Theory and New Primitives for Safely Connecting Routing Protocol Instances

Franck Le
Carnegie Mellon University
franckle@cmu.edu

Geoffrey G. Xie
Naval Postgraduate School
xie@nps.edu

Hui Zhang
Carnegie Mellon University
hzhang@cs.cmu.edu

ABSTRACT

Recent studies have shown that the current primitives for connecting multiple routing protocol instances (OSPF 1, OSPF 2, EIGRP 10, etc.) are pervasively deployed in enterprise networks and the Internet. Furthermore, these primitives are extremely vulnerable to routing anomalies (route oscillations, forwarding loops, etc.) and at the same time too rigid to support some of today's operational objectives. In this paper, we propose a new theory to reason about routing properties across multiple routing instances. The theory directly applies to both link-state and vector routing protocols. Each routing protocol still makes independent routing decisions and may consider a combination of routing metrics, including bandwidth, delay, cost, and reliability. While the theory permits a range of solutions, we focus on a design that requires no changes to existing routing protocols. Guided by the theory, we derive a new set of connecting primitives, which are not only provably safe but also more expressive than the current version. We have implemented and validated the new primitives using XORP. The results confirm that our design can support a large range of desirable operational goals, including those not achievable today, safely and with little manual configuration.

Categories and Subject Descriptors:
C.2.6 [Computer-Communication Networks]: Internetworking
General Terms: Design, Theory
Keywords: Connecting primitives, route redistribution, route selection

1. INTRODUCTION

Recent empirical studies [24, 21, 5] challenge the traditional, simple "BGP over your favorite IGP" view of the Internet routing architecture. As illustrated in Figure 1, they reveal that the Internet routing landscape is in reality much more complex. ISPs and enterprise networks deploy tens to hundreds of routing protocol instances simultaneously [21, 5], and those routing instances are oftentimes interconnected in diverse ways [24]. In a recent study [5], the authors found that 57% of the analyzed networks have more than three routing instances, which is greater than a single IGP and EGP, and discovered both enterprise and university networks with more than ten instances. Former studies [21, 24] have also confirmed the prevalance of routing instances and exposed networks with even hundreds of routing instances. There are several reasons for these sophisticated routing designs: the need to route traffic based on metrics other than hop count, the desire for autonomy between departments of a same company [21], the requirement to filter route announcements [8], scalabity [21] and economical reasons [5].

It has been observed that the connecting primitives, which run on the border routers (e.g., A and B in Figure 1) and govern the interactions between the routing protocol instances, play critical roles in implementing the sophisticated routing designs. Even in the simplest "BGP over IGP" scenarios, those primtives are actually required to inject IGP or static routes into BGP. More importantly, operators use them to not simply interconnect routing protocol instances but also achieve critical design objectives (e.g., domain backup, shortest path routing across instances) that are infeasible using routing protocols (e.g., BGP) alone [21].

Currently, the primitives responsible for interconnecting routing instances consist of the so-called *route selection* and *route redistribution* procedures [9, 8]. Consider routers A and B in Figure 1. They are border routers in the sense that they belong to multiple routing protocol instances at the same time. Router A belongs to three routing protocol instances (BGP, OSPF 100, and RIP) and runs a separate routing process for each of them. In contrast, router B is a member of two different OSPF instances. When a border router (e.g., A) receives routes, to the same destination prefix, from multiple routing processes (e.g., BGP, OSPF 100, RIP), the border router cannot directly compare the routes as each routing instance typically has its own metrics. For example, RIP relies on a hop-count, whereas OSPF routes have a type (intra-area, inter-area, external type 1, external type 2) and a cost. The border router uses the *route selection* procedure to rank routes received from different routing processes and to determine which one to install in its forwarding table. As for *route redistribution*, this procedure is required to exchange routing information between routing instances. By default, routing processes of different protocol instances do not exchange routing information even though they are on the same border router. Route redistribution must be explicitly enabled through router configuration. For example, the OSPF 200 and OSPF 300 instances will not exchange routes unless route redistribution between the two instances is configured on router B. Current operational networks rely heavily on these two procedures. A recent study [21] analyzed the usage of route redistribution in more than 1600 networks, and revealed that 99.9% of them depend on it.

Despite the prevalence and importance of the connecting primi-

SIGCOMM'10, August 30–September 3, 2010, New Delhi, India.

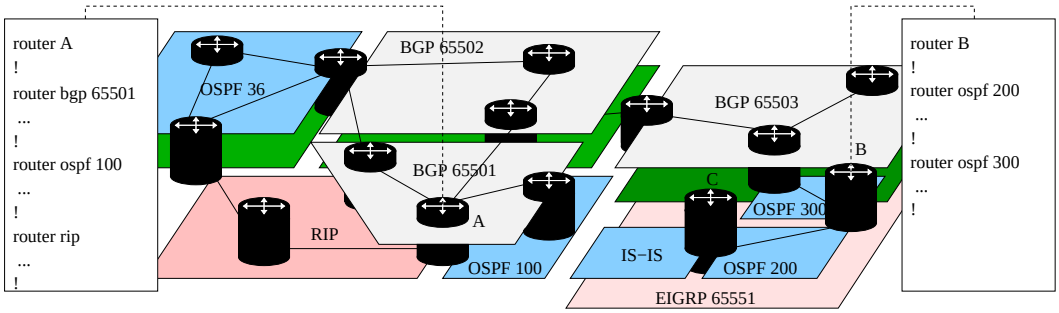

Figure 1: A typical slice of Internet routing landscape. Though abstracted, it still shows formidable complexity.

tives, it has been shown that the current mechanisms are extremely prone to misconfigurations [20, 22] and such errors are likely the root causes of many reported forwarding loops, route oscillations, prefix hijacks, and non-deterministic path problems [23]. In response, several analytical models [20, 22, 23] have been developed enabling rigorous analyses of the current route selection and route redistribution procedures, and the formulation of practical configuration guidelines. However, adding band-aids to current mechanisms presents severe limitations. Configuration guidelines introduce new restrictions on setting parameters and, therefore, reduce the flexibility of the primitives and their power to implement operational goals. Operators have reported that the current primitives, even without any restriction, are already too rigid to support some desirable routing policies [21]. The existing analytical models are too tied to the current mechanisms and as such, do not provide insights for new designs.

We believe that the Internet will remain a myriad of routing protocol instances and that the primitives responsible for connecting different routing protocol instances will continue to play a crucial role in the Internet routing architecture. One single routing instance is unlikely to satisfy all operational requirements. The driving forces behind the current prevalence of routing protocol instances, including the distinction between IGP and EGP functionality, the requirement to route traffic based on different metrics, and the desire for autonomy between sites branches or departments, are likely to persist. In fact, the number of routing protocol instances may even grow with the emergence of new technologies (e.g., wireless networks, ad-hoc networks, vehicular networks, sensor networks, etc.) as each of them presents unique characteristics and may require distinct routing protocols. In this context, operators need a safe way to connect routing instances.

This situation brings up a fundamental open question: Can we design a set of connecting primitives that both guarantee routing correctness (i.e., always converge to loop-free forwarding paths regardless of how they are configured) and increase the offered degree of expressiveness allowing operators to fulfill their requirements? To answer the question, we need a theory to reason about routing across multiple routing protocol instances.

In this paper, we present such a theory for reasoning about routing correctness in networks with multiple interconnected routing instances. From the theory, we then derive a new set of primitives to connect routing protocol instances. While the theory permits a wide range of design options, we focus on a design characterized by no changes to existing routing protocols. We implemented the new primitives in XORP routing software [3]. The results confirm that our proposed design allows operators to safely implement a large range of desirable design objectives, including those not feasible today. Our contributions are three-fold:

1. We have developed a new formal framework to reason about routing properties in networks with *multiple* interconnected routing protocol instances. By adding the formalism of conversion functions to the theory of routing algebras, we are able to abstract the functional requirements of connecting routing protocol instances. The framework is able to model the current route selection and route redistribution procedures and also provide insights for a clean slate design of these mechanisms. The key result is a set of sufficient conditions for guaranteeing safe routing, and optimal paths, across multiple routing instances.

2. Guided by the new theory, we have created a new design of connecting primitives. In contrast to the current approach, our design guarantees routing safety regardless of configuration errors and, moreover, supports a large range of operational goals. The solution is desirable and feasible for many networks, including individual ISP networks, and large enterprise networks. To deploy the new primitives likely requires router software upgrade. However, no modification to any of the existing routing protocols is necessary. As such, the scope of router software upgrade will be limited to a relatively small number of border routers.

3. We have implemented the new primitives into XORP and conducted experiments to validate the ability of these primitives to support several design goals considered important by the operational community. The results are encouraging. While some of the goals are not feasible today, our implementation show the new primitives are able to support them without requiring elaborate configurations.

The rest of the paper is structured as follows. Section 2 provides a brief description of the existing route selection and route redistribution procedures. Section 3 presents the newly proposed theory to reason about routing across multiple routing protocol instances. Section 4 identifies sufficient conditions to guarantee correct routing and optimal paths across routing protocol instances. From the theory, Section 5 derives new primitives. Section 6 presents our implementation. Section 7 illustrates the expressiveness of the design. Section 8 presents related work, and finally, Section 9 discusses future work.

