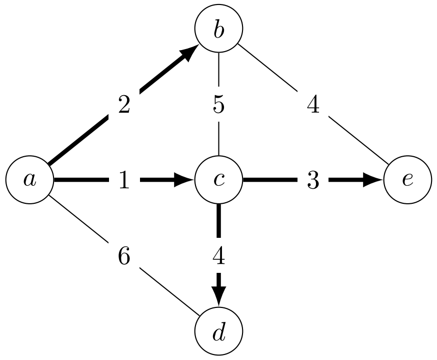
Among other things, we believe the paper reveals some of the complexities associated with Internet routing protocols. Section 8 considers possible applications to protocol design as well as open problems.

2. ALGEBRAIC PATH PROBLEMS

The algebraic approach to describing and solving path problems in graphs is well developed in Operations Research where it is applied to wide variety of problems including scheduling and planning. An excellent survey can be found in [7]. Algebraic models can also capture a broad range of routing metrics. We recommend [3] as an introduction to algebraic techniques for network routing.

The classical shortest-paths problem can be generalized to a large class of algebraic structures called *semirings*, of the form $(S,\ \oplus,\ \otimes,\ \overline{0},\ \overline{1})$. Figure 2 presents three simple semirings. Semirings must obey the axioms of Figure 3. For the classical shortest-paths semiring the additive operation is $\oplus = \min$ and the multiplicative operation is $\otimes = +$. The additive identity is $\overline{0} = \infty$, while the multiplicative identity is $\overline{1} = 0$. Thus the standard abstract notations may seem strangely perverse, until we realize that, historically, this derives from viewing semirings as generalizing *rings*, and the classical ring from linear algebra is $(\mathbb{R}, +, \times, 0, 1)$. Indeed, many path finding algorithms using semirings have analogs in linear algebra. For example, Bellman's algorithm is closely related to Jacobi's method from linear algebra ([7], Chapter 4).

Given a semiring $(S, \oplus, \otimes, \overline{0}, \overline{1})$ and a graph $G = (V,\ E)$, a *weight function* for G is a mapping $w \in E \rightarrow S$. We will represent this function with an *adjacency*



(a) Shortest paths from node a in bold

$$\mathbf{A} = \begin{array}{c|ccccc} & a & b & c & d & e \\ \hline a & \infty & 2 & 1 & 6 & \infty \\ b & 2 & \infty & 5 & \infty & 4 \\ c & 1 & 5 & \infty & 4 & 3 \\ d & 6 & \infty & 4 & \infty & \infty \\ e & \infty & 4 & 3 & \infty & \infty \end{array}$$

(b) Adjacency matrix

$$\mathbf{A}^* = \begin{array}{c|ccccc} & a & b & c & d & e \\ \hline a & 0 & 2 & 1 & 5 & 4 \\ b & 2 & 0 & 3 & 7 & 4 \\ c & 1 & 3 & 0 & 4 & 3 \\ d & 5 & 7 & 4 & 0 & 7 \\ e & 4 & 4 & 3 & 7 & 0 \end{array}$$

(c) Solution matrix

Figure 4: Example of finding best paths using the sp semiring. The bold arrows indicate the shortest-paths tree rooted at node a.

(SR.A.ASSO)	$a \oplus (b \oplus c) = (a \oplus b) \oplus c$
(SR.A.COMM)	$a \oplus b = b \oplus a$
(SR.A.ID)	$a \oplus \overline{0} = \overline{0} \oplus a = a$
(SR.M.ASSO)	$a \otimes (b \otimes c) = (a \otimes b) \otimes c$
(SR.M.ID)	$a \otimes \overline{1} = \overline{1} \otimes a = a$
(SR.M.ANNI)	$a \otimes \overline{0} = \overline{0} \otimes a = \overline{0}$
(SR.L.DIST)	$a \otimes (b \oplus c) = (a \otimes b) \oplus (a \otimes c)$
(SR.R.DIST)	$(b \oplus c) \otimes a = (b \otimes a) \oplus (c \otimes a)$

Figure 3: Semiring (sr) axioms for structures of the form $(S, \oplus, \otimes, \overline{0}, \overline{1})$. All free variables are universally quantified over S. Abbreviations: additive (a), multiplicative (m), associative (asso), commutative (comm), identity (id), annihilator (anni), left (l), right (r), distributive (dist).

matrix $\mathbf{A}$, where $\mathbf{A}(u,v) = w(u,v)$ when $(u,v) \in E$ and $\mathbf{A}(u,v) = \overline{0}$ when $(u,v) \notin E$. If $p = v_0, v_1, \cdots, v_k$ is a path in G, then the weight of p, denoted $w(p)$, is the product of arc weights,

$$w(v_o, v_1) \otimes w(v_1, v_2) \otimes \cdots \otimes w(v_{k-1}, v_k).$$

The "empty path" from a node v to itself using no arcs is given the weight $\overline{1}$.

For most semirings useful in networking, and all of the examples in Figure 2, the $\oplus$ operation is *idempotent* — that is, $\forall s \in S : s \oplus s = s$. In such cases, the relation defined as

$$a \leq b \;\equiv\; a = a \oplus b \tag{1}$$

is a partial order. When $\oplus$ is also *selective* (i.e., $\forall s, t \in S : s \oplus t \in \{s,\ t\}$), then $\oplus$ represents a total order (again, this is true of all examples in Figure 2).

2.1 The all-pairs problem

Given a weighted graph, the *all-pairs best-paths problem* is to find the best path weight over all possible paths from i to j for all $i, j \in V$. Put another way, we would like to find a matrix $\mathbf{A}^*$ such that

$$\mathbf{A}^*(i,\ j) = \bigoplus_{p \in \mathcal{P}(i,\ j)} w(p). \tag{2}$$

where $\mathcal{P}(i,\ j)$ is the set of all paths from node i to node j. Note that in general the $\mathbf{A}^*$ may not exist, or may not be computable. We can read Equation 2 as seeking *globally optimal* paths weights, since $\oplus$ represents minimization with respect to $\leq$.

Figure 4(a) presents a simple example using the shortest-paths semiring **sp**. This graph is represented with the adjacency matrix $\mathbf{A}$ in Figure 4(b). The matrix $\mathbf{A}^*$ is presented in Figure 4(c).

Any semiring S can be used to define a semiring of $n \times n$ matrices over S. If $\mathbf{X}$ and $\mathbf{Y}$ are two matrices, then

$$(\mathbf{X} \oplus \mathbf{Y})(i,j) = \mathbf{X}(i,\ j) \oplus \mathbf{Y}(i,\ j), \tag{3}$$

and

$$(\mathbf{X} \otimes \mathbf{Y})(i,j) = \bigoplus_{1 \leq q \leq n} \mathbf{X}(i,q) \otimes \mathbf{Y}(q,j). \tag{4}$$

We often write $\mathbf{XY}$ instead of $\mathbf{X} \otimes \mathbf{Y}$.

The matrix $\mathbf{A}^*$ can also be specified as the solution of the matrix equation

$$\mathbf{X} = \mathbf{AX} \oplus \mathbf{I}. \tag{5}$$

With semirings, $\mathbf{X} = \mathbf{A}^*$ is a in some sense the best solution to Equation 5 (see [7]).

