The Impact of Communication Models on Routing-Algorithm Convergence

Aaron D. Jaggard
Rutgers University
adj@dimacs.rutgers.edu

Vijay Ramachandran
Colgate University
vijayr@cs.colgate.edu

Rebecca N. Wright
Rutgers University
rebecca.wright@rutgers.edu

Abstract

Autonomous routing algorithms, such as BGP, are intended to reach a globally consistent set of routes after nodes iteratively and independently collect, process, and share network information. Generally, the important role of the mechanism used to share information has been overlooked in previous analyses of these algorithms. In this paper, we explicitly study how the network-communication model affects algorithm convergence. To do this, we consider a variety of factors, including channel reliability, how much information is processed from channels, and how many channels are processed simultaneously. Using these factors, we define a taxonomy of communication models and identify particular models of interest, including those used in previous theoretical work, those that most closely model real-world implementations of BGP, and those of potential interest for the design of future routing algorithms. We characterize an extensive set of relationships among models in our taxonomy and show that convergence depends on the communication model in nontrivial ways. These results highlight that certain models are best for proving conditions that guarantee convergence, while other models are best for characterizing conditions that might permit nonconvergence.

1. Introduction

Autonomous routing protocols (epitomized by BGP [14] but also including, *e.g.*, SPVP [9] and protocols in the Metarouting framework [10]) are distributed algorithms that establish network connectivity when coordination among network entities is low. Much work has been done to analyze the behavior of such protocols, including giving examples of possible protocol divergence and proving sufficient conditions for protocol convergence. This work has used a variety of models for the communication between protocol participants; however, the possible impact of the communication model on the theoretical results has generally been neglected, so it was previously unknown whether this might be a significant factor. We address this issue comprehensively by defining a taxonomy of communication models for routing protocols and showing that theoretical convergence results depend on the choice of communication model. We then consider the possible pairs of communication models from our taxonomy and prove an extensive set of results showing when protocol convergence or divergence results proved in one of the models necessarily hold in the other. Our work thoroughly explores the design space defined by our taxonomy; different points in this space correspond to different models used in previous research and also to different configuration settings in real-world networks. Here we identify previously unstudied models that we believe are important because they both better capture the flexibility of actual protocols and have a strong ability to simulate algorithm behavior in other models.

Distributed autonomous routing algorithms iteratively compute path assignments. They promote a consistent view of the network without overwhelming it with unnecessary information. The essential steps in such an algorithm can be abstractly captured as the following actions, repeatedly performed by each node:

Action (1): Collect updates of path information describing feasible paths to a destination node d.

Action (2): Choose the most preferred path (according to local policy) from the paths known at the node.

Action (3): Announce any changes in path choice to neighboring nodes.

However, as we investigate, these descriptions of actions do not specify some aspects of the communication model that can have an impact on an algorithm's outcome.

Protocol designers and network operators hope that when a distributed autonomous routing protocol is run, it will eventually converge to a stable and consistent path assignment. However, previous work ([16], followed by a line of work analyzing BGP convergence, including [4], [6], [8], [9], [15]) has shown that divergence, caused by routing oscillations, is possible. Different work in this area has made different assumptions about how nodes execute Actions (1)–(3) above. Furthermore, these differences can be realized in practice (*e.g.*, the BGP specification [14] allows flexibility in how actions (1) and (3) are carried out); as we discuss below, we may view these differences as assuming different

Partially supported by the DIMACS Special Focus on Algorithmic Foundations of the Internet, by NSF awards CNS–0753061, CNS–0753492, and DMS–0239996, and by ONR award N00014–05–10818.

Appears in *Proceedings of the 29th International Conference on Distributed Computing Systems (ICDCS)*, 2009.

properties of the communication channels between nodes.

Another important consideration is that routing algorithms may be run on networks that do not guarantee reliable message delivery (*e.g.*, in many wireless networks, or if TCP is not used). In such cases, communications between nodes running a protocol can be arbitrarily delayed or even lost. Thus, we consider here both differences in how Actions (1)–(3) are carried out as well as the reliability of message delivery.

More specifically, in this paper we define and explore the space of communication models using three dimensions that correspond to answers to the following questions:

- Are updates reliably delivered?
- From how many neighbors are updates collected in Action (1) above?
- How many updates from each neighbor are collected in Action (1) above?