2. BACKGROUND: CURRENT PRIMITIVES

This section briefly describes the current implementation of the two primitives, namely route selection and route redistribution, that govern the interactions between routing protocol instances. It should be noted that all discussions in the paper are with respect to a single destination prefix, denoted by P, unless noted otherwise.

```
1   interface ethernet 0
2    ip address 192.1.1.1 255.255.255.0
3   !
4   interface ethernet 1
5    ip address 192.1.2.1 255.255.255.0
6   !
7   router rip
8    network 192.1.1.0
9    distance 100
10  !
11  router ospf 100
12   network 192.1.2.0 255.255.255.0 area 0.0.0.0
13   default-metric 100
14   redistribute rip metric 200 metric-type 1 subnets
15  !
```

Figure 2: Excerpt of a router configuration file illustrating the current IOS commands for route selection and route redistribution.

Route selection: A router that runs multiple instances of different routing protocols (EIGRP, BGP, OSPF, etc.) or multiple instances of a same routing protocol (e.g., OSPF 100, OSPF 200, etc.) creates a separate routing process for each of them. In the rest of this paper, we will more formally say that two routing processes belong to the same routing protocol instance when the two processes are each on a different router, run the same routing protocol and exchange routing information through it.

For the destination prefix P, each routing process selects one best route, from both the received updates and the local information, using a protocol specific algorithm: E.g., RIP simply compares the hop count while BGP uses an elaborate path ranking procedure. Then, if more than one routing process offer[s] a route to P, the router must perform a *route selection* procedure to determine which one to install in the Forwarding Information Base (FIB). This decision is currently based on a configurable parameter called Administrative Distance (AD) [9], with the preference given to the route with the lowest AD value. By default, in Cisco routers, RIP processes have an AD of 120 whereas OSPF processes have an AD of 110. As such, unless the AD values are overridden, when receiving both a RIP route and an OSPF route to the same destination prefix, a router prefers and installs the OSPF route in its FIB.

Route redistribution: Route redistribution allows operators to exchange routing information across routing instances. One complication is that routing protocols use different types of routing metrics. For example, RIP uses a single metric (hop count) while EIGRP relies on a weighted sum of bandwidth, delay, reliability, and load. The current route redistribution procedure handles this incompatibility in a crude fashion. It resets the metric of a redistributed route to either a constant default value or a fixed value manually configured by the operator. In either case, the new metric values typically have no relation to the route's original metric values.

Configuration commands: Each router vendor has its own configuration language. We focus on the Cisco IOS commands for illustration purposes. The syntax may differ across router vendors but the functions remain similar. Currently, configuring route selection and route redistribution on Cisco routers mainly involves three IOS commands. Each command allows a number of variants and options. Figure 2 illustrates an example use of these commands. The router has two interfaces, and runs two routing processes: RIP on the first interface, and OSPF on the second one.

1. The `distance` command (line 9) allows operators to override the default administrative distance of a routing process. In the depicted example, the administrative distance of RIP is set to 100, which is lower than the default administrative distance value of OSPF (110). Consequently, when receiving routes to the same destination from both RIP and OSPF, the router will select the RIP route.

2. The `redistribute` command (line 14) inside the OSPF command block activates route redistribution from RIP into the OSPF process. When configuring BGP, one may also use the `network` command to activate redistribution from *any* source (e.g., static, RIP, etc.) into BGP. Route filters can be applied to a `redistribute` or BGP `network` command to restrict the redistribution to a specific subset of routes. The `redistribute` command has protocol-specific options. For example, in the depicted example, the `metric-type` command is specific to OSPF, which mandates the routes to be advertised as "external type 1". A route can be injected into OSPF as either an external type 1 or an external type 2 route. The two types differ in the way their costs will be calculated as they propagate inside the OSPF routing instance. The cost of a type 1 route will be dynamic, with the costs of the internal links added to the metric value assigned at the time of redistribution. In contrast, the cost of a type 2 route remains fixed regardless how many internal links it contains. In addition, a type 1 route is always preferred to a type 2 route.

3. The `default-metric` command (line 13) allows operators to configure a new default metric value for all route redistributions to a routing process. In addition, the `metric` option (line 14) may be used to override this default metric value for redistribution from a particular source. In the example, routes from the RIP routing process are injected into the OSPF process with an initial OSPF cost of 200 instead of the default value of 100.

In summary, the AD parameter (in route selection) and the metrics of newly redistributed routes (for route redistribution) are mainly set to arbitrary constant values, independently of the route's original attributes. As a result, information related to the initial routes (e.g., relative preference) may be lost potentially leading to persistent forwarding loops, permanent route oscillations and other unacceptable outcomes [22, 23].

3. A THEORY FOR MULTI-INSTANCE ROUTING

Although a considerable body of research has been devoted to the correctness of routing, most prior work concentrated on the behaviors of one specific routing protocol (e.g., RIP, OSPF, or BGP) at a time. In contrast, this section presents a general framework to study routing properties *across multiple routing protocol instances*.

Inspired by the seminal work of Griffin and Sobrinho on Metarouting [28, 29, 17], the proposed theory models routing protocols as algebras. Such an abstraction allows us to leave out the myriad of algorithmic details of different routing protocols and focus on crucial correctness requirements such as convergence and loop freedom. Furthermore, general results which are applicable to both existing and future routing protocols may be obtained from the theory.

Section 3.1 provides a brief overview of the most relevant results on routing algebras. Then, Section 3.2 introduces the new notion of *conversion functions* to model the interactions between routing algebras and extend the focused analysis to a network with multiple routing protocol instances.

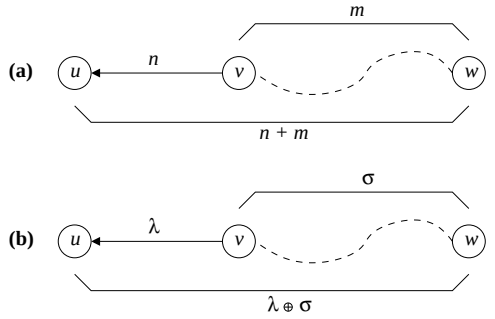

Figure 3: Illustration of similarity between (a) "classic" shortest path, and (b) routing algebra.

3.1 Background: Routing algebras

Routing algebras can be viewed as an abstraction and generalization of shortest path routing [7, 13, 15, 14, 17]. As illustrated in Figure 3 [17], each route has a signature ($\sigma \in \Sigma$) to model its relative precedence, and the notion of link weights is generalized to policy labels. When a route with signature σ is extended over a link ("u - v" in this example), with policy label $\lambda \in L$, the route's new signature becomes $(\lambda \oplus \sigma) \in \Sigma$. In other words, a signature represents the set of a route's attributes, a label represents the set of routing policies when a route is propagated over a given link, and $\oplus$ symbolizes the application of the routing policies to a route.

More formally, an algebra A is defined by a tuple (L, Σ, ϕ, $\oplus$, $\preceq$) [28, 29, 17] with ϕ being a special signature indicating a prohibited path. $\oplus$ is a mapping from $L \times \Sigma$ into Σ. The relation $\preceq$ is called a *preference relation* and creates a total pre-order over Σ. It allows routers to rank routes from A: If two routes have signatures α and β, (α, $\beta \in \Sigma$) and $\alpha \preceq \beta$, the route with signature α is preferred to the one with signature β. If $\alpha \preceq \beta$ and $\beta \preceq \alpha$, then we say that α and β are *equally prefered* (noted $\alpha \sim \beta$). Prohibited paths – paths with signature ϕ – are not further extended and $\forall \sigma \in \Sigma \setminus \{\phi\}$, $\sigma \prec \phi$.

The relation $\preceq$, being a total pre-order over Σ, satisfies the following properties:
(Reflexivity) $\forall \sigma \in \Sigma, \sigma \preceq \sigma$.
(Transitivity) $\forall \sigma_1, \sigma_2, \sigma_3 \in \Sigma$,
if $\sigma_1 \preceq \sigma_2$ and $\sigma_2 \preceq \sigma_3$, then $\sigma_1 \preceq \sigma_3$.
(Totality) $\forall \sigma_1, \sigma_2 \in \Sigma$, $\sigma_1 \preceq \sigma_2$ or $\sigma_2 \preceq \sigma_1$.
The relation $\preceq$ is not necessarily antisymmetric, i.e., for σ_1, σ_2 in Σ, $\sigma_1 \preceq \sigma_2$ and $\sigma_2 \preceq \sigma_1$ do not imply $\sigma_1 = \sigma_2$. This relaxation allows us to enlargen the scope of covered routing protocols. In particular, the framework can include path vectoring routing protocols. To illustrate it, we assume that a signature σ consists of a sequence of identifiers (e.g., router identifiers or BGP Autonomous System Numbers), and $\sigma_1 \preceq \sigma_2$ if σ_1 has a shorter sequence of identifiers than σ_2: For example, (27, 36, 45) $\preceq$ (117, 234, 54, 810). We note that (10, 20, 30) $\preceq$ (50, 30, 80) and (50, 30, 80) $\preceq$ (10, 20, 30) but (10, 20, 30) $\neq$ (50, 30, 80). This operation is similar to the BGP AS PATH.

To serve as an example of a routing algebra, the RIP routing protocol can be modeled by the following one: L = {1, 2, ..., 16}, Σ = {1, 2, ..., 16}, ϕ = 16, "$\preceq$" = "$\leq$", and $\oplus$ defined as $\lambda \oplus \sigma = \min(\lambda + \sigma, \phi)$. Each hop in a path is assigned a configurable hop count which can take any value from 1 to 16. When a router receives a routing update, it adds its hop count to the metric value, and routes with hop count of 16 or more are prohibited and not propagated. In this specific case, L = Σ. However, this may not always be the case. As another example of routing algebras, a routing protocol that selects the path with maximum available bandwidth can be modeled with ($\oplus$, $\preceq$) = (min, $\geq$).