2.2 Algorithms

Letting $\mathbf{A}^k$ represent k-fold matrix product of $\mathbf{A}$, it is not difficult to show that $\mathbf{A}^k(i,\ j)$ is the best weight over all paths from i to j having exactly k arcs. Therefore, when $\mathbf{A}^*$ exists it can be expressed as

$$\mathbf{A}^* = \mathbf{I} \oplus \mathbf{A} \oplus \mathbf{A}^2 \oplus \cdots \oplus \mathbf{A}^k \oplus \cdots \qquad (6)$$

For many semirings useful in networking (and all of the examples in Figure 2) the multiplicative identity is also an annihilator for $\oplus$:

$$(\textsc{sr.a.anni}) \qquad s \oplus \overline{1} = \overline{1} \oplus s = \overline{1}$$

In this case we need not inspect paths containing loops, so

$$\mathbf{A}^* = \mathbf{I} \oplus \mathbf{A} \oplus \mathbf{A}^2 \oplus \cdots \oplus \mathbf{A}^{n-1}. \qquad (7)$$

There are many algorithms for computing $\mathbf{A}^*$ more efficiently.

The matrix $\mathbf{A}^*$ can be computed with distributed algorithms. With **link-state** protocols each node j knows $\mathbf{A}(j, \_)$ and this information is flooded throughout the network. Eventually, each node i learns $\mathbf{A}$ and then computes one row $\mathbf{A}^*(i,\ \_)$ of $\mathbf{A}^*$ using an algorithm such as Dijkstra's.

Another approach is to distribute the computation, keeping data local, which typically employs algorithms in the Bellman-Ford family often called **distance vector** or **path vector** algorithms. At a high level of abstraction, we can view these algorithms as implementing an iterative method of computing $\mathbf{A}^*$ from Equation 7:

$$\begin{aligned} \mathbf{A}^{[0]} &= \mathbf{I} \\ \mathbf{A}^{[k+1]} &= \mathbf{A}\mathbf{A}^{[k]} \oplus \mathbf{I}. \end{aligned}$$

It is easy to see, courtesy of the distributivity axioms, that $\mathbf{A}^* = \mathbf{A}^{[n-1]}$ (assuming $\textsc{sr.a.anni}$). Note that for $i \neq j$ we have

$$\mathbf{A}^{[k+1]}(i,\ j) = \bigoplus_q \mathbf{A}(i,\ q)\mathbf{A}^{[k]}(q, j),$$

which says that at each iteration node i constructs the best paths it can given the best paths at its neighbors. Of course this abstract view of vectoring protocols ignores many important implementation details (session management, hard-state vs. soft state, and so on). In particular, each algorithm can be modified to record path information, which we are ignoring here. For example, Internet routing protocols compute *next-hop* routers for best paths.

Algorithms in the distributed Bellman-Ford family exhibit a problem often referred to as *counting to infinity.* This can be easily explained algebraically. Suppose the Bellman-Ford iteration starts in an arbitrary state represented by the matrix $\mathbf{Z}$,

$$\begin{aligned} \mathbf{A}_{\mathbf{Z}}^{[0]} &= \mathbf{Z} \\ \mathbf{A}_{\mathbf{Z}}^{[k+1]} &= \mathbf{A}\mathbf{A}_{\mathbf{Z}}^{[k]} \oplus \mathbf{I}. \end{aligned}$$

By induction we can see that for all k we have

$$\mathbf{A}_{\mathbf{Z}}^{[k+1]} \ = \ \mathbf{A}^k\mathbf{Z} \ \oplus \ \mathbf{A}^{[k]},$$

and so q such that $n \leq q$ we have

$$\mathbf{A}_{\mathbf{Z}}^{[q]} \ = \ \mathbf{A}^{q-1}\mathbf{Z} \ \oplus \mathbf{A}^*.$$

Now suppose that $\mathbf{Z}$ represents the best paths of the network just before some link failures, and $\mathbf{A}$ is the new adjacency matrix after the failures. It could be that there are two nodes i and j such that $\mathbf{A}^*(i,\ j)$ is much larger than $\mathbf{A}^{q-1}\mathbf{Z}(i,\ j)$, so the iteration will continue until $\mathbf{A}^*(i, j)$ wins. This may never happen if the failures partitioned the network and i and j are no longer connected.

One solution is to make infinity very small, as with RIP. Another approach is to add paths (such as with BGP's ASPATH) and modify the algorithm to only consider loop-free paths. This works because for $n - 1 < k$ the each entry in the matrix $\mathbf{A}^k$ represents the weight of a path with a loop. A third approach is used in Cisco's EIGRP — the iteration is performed with additional communication between neighbors to ensure only consistent states are explored [6].

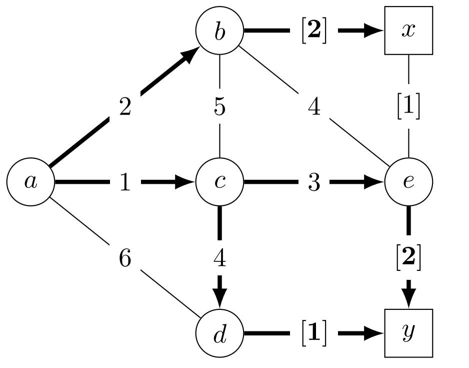
3. PATH VS. ROUTING METRICS

Close inspection of Internet routing protocols reveals that they often employ *distinct* metrics for selecting *paths* and for selecting *routes.* This is especially true for current link-state protocols. However, ongoing work in the IETF associated with the Locator/ID Separation Protocol (LISP) is attempting to introduce the path/route distinction into BGP (see [14] for an introduction). For current link-state protocols the interaction of path and route selection is described informally in an RFC or buried in code. We need to make this interaction explicit in a way that integrates well with an algebraic theory of routing. We adopt a solution recently proposed [4] which employs *semi-modules* to model this distinction.

Consider the network of Figure 5 where two external destinations, x and y, are attached to the graph of Figure 4. Using the terminology of LISP, it might be helpful to think of x and y as *identifiers* and the node names $\{a, b, c, d, e\}$ as *locators.* Each attachment is associated with a *mapping metric* m, which is presented as $[m]$ to distinguish it from the routing metric.

Figure 5(a), (b), and (c) present the simplest case where attachment metrics are integers used simply to extend shortest-paths weights. Figure 5(a) shows the shortest paths in bold arrows from node a, while Figure 5(b) captures the associated path weights in matrix form. Finally, Figure 5(c) presents the (network-wide) forwarding tables associated with these paths. For example, the forwarding table at node c would traditionally be presented as

destination	next-hop(s)
x	e
y	$d,\ e$



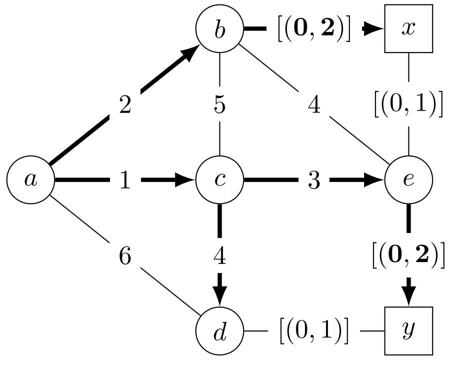
(a) Simple paths from node a

$$\mathbf{R}_{\mathrm{sim}} = \begin{array}{c} \\ a \\ b \\ c \\ d \\ e \end{array} \begin{array}{c} \begin{array}{cc} x & y \end{array} \\ \left[\begin{array}{cc} 4 & 6 \\ 2 & 6 \\ 4 & 5 \\ 8 & 1 \\ 1 & 2 \end{array} \right] \end{array}$$