The space defined by these dimensions provides a taxonomy that systematically captures various combinations of assumptions about synchrony, atomicity, and network delays. Several points in the taxonomy map to realistic differences in current and future network infrastructure (*e.g.*, underlying transport-level protocols, next-generation physical-layer properties, and protocol timer settings). The space also includes both reliable and unreliable channels. To our knowledge, this is the first work to consider the algorithmic properties of interdomain routing over unreliable channels.

Considering the various communication models in our taxonomy, we are particularly interested in whether a routing algorithm that is guaranteed to converge in a network (perhaps satisfying certain conditions) using one model is also guaranteed to converge if that network uses a different communication model. To this end, we formally define different notions of realizing an execution of one model in another and prove an extensive set of realization relationships between different pairs of models. We show that for many pairs of the models in our taxonomy, all executions (in particular, all divergent executions) in one model can be realized in some form in the other; however, we also show that there are some pairs of models for which this is not true. This work enriches the important line of work understanding the convergence behavior of BGP and other networking protocols, *e.g.*, [3]–[6], [8]–[10], [15].

Our results imply that there are "weak" models that are useful for exhibiting oscillations (so that divergence is then possible in all other models as well) and "strong" models that are useful for proving guarantees of convergence (because guarantees in those models carry over to other models as well). We demonstrate that previous analyses of BGP in particular, while not explicitly considering the effects of communication models, satisfy our guidelines to ensure applicability across communication models.

By demonstrating which models are appropriate for which types of analysis results, our work can guide the future analysis of routing protocols. Our results here also contribute to protocol analysis by demonstrating that some types of divergence are not possible in some models and by identifying important models in our taxonomy that are realistic and powerful but that have not been considered before. Finally, our results imply that, in this framework, reliable channels offer little benefit over unreliable channels for the purpose of guaranteeing convergence. However, always having access to the current network state (instead of sequentially processing update messages from a queue) can help guarantee convergence.

To summarize, our main contributions are as follows:

- We define a taxonomy of communication models (Sec. 2.2).
- We identify particular points of interest in our taxonomy, including "polling," "message-passing," and "queueing" models (Sec. 2.3).
- We prove an extensive set of relationships between the different models in our taxonomy (Sec. 3).
- Our results demonstrate that convergence, including for well-known examples from the existing literature, *does* depend on the choice of communication model.

We begin with some background and technical preliminaries in Sec. 2.1. We discuss related work, including the relationship of our work to classical results about modeling in the distributed computing literature, in Sec. 4.

2. Communication Models

2.1. Problem basics and algorithm

We focus on using distributed autonomous routing algorithms to solve the Stable Paths Problem (SPP) [9], an abstract representation of the interdomain-routing problem. An instance of SPP contains an undirected graph $G = (V, E)$ with a distinguished *destination node* d and, for each node $v \in V$, a set of *permitted paths* $\mathcal{P}_v$, which is a subset of all simple paths from v to d, and a ranking function $\lambda_v : \mathcal{P}_v \to \mathbb{N}$ indicating v's preference for each permitted path. Paths with lower rank are more preferred. Ties in ranking are not permitted except when two paths go through the same neighbor. The problem is to find a *path assignment* $\pi = \{\pi_v\}_{v \in V}$ that, for each $v \neq d$, is (1) *consistent*—assuming that the next hop along π_v is u, we have that $\pi_v = v\pi_u$ (if v extends a path from u to d, then that path from u to d is assigned to u)—and (2) *stable*—for all neighbors $w \neq u$ of v, $\lambda_v(v\pi_u) < \lambda_v(v\pi_w)$, *i.e.*, π_v is more preferred than any other $v\pi_w$. We assume that $\pi_d = d$.

Let $\mathcal{C}$ be the set of communication channels that might be used; for each edge $\{u, v\}$ in the undirected instance graph, $\mathcal{C}$ contains directed channels (u, v) and (v, u). We assume

that each channel is FIFO, so that messages that are written by u (and no other party) to the channel (u, v) and are not dropped by the channel are processed by v in the same order in which they were written. We also assume that a single communication model is used throughout the network.