	SM	I	$\oplus$ is assoc.
Vectoring	✓		
Link-state with Dijkstra's alg.	✓	✓	✓

Table 1: Sufficient conditions for correctness of routing.

We consider m routing instances $\{1, \ldots, m\}$. We denote $r.i$ the routing process of routing instance i hosted at router r. Each routing instance i is represented by a distinct finite algebra $A_i = (L_i, \Sigma_i, \phi_i, \oplus_i, \preceq_i)$. We model each adjacency between two routers by a distinct edge between them. In the rest of this paper, we may loosely use the terms routing algebras, routing instances and routing processes interchangeably depending on the context.

Previous papers by Sobrinho [28, 29] have identified sufficient conditions for routing correctness for both vectoring and link state routing protocols. These properties are summarized in Table 1 [17]. First, a routing algebra A_i satisfies the Strict Monotonicity (or simply SM) property if the following condition holds:
(SM) $\forall l \in L_i, \forall \sigma \in \Sigma_i \setminus \{\phi_i\}, \sigma \prec_i (l \oplus_i \sigma)$.

Simply put, SM requires that the preference of any route strictly decreases each time it is propagated by a router. SM by itself is a sufficient condition for routing correctness for a vectoring protocol [29]. For link-state protocols, additional properties are needed. In particular, a routing algebra A_i satisfies the isotonicity (I) property[1] if both of the following conditions hold:
(Left-Isotonicity) $\forall l \in L_i, \forall \sigma_1, \sigma_2 \in \Sigma_i$,
if $\sigma_1 \preceq_i \sigma_2$, then $l \oplus_i \sigma_1 \preceq_i l \oplus_i \sigma_2$.
(Right-Isotonicity) $\forall \sigma_1, \sigma_2, \sigma_3 \in \Sigma_i$,
if $\sigma_1 \preceq_i \sigma_2$, then $\sigma_1 \oplus_i \sigma_3 \preceq_i \sigma_2 \oplus_i \sigma_3$.

Isotonicity means that the preference order between two routes is preserved when they both are prepended by, or extended over, a common link. As shown in Table 1, both I and SM are needed to ensure the correctness of a link-state protocol. In addition, left-isotonicity is a sufficient condition to guarantee optimal paths for vectoring routing algebras, and a stronger condition of full isotonicity is sufficient for link-state algebras to have the same property [28].

Prior work used this elegant framework to analyze BGP and design new routing protocols through composition of routing algebras that are simple and conform to the sufficient conditions for correctness. However, the framework only applies to a network with a single routing protocol instance, i.e., every router in that network must run a single, identical routing protocol. The next section extends the framework to routing across multiple routing instances.

3.2 Conversion functions

We observe that at the heart of inter-routing-process route selection and route redistribution procedures are two types of *necessary* routing metric conversions. For route selection, metric conversions are required to establish a common ground to compare routes from different routing processes. For route redistribution, metric conversions are effectively performed when assigning metric values to redistributed routes in the target routing processes. For example, the current route redistribution procedure resets the metrics of newly redistributed routes to constant values, either by default or as specified by operators. Such redistribution can be represented by constant conversion functions.

[1]We adopt the terminology proposed in [28, 29] herein. Other works have used the terms *monotonicity* and *nondecreasing* in place of *isotonicity* and *monotonicity*, respectively.

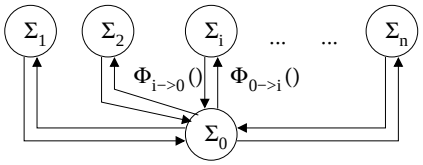

Figure 4: Illustration of the relations between the individual routing instance signature spaces $\{\Sigma_1, \Sigma_2, ..., \Sigma_m\}$ and the universal metric space Σ_0.

Therefore, we propose to extend the routing algebra framework as follows. We treat each routing instance as a separate routing algebra, and model the interactions between these routing protocol instances with a set of metric *conversion functions* between routing algebras. To be more scalable, we conceptualize the metric conversions between routing algebras as indirect, via a hypothetical common algebra with a universal metric (signature) space, as shown in Figure 4. We define the connections between each algebra A_i and the common algebra with a pair of conversion functions $\Phi_{i\to 0}()$ and $\Phi_{0\to i}()$.

To illustrate the utility of these conversion functions in *comparing routes from different routing instances*, consider a router r that receives routes through k distinct routing processes ($r.1$, $r.2$, ... $r.k$). (See Figure 5.) Each routing process $r.i$ selects its best route according to a routing process specific ranking algorithm, and presents its most preferred route to the inter-routing process route selection algorithm. We note σ_i the signature of the best route from routing process $r.i$. The signatures of these best routes (σ_1, σ_2, ..., σ_k) presented by the k routing processes belong to different signature domains, and as such, cannot be directly compared. They must first be converted, through the conversion functions $\Phi_{1\to 0}()$, $\Phi_{2\to 0}()$,..., $\Phi_{k\to 0}()$, into the common *universal metric* space. Then, with the signatures being in the same unit, a best route among all the presented routes can be selected and installed in the router's Forwarding Information Base (FIB) for forwarding purposes.

Each routing instance i may connect with other routing instances via multiple border routers. In the general case, a routing instance may use different conversion functions at different border routers. For example, the current inter-routing-process route selection procedure allows operators to override the default administrative distances of the routing processes for each router. To represent this per-router behavior, we make the router r an argument of the conversion function $\Phi_{i\to 0}()$. As such, $\Phi_{i\to 0}()$ maps elements from $\mathcal{R}$ $\times$ Σ_i into Σ_0 where $\mathcal{R}$ represents the set of border routers in the network, and Σ_0 the universal metric space.

To further motivate the use of conversion functions to abstract the *exchange of routing information across routing instances*, consider Figure 6 where a route is propagated from router A to B, then to C, and finally back to A. Observe that there are two types of route propagation: (1) routes may be propagated between routing processes of a same routing instance, and (2) routes may be redistributed between different routing processes (and thus different routing instances) on a same router.

(1) Route propagation within each routing instance i can be fully modeled by the routing algebra operators (i.e., $\oplus_i$) for the routing instance [17]. In Figure 6, when $B.RIP$ receives a route from $A.RIP$ with hop-count σ_1, and $B.RIP$ re-advertises the route over the link with weight of λ_2, the route has a new signature $\sigma_2 = \lambda_2 \oplus_{RIP} \sigma_1 = \lambda_2 + \sigma_1$.

(2) To explain route propagation across routing instance boundaries,

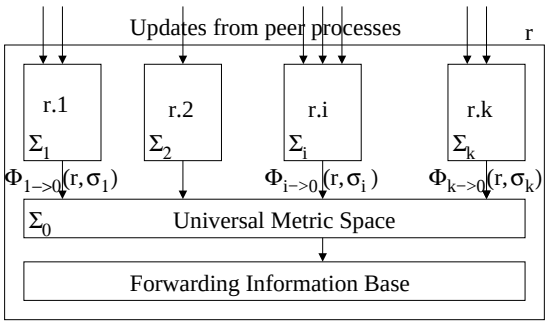

Figure 5: Ranking of routes received from different routing instances at a router. Each instance's process selects its most preferred route using an instance specific ranking algorithm and offers only its best route to the inter-routing-process route selection, which must rely on the conversion functions to map the signatures of all offered routes into a common metric space before ranking them.

suppose that the RIP route with signature $\sigma_2(\in \Sigma_{RIP})$ is redistributed from the RIP routing process into the EIGRP routing process at the border router C. The operation $\oplus_{EIGRP}$ will define the signature while the route is being advertised in the EIGRP routing instance. However, while $\oplus_{EIGRP}$ maps elements of $L_{EIGRP} \times \Sigma_{EIGRP}$ into Σ_{EIGRP}, σ_2 is from Σ_{RIP}. We need to convert the signature from Σ_{RIP} into Σ_{EIGRP} through the application of two conversion functions: $\Phi_{0\to EIGRP}(r, \Phi_{RIP\to 0}(r, \sigma_2))$. As such, the propagation of routes between routing instances at a border router can be seen as over a virtual inter-routing-process link labeled with conversion functions, and the new signature is well defined.

More formally, let $A_0 = (L_0, \Sigma_0, \phi_0, \oplus_0, \preceq_0)$ denote the common algebra, where Σ_0 represents the domain of signatures in the universal metric space, and the relation $\preceq_0$ is a total pre-order over Σ_0. For brevity, let $L = L_1 \cup L_2 \cup ... \cup L_m$, and $\Sigma = \Sigma_1 \cup \Sigma_2 \cup ... \cup \Sigma_m$.

Definition 1: Each algebra A_i ($i \in [1, m]$) is associated with two conversion functions:
1) $\Phi_{i\to 0}()$: $\mathcal{R} \times \Sigma_i \to \Sigma_0$
2) $\Phi_{0\to i}()$: $\mathcal{R} \times \Sigma_0 \to \Sigma_i$

Definition 2: The binary relation $\preceq^r$ over Σ is defined as:
$\forall\, r \in \mathcal{R}, \forall \alpha, \beta \in \Sigma$,
$\exists i, j \in [1, m]$ such that $\alpha \in \Sigma_i, \beta \in \Sigma_j$, then

$$\alpha \preceq^r \beta \stackrel{def}{=} \begin{cases} \alpha \preceq_i \beta & \text{if } i = j \\ \Phi_{i\to 0}(r, \alpha) \preceq_0 \Phi_{j\to 0}(r, \beta) & \text{else} \end{cases}$$

The relation $\preceq^r$ allows router r to rank any set of routes. If two candidate routes are from the same routing process ($r.i$), the routing instance specific best path selection algorithm ($\preceq_i$) determines the best route. If routes are from different routing processes, their signatures are first converted into the universal metric space using their respective conversion functions, and the total pre-order $\preceq_0$ over Σ_0 defines their ranking at router r.