(b) Matrix of simple routing metrics

$$\mathbf{T}_{\mathrm{sim}} = \begin{array}{c|cc} & x & y \\ a & \{b\} & \{c\} \\ b & \{x\} & \{e\} \\ c & \{e\} & \{d,\ e\} \\ d & \{c\} & \{y\} \\ e & \{x\} & \{y\} \end{array}$$

(c) Simple forwarding tables (next-hops)



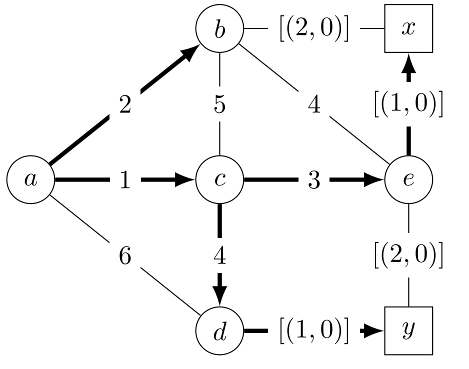
(d) Hot-potato paths from node a

$$\mathbf{R}_{\mathrm{hot}} = \begin{array}{c|cc} & x & y \\ a & (2,2) & (4,2) \\ b & (0,2) & (4,2) \\ c & (3,1) & (3,2) \\ d & (7,1) & (0,1) \\ e & (0,1) & (0,2) \end{array}$$

(e) Matrix of hot-potato routing metrics

$$\mathbf{T}_{\mathrm{hot}} = \begin{array}{c|cc} & x & y \\ a & \{b\} & \{c\} \\ b & \{x\} & \{e\} \\ c & \{e\} & \{e\} \\ d & \{c\} & \{y\} \\ e & \{x\} & \{y\} \end{array}$$

(f) Hot-potato forwarding tables (next-hops)



(g) Cold-potato paths from node a

$$\mathbf{R}_{\mathrm{cold}} = \begin{array}{c|cc} & x & y \\ a & (1,4) & (1,5) \\ b & (1,4) & (1,7) \\ c & (1,3) & (1,4) \\ d & (1,7) & (1,0) \\ e & (1,0) & (1,7) \end{array}$$

(h) Matrix of cold-potato routing metrics

$$\mathbf{T}_{\mathrm{cold}} = \begin{array}{c|cc} & x & y \\ a & \{c\} & \{c\} \\ b & \{e\} & \{a\} \\ c & \{e\} & \{c\} \\ d & \{c\} & \{y\} \\ e & \{x\} & \{c\} \end{array}$$

(i) Cold-potato forwarding tables (next-hops)

Figure 5: Illustrating the difference between path metrics and routing metrics. Destinations x and y are attached to the graph of Figure 4 and three different routing and forwarding tables are constructed from the same set of shortest paths. In each scenario routes from node a to destinations x and y, are shown in bold arrows. The matrix of routing metrics $\mathbf{R}$ can be specified as $\mathbf{R} = \mathbf{A}^* \rhd \mathbf{M}$ using an appropriately defined semi-module $(U, \Box, \rhd, \hat{0})$ over the semiring sp. Here M is an appropriately constructed *mapping matrix* representing the attachment of destinations x and y to the graph. (The definitions can be found in Figure 6.) Section 3.1 shows that the matrix of routing metrics can be defined as the solution to the matrix equation $\mathbf{R} = (\mathbf{A} \rhd \mathbf{R}) \Box \mathbf{M}$. Section 5 explains how OSPF and IS-IS each employ similar techniques.

	Simple	Hot-potato	Cold-potato
$(U,\ \hat{0})$	$(\mathbb{N}^{\infty},\ \infty)$	$((\mathbb{N}\times\mathbb{N})\cup\{\infty\},\ \infty)$	$((\mathbb{N}\times\mathbb{N})\cup\{\infty\},\ \infty)$
$\square$	min	$\min \vec{\times} \min$	$\min \vec{\times} \min$
$\rhd$	$+$	$s \rhd_{\text{hot}} (t,\ u) = (s+t,\ u)$	$s \rhd_{\text{cold}} (u,\ t) = (u,\ s+t)$
$\mathbf{M}$	$\mathbf{M}_{\text{sim}} = \begin{array}{cc} & \begin{array}{cc} x & y \end{array} \\ \begin{array}{c} a \\ b \\ c \\ d \\ e \end{array} & \left[\begin{array}{cc} \infty & \infty \\ 2 & \infty \\ \infty & \infty \\ \infty & 1 \\ 1 & 2 \end{array}\right] \end{array}$	$\mathbf{M}_{\text{hot}} = \begin{array}{cc} & \begin{array}{cc} x & y \end{array} \\ \begin{array}{c} a \\ b \\ c \\ d \\ e \end{array} & \left[\begin{array}{cc} \infty & \infty \\ (0,2) & \infty \\ \infty & \infty \\ \infty & (0,1) \\ (0,1) & (0,2) \end{array}\right] \end{array}$	$\mathbf{M}_{\text{cold}} = \begin{array}{cc} & \begin{array}{cc} x & y \end{array} \\ \begin{array}{c} a \\ b \\ c \\ d \\ e \end{array} & \left[\begin{array}{cc} \infty & \infty \\ (2,0) & \infty \\ \infty & \infty \\ \infty & (1,0) \\ (1,0) & (2,0) \end{array}\right] \end{array}$

Figure 6: Semi-modules and mapping matrices for each scenario of Figure 5.

(LSM.A.ASSO)	$u \square (v \square w) = (u \square v) \square w$
(LSM.A.COMM)	$u \square v = u \square w$
(LSM.A.ID)	$u \square \hat{0} = \hat{0} \square u = u$
(LSM.M.ASSO)	$(a \otimes b) \rhd u = a \rhd (b \rhd u)$
(LSM.M.ID)	$\overline{1} \rhd u = u$
(LSM.M.ANNI)	$a \rhd \hat{0} = \overline{0} \rhd u = \hat{0}$
(LSM.L.DIST)	$a \rhd (u \square v) = (a \rhd u) \square (a \rhd v)$
(LSM.R.DIST)	$(a \oplus b) \rhd u = (a \rhd u) \square (b \rhd u)$

Figure 7: Axioms for left semi-module (sr) of the form $(U, \square, \rhd, \hat{0})$ over a semiring $(S, \oplus, \otimes, \overline{0}, \overline{1})$. All axioms are universally quantified over $a, b \in S$ and $u, v, w \in U$.

Figure 5(d), (e), and (f) present a more interesting and very familiar scenario called *hot-potato* routing. Here routing metrics are of the form $(s,\ u)$, where s represents a path metric and u represents an attachment metric. Such metrics are compared *lexicographically*, with path metric being more significant. For example, consider how node c selects a route to destination x. The routing metric associated with the route from c to x via b is $(3, 2)$, while the routing metric associated with the route from c to x via e is $(3, 1)$. Since $(3, 1) < (3, 2)$ lexicographically, node e is c's hot-potato egress point to destination x.

Figure 5(g), (h), and (i) present *cold-potato* routing, where routing metrics are of the form $(u,\ s)$, again compared lexicographically left-to-right. In this case, the attachment metric is more significant than the path metric. For example, to reach destination x every node v will construct a route metric of the form $(1,\ s_v)$, where s_v is the shortest path weight from v to e.

The main idea of [4] is that the matrix of routing metrics $\mathbf{R}$, such as those of Figure 5, can be described using *semi-modules*. Given a semiring S of path metrics, a (left) semi-module over S is a structure of the form $(U, \square, \rhd, \hat{0})$ that satisfies the axioms of Figure 7. Just as semirings generalize rings, semi-modules over semirings generalize modules over rings, which in turn generalize vector spaces over the classical ring $(\mathbb{R}, +, \times, 0, 1)$. In that familiar case, U is the set of n-vectors over $\mathbb{R}$, $\square$ is vector addition, and $\rhd$ is scalar multiplication (the axioms are explained in [7], but not applied to routing).