Definition 2.1 (Components of network state)**.** We keep track of various components of the network state, although for relationships between models we focus on the path assignments in particular. Each aspect of state depends on the step t of the algorithm's execution. The components we consider are:

Path assignments. $\pi_v(t)$ is the path to the destination that v chooses at the end of step t; we let $\pi(t) = \{\pi_v(t)\}_{v \in V}$ be the collection of all path assignments. Note that $\pi_v(t+1) = \pi_v(t)$ unless v runs the update algorithm in step $t+1$. We let $\pi_v(0) = \epsilon$ for $v \neq d$ and $\pi_d(0) = d$.

Known routes. After the execution of the first part of the algorithm in step t, $\rho_v(c; t)$ contains the contents of the last update that v successfully processed from the channel c. $\rho_v(c; t+1) = \rho_v(c; t)$ unless v updates from channel c in step $t+1$. We let $\rho_v(c; 0) = \epsilon$ for every $v \in V$ and $c \in \mathcal{C}$.

Channel contents. For a channel $c = (u, v)$, we let $c(t)$ be the contents of c at the beginning of step t; let $m_c(t)$ be the number of messages in c at the beginning of step t. We will use $c_i(t)$ to denote the i^{th} message in c at the beginning of step t, where the first message is the oldest. We let $c(0) = \emptyset$ for every $c \in \mathcal{C}$.

In each iterative step of the algorithm, nodes (1) collect information from channels, (2) choose route objects based on their ranking function and permitted paths, and (3) share information by writing route objects to channels. The communication models we study here affect only the first of these three actions. Note that our definition of autonomous routing algorithm and our results are broad enough to cover any protocol that can be abstractly modeled by the Simple Path-Vector Protocol (SPVP) [9], including BGP.

We use an *activation sequence* to specify the nodes involved in each round and how the first action is executed; an activation sequence determines the algorithm's execution. We now define the most general notion of activation sequence; the different models we consider can be viewed as different restricted classes of activation sequences.

Definition 2.2. An activation sequence α is a function on the non-negative integers that assigns to each $t \in \mathbb{Z}_{\geq 0}$ a quadruple (U, X, f, g) such that:

- $U \subseteq V$ is the set of nodes that will update in step t;
- $X \subseteq \mathcal{C}$ is the set of channels that will be updated in step t. For each $c = (u, v) \in \mathcal{C}$, we require that $v \in U$, *i.e.*, the receiving end of each channel is one of the nodes that is updating in step t;
- $f : X \to \mathbb{Z}_{\geq 0} \cup \{\infty\}$ indicates how many messages from each channel should be processed. For $c = (u, v) \in X$, v will process $f(c)$ messages from c (if $f(c) = \infty$, then v will process all messages in the channel); and
- $g : X \to \mathcal{P}(\mathbb{Z}_{>0})$ indicates which, if any, messages will be dropped from each channel; the elements of $g(c)$ are the indices of the messages that will be dropped so if, *e.g.*, $2 \in g(c)$, then the second message in c will be dropped. We require that if $f(c) = 0$, then $g(c) = \emptyset$, and if $0 < f(c) < \infty$, then $g(c) \subseteq \{1, 2, \ldots, f(c)\}$.

Definition 2.3 (Algorithm execution using a general activation sequence)**.** Given a network instance and an activation sequence α, the iterative routing algorithm executes as follows, starting with $t = 0$.

1) Let $(U, X, f, g) = \alpha(t)$.
2) For each $v \in U$ and $u \in \mathcal{N}(v)$ such that $(u, v) \in X$
   a) Let $c = (u, v)$
   b) If $f(c) = \infty$, let $i = m_c(t)$; if $f(c) < \infty$, let $i = \max\{f(c), m_c(t)\}$.
   c) If $\{1, 2, \ldots, i\} \setminus g(c) \neq \emptyset$, let j be the largest element of this set, and let $\rho_v(c; t)$ be the route in the j^{th} message in c. If $\{1, 2, \ldots, i\} \setminus g(c) = \emptyset$, let $\rho_v(c; t) = \rho_v(c; t-1)$.
   d) Delete the first i messages from c, and set $m_c(t+1) = m_c(t) - i$.
3) For each $v \in U$, set $\pi_v(t)$ to be the most preferred path from the set $\{\rho_v((u, v); t) \cap \mathcal{P}_v \mid u \in \mathcal{N}(v)\}$ if $v \neq d$, and let $\pi_v(t) = d$ otherwise.
4) For each $v \in U$ and $u \in \mathcal{N}(v)$, if $\pi_v(t) \neq \pi_v(t-1)$ and if prescribed by export policy, write the path $\pi_v(t)$ to the channel (v, u) and increase $m_{(v,u)}(t+1)$ by one.
5) Increment t and repeat from Step 1.