Definition 3: The operator $\oplus^r : L \times \Sigma \to \Sigma$ is defined as:
$\forall\, r \in \mathcal{R}, \forall \lambda \in L, \forall \sigma \in \Sigma$,
$\exists j, i \in [1, m]$ such that $\lambda \in L_j, \sigma \in \Sigma_i$, then

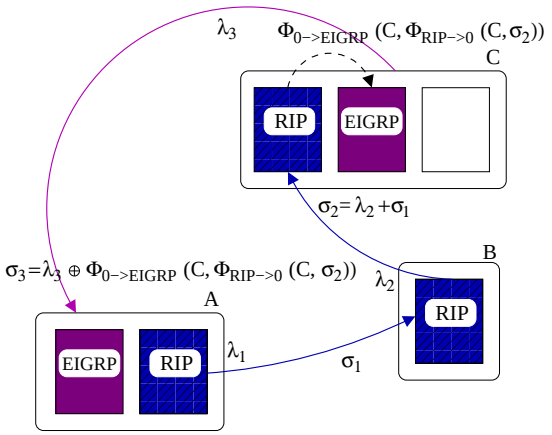

Figure 6: Propagation of routes: A route can be propagated within the same routing instance, or into a different routing instance at a border router (e.g., C in this figure).

$$\lambda \oplus^r \sigma \stackrel{def}{=} \begin{cases} \lambda \oplus_j \sigma & \text{if } i = j \\ \lambda \oplus_j \Phi_{0\to j}(r, \Phi_{i\to 0}(r, \sigma)) & \text{else} \end{cases}$$

The operator $\oplus^r$ specifies the signature of a route further propagated by router r. If a route is further propagated in the same routing instance, the new signature is determined as $\lambda \oplus_j \sigma$ as previously shown [17]. If the route is redistributed into a different routing instance, the initial signature must first be converted into a signature of the target routing instance, before the operation $\oplus_j$ can be applied.

We note that today's route selection and route redistribution procedures, as described in Section 2, can be modeled by constant conversion functions as follows.

• A two dimensional universal metric space $\Sigma_0 = [1, \ldots, 255] \times [1, \ldots, m]$, where the first dimension models the AD value space and the second an enumeration of all the routing instances in the network.

• $\preceq_0$ defined as $\forall (x,i),(y,j) \in \Sigma_0$, $(x,i) \preceq_0 (y,j)$ if and only if $x \leq y$ (with $\leq$ being the standard ordering between integers).

• $\forall \sigma = (255, i) \in \Sigma_0$, $\sigma = \phi_0$, i.e., prohibited path.

• $\forall i \in [1,m], \forall r \in \mathcal{R}, \forall \sigma \in \Sigma_i$: $\Phi_{i\to 0}(r,\sigma) = (AD(r,i), i)$, where $AD(r,i)$ represents the default or configured administrative distance for $r.i$.

• $\forall j \in [1,m], \forall r \in \mathcal{R}, \forall \sigma = (x,i) \in \Sigma_0$: $\Phi_{0\to j}(r,\sigma) = metric(r,i,j)$, where $metric(r,i,j)$ is the default or configured constant metric assigned to routes redistributed from $r.i$ into $r.j$.

This model permits us to predict – given a fixed set of input routes – the forwarding state at a router with the current design. However, the model is limited in its ability to infer end-to-end forwarding paths without additional non-trivial work to take into consideration the timing of route propagation (and possibly race conditions) and incomplete knowledge of external routes. Fortunately, as detailed in the next section, this framework allows us to identify sufficient conditions for network-wide routing safety based on only conversion function definitions per routing instance.

4. SUFFICIENT CONDITIONS

The previous section introduced the notion of conversion functions to model and reason about the properties of the connecting primitives. Under this framework, the initial question on whether we can design safer and more expressive primitives hinges on the following question: Are there conditions that are sufficient for the conversion functions to guarantee routing safety? The answer is positive and this section presents a set of such conditions. The discussion is focused on a special case where for each routing instance, the conversion functions are identical across its border routers. Formally, $\forall r_1, r_2 \in \mathcal{R}$, $\forall i \in [1,m]$,
$\forall \sigma \in \Sigma_i$, $\Phi_{i\to 0}(r_1,\sigma) = \Phi_{i\to 0}(r_2,\sigma)$, and
$\forall \sigma \in \Sigma_0$, $\Phi_{0\to i}(r_1,\sigma) = \Phi_{0\to i}(r_2,\sigma)$.

Consequently, $\oplus^r$ and $\preceq^r$ are the same across all routers. Thus, in the rest of of the paper, we simplify the notation by removing the argument r from the conversion functions, and the superscript from the operators.

For ease of exposition, the discussion is divided into two parts: first for *unary* routing algebras and then for the more general case of *n-ary* lexicographic products of sub-algebras. We define unary algebras as algebras that use a single attribute to determine their best path. An example is the RIP protocol which selects the route with the lowest hop count. In contrast, n-ary lexicographic products of sub-algebras perform a lexicographic comparison of up to n attributes. For example, the BGP best path selection algorithm is based on a lexicographic ordering of the *local-preference*, the *AS-PATH length*, the *origin type*, and other additional attributes. For brevity, we only present the results for unary algebras in detail.

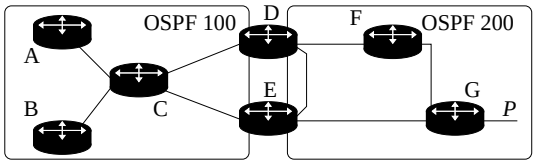

Figure 7: Modes of redistribution: Routes can be redistributed in either (1) a vectoring or (2) a link-state mode between different instances.

4.1 Two modes of route redistribution

The current route redistribution procedure injects a route into a new routing instance in a vectoring mode because the new routing information mainly consists of the destination prefix and some metrics (to rank routes). Theoretically, the exchange of routing information between routing instances can also be performed in a link-state manner whereby one routing instance passes on its entire link state database to another routing instance at a border router. In fact, this mode has an advantage over the current approach as demonstrated in the following example. Consider the network depicted in Figure 7 which consists of two instances of OSPF (OSPF 100, OSPF 200). Suppose the border routers (D, E) are configured to redistribute routes from OSPF 200 into OSPF 100. Let us focus on a subnet with prefix P connected to router G. When routers in the OSPF 100 instance (e.g., A) run Dijkstra's algorithm, they will not have a complete view of the entire network topology with the current design; instead, they only see two possible egress routers (D, E) to reach P, knowing little about the subpaths inside OSPF 200. In contrast, if the route redistribution were performed in a link-state mode, routers in OPSF 100 would have a complete view of the topology across both OSPF instances and thus be able to find end-to-end disjoint paths, to P.

Intuitively, for the vectoring mode, the key property to preserve in order to guarantee convergence and loop-free forwarding paths is SM. For the link-state mode, additional properties are required as indicated by Table 1.

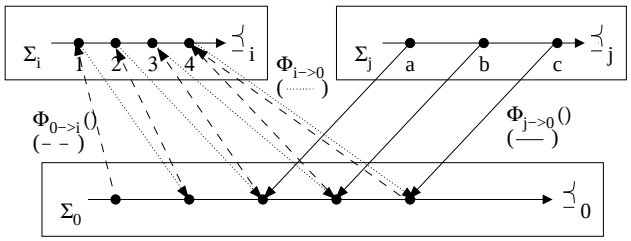

Figure 8: Illustration of Condition 1.

4.2 Safety condition for vectoring mode

Condition 1: $\forall i \in [1, m]$,
(a) $\Phi_{i\to 0} : \Sigma_i \to \Sigma_0$ is strictly increasing, i.e.,
$\forall \sigma_1, \sigma_2 \in \Sigma_i, \sigma_1 \prec_i \sigma_2 \Rightarrow \Phi_{i\to 0}(\sigma_1) \prec_0 \Phi_{i\to 0}(\sigma_2)$
(b) $\forall \sigma \in \Sigma_0$, $\sigma \preceq_0 \Phi_{i\to 0} \circ \Phi_{0\to i}(\sigma)$
where $\Phi_{i\to 0} \circ \Phi_{0\to i}(\sigma) = \Phi_{i\to 0}(\Phi_{0\to i}(\sigma))$

This condition stipulates that the conversion function $\Phi_{i\to 0}()$ maps distinct signatures of Σ_i into distinct values of Σ_0 in an order preserving manner (Condition 1a). In addition, the preference of a route should not decrease as it is redistributed (Condition 1b). Figure 8 illustrates the above conditions: The pair of conversion functions ($\Phi_{i\to 0}()$, $\Phi_{0\to i}()$) satisfies Condition 1.

Lemma 1: *Condition 1 guarantees that the relation $\preceq$ is a total pre-order over the set of signatures Σ.*

To prove Lemma 1, we need to show the relation $\preceq$ is reflexive, transitive, and total. For brevity, we omit the details of the proof.