As a semiring S can be lifted to a semiring of $n \times n$-matrices, so a semi-module U over S can be lifted to the set of $n \times m$-matrices over U, and this turns out to be another semi-module over the semiring of matrices. Lifting $\square$ to matrices is point-wise as with lifting $\oplus$. If $\mathbf{A}$ is an $n \times n$-matrix over S and $\mathbf{M}$ is an $n \times m$-matrix over U, then $\mathbf{A} \rhd \mathbf{M}$ is an $n \times m$-matrix over U defined as

$$(\mathbf{A} \rhd \mathbf{M})(i,\ d) = \mathop{\Large\square}_{1 \le q \le n} \mathbf{A}(i,\ q) \rhd \mathbf{M}(q,\ d). \qquad (8)$$

With the appropriately defined semi-module, each matrix of routing metrics of Figure 5 can be defined as

$$\mathbf{R} = \mathbf{A}^* \rhd \mathbf{M}, \qquad (9)$$

where $\mathbf{A}$ is the adjacency matrix (over semiring S) for the router graph, and $\mathbf{M}$ is an appropriately constructed *mapping* matrix that captures the attachment of destinations to the router graph. From Equation 8, this gives us

$$\mathbf{R}(i,\ d) = \mathop{\Large\square}_{1 \le q \le n} \mathbf{A}^*(i,\ q) \rhd \mathbf{M}(q,\ d). \qquad (10)$$

Note that the best *routes* are selected using $\square$, not $\oplus$. If an egress node q is associated with the best routing metric $\mathbf{R}(i,\ d)$, then we can be sure that the best paths, according to $\mathbf{A}^*$, are used to reach q.

Figure 6 presents the semi-modules and mapping matrices for each of the three scenarios of Figure 5. For example, $\mathbf{R}_{\text{cold}} = \mathbf{A}^* \rhd_{\text{cold}} \mathbf{M}_{\text{cold}}$ can be easily verified.

3.1 Algorithms

The matrix of routing metrics can be specified as the solution to the following matrix equation.

$$\mathbf{R} = (\mathbf{A} \rhd \mathbf{R}) \square \mathbf{M}. \qquad (11)$$

It is not hard to show that this equation is solved by $\mathbf{R} = \mathbf{A}^* \rhd \mathbf{M}$, assuming $\mathbf{A}^*$ exists:

$$\begin{array}{cll} & (\mathbf{A} \rhd (\mathbf{A}^* \rhd \mathbf{M})) \,\Box\, \mathbf{M} & \\ = & (\mathbf{A} \otimes \mathbf{A}^*) \rhd \mathbf{M})) \,\Box\, \mathbf{M} & (\text{by } \textsc{lsm.m.asso}) \\ = & ((\mathbf{A} \otimes \mathbf{A}^*) \oplus \mathbf{I}) \rhd \mathbf{M} & (\text{by } \textsc{lsm.r.dist}) \\ = & \mathbf{A}^* \rhd \mathbf{M} & (\text{by definition of } \mathbf{A}^*) \end{array}$$

The matrix $\mathbf{R}$ can also be computed with a a Bellman-Ford iteration

$$\begin{array}{rcl} \mathbf{R}^{[0]} & = & \mathbf{M}, \\ \mathbf{R}^{[k+1]} & = & (\mathbf{A} \rhd \mathbf{R}^{[k]}) \,\Box\, \mathbf{M}. \end{array}$$

In this case we have, again courtesy of the distributivity axioms, that $\mathbf{R}^{[n-1]} = \mathbf{A}^* \rhd \mathbf{M}$. Distributed implementations (distance-vector and path-vector routing) require additional path information to avoid counting-to-infinity problems.

In a link-state approach, each node q would flood the entries of the entries of $\mathbf{M}(q, \_)$ in addition to $\mathbf{A}(q, \_)$. Node i computes $\mathbf{A}^*(i, \_)$ as before, and then computes $\mathbf{R}(i, d)$ as in Equation 10.

4. GLOBAL VS. LOCAL OPTIMALITY

It is well known that protocols such as EIGRP [8] and BGP [18] violate some of the algebraic rules of the previous sections. In particular the distributivity rules are often violated. We will see in Section 5 that this is true even of OSPF and IS-IS.

With the loss of distributivity, some of the arguments of previous sections no longer hold. In particular, the routing matrix $\mathbf{R}$ in the equations

$$\begin{array}{lll} \text{(I)} & \mathbf{X} = \mathbf{AX} \oplus \mathbf{I} & \mathbf{F} = \mathbf{X} \rhd \mathbf{M} \\ \text{(II)} & \mathbf{F} = (\mathbf{A} \rhd \mathbf{F}) \,\Box\, \mathbf{M} & \\ \text{(III)} & \mathbf{X} = \mathbf{AX} \oplus \mathbf{I} & \mathbf{F} = (\mathbf{X} \rhd \mathbf{F}) \,\Box\, \mathbf{M} \end{array}$$

may now have *distinct* solutions. Informally this means that a "link-state" solution (I) may be different from a "path vector" solution (II) for the *same* adjacency matrix and mapping matrix.

The results of [18] can be used to show that the iterative method (Bellman-Ford algorithms) will converge *strictly inflationary* holds in the algebraic structure. That is, when the following rule holds.

$$\begin{array}{ll} (\textsc{sr.s.inf}) & a \neq \hat{0} \rightarrow a <_{\oplus} b \otimes a \\ (\textsc{lsm.s.inf}) & u \neq \hat{0} \rightarrow u <_{\Box} s \rhd u \end{array}$$

where $x \leq_{\oplus} y \equiv x = x \oplus y$ and $u \leq_{\Box} w \equiv u = u \Box w$. Again, the iteration is restricted to simple paths (so an implementation needs to keep track of paths and not consider those paths with loops). Equation 5 and Equation 11 can be read as the specification of a *local optima.*

By (hop-by-hop) *Forwarding Consistency* (FC) we mean that every path associated with $\mathbf{F}(i,\ d)$ that transits node $j \neq i$ is an extension to some path associated with $\mathbf{F}(j,\ d)$. It is not hard to see that solutions to Equation 11 (that is Equation (II) above) will always be FC exactly when the algebra is strictly inflationary, even when distributivity is violated. However, if any distributivity is violated, then solving Equations (I) or (III) can lead to a loss of FC. Put another way, link-state routing seems to be inherently susceptible to violations of FC when used in conjunction with $\rhd$ functions that violate distributivity. This suggests that hop-by-hop forwarding should be replaced with some kind of tunnelling in such situations to avoid this kind of problem.

5. IS-IS AND OSPF

In this section we present an algebraic framework to model OSPF [15, 16] and IS-IS [1] protocols. We do this to demonstrate that these protocols are not simple, since this will complicate the AD/RR picture when these protocols are participating. In addition, we see that both OSPF and IS-IS employ *internally* mechanisms very similar to RR/AD.

5.1 Lexicographic amalgamation

Route selection based on AD and the preference for intra-area over inter-area routes in OSPF and IS-IS can be modeled using a lexicographic choice.