Definition 2.4 (Fair activation sequence)**.** We say that an activation sequence α is *fair* if every node tries to read each of its channels infinitely often and, if a message is dropped from channel c (a possibility when unreliable channels are used), then there is a later message on c that is not dropped.

We will restrict our attention to fair activation sequences.

Definition 2.5 (Convergence of an activation sequence)**.** An activation sequence α *converges to the path assignment* π if, for the induced sequence of path assignments $\pi(t)$, there exists some t^* such that for any $t' > t^*$, $\pi(t') = \pi$. We write $\lim_{t \to \infty} \pi(t) = \pi$, and say α *converges* if the limit exists.

2.2. Dimensions of the model space

The quadruples that appear in a general activation sequence suggest four dimensions that might be considered in studying different models of communication: channel reliability, the number of neighbors processed at each step, the

number of messages processed per channel, and the number of nodes updating per step. We present these dimensions and points along those dimensions in order to describe a complete space. Our results focus on relationships among a natural subset of points in this space.

Definition 2.6 (Dimensions of the model space)**.** The four dimensions we study are:

Channel reliability: In general, channels may have some probability at which messages are lost and may deliver messages out-of-order, *e.g.*, if the underlying transport mechanism cannot be TCP. In this paper, we assume FIFO channels and we consider channels that are either reliable or unreliable in the following sense:

- R **(Reliable)** Every message placed in a channel (u, v) by u is always read by v, *i.e.*, the functions g_v in the fourth component of an activation sequence entry are always identically equal to $\emptyset$.
- U **(Unreliable)** Some messages placed in channels are not read, *i.e.*, the functions g_v need not be identically equal to $\emptyset$.

Number of neighbors processed: Each model specifies how many channels a node should process when it updates. Practically, this dimension affects whether or not announcement import and export should be treated as an atomic action. The possible values are:

- E **(Every)** Whenever a node updates, it processes messages from every one of its neighbors, *i.e.*, $X_v = \mathcal{N}(v)$ for every $v \in U$.
- M **(Multiple)** Whenever a node updates, it processes messages from some subset of its neighbors (potentially multiple neighbors), including the possibility of processing *no* channels and that of processing *all* channels; this imposes no additional restriction on X_v.
- 1 Whenever a node updates, it processes messages from exactly one of its neighbors, *i.e.*, $|X_v| = 1$ for every $v \in U$.

Number of messages processed per channel: Each model specifies how many messages a node should read from a channel when it processes that channel. Practically, this dimension covers variation on link delay (which may cause different schedules for arrival of messages) and protocol-timer settings, *e.g.*, the BGP MRAI timer [14], which sets the amount of time a router must wait before sending out a current-route update.

- A **(All)** Whenever a node v updates and is assigned to process messages from a neighbor u, v processes all of the messages in this channel, *i.e.*, $f_v \equiv \infty$.
- S **(Some)** There are no restrictions on the number of messages that a node processes from each of its neighbors when it updates.
- F **(Forced)** Each node is forced to process at least one message from each of the neighbors from which it updates (although this set may omit some neighbors), but it may process multiple messages from each neighbor.
- O **(One)** Whenever a node v updates and is assigned to process messages from a neighbor u, it processes exactly one message from this channel, *i.e.*, $f_v \equiv 1$. If unreliable channels are used, this message may be lost.

Number of nodes updating: An activation sequence must specify the set of nodes that update at each step (the *activated* nodes). Various possible values are:

- **Every** Every node updates at every step, *i.e.*, $U = V$;
- **Unrestricted** There are no restrictions on which nodes do or do not update, although to avoid trivial steps we assume $U \neq \emptyset$; and
- **One** Exactly one node updates at each step, *i.e.*, $|U| = 1$.