Theorem 1: *If all algebras $A_1, A_2, ..., A_m$ are SM, then Condition 1 is a sufficient condition to guarantee the preservation of the SM property within and across the algebras, i.e., $\forall \lambda \in L$, $\forall \sigma \in \Sigma$, $\sigma \prec (\lambda \oplus \sigma)$, and thus guarantees routing safety.*

PROOF. To demonstrate that Condition 1 is a sufficient condition to preserve SM across the algebras, we assume a router receiving a route with signature σ, and we assume that the route is extended to an arc with a label λ. We show that assuming that all algebras $A_1, A_2, ..., A_m$ are SM, and that the conversion functions are compliant with Condition 1, then the extended route has a strictly lower preference than the initial route.

First, because $\sigma \in \Sigma$, and $\lambda \in L$, there exists $i, j \in [1, m]$ such that $\sigma \in \Sigma_i$ and $\lambda \in L_j$. We then distinguish two cases:
Case 1: $i = j$. The initial route is extended into the same routing algebra A_i. Then, since A_i is SM, we conclude that $\sigma \prec_i \lambda \oplus_i \sigma$, i.e., $\sigma \prec \lambda \oplus \sigma$ (according to Definitions 2 and 3).
Case 2: $i \neq j$. The initial route from A_i is extended into a different algebra A_j. Since A_j is SM, when the signature $\Phi_{0\to j} \circ \Phi_{i\to 0}(\sigma)$ is extended over the arc with label λ, its preference stricly decreases:
$\Phi_{0\to j} \circ \Phi_{i\to 0}(\sigma) \prec_j \lambda \oplus_j \Phi_{0\to j} \circ \Phi_{i\to 0}(\sigma)$
Then, by Condition 1(a),
$\Phi_{j\to 0} \circ \Phi_{0\to j} \circ \Phi_{i\to 0}(\sigma) \prec_0 \Phi_{j\to 0}(\lambda \oplus_j \Phi_{0\to j} \circ \Phi_{i\to 0}(\sigma))$
From Condition 1(b), we also have
$\Phi_{i\to 0}(\sigma) \preceq_0 \Phi_{j\to 0} \circ \Phi_{0\to j} \circ \Phi_{i\to 0}(\sigma)$
Since $\preceq_0$ is transitive, from the two above inequations
$\Phi_{i\to 0}(\sigma) \prec_0 \Phi_{j\to 0}(\lambda \oplus_j \Phi_{0\to j} \circ \Phi_{i\to 0}(\sigma))$
By definition of $\oplus$, we get
$\Phi_{i\to 0}(\sigma) \prec_0 \Phi_{j\to 0}(\lambda \oplus \sigma)$
Finally, by definition of $\preceq$, we conclude
$\sigma \prec \lambda \oplus \sigma$ □

Theorem 1 is an important result as it may allow us to classify certain conversion functions as safer than others. For example, it is straightforward to show that the constant conversion functions listed in Section 3.2 for modeling the current connecting primitives do not satisfy Condition 1. More important, this type of sufficient condition may guide us to design better connecting primitives.

We note that although we conceptualized the redistribution of a route as propagating the route over a virtual link in the previous section, route redistribution does not need to be SM. This is because the redistributed route is not eligible to be installed in the local FIB [22, 23].

4.3 Optimality condition for vectoring mode

Condition 2: $\forall i \in [1, m], \Phi_{0\to i} : \Sigma_0 \to \Sigma_i$ is increasing, i.e., $\forall \sigma_1, \sigma_2 \in \Sigma_0, \sigma_1 \preceq_0 \sigma_2 \Rightarrow \Phi_{0\to i}(\sigma_1) \preceq_i \Phi_{0\to i}(\sigma_2)$

Theorem 2: *If all algebras $A_1, A_2, ..., A_m$ are left-isotone, then Conditions 1 and 2 guarantee the preservation of the left-isotonicity property within and across the algebras, and thus guarantee path optimality.*

To prove the above result, we assume $l \in L$, and $\sigma_1, \sigma_2 \in \Sigma$ with $\sigma_1 \preceq \sigma_2$. There exists $i, j, k \in [1, m]$ such that $l \in L_i$, $\sigma_1 \in \Sigma_j$ and $\sigma_2 \in \Sigma_k$. We enumerate all possible cases (e.g., $i = j = k$, $i = j \neq k$, etc.), show that in each case, we have $l \oplus \sigma_1 \preceq l \oplus \sigma_2$. The details are omitted for brevity.

We note that if an algebra (e.g., EIGRP) violates the sufficient conditions and does not provide globally optimal paths [15], then we cannot provide global optimal paths across the routing instances.

4.4 Conditions for link-state mode

Condition 3:
(a) $\forall i \in [1, m]$, $\Phi_{i\to 0}$ is bijective and $\Phi_{0\to i} = \Phi^{-1}_{i\to 0}$
(b) $\forall i \in [1, m]$, $\Phi_{i\to 0}$ is homomorphic, i.e., $\forall \sigma_1, \sigma_2 \in \Sigma_i$,
$\Phi_{i\to 0}(\sigma_1 \oplus_i \sigma_2) = \Phi_{i\to 0}(\sigma_1) \oplus_0 \Phi_{i\to 0}(\sigma_2)$
(c) $\forall i \in [1, m]$, $\Phi_{0\to i}$ is homomorphic, i.e., $\forall \sigma_1, \sigma_2 \in \Sigma_0$,
$\Phi_{0\to i}(\sigma_1 \oplus_0 \sigma_2) = \Phi_{0\to i}(\sigma_1) \oplus_i \Phi_{0\to i}(\sigma_2)$

Proposition: *If all algebras A_0, A_1, ..., A_m are isotone, and for every i in [1,m], $\oplus_i$ is associative, then Conditions 1 to 3 guarantee the isotonicity property across the algebras and the associativity of $\oplus$, and thus guarantee routing safety and path optimality.*

To prove it, we assume $\sigma_1, \sigma_2, \sigma_3 \in \Sigma$. There exists $i, j, k \in [1, m]$ such that $\sigma_1 \in \Sigma_i$, $\sigma_2 \in \Sigma_j$ and $\sigma_3 \in \Sigma_k$. We enumerate all possible cases (e.g., $i = j = k$, $i = j \neq k$, etc.), and show that in each case, $\sigma_1 \preceq \sigma_2 \Rightarrow \sigma_1 \oplus \sigma_3 \preceq \sigma_2 \oplus \sigma_3$, and $(\sigma_1 \oplus \sigma_2) \oplus \sigma_3 = \sigma_1 \oplus (\sigma_2 \oplus \sigma_3)$. Again, the details are omitted.

The above conditions imply the algebras to be isomorphic to be connected in a link-state mode. It will be interesting to investigate whether these conditions can be weakened. We leave it for future work (Section 9).

4.5 Generalization to n-ary lexicographic products

The framework naturally extends to the more general case of n-ary lexicographic products of sub-algebras, by defining each A_i as the lexicographical product of n unary routing algebras: $\otimes(A_{i1}, A_{i2}, \ldots, A_{in})$ [18]. Formally, the ranking procedure of n-ary lexicographic products is defined as: $\forall\, i \in [1, m]$, $A_i = \otimes(A_{i1}, A_{i2}, \ldots, A_{in})$ with $\forall\, i \in [1, m]$, $\forall\, d \in [1, n]$, A_{id} being an unary routing algebra (L_{id},Σ_{id},ϕ_{id},$\oplus_{id}$, $\preceq_{id}$), and the relation $\preceq_i$ over Σ_i is defined as: $\forall\, \alpha = (\alpha_1, \alpha_2, \ldots, \alpha_n) \in \Sigma_i = (\Sigma_{i1}, \Sigma_{i2}, \ldots, \Sigma_{in})$,

$\forall\ \beta = (\beta_1, \beta_2, \ldots, \beta_n) \in \Sigma_i = (\Sigma_{i1}, \Sigma_{i2}, \ldots, \Sigma_{in})$,
$\alpha \preceq_i \beta \Leftrightarrow \exists e \in [1,n], \forall d < e, \alpha_d \sim_{id} \beta_d$ and ($\alpha_d \prec_{id} \beta_d$ or $(e = n$ and $\alpha_n \preceq_{in} \beta_n)$).

Section 3 defines a pair of conversion functions for each unary algebra. In this case, n pairs of conversion functions must be defined for A_i because each of its component A_{id}, $d = 1, ..., n$, requires a pair: (1) $\Phi_{id \to 0d}$: $\Sigma_{id} \to \Sigma_{0d}$, and (2) $\Phi_{0d \to id}$: $\Sigma_{0d} \to \Sigma_{id}$.

We can prove that if each pair of conversion functions satisfies Condition 1, and the collection of n sub-algebras satisfies the following condition, SM is preserved and thus, routing is safe across the routing instances when route redistribution is performed in a vectoring mode.

Condition 4: $\exists e \in [1,n], \forall i \in [1,m], \forall d < e$, A_{id} is either M or SM, and A_{ie} is SM.

$A_i = \otimes \quad (\underbrace{A_{i1}}_{\text{(M)}}, \underbrace{A_{i2}}_{\text{(M)}}, \underbrace{\ldots}_{\text{(M)}}, \underbrace{A_{ie}}_{\text{(SM)}}, \underbrace{\ldots}_{\text{Dont}}, \underbrace{A_{in}}_{\text{care}})$

5. DESIGN OF NEW PRIMITIVES

The theory presented in the previous two sections opens up new design possibilities, including the creation of a new class of connecting primitives that inherently conforms to the identified safety conditions. This section describes the details of one such design. While we can formally establish that the new primitives guarantee safety, the proof is omitted for space reasons.