Suppose we have routing algebras $(U_i, \Box_i, \rhd_i, \hat{0}_i)$ for $0 \leq i \leq m$. We define the *lexicographic amalgamation* of the additive components to be $(U, \Box, \hat{0})$ as follows:

$$U = \{(i,\ u) \mid u \in U_i \wedge u \neq \hat{0}_i\} \cup \{\hat{0}\}$$

where $x \Box \hat{0} = \hat{0} \Box x = \hat{0}$ and

$$(i,\ u) \,\Box\, (j,\ v) = \begin{cases} (i,\ u) & (i < j) \\ (j,\ v) & (j < i) \\ (i,\ u \,\Box_i\, v) & (i = j) \end{cases}$$

Note that this is very much like the lexicographic product [9] except here the type of u in the pair $(i,\ u)$ depends on the *value* of the first component. This might be called a *dependent* lexicographic product. We can think of this construction as *tagging* the elements of the U_i so that (non-$\hat{0}_i$) element of U_i is preferred over every element of U_j for all $i < j$.

If $u \in U_i$, then we will use the notation $\langle i,\ u\rangle \in U$ to mean

$$\langle i,\ u\rangle \equiv \begin{cases} (i, u) & \text{if } u \neq \hat{0}_i \\ \hat{0} & \text{otherwise} \end{cases}$$

For matrix $\mathbf{X}$ the notation $\langle i,\ \mathbf{X}\rangle$ denotes that matrix with entries $\langle i,\ \mathbf{X}\rangle(i, j) = \langle i,\ \mathbf{X}(i, j)\rangle$.

5.2 Ships-in-the-night routing

We need an algebraic way of solving multiple independent path problems over the same graph. This will be used to model independent routing protocols, or distinct algorithms within the same protocol (as with IS-IS and OSPF in the next section).

Suppose a network is partitioned into m regions and region i is using the semiring $(S_i, \oplus_i, \otimes_i, \overline{0}_i, \overline{1}_i)$ as its path algebra. We then define the *ships-in-the-night* routing algebra $(S, \oplus, \otimes, \overline{0}, \overline{1})$ for the entire network as follows:

$$\begin{aligned} S &\equiv S_1 \times S_2, \times \cdots \times S_m \\ \overline{0} &\equiv (\overline{0}_1,\ \overline{0}_2, \cdots, \overline{0}_m) \\ \overline{1} &\equiv (\overline{1}_1,\ \overline{1}_2, \cdots, \overline{1}_m) \end{aligned}$$

$$\begin{aligned} &(s_1,\ s_2, \cdots, s_m) \oplus (t_1,\ t_2, \cdots, t_m) = \\ &\qquad (s_1 \oplus_1 t_1,\ s_2 \oplus_2 t_2, \cdots, s_m \oplus_m t_m) \\ &(s_1,\ s_2, \cdots, s_m) \otimes (t_1,\ t_2, \cdots, t_m) = \\ &\qquad (s_1 \otimes_1 t_1,\ s_2 \otimes_2 t_2, \cdots, s_m \otimes_m t_m) \end{aligned}$$

It is easy to show that $(S, \oplus, \otimes, \overline{0}, \overline{1})$ is a semiring, but *partially ordered.* This lack of totality helps explain why the direct product is rarely encountered in the network routing literature.

We represent the ships-in-the-night adjacency matrix $\mathbf{A}$ as a combination of of individual adjacency matrices $\mathbf{A}_i$, and use the notation

$$\mathbf{A} = \llparenthesis \mathbf{A}_1, \mathbf{A}_2, \cdots, \mathbf{A}_m \rrparenthesis$$

where

$$\mathbf{A}(i,j) = (\mathbf{A}_1\langle i,j\rangle, \mathbf{A}_2\langle i,j\rangle, \cdots, \mathbf{A}_m\langle i,j\rangle)$$

and

$$\mathbf{A}_k\langle i,j\rangle \equiv \begin{cases} \mathbf{A}_k(i,j) & \text{if } i \text{ and } j \text{ are in the domain of } \mathbf{A}_k \\ \overline{0}_k & \text{otherwise} \end{cases}$$

It is then not hard to check that

$$\mathbf{A}^* = \llparenthesis \mathbf{A}_1^*, \mathbf{A}_2^*, \cdots, \mathbf{A}_m^* \rrparenthesis,$$

which can be interpreted as a single matrix representing the computation of m sub-matrices in parallel. Of course the the set of matrix indices may need to be adjusted properly to make this work, but we ignore the details here.

5.3 OSPF and IS-IS metrics

IS-IS		OSPF	
L1	$u = (1, x)$	intra-area	$u = (1, x)$
L2/L1→L2	$u = (2, x)$	inter-area	$u = (2, x)$
L2→L1	$u = (3, x)$	internal	$(0, u)$
internal	$(0, u)$	type-I external	$(1, u)$
external	$(0, v, u)$	type-II external	$(2, v, u)$

Figure 8: OSPF and IS-IS routes and their preference, ordered lexicographically from top to bottom.

Both OSPF and IS-IS can be used to partition a network into *regions*, called *levels* in IS-IS and *areas* in OSPF. Although many think of these protocols as simple shortest path protocols, based on the semiring **sp**, the metrics employed are actually quite complex as is illustrated in Figure 8. Both protocols use "nested" amalgamated lexicographic constructions to rank distinct kinds of internal routes (denoted by u in the Figure), as well as making a distinction between internal and external routes. OSPF type-II external external routes and IS-IS external routes both employ a *cold potato* choice — the external metric (actually originating from configuration) is more significant than the internal metric.

5.4 IS-IS routing algebra

We will take a closer look at IS-IS. For a similar description of OSPF as well as example configurations, see the first author's dissertation [2].

IS-IS computes three distinct types of *internal* routes – L1 routes, L2 or L1→L2 routes, and L2→L1 routes – and prefers L1 routes over L2 or L1→L2 routes which are preferred over L2→L1 [13].

In IS-IS the same shortest-paths algebra is used in all the regions including the backbone. However, they treat backbone routes differently from non-backbone routes. We use one ships-in-the-night algebra for all the non-backbone areas and represent the all-region adjacency matrix as follows:

$$\mathbf{A}_\mathrm{r} = \llparenthesis \mathbf{A}_1, \mathbf{A}_2, \cdots, \mathbf{A}_m \rrparenthesis.$$

Suppose, $\mathbf{A}_\mathrm{b}$ represents the adjacency matrix of the backbone region. We then represent the adjacency matrix of an OSPF or IS-IS network as follows:

$$\begin{aligned} \mathbf{A} &= \llparenthesis \mathbf{A}_\mathrm{r}, \mathbf{A}_\mathrm{b} \rrparenthesis \\ &= \llparenthesis \llparenthesis \mathbf{A}_1, \mathbf{A}_2, \cdots, \mathbf{A}_m \rrparenthesis, \mathbf{A}_\mathrm{b} \rrparenthesis \end{aligned}$$

$$\begin{aligned} (\vec{s}_\mathrm{r}, s_\mathrm{b}) \rhd_\mathrm{I} (1, x) &= \mathop{\Box}_{j=1}^{m} \langle 1,\ s_j + x\rangle \;\Box\; \langle 2,\ s_\mathrm{b} + x\rangle \\ (\vec{s}_\mathrm{r}, s_\mathrm{b}) \rhd_\mathrm{I} (2,\ x) &= [\vec{s}_\mathrm{r} \rhd_{1,3} \rho_3(2, x)] \\ (\vec{s}_\mathrm{r}, s_\mathrm{b}) \rhd_\mathrm{I} (3,\ x) &= \infty \end{aligned}$$

$$\begin{aligned} (\vec{s}_\mathrm{r}, s_\mathrm{b}) \rhd_\mathrm{E} (1,\ u) &= \langle 1,\ (\vec{s}_\mathrm{r}, s_\mathrm{b}) \rhd_\mathrm{I} u\rangle \\ (\vec{s}_\mathrm{r}, s_\mathrm{b}) \rhd_\mathrm{E} (2,\ v,\ u) &= \langle 2,\ v,\ (\vec{s}_\mathrm{r}, s_\mathrm{b}) \rhd_\mathrm{I} u\rangle \end{aligned}$$

Figure 9: Multiplicative operations of IS-IS forwarding algebra.