In the remainder of this paper, we require exactly one node to update at each step. That is, we do not consider variations in this dimension.

The symbols for each option in the first three dimensions (R, E, A, *etc*.) are used to abbreviate the combinations; thus RMO is the model that uses reliable channels in which the node that updates in a given step processes an arbitrary set of its incoming channels and processes exactly one message from each of these channels.

2.3. Specific models

We now highlight some models that are of particular interest because of their previous use in research, their fidelity to the current BGP specification [14], or their potential for guiding the design of new routing protocols. (We note that although the BGP specification [14] assumes that BGP runs on TCP, thus ensuring reliable delivery, protocols like BGP are being designed for various purposes [10]. Our work permits analysis of convergence even when only unreliable channels are available.) As noted above, we restrict our attention to models in which exactly one node updates in every step, *i.e.*, $|U| = 1$ in every activation sequence quadruple.

2.3.1. Polling models. Informally, in "polling" models, nodes learn the current state of (some or all of) their neighbors in the first algorithm action when activated. The REA polling model was used in [3], [4]; as we show, this allows the hardness results in that work to apply regardless of model used. The REA polling model has also been used in the application of mechanism design to routing (*e.g.*, [7], [12]), although the primary focus of that work is not algorithm convergence.

Because all messages in a channel are processed before the active node chooses its best route, the active node essentially ignores any intermediate messages and only uses each neighbor's most recent announcement. (We note that in settings in which nodes can misbehave, nodes might use the additional information from intermediate messages to modify their actions; otherwise, only the most recent information is useful in route choice.)

Because of the constraints on activation sequences that define polling models, we may simply abbreviate the elements of the activation sequence as the updating node and the neighbors from which it updates. If REA is being used, we may further abbreviate this to just the node itself (writing, *e.g.*, $\alpha(t) = v$), because all channels are read.

Informally, we call R1A "poll one," RMA "poll some," and REA "poll all."

2.3.2. Message-passing models. "Message-passing" models were used in the original definition and proof of NP-completeness of SPP [9]. In these models, when a node is assigned to update, it reads (or possibly drops if unreliable channels are used) one message from each channel being read. As we show in Sec. 3, any nonconvergent algorithm execution in any of the other models we consider can also occur in the message-passing models R1O and RMO (but not necessarily in the model REO, in which a node essentially acts on the first message in every nonempty channel simultaneously). The message-passing model R1O can be thought of as an "event-driven" model, in which nodes respond individually to each incoming update, or in which node activation is triggered by an announcement.

2.3.3. Queueing models. The "queueing" models RMS and UMS have not, to our knowledge, been used in earlier work on path-vector protocols. We believe these are of particular interest because the flexibility of configuration parameters in the BGP specification [14] suggests that these models most naturally correspond to correct operation of BGP on the Internet. Furthermore, we show that they are very strong in their ability to realize the other models in our taxonomy.

There is one queueing model for each option of channel reliability; each model then allows any number of channels to be processed in a single step, and any number of messages to be processed from each channel (so that X and f are unrestricted and either $g \equiv \emptyset$ or g is also unrestricted).

3. Realization Relationships between Communication Models

In this section, we analyze the algorithm executions that can occur in the various models generated by the dimension values outlined in the previous section. We begin by defining types of relationships between models and then go on to prove how an algorithm execution in one model can be seen as an "equivalent" execution in another.

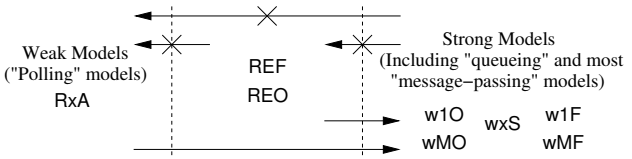

Figure 1. Highlights of the major results separating different classes of communication models. Arrows denote preservation of the oscillations of one collection of models by another; the models on the right capture all the oscillations of the other models shown. The crossed-out arrows indicate that the models on the right can oscillate in ways the models in the middle and on the left cannot; similarly, the models in the middle can oscillate in ways those on the left cannot. Taken together, these show that the models in the left and middle sections are strictly weaker than the models on the right.