5.1 Design considerations

Trading autonomy for expressiveness: This tradeoff is similar to what has been discovered for BGP [11]. By resetting the metric of a redistributed route in the new routing instance, the current design offers a high degree of autonomy for the participating routing instances. However, such metric reset results in a considerable loss of information which prevents network operators from achieving desirable design objectives. In this design, we choose conversion functions strictly according to safety conditions, which will introduce a strong dependency in the ranking of routes in different routing instances. The resulting solution is applicable to only networks where the routing instances are managed by a single operator or a team of cooperative operators. Many networks, including individual ISP networks, enterprise networks, campus networks, and military networks, fall into this category. In fact, our design is more desirable for such networks because it is more expressive and supports a large range of important operational goals, including those not achievable today. We will substantiate this point in Section 7.

Designing for incremental deployment: To enable incremental deployment, we target a design that requires no modification to existing routing protocols (e.g., BGP, EIGRP, OSPF, RIP, IS-IS). This decision introduces some complications that constrain our design space as follows. The theory assumes each routing protocol instance to be correct and concentrates on the requirements for conversion functions. As such, the theory requires each routing instance to satisfy the safety conditions (as defined in Sections 3 and 4). However, there are two clear violations of such conditions by current routing protocols.

First, OSPF External Type 2 routes do not comply with the SM condition: As explained in Section 2, OSPF does not increase the cost of an External Type 2 route while the route propagates. We solve this problem by imposing the following OSPF-specific restriction on the design:

1. Routes redistributed into OSPF are always set to External Type 1.

Second, prior work [17] has shown that BGP is not SM but can result in various routing instabilities. Making BGP compliant with safety conditions is more difficult to achieve since even the first attribute, *local-preference*, of the BGP route ranking criteria, is neither SM nor M. As such, BGP cannot be rendered safe by simply discarding specific options. Instead, we impose the following BGP-specific restrictions:

2. BGP routes are selected only if no route is offered by other protocols.

3. Routes from BGP cannot be redistributed into a non-BGP protocol instance.

These two restrictions enforce SM between non-BGP and BGP instances, and ultimately guarantee correct routing so long as the BGP configurations do not result in anomalies. Although the restrictions do not exist today, they do not prevent common design objectives from being accomplished. The first restriction makes BGP routes the least preferred routes. Indeed, when a router receives routes from both BGP and an IGP to a same destination prefix, the router should typically prefer the more direct internal route learned from the IGP over the external BGP route. This is because sending traffic through external networks (e.g., providers) can cost money. The question of whether the second restriction would prevent existing design objectives from being achieved is a more complicated one, and we defer the answer to Section 7.

Finally, we note that recent proposals [17] have suggested modifications to BGP which would guarantee important properties (e.g., SM), while at the same time, still support existing policies (e.g., customer-provider, peering, and sibling relationships). The modifications if implemented would eliminate the need to treat BGP separately.

5.2 Universal metric space

All non-BGP routing protocols (OSPF, IS-IS, RIP, EIGRP, static routes) can be unified under the following 2-ary metric space: {*type*, *cost*}. Derived from the theory, we make conversion functions an *explicit component* of the design. We treat each non-BGP protocol as a 2-ary algebra, where the first attribute is the route *type* and the second attribute is the route *cost*. We define the following universal metric space for the design of conversion functions:
1. Type: $\Sigma_{0,1} = \{\text{A, B, C}\}$. The universal domain for the type consists of three permitted elements, and is totally ordered with type A being preferred to type B which is in turn preferred to type C.
2. Cost: $\Sigma_{0,2} = \{1, 2, 3, \ldots, 2^{32} - 1\}$. The universal domain for the cost metric consists of the set of integers from 1 to $2^{32} - 1$ and is totally ordered by the arithmetic operator $\leq$.

5.3 New connecting primitives

When comparing routes received from different routing protocol instances, our design maps the type and cost of each route into the universal metric space according to the default conversion functions shown in Table 2[2]. It then ranks the routes based on their ordering in the universal metric space. Since RIP does not define a route type, all RIP routes are effectively of the same type "RIP". The same applies to static routes. The default conversion functions for the cost dimension (e.g., $x \to x^8$ for RIP) are designed to scale the metric space (e.g., 4-bit for RIP) of each protocol to the 32-bit value range of the universal metric space. For example, an

[2]Details for IS-IS are omitted because of the protocol's similarities with OSPF. Prohibited signatures are not listed for brevity.

Attribute	Protocol	To universal domain $\Phi_{protocol \to 0}$	From universal domain $\Phi_{0 \to protocol}$
Type	OSPF	intra-area → A inter-area → B external type 1 → C	* → external type 1
	RIP	RIP → C	* → RIP
	EIGRP	internal → B external → C	* → external
	Static	static → C	NA
Cost	OSPF	$x \to x^2$	$x \to$ ceiling($\sqrt{x}$)
	RIP	$x \to x^8$	$x \to$ ceiling($\sqrt[8]{x}$)
	EIGRP	$x \to x$	$x \to$ ceiling(x)
	Static	$x \to x$	NA

Table 2: Default conversion functions. The symbol "*" represents any permitted value.

OSPF route of type "intra-area" and cost "30" would be mapped into type "A" and cost "900" in the universal metric space. Similarly, an EIGRP route of type "internal" and metric "65345" would be mapped into type "B" and cost "65345". Since type A routes are preferred over type B routes, the OSPF route would be preferred.

The conversion functions in the other direction (i.e., from universal metric space to a protocol specific metric space) are needed for route redistribution. For example, let us assume that the OSPF route in the example above is being redistributed into RIP. It would be given a RIP hop-count of 3 because *ceiling*($\sqrt[8]{900}$) = 3.

It is straightforward to show that the default conversion functions comply with the identified safety conditions. As discussed in Section 7, these functions are sufficient for most of today's operational goals. Network operators may customize the conversion functions based on operational objectives subject to these constraints: (i) the new conversion functions comply with the safety conditions as defined in Section 4, and (ii) two routing processes at border routers must be configured with the same conversion functions if they belong to the same routing instance.

New route selection procedure: Like before, each routing process first determines a best route within its own RIB. For example, among all the received BGP routes, the BGP best path selection algorithm would choose a single most preferred BGP route. We note that currently, routers can run at most one instance of RIP and BGP but can run multiple processes of OSPF and EIGRP. Consequently, after each routing process has determined its best route, a router obtains at most one BGP route, but may receive multiple OSPF routes, each from a different OSPF process. To select one among them for the Forwarding Information Base (FIB), a router applies the following ranking rules in our design:

Step 1. Protocol: Prefer non-BGP (i.e., EIGRP, OSPF, RIP, static) routes to BGP route.

Step 2. Type: If multiple non-BGP routes are available, prefer type *A* routes, then type *B* routes, and type *C* last.

Step 3. Cost: Among non-BGP routes of the preferred route type, prefer the route with the lowest cost.

If only one route is in consideration and is from BGP, the process stops after rule 1 and selects the BGP route. Otherwise, it follows the ordering in the 2-ary universal space. Again, *step 1* of the proposed route selection procedure is created to handle the special case of BGP and to enforce the previous restrictions. Similarly, the following route redistribution procedure treats BGP differently.

New route redistribution procedure: The theory allows the redistribution to be performed in either a vectoring or a link-state manner. In this design, for brevity, we restrict route redistribution to the vectoring mode.

For redistribution between non-BGP instances (e.g., from OSPF into RIP), the metrics of the redistributed routes are decided by the conversion functions.

We disallow any redistribution from BGP into a non-BGP protocol instance as part of the BGP-specific restrictions defined in the beginning of this section. We allow great flexibility for redistribution into BGP since this is not part of the BGP-specific restrictions. When a route is redistributed into BGP, its BGP attributes (e.g., local preference, AS-PATH, MED, community, etc.) can be set to any value as long as they do not cause routing anomalies within BGP. This flexibility allows the new primitives to preserve the current levels of autonomy, expressiveness and privacy between BGP networks. Any policy (e.g., customer, provider, peer relationships) currently implemented between BGP networks can still be accomplished. In addition, networks administered by different authorities and connected through BGP do not need to share more information than today. In particular, they do not need to exchange information on the conversion functions.

6. IMPLEMENTATION

We have implemented the new connecting primitives into the XORP [3] routing software, version 1.6. Default conversion functions are exactly as defined in Section 5. It is straightforward to configure the new primitives because they primarily have a single tuning knob: the option to customize the conversion functions.

Custom conversion functions can be specified either in a static configuration file – which is loaded at XORP startup time – or dynamically through the XORP command line interface. They are defined through self-contained policy statements[3] such as the one shown below.

```
policy{
       f_rip_to_universal_type:  c
       f_rip_to_universal_cost_a:  1
       f_rip_to_universal_cost_b:  0
       f_rip_to_universal_cost_n:  7
}
```

The system verifies that the definitions comply with the sufficient conditions for safety before accepting them.

For the *cost* dimension, the conversion functions from a routing protocol specific metric (e.g., RIP hop-count) to the universal space are restricted to the form of "$a \times x^n + b$" (with a, n, and b being integer parameters), and the conversion functions in the other direction are simply the inverse. This restriction facilitates a straightforward algorithm to verify that the conversion functions comply with the desired conditions, while still enabling a wide range of operational goals, as illustrated in the next section. In future work, we will attempt to understand the full implication of this restriction and explore other forms of conversion functions.