Figure 9 presents the multiplicative component. It is defined in terms of a $\rhd_\mathrm{I}$ for internal routes and a $\rhd_\mathrm{E}$ for external routes. We also need to define "redistribution" functions for the IS-IS forwarding algebra —

define $\rho_k(i,\ x)$ for $k > i$ as follows:

$$\rho_2(1,\ x) = (2,\ x)$$

$$\rho_3(2,\ x) = (3,\ x)$$

Since IS-IS uses the same shortest-paths algebra at different levels, we do not need type coercion functions for redistributing routes from one level into another.

Let us look at each of the definitions of Figure 9 to see what it does in IS-IS routing. The first multiplicative operation says that when a L1 route is extended by a L1 link, it remains L1 route but when a L1 route is injected into the backbone and is extended by a L2 link, it becomes a backbone route. The second multiplicative operation defines how to inject routes downward in the preference level; that is, when a backbone route (L2 or L1→L2) is leaked down to a non-backbone area and is extended by an non-backbone link, it becomes a leaked route. Note that injecting routes downward does not need to introduce a new preference level. Integrated IS-IS gives *leaked routes* a different preference level in order to avoid any conflict with the original IS-IS route selection. Finally, the third multiplicative operation says that a leaked route cannot be be injected back into the routing system.

In order to compute forwarding table entries with external routes, IS-IS needs to define two RR functions – ρ_{int} for injecting external routes with internal metrics and ρ_{ext} for injecting external routes with external metrics. We can define ρ_{int} for IS-IS as follows

$$\rho_{\text{int}}(v) = (1,\ c_{\text{i}}(v)),$$

where v is an external route and $c_{\text{i}} : V \to U$ is a coercion function and ρ_{ext} as

$$\rho_{\text{ext}}(v) = (2,\ v,\ u^{'})$$

for cold-potato forwarding, where $u^{'}$ is a locally assigned internal metric.

With the forwarding algebra defined above, IS-IS builds network-wide forwarding tables by solving the semi-module equation of the form

$$\mathbf{F} = \llparenthesis \mathbf{A}_1^*,\ \mathbf{A}_2^* \rrparenthesis \rhd_{\text{E}} \mathbf{F} \,\Box_{\text{E}}\, \mathbf{M} \tag{12}$$

using the iterative algorithm III discussed in Section 3. Note that the hub-and-spoke topology of areas makes an IS-IS network a 3-hop network and the forwarding algebra defined above captures this constraint. For instance, a L1 route can only be extended by a L2 route, which results a L1→L2 route. A L2/L1→L2 route can only be extended by a L1 route, which results in a leaked route. A a leaked route cannot be extended further limiting IS-IS route computations within three hops. These topological restrictions obviate the need for paths in this distributed Bellman-Ford computation.

6. EXPLORING RR/AD DESIGNS

We consider only two protocols, P_1 and P_2, which interact through the FIB. Each node i is computing one row $\mathbf{F}(i,\ \_)$ of the network-wide forwarding matrix $\mathbf{F}$. Our observations extend in a natural way when there are more than two protocols.

Figure 10 illustrates two routers, i and j, that are both running protocols P_1 and P_2. Protocol P_1 is constructing RIB $\mathbf{R}_1$ while protocol P_2 is constructing RIB $\mathbf{R}_2$. We attempt to model this picture with a high-level, network-wide, algebraic model.

First, we assume that in each case the mapping matrices come from routes redistributed from the FIB. That is,

$$\begin{aligned}\mathbf{M}_1 &= \mathbf{U}_1 \lhd_1 \mathbf{F}\\ \mathbf{M}_2 &= \mathbf{U}_2 \lhd_2 \mathbf{F}\end{aligned}$$

where, for $p \in \{1,\ 2\}$, we want

$$\mathbf{M}_p(q,\ d) = \mathbf{U}_p(q,\ q) \lhd_p \mathbf{F}(q,\ d),$$

for a (diagonal) configuration $\mathbf{U}_p(q,\ q)$ and a *redistribution function* $\lhd_p$ used to model the translation of FIB routes to mappings of the appropriate type for protocol p ($\mathbf{U}$ stands for $\mathbf{U}$p). Note that we could "disaggregate" each node in Figure 10, distributing the RIBs and FIBs to distinct nodes in the graph. However, we will leave such complications to future work.

In addition, we want the FIB to depend on routes exported by the two protocols,

$$\mathbf{F} \;=\; (\mathbf{D}_1 \lhd^1 \mathbf{R}_1) \,\Box_0\, (\mathbf{D}_2 \lhd^2 \mathbf{R}_2) \,\Box_0\, \mathbf{F}_0$$

where $\mathbf{F}_0$ contains, in some way, the *connected* routes, $\Box_0$ represents routes selection on FIB routes, and (diagonal) configuration $\mathbf{D}_p(q,\ q)$ and *registration function* $\lhd^p$ are used to model the translation of RIB routes of protocol p to FIB routes ($\mathbf{D}$ stands for $\mathbf{D}$own).

Our design problem can be summarized as follows. Given protocols protocols P_1 and P_2 we would like to define the type and ordering of FIB routes together with redistribution and registration functions so that this system always converges to a unique solution. As argued in Section 1, we feel a *loosely coupled approach* is most appropriate for the design of a general RR/AD mechanism for Internet routers. This means that RR/AD designers and implementors may have to know a great deal about the protocols they intend to support, but that routing protocol designers can independently develop their systems with little or no coordination.

6.1 Lacking theoretical framework

The design space captured in Figure 10 is enormous. Can we use routing theory to make progress? Sadly, we believe that *current* routing theory (for example, [17]) does not help us directly. To see this, consider the special case where both protocols are computed by an it-

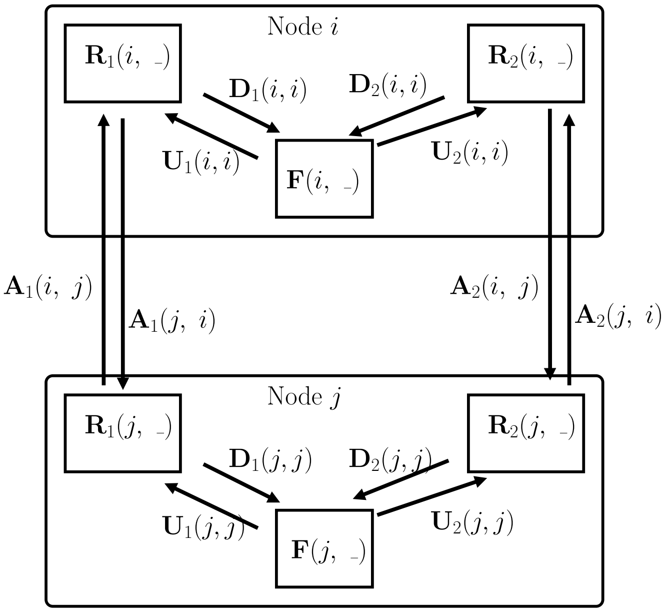


Figure 10: General picture of route redistribution between two protocols constructing RIBs $\mathbf{R}_1$ and $\mathbf{R}_2$. The FIB $\mathbf{F}$ receives routes from the protocols, selects the best routes for forwarding, and redistributes routes back to the protocols.