Figure 1 gives a high-level overview of our main results. Arrows extend from one collection of models to another if the second collection preserves the oscillations of the first collection of models (in the sense of Def. 3.1); crossed-out arrows indicate that such preservation does not hold. In particular, the models on the right, which include the queueing models and most of the message-passing models, can exhibit all of the oscillations seen in the models on the left and in the middle. In fact, one of the main results of this section is that these models capture the protocol executions of every other model in the space (including those not depicted here). The models on the right can oscillate in ways that the models in the middle cannot, and the models on the right and in the middle can oscillate in ways that the polling models on the left cannot. As a result, the models on the right are strictly stronger than all the other models depicted. Our full results, including some not depicted in Fig. 1, are described below; additionally, the exact relationships between some of the models in the collections shown is still open. (Here, w ranges over $\{\mathsf{R}, \mathsf{U}\}$ and x ranges over $\{1, \mathsf{M}, \mathsf{E}\}$.)

3.1. Notions of realization

In studying the effects of communication models, we are interested in the following relationship between models.

Definition 3.1 (Oscillation preservation)**.** We say that model B *preserves the oscillations of* model A if the existence of a nonconvergent activation sequence α in A for some network instance I implies that there exists an activation sequence α' in B for network instance I that does not converge.

Note that this definition does not insist that the set of nodes involved in an oscillation in A be identical to the set of nodes in an oscillation in B, but just that whenever A has some oscillation, B does as well.

While the oscillation-preserving relationship is weaker than most of the relationships that we actually prove, it captures the essential impact of model choice on convergence. Most of our results involve the following realization relations, which imply oscillation preservation.

Definition 3.2 (Execution realization)**.** We say that model B *realizes the executions of* model A (*exactly*, *exactly with repetition*, or *as subsequences*) if, for every network instance using B and activation sequence α (in B), there exists an activation sequence α' in the A such that, if $\{\pi(t)\}_t$ and $\{\pi'(t)\}_t$ are the path-assignment sequences induces by α and α', one of the following holds:

Exact realization: $\forall t$, $\pi'(t) = \pi(t)$, *i.e.*, the sequences are the same.

Exact realization with repetition: $\exists f : \mathbb{N} \to \mathbb{N}$ such that $\forall i, j$, $i < j \Rightarrow f(i) < f(j)$ and $f(t) \leq k < f(t+1) \Rightarrow \pi'(k) = \pi(t)$, *i.e.*, $\{\pi'(t)\}_t$ is obtained from $\{\pi(t)\}_t$ by replacing each $\pi(t)$ with one or more occurrences of $\pi(t)$.

Realization as a subsequence: $\exists f : \mathbb{N} \to \mathbb{N}$ such that $\forall i, j$, $i < j \Rightarrow f(i) < f(j)$ and $\forall t$, $\pi'(f(t)) = \pi(t)$, *i.e.*, $\{\pi(t)\}_t$ is a subsequence of $\{\pi'(t)\}_t$.

Saying that model A realizes model B (in one of these senses) could also be thought of as asserting that for every execution of a system under model B, there is an execution of the system under model A such that a specific type of simulation relation (as in, *e.g.*, [13]) can be defined from the first execution to the second.

It is straightforward from these definitions that exact realization implies exact realization with repetition, which in turn implies realization as a subsequence. If one model realizes the executions of another, in any of these senses, it immediately follows that the first model preserves the oscillations of the second model.

3.2. Foundational positive results

Here we present our positive results, which show that certain models may be realized in various senses by other models. We build on these results by combining them with each other, our negative results in Sec. 3.3, and additional arguments to obtain a broad range of relationships as described in Sec. 3.5. We have omitted some proofs; these are available in the longer version of the paper [11]. We start with some general exact realizations.

Proposition 3.3. *For every* $w \in \{\mathsf{R}, \mathsf{U}\}$, $x \in \{1, \mathsf{M}, \mathsf{E}\}$, *and* $y \in \{\mathsf{O}, \mathsf{S}, \mathsf{F}, \mathsf{A}\}$*:*

1) $\mathsf{U}xy$ *exactly realizes* $\mathsf{R}xy$*;*

2) $wx\mathsf{S}$ *exactly realizes* $wx\mathsf{F}$*;*

3) $wx\mathsf{F}$ *exactly realizes* $wx\mathsf{O}$ *and* $wx\mathsf{A}$*; and*

4) $w\mathsf{M}y$ *exactly realizes* $w1y$ *and* $w\mathsf{E}y$*.*

Proposition 3.4. *For every* w, $w\mathsf{ES}$ *exactly realizes* $w\mathsf{MS}$*.*

The following general result involves realization with repetition. In light of Prop. 3.8, it cannot be strengthened, at this level of generality, to use exact realization.