It should be noted that neither EIGRP nor IS-IS is part of the implementation. EIGRP is a CISCO proprietary protocol and not supported by XORP. The XORP community has plans to add IS-IS to XORP but it is yet to be accomplished as of version 1.6.

[3]The syntax is similar to that of JUNOS. We note that an implementation in a Cisco IOS like environment would also be straightforward: Replace the existing *distance* and *default-metric* commands with a couple of new commands for customizing the conversion functions, and modify the *redistribute* command to remove those metric related parameters.

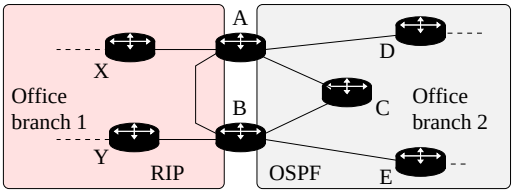

Figure 9: Illustration of domain backup.

7. VALIDATION OF EXPRESSIVENESS

We define expressiveness broadly as the ability of the primitives to support operational goals. In the section, we consider five operational objectives. The first three are regarded important requirements by operators [21], and we have conducted validation experiments with our XORP implementation for each of them. The fourth objective addresses the implications of the three restrictions introduced in Section 5.1 on the expressiveness of our design. Finally, the last objective illustrates the flexibility with which one can derive connecting primitives from the proposed theory.

In a related note, as explained in the last paragraph of Section 5, our design preserves the current levels of autonomy, privacy and expressiveness between BGP networks.

7.1 Domain backup

Domain backup designates the ability for a network to preserve reachability even in the event of a routing instance partition, through alternate physical paths traversing other routing instances. To illustrate the property, consider the network from Figure 9. It consists of two office branches, each running its own routing instance (RIP, OSPF). In the failure of router C, link A-C, or link B-C, the routers D and E can no longer directly communicate despite the existence of a physical path (D-A-B-E) between the two routers. By default, the path D-A-B-E is not offered as it traverses a different routing instance (RIP). To make this path available, mutual route redistribution should be enabled between OSPF and RIP at the border routers B and A, respectively. However, route redistribution at multiple points can easily result in routing anomalies [8]. Hence, to support domain backup, current route redistribution solutions require specific physical topologies and complex policies [21]: The routing instances must be connected in a star topology, and domain backup is provided *only* to the leaf routing instances.

In contrast, the new primitives can offer domain backup to every routing instance with no restriction on physical topology. We implemented the network in Figure 9. The border routers A and B perform mutual route redistribution between the RIP and OSPF instances, with the default conversion functions. We observed that, in the absence of failure, E receives two paths to D: E-B-C-A-D and E-B-A-D. E selects the first path to forward traffic to D as it is an intra route (type A in the universal metric space) whereas the other route is external (type C). Then, we simulated a failure of router C, link A-C, or link B-C. Despite the partition, routers E and D still preserved their connectivity through the path E-B-A-D.

7.2 Router-level shortest path routing across IGP instances

Router-level shortest path routing across IGP instances designates the ability for a pair of end hosts in different IGP instances to route traffic to each other along the shortest path. Today, this property is supported but only between OSPF instances. IOS provides the option to preserve the cost of a route redistributed from one

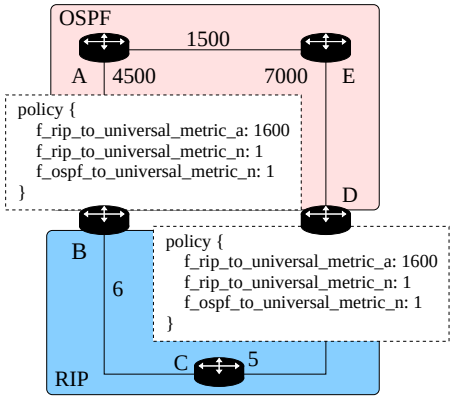

Figure 10: Illustration of router-level shortest path routing across OSPF and RIP instances.

OSPF instance into another OSPF instance. However, the current primitives do not permit router-level shortest path routing between two instances of different protocol types (e.g., OSPF and RIP). In such a setting, the cost of a redistributed route is set to a value with no relation to the cost of the initial route. The cost information to the destination is therefore lost at the first redistribution point. Even when redistributing between OSPF instances, the existing procedure has the following additional limitation. When a route from an OSPF instance is redistributed into a different OSPF instance , the cost can only be set to either an arbitrary value that is independent from the initial cost, or its original value. The current route redistribution procedure does not permit the cost of redistributed routes to be modified between the instances through a function. However, operational networks may rely on the OSPF cost to reflect the physical distance between routers. When different units are used in each instance (e.g., miles versus meters), router-level shortest path routing is not possible.

In contrast, with the new design, operators can specify their own conversion functions. The new primitives enable router-level shortest path routing between any pair of IGP instances. We implemented the network of Figure 10. It consists of two routing protocol instances: OSPF and RIP. Their routing metrics represent the physical distance in different units (e.g., meters and miles, respectively.) The described conversion functions at the border routers (B, D) allow geographical shortest path routing across the instances, despite the usage of different units in each domain. We modified the costs of the different links, and the shortest path (e.g., between A and C) was consistently selected.

7.3 Traffic engineering

Current traffic engineering techniques are only applicable within one IGP routing protocol instance [12] or between BGP domains [26]. Our primitives, with their support for router level shortest path routing, naturally extends traffic engineering across multiple routing instances without requiring any additional coordination between the instances.

To illustrate the existing limitations and newly supported capabilities, consider the example network depicted in Figure 11. The network is composed of three routing protocol instances (OSPF 10, RIP, OSPF 20). Network operators frequently adjust the IGP weights to minimize congestion. The adjustments of IGP weights aim at redirecting traffic over less congested links. However, this technique is currently applicable only within a single routing protocol instance. We assume that the link B-C is congested and its

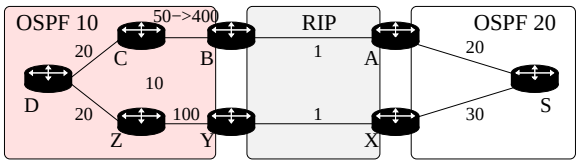

Figure 11: Illustration of traffic engineering across routing protocol instances.

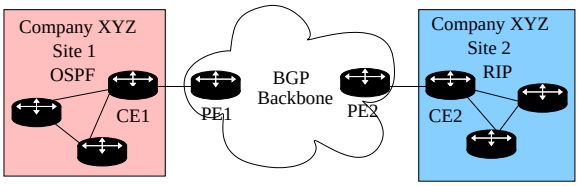

Figure 12: Illustration of typical VPN scenario.

Attribute	Protocol	To universal domain $\Phi_{protocol \to 0}$	From universal domain $\Phi_{0 \to protocol}$
Protocol	OSPF	OSPF $\to$ 254	* $\to$ OSPF
	RIP	RIP $\to$ 254	* $\to$ RIP
	EIGRP	EIGRP $\to$ 254	* $\to$ EIGRP
	static	static $\to$ 254	NA

Table 3: Additional default conversion functions for the new *protocol* attribute.

weight is therefore increased to a larger value. The goal is for senders (e.g., S) to select the less resource constrained paths (e.g., S-X-Y-Z-D). However, because redistributed routes are assigned a static metric values (e.g., at B, Y, A and X), the initial weight information is lost and senders may still select the congested paths (e.g., S-A-B-C-D). Although the metrics of redistributed routes could be updated at the border routers B and Y in times of congestion, the operators of OSPF 10 may have no control over the border routers A and X. As a consequence, congestion cannot be minimized across multiple routing instances. In comparison, our implementation eliminates this limitation.

We implemented the network in Figure 11. We did not modify the default conversions at the border routers. We observed that with the cost of B-C set to 50, S received two paths to D: D-C-B-A-S with a cost of 276, and D-Z-Y-X-S with a cost of 655. Because of its lower cost, S selected the first path to forward traffic to D. Then, after increasing the cost of B-C to 400, the cost of the route redistributed by A into OSPF was updated to 1296. Consequently, S switched to the second path to forward its traffic to D.

7.4 Virtual private networks

Section 5.1 introduces a limitation to BGP: Routes cannot be redistributed from BGP into an IGP. An important question that naturally ensues is: Does this new restriction prevent existing design objectives from being achieved?

Empirical studies [21] have revealed that network often inject routes from BGP into an IGP, especially in VPN deployments. As illustrated in Figure 12, a company (e.g., XYZ) may have multiple sites (e.g., Site 1, Site 2) with their own routing protocol (e.g., RIP, OSPF). To allow connectivity between the sites, the company relies on a service provider backbone. Routes from one site (e.g., OSPF routes from Site 1) are first redistributed into the backbone (i.e., BGP cloud) at an provider edge (PE) router (e.g., PE 1). The routes are propagated through the BGP backbone, and then redistributed from BGP into the IGP of each remote site (e.g., RIP from Site 2) at the connecting PEs (e.g., PE 2).