erative Bellman-Ford algorithm so that we can describe the system as the simultaneous solution of the following three equations.

$$\begin{array}{lrcl} (a) & \mathbf{R}_1 & = & (\mathbf{A}_1 \rhd_1 \mathbf{R}_1) \Box_1 (\mathbf{U}_1 \lhd_1 \mathbf{F}) \\ (b) & \mathbf{R}_2 & = & (\mathbf{A}_2 \rhd_2 \mathbf{R}_2) \Box_2 (\mathbf{U}_2 \lhd_2 \mathbf{F}) \\ (c) & \mathbf{F} & = & (\mathbf{D}_1 \lhd^1 \mathbf{R}_1) \Box_0 (\mathbf{D}_2 \lhd^2 \mathbf{R}_2) \Box_0 \mathbf{F}_0 \end{array}$$

In order to apply the theory of [17] we would need to represent the network routing state in a matrix $\mathbf{X}$ over some type, construct a selection operation $\Box$ and a policy operation $\rhd$, a network configuration matix $\mathbf{A}$, an initial mapping matrix $\mathbf{M}$. We then would have to show that these operations have the properties that ensure

$$\mathbf{X} = \mathbf{A} \rhd \mathbf{X} \Box \mathbf{M}, \tag{13}$$

has a unique solution that can be arrived at with a modified Bellman-Ford algorithm.

How could we form such an $\mathbf{X}$? One obvious choice for $\mathbf{X}$ is the direct product construction described in Section 5.2,

$$\mathbf{X} = (\!|\mathbf{R}_1,\ \mathbf{R}_2,\ \mathbf{F}|\!).$$

We would then need to define $\mathbf{A}$ and $\rhd$ such that

$$\mathbf{A} \rhd (\!|\mathbf{R}_1,\ \mathbf{R}_2,\ \mathbf{F}|\!) = (\!|\mathbf{R}_1',\ \mathbf{R}_2,'\ \mathbf{F}'|\!),$$

where

$$\begin{array}{rcl} \mathbf{R}_1' & = & (\mathbf{A}_1 \rhd_1 \mathbf{R}_1) \Box_1 (\mathbf{U}_1 \lhd_1 \mathbf{F}) \\ \mathbf{R}_2' & = & (\mathbf{A}_2 \rhd_2 \mathbf{R}_2) \Box_2 (\mathbf{U}_2 \lhd_2 \mathbf{F}) \\ \mathbf{F}' & = & (\mathbf{D}_1 \lhd^1 \mathbf{R}_1) \Box_0 (\mathbf{D}_2 \lhd^2 \mathbf{R}_2) \end{array}$$

So far we have not succeeded in making such a construction work, and we leave it as an open problem.

But suppose that we could make some definitions work. A large problem remains in that the structures $(\!|\mathbf{R}_1,\ \mathbf{R}_2,\ \mathbf{F}|\!)$ are *partially ordered*, while all of our theory currently requires either a total order or a total pre-order. Thus we would still be stuck.

One more word about the design space. We will assume here that both protocols share the same low-level addressing and forwarding model (such IP's hop-by-hop forwarding). A very interesting dimension to the design space, which is also left for future work, is to consider the interaction of routing protocols with *distinct* forwarding techniques (for example, hop-by-hop forwarding and tunneling).

6.2 Special case: Administrative Distance (AD)

Given our current lack of knowledge it seems that all we can do is make some assumptions and look at special cases that allow us to reason about this complex system. A good place to start is with the current use of Administrative Distance (AD), which can be modeled abstractly using lexicographic amalgamation described in Section 5.1. This clearly supports a loosely coupled design since no relationship between the participating metrics is assumed.

Suppose that connected routes are most preferred, then routes from protocol P_1, then routes from protocol P_2. We can easily define the registration functions so that

$$(c)\quad \mathbf{F} \;=\; \langle 1,\ \mathbf{R}_1\rangle \Box \langle 2,\ \mathbf{R}_2\rangle \Box \langle 0,\ \mathbf{M}_0\rangle$$

For a loosely coupled design it is hard to imagine doing anything other than breaking the dependency between $\mathbf{R}_1$ and $\mathbf{R}_2$ by allowing routes to flow in only one direction with respect to the ordering. But once we have fixed P_1 routes as being more preferred (for whatever reason), allowing routes to flow "downward" from P_2 to P_1 causes a direct conflict at the FIB level. Therefore, we will only allow redistribution "upwards" in the ordering. That is, we need to define the functions $\lhd_p$ so that

$$\begin{array}{rcccl} \mathbf{M}_1 & = & \mathbf{U}_1 \lhd_1 \mathbf{F} & = & \mathbf{U}_1 \lhd_1' \langle 0,\ \mathbf{M}_0\rangle) \\ \mathbf{M}_2 & = & \mathbf{U}_2 \lhd_2 \mathbf{F} & = & \mathbf{U}_2 \lhd_2' (\langle 1,\ \mathbf{R}_1\rangle \Box \langle 0,\ \mathbf{M}_0\rangle)) \end{array}$$

for some (newly defined) functions $\lhd_p'$.

Now, in a loosely coupled design how could we define $\lhd_2'$? We cannot require the designers of protocol P_2 to know the details of the metrics of protocol P_1, so the only reasonable solution is to define $\lhd_2'$ (at the element level, not the matrix level) as

$$\begin{aligned} t \lhd_2' \bar{0} &= \bar{0} \\ t \lhd_2' (0, x) &= P_2\text{'s way of accepting a connected route.} \\ t \lhd_2' (1, x) &= \text{some constant value extracted from } t. \end{aligned}$$

In fact, this is essentially what vendors do today. That is, metrics are *not* translated in an order preserving manner from one protocol to another. Contrary to the NRP paper, we feel this is a good design choice.

With these definitions we can imagine solving the system when both protocols are based on path-vectoring.

$$\begin{aligned} (a.pv) \quad \mathbf{R}_1 &= (\mathbf{A}_1 \rhd_1 \mathbf{R}_1) \Box_1 \mathbf{M}_1 \\ (b.pv) \quad \mathbf{R}_2 &= (\mathbf{A}_2 \rhd_2 \mathbf{R}_2) \Box_2 \mathbf{M}_2 \\ (c.pv) \quad \mathbf{F} &= \langle 1,\ \mathbf{R}_1 \rangle \Box \langle 2,\ \mathbf{R}_2 \rangle \Box \langle 0,\ \mathbf{M}_0 \rangle \end{aligned}$$

When there are no changes in the network we would expect protocol P_1 to converge first, then protocol P_2, then the FIB $\mathbf{F}$ would stabilize. A similar situation can be imagined with both protocols using a link-state approach. In this case we would have

$$\begin{aligned} (a.ls) \quad \mathbf{R}_1 &= \mathbf{A}_1^* \rhd_1 (\mathbf{U}_1 \lhd_1 \mathbf{M}_1) \\ (b.ls) \quad \mathbf{R}_2 &= \mathbf{A}_2^* \rhd_2 (\mathbf{U}_2 \lhd_2 \mathbf{M}_2) \\ (c.ls) \quad \mathbf{F} &= \langle 1,\ \mathbf{R}_1 \rangle \Box \langle 2,\ \mathbf{R}_2 \rangle \Box \langle 0,\ \mathbf{M}_0 \rangle. \end{aligned}$$

But note that following the discussion of Section 4 there is no reason to expect that these two scenarios will result in the same stable routing solutions.