Theorem 3.5. *For every* $w \in \{\mathsf{R}, \mathsf{U}\}$, $y \in \{\mathsf{O}, \mathsf{S}, \mathsf{F}, \mathsf{A}\}$, $w1y$ *realizes* $w\mathsf{M}y$ *with repetition.*

Proof: For fixed $w \in \{\mathsf{R}, \mathsf{U}\}$ and $y \in \{\mathsf{O}, \mathsf{S}, \mathsf{F}, \mathsf{A}\}$, let α be a fair activation sequence in $w\mathsf{M}y$. For each element $(\{v\}, X, f, g)$ in $\{\alpha(t)\}$ we produce a sequence of quadruples that are legal entries in activation sequences (for this network) in $w1y$; replacing each element of $\{\alpha(t)\}$ by its corresponding sequence produces a complete, valid activation sequence in $w1y$ that meets the claimed conditions. We assume that all channels in the $w1y$ system contain exactly the messages, and in the same order, that appear in the $w\mathsf{M}y$ system; this is true before the system starts, and each sequence of steps in the $w1y$ system (corresponding to a single step in the $w\mathsf{M}y$ system) preserves this.

Let c be the channel from which v learns the path it selects after step t (in the $w\mathsf{M}y$ system) and let d be the channel from which it learns the path it selects after step $t-1$ (if $t > 0$). Order the channels in X as $c_1, \ldots, c_k$ so that, if $c \neq d$, $c_1 = c$ (if $c \in X$) and $c_k = d$ (if $d \in X$); if $c = d \in X$, then let $c_1 = c$ if the path chosen after step t is higher-ranked than the path chosen after step $t-1$, and let $c_k = d$ if the opposite is true. The requisite sequence of quadruples is $(\{v\}, \{c_1\}, f'_1, g'_1), \ldots, (\{v\}, \{c_k\}, f'_k, g'_k)$ where $f'_i(c_i) = f(c_i)$ and $g'_i(c_i) = g(c_i)$. Before proceeding through any of these steps, the $w1y$ system (by induction) uses the path $P = \pi_v(t-1)$. After proceeding through these steps, the $w1y$ system uses the path $Q = \pi_v(t)$ because it knows the same set of routes that the $w\mathsf{M}y$ system does after step t. We claim that, if $P \neq Q$, then at some point in this sequence of steps, the new system switches from P to Q and that this is the only change in path assignment that it makes. If v did not already know Q before it starts this sequence of steps, it learns Q in the first step unless it prefers P to Q and it learns both P and Q from the same channel c. (In that case, all paths that v learns in the intermediate steps are less preferred than Q, and thus P, which it knows about until the final step, when it learns—and switches to—P; this case thus satisfies the claim.) Any new paths that v learns in intermediate steps must be less preferred than Q; if there are old paths that are more preferred than Q, these are less preferred than P and are thus never chosen (because P is then chosen until the last step). Thus the claim holds. □

We also have the following positive results. In view of Cor. 3.14, the one for reliable channels cannot be strength-

ened to realization with repetition.

Proposition 3.6. R1O *realizes* R1S *as a subsequence, and* U1O *realizes* U1S *with repetition.*

Proof: For the reliable channel case, we let α be a fair activation sequence in R1S. For each element $(\{v\},\{c\},f,g)$ in $\{\alpha(t)\}$ we define a sequence of quadruples that are legal entries in activation sequences (for this network) in R1O; replacing each element of $\{\alpha(t)\}$ by its corresponding sequence produces a valid activation sequence in R1O that meets the claimed conditions. In the R1O system, we informally "flag" some messages to indicate the ends of sequences of messages that are processed in a single update in the R1S system.