The fact that the new primitives prevent this type of redistribution may therefore be a serious impediment to its adoption. However, it turns out that the same objective can readily be achieved without any redistribution from BGP into IGP. For simple scenarios, a customer edge (CE) router (e.g., CE 1) can originate a default route in the respective site's IGP (e.g., Site 1's OSPF), and be configured with a static route pointing to the connecting PE (e.g., PE 1) for the default route (0.0.0.0/0). As an alternative, BGP can also be deployed in the company's sites. Then, operators simply need to redistribute the IGP routes into BGP at each site, and control the route propagation through BGP policies. In future work, we will seek to further understand the full impact of the OSPF and BGP restrictions stipulated in our design on network operations.

7.5 Strict preference policy

The current route selection allows routers to strictly prefer routes from one protocol instance over another, e.g., *"Always prefer OSPF routes to RIP routes"*. This type of policy might be useful to implement blackholes (e.g., in the event of DDoS). This section illustrates how our design can be extended to support such strict preference policy. Every non-BGP routing protocol instance is modeled as a 3-ary routing algebra: {*protocol*, *type*, *cost*}. The new *protocol* attribute is first considered when comparing non-BGP routes. Its has an integer range from 1 to 255 in the universal metric space, with 255 corresponding to the prohibited path. EIGRP, OSPF, RIP and static routes are defined to be of *protocol* type "EIGRP", "OSPF", "RIP" and "static", respectively. Table 3 presents the additional default conversion functions. All protocols are equally preferred by default. This design extension supports strict preference policies in addition to the previously presented objectives. To specify a strict preference for a protocol instance, simply override that instance's conversion function in the *protocol* dimension at all its border routers, e.g., from "OSPF $\to$ 254" to "OSPF $\to$ 10". Doing so will not result in routing anomalies as long as the new set of conversion functions conform to Condition 1.

8. RELATED WORK

A large body of work exists on the correctness of routing. However, most of prior work considered a specific protocol at a time. For RIP, the focus was on solving the "count to infinity" problem. For OSPF, special attention was given to its stability issues [4, 27]. For BGP, various causes for potential routing anomalies have been identified, followed by the development of thorough analytical models and solutions. The insights gained from these efforts led researchers to explore design principles towards the creation of a safer inter-domain protocol [16, 19, 10, 11] and more abstractly, develop unifying algebraic frameworks identifying fundamental properties a vector or link-state routing protocol must satisfy to ensure correct behaviors [29, 17]. In fact, prior to these results, researchers had already started adopting an algebraic approach to routing [7, 13, 14]. However, these algebraic structures may abstract away too many routing protocol specific dynamics and some possess properties that are not realistic for contemporary routing protocols. For example, neither IGRP nor BGP satisfies the distributivity property required by dioids.

For routing across multiple routing protocol instances, several analytical models were recently introduced [22, 23], enabling a rigorous analysis of the current design of connecting primitives and exposing its deficiencies. These models also made the formulation of practical configuration guidelines possible [20]. However,

this approach is inherently backward-looking. The models only apply to existing solutions, and the derived guidelines further restrict the expressiveness of the already rigid current primitives. Recently, two researchers [6] have proposed a new algebraic approach based on idempotent semirings to model routing, including the case across multiple routing instances. While the approach is general and promising, it models only route redistribution, not AD, nor the interaction of route redistribution and AD, which was left as future work. The closest related work to our proposal is the "metric transformations" introduced by Mills and Braun in 1987 [25]. This concept is similar to our notion of "conversion functions" as it permits to exchange routes between routing instances and derive the metric of the newly redistributed routes. Understandably, Mills and Braun focused on routing protocols with only a single metric (e.g., RIP) and with a distance vector mode of computation between the routing instances. In comparison, our formal framework applies to routing protocols with multiple criteria, allows a link-state mode of redistribution across the routing instances, and identifies sufficient conditions for routing safety and optimal paths.

9. DISCUSSION

Several important questions still need to be investigated. On the theory front, can we find safety conditions without requiring all border routers of the same routing instance to use identical conversion functions? As the fine-grained effects of different path computation algorithms on routing correctness are better understood [30], can we weaken the sufficient conditions for the conversion functions? In addition, we note that while the notion of conversion functions is general and can model the existing mechanisms, our theory concentrates on routing protocols that rank routes based on a lexical product of multiple attributes. All existing routing protocols indeed fall into this category. However, should new routing protocols with different ways of ranking routes emerge, new sufficient conditions may need to be derived. On the design front, there may be important operational requirements that are little known outside the selected operational communities. How do we collect them if this is indeed the case? Furthermore, how can we anticipate requirements that may arise in the future?

10. CONCLUSION

We have presented a new theory to reason about the safety of routing across multiple routing instances. In addition, we identify as set of conditions for the connecting primitives to guarantee correct routing and optimal paths. The second part of the paper describes an application of the theory to create a new set of connecting primitives that are much safer and more flexible than the currently deployed version. We assumed no changes to the specifications of the existing routing protocols, and we demonstrate that with very minimum changes to how they should be configured, the new primitives can not only support existing operational objectives but also enable new functions that are important but not feasible today, all the while guaranteeing routing safety. In the big picture, our effort can be viewed as another example that underscores the importance and feasibility of principled design, which we believe can help all phases of network operations.

11. ACKNOWLEDGEMENT

We thank the SIGCOMM TPC and Joao Sobrinho for their constructive comments on early drafts of the work. Yi Zhuang and Aditya Bhave contributed to the XORP implementation. This research was partially supported by the NSF under the 100x100 project [1] (ANI-0331653), the 4D project [2] (ANI-0520187 & ANI-0520210), grant CNS-0721574 and a graduate research fellowship. Views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of NSF or the U.S. government.

12. REFERENCES

[1] 100x100 Clean Slate Project. www.100x100network.org.
[2] 4D Project. www.cs.cmu.edu/∼4D.
[3] XORP: eXtensible Open source Routing Platform. www.xorp.org.
[4] A. Basu and J. G. Riecke. Stability Issues in OSPF Routing. In *ACM SIGCOMM*, 2001.
[5] T. Benson, A. Akella, and D. Maltz. Unraveling the Complexity of Network Management. In *USENIX NSDI*, 2009.
[6] J. N. Billings and T. G. Griffin. A Model of Internet Routing Using Semi-modules. In *International Conference on Relational Methods in Computer Science*, 2009.
[7] B. Carré. *Graphs and Networks*. Oxford University Press, 1979.
[8] Cisco. OSPF Redistribution Among Different OSPF Processes, 2006.
[9] Cisco. What Is Administrative Distance?, March 2006.
[10] N. Feamster, H. Balakrishnan, and J. Rexford. Some Foundational Problems in Interdomain Routing. In *HotNets*, 2004.
[11] N. Feamster, R. Johari, and H. Balakrishnan. Implications of Autonomy for the Expressiveness of Policy Routing. In *ACM SIGCOMM*, 2005.
[12] B. Fortz, J. Rexford, and M. Thorup. Traffic Engineering With Traditional IP Routing Protocols. In *IEEE Communication Magazine*, 2002.
[13] M. Gondran and M. Minoux. *Graphs and Algorithms*. Wiley, 1984.
[14] M. Gondran and M. Minoux. *Graphs, Dioids, and Semirings : New Models and Algorithms*. Springer, 2008.
[15] M. G. Gouda and M. Schneider. Maximizable Routing Metrics. In *IEEE ICNP*, 1998.
[16] T. G. Griffin, A. D. Jaggard, and V. Ramachandran. Design Principles of Policy Languages for Path Vector Protocols. In *ACM SIGCOMM*, 2003.
[17] T. G. Griffin and J. L. Sobrinho. Metarouting. In *ACM SIGCOMM*, 2005.
[18] A. Gurney and T. G. Griffin. Lexicographic Products in Metarouting. In *ICNP*, 2007.
[19] A. D. Jaggard and V. Ramachandran. Robustness of Class-Based Path-Vector Systems. In *IEEE ICNP*, 2004.
[20] F. Le and G. Xie. On Guidelines for Safe Route Redistributions. In *ACM INM Workshop*, 2007.
[21] F. Le, G. Xie, D. Pei, J. Wang, and H. Zhang. Shedding Light on the Glue Logic of the Internet Routing Architecture. In *ACM SIGCOMM*, 2008.
[22] F. Le, G. Xie, and H. Zhang. Understanding Route Redistribution. In *IEEE ICNP*, 2007.
[23] F. Le, G. Xie, and H. Zhang. Instability Free Routing: Beyond One Protocol Instance. In *ACM CoNEXT*, 2008.
[24] D. Maltz, G. Xie, J. Zhan, H. Zhang, G. Hjalmtysson, and A. Greenberg. Routing Design in Operational Networks: A Look from the Inside. In *ACM SIGCOMM*, 2004.
[25] D. Mills and H. Braun. The NSFNET Backbone Network. In *ACM SIGCOMM*, 1987.
[26] B. Quoitin, C. Pelsser, L. Swinnen, O. Bonaventure, and S. Uhlig. Interdomain Traffic Engineering with BGP. In *IEEE Communication Magazine*, 2003.
[27] A. Shaikh, C. Isett, A. Greenberg, M. Roughan, and J. Gottlieb. A Case Study of OSPF Behavior in a Large Enterprise Network. In *IMW*, 2002.
[28] J. Sobrinho. Algebra and Algorithms for QoS Path Computation and Hop-by-Hop Routing in the Internet. In *IEEE INFOCOM*, 2001.
[29] J. L. Sobrinho. Network Routing With Path Vector Protocols: Theory and Applications. In *ACM SIGCOMM*, 2003.
[30] J. L. Sobrinho and T. G. Griffin. Routing in Equilibrium. In *Mathematical Theory of Networks and System*, 2010.