Note that proving convergence for the RR/AD mechanism now reduces to proving convergence separately for the protocols involved. Imagine for a moment that this were not the case. This would mean that a collection of routing protocols that are “‘well behaved” individually might collectively cause a RR/AD system to misbehave. This is exactly the kind of problem that would probably arise in the tightly coupled design described by NRP.

7. NRP REVISITED

We have modified the notation used in the NRP paper ([12]) to ease comparison.

We start with an ordered structure $(U, \leq)$ where $\leq$ is a total order. The NRP model is based on what we call an *embedded amalgamation* of ordered structures $(S_i, \leq_i, \otimes_i)$, $1 \leq i \leq k$, where each order $\leq_i$ is total. Suppose there are *embeddings* $e_i \in S_i \rightarrow U$. In addition, there are *projections* $p_i \in U \rightarrow S_i$. The *coercion* function $c_i^j \in S_i \rightarrow S_j$ is defined as $c_i^j(s) = p_j(e_i(s))$.

The amalgamated carrier is defined as the disjoint union of all S_i, $S = \biguplus_i S_i$. An amalgamated order is defined over S as

$$a \leq b \quad = \quad \begin{cases} a \leq_i b & \forall a, b \in S_i \\ e_j(a) \leq e_i(b) & \forall a \in S_j \wedge \forall b \in S_i \end{cases}$$

Finally, the multiplicative operation is defined as

$$a \rhd b \quad = \quad \begin{cases} a \otimes_i b & a,\ b \in S_i \\ a \otimes_j c_i^j(b) & i \neq j \wedge a \in S_j \wedge b \in S_i \end{cases}$$

Furthermore, postulate that

(a) e_i preserve strict order, $a <_i b \rightarrow e_i(a) \leq e_i(b)$,

(b) $a \leq e_i(p_i(a))$, for all $a \in U$.

The main result of NRP is that when (a) and (b) hold, and each S_i is strictly inflationary, then $(S,\ \leq,\ \rhd)$ is strictly inflationary. The results of [18] are then invoked.

We have argued in Section 6.1 that the theory of [18] does not directly apply here. We believe that is true for the NRP approach as well. The problem is that the *entire* routing state of the network must be modeled in the Bellman-Ford iterations, and this is not done in [12]. Furthermore, the NRP paper claims that its approach would require no modifications to existing protocols. Yet the results of [18] critically depend on paths used in the Bellman-Ford iterations to prevent consideration of paths with loops. No such paths are added to the algorithms of NRP.

In addition, the application of [18] implies that the NRP approach would use a Bellman-Ford iterations to reach a solution. But as argued in Section 6.2, there is no guarantee that this is the solution that the *unmodified* protocols would reach.

The existence of $(U,\ \leq)$ with mappings e_i and p_i that satisfy conditions (a) and (b) is in fact a *very strong set of assumptions*. This requires a *tightly coupled* approach to design, which we have argued against.

Our amalgamated lexicographic model imposes no such constraints and is therefore better suited to model real-world protocols. It is hard to imagine how we would find an embedded amalgamation that unifies such algebraically distinct metrics as those of EIGRP and OSPF or IS-IS from the Section 5. The NRP paper models OSPF and IS-IS as simply computing shortest paths.

8. RECOMMENDATIONS, PROBLEMS

The work of [10, 11] has described scenarios where network engineers have employed the RR/AD mechanism to implement complex schemes of region backup and other types of multi-protocol protection against failures. However, we would argue that just because RR/AD is pressed into service to solve problem X, this does not mean that in designing a new RR/AD mechanism we must embrace solving X as a design goal. The problem that network engineers face is that the have *too few* tools to solve their problems and they simply must make do with the limited tools at hand.

Our main recommendation is to keep the RR/AD as simple as possible, as loosely coupled as possible, and avoid overloading it with additional functionality. If region backup is needed, perhaps a new IGP would be a better solution — an intra-domain routing protocols especially designed to provide inter-protocol backup (a BGP for IGPs?). Attempting to solve the region backup problem with RR/AD will inevitably lead to a tightly coupled design.

The framework presented here is by no means complete. Most importantly, it seems to us that a rigorous theory of *forwarding* (both hop-by-hop and tunneled) must be fully incorporated into the routing theory. The routing theory tells us which routes (and paths) are selected, but not *how* they are used by the low-level forwarding engine. And problems related to forwarding interactions seem to be at the heart of most problems described [10, 11].

In addition, our current model is *flat* in the sense that it does not account of route aggregation and other types of hierarchical routing. We do not account for routing adjacencies that arise from the connectivity provided by *lower-level* protocols. For example, internal BGP (iBGP) adjacencies (and forwarding paths) depend on the solutions provided by IGP routing. Another example would be PNNI-like routing, where sub-domains of arbitrary nesting can be treated as nodes.

Acknowledgment

This work was part of the first author's doctoral dissertation at the Computer Laboratory, University of Cambridge, which was supported by grants from Boeing and Cisco Systems. His current work is partly funded by the EU FP7 CHANGE (257422) project. The second author is partially supported by those grants and EPSRC (UK) grant EP/F002718/1. Both authors would like to thank the CoNEXT reviewers for their helpful comments.

9. REFERENCES

[1] Intermediate System to Intermediate System (IS-IS) Intra-domain Routing Information Exchange Protocol. International Standard ISO/IEC 10589:2002(E), November 2002.

[2] M. A. Alim. On the interaction of internet routing protocols. University of Cambridge, 2011. Doctoral disseration.

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

[4] J. N. Billings and T. G. Griffin. A Model of Internet Routing using Semi-modules. In *RelMiCS/AKA 11 2009*, Doha, Qatar, November 2009.

[5] J. Day. *Patterns in Network Architectures : A return to fundamentals.* Prentice Hall, 2008.

[6] J. Garcia-Luna-Aceves. Loop-free routing using diffusing computations. *IEEE/ACM Transactions on Networking*, 1(1), 1993.

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

[8] M. G. Gouda and M. Schneider. Maximizable routing metrics. *IEEE/ACM Transactions on Networking*, 11(4):663–675, August 2003.

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

[10] F. Le, G. Xie, D. Pei, J. Wang, and H. Zhang. Shedding light on the glue logic of the internet routing architecture. In *Proc. ACM SIGCOMM*, 2008.

[11] F. Le, G. Xie, and H. Zhang. Understanding route redistribution. In *Proc. Inter. Conf. on Network Protocols*, 2007.

[12] F. Le, G. Xie, and H. Zhang. Theory and new primitives for safely connecting routing protocol instances. In *Proc. ACM SIGCOMM*, August 2010.

[13] T. Li, T. Przygienda, and H. Smit. Domain-wide prefix distribution with two-level IS-IS. RFC 2966, October 2000.

[14] D. Meyer. The locator identifier separation protocol (LISP). *Cisco Protocol*, 11(1):23–36, 2008.

[15] J. Moy. *OSPF: Anatomy of an Internet Routing Protocol.* Addison-Wesley, 1998.

[16] J. Moy. *OSPF: Complete implementation.* Addison-Wesley, 2000.

[17] J. L. Sobrinho. Network routing with path vector protocols: Theory and applications. In *Proc. ACM SIGCOMM*, September 2003.

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