If $f(c) > 0$, let $k \geq 1$ be the number of messages in c, after the steps in the R1O system that correspond to steps $0,\ldots,t-1$ in the R1S system, up to and including the $f(c)^{\text{th}}$ "flagged" message. Replace $(\{v\},\{c\},f,g)$ with k copies of $(\{v\},\{c\},f',g')$, where $f'(c)=1$ and $g'(c)=\emptyset$ as required by R1O. Finally, flag the update messages that v sends after processing the k^{th} of these updates. If $f(c)=0$, then we delete $(\{v\},\{c\},f,g)$ and do not replace it with anything. The resulting sequence is a legal activation sequence in R1O. The R1O system may have various path assignments while processing the k update commands, but after processing the final one, the set of known paths at v (and at all other nodes, none of which have updated during this time) is the same as at v in the R1S system after step t, so the path assignments are the same. As a result, we see that R1O realizes R1S as a subsequence.

For unreliable channels, given a fair activation sequence α in U1S, we replace $(\{v\},\{c\},f,g)$ in $\{\alpha(t)\}$ with nothing if $f(c)=0$ and, if $f(c)=k>0$, with $(\{v\},\{c\},f'_1,g'_1),\ldots,(\{v\},\{c\},f'_k,g'_k)$, where $f'_i(c)=1$ for every i and $g'_i(c)=\emptyset$ if i is the largest element of $\{1,\ldots,k\}$ that is not in $g(c)$ and $g'_i(c)=\{1\}$ otherwise (including if no such integer exists). This means that the only messages not dropped in the U1O system are exactly the ones that the U1S system actually uses (*i.e.*, that contribute known paths at the end of a step); thus the U1O system may repeat path assignments, but it does not transition through other path assignments. This shows that U1O realizes U1S with repetition. □

The following property is important because it demonstrates realization of the unreliable case by the reliable case; this leads to the fact that dropping reliability from the strong models does not, by itself, introduce new oscillations. The proof of this theorem relies on the fact that every dropped message is followed (eventually) by a non-dropped message in the definition of a fair activation sequence. Without this assumption, however, R1S would not even preserve the oscillations of U1O.

Theorem 3.7. R1S *exactly realizes* U1O.

Proof: Let α be a fair activation sequence in U1O; each element is of the form $(\{v\},\{c\},f,g)$, where $f(c)=1$ and $g(c)$ is either $\emptyset$ or $\{1\}$. Construct α' by: replacing all elements $(\{v\},\{c\},f,g)$ for which $g(c)=\{1\}$ with $(\{v\},\{c\},f',g')$, where $f'(c)=0$ and $g'(c)=\emptyset$, and replacing those elements for which $g(c)=\emptyset$ with $(\{v\},\{c\},f',g')$, where: $f'(c)\geq 1$ equals the number of elements $(\{v\},\{c\},f,g)$ in α (including the current one) since the previous such element with $g(c)=\emptyset$; and $g'(c)=\emptyset$. This is then a fair activation sequence in R1S, and the sequence of path assignments it induces is the same one induced by α, because in any channel there is always a non-dropped message after every dropped message. □

3.3. Negative results

Our negative results show that certain models cannot be realized, in various senses, by other models. These are proved by exhibiting a network and an activation sequence from the first model that produces behavior that provably cannot be seen in the second model (either at all, to show that oscillations cannot be preserved, or without the addition of repeated or other states, to show that certain other realization relations do not hold). Due to space considerations, we omit these examples and the associated proofs; these can be found in the longer version of the paper [11].

Proposition 3.8. REO *cannot be exactly realized in* R1O.

Theorem 3.9. *The oscillations of* R1O *are not preserved by* REO, REF, R1A, RMA, *and* REA.

Theorem 3.10. *The oscillations of* REO *and* REF *are not preserved by* R1A, RMA, *and* REA.

Proposition 3.11. REA *cannot be realized with repetition in* R1O.

Proposition 3.12. REA *cannot be exactly realized by* R1S.

Proposition 3.13. REO *cannot be exactly realized by* R1S.

3.4. Transitivity

As noted in Sec. 3.1, exact realization is stronger than realization with repetition, which in turn is stronger than realization as a subsequence, which in turn is stronger than the preservation of oscillations. If a model M_2 realizes a model M_1 in sense R_1, and M_3 realizes model M_2 in sense R_2, then we have that M_3 must also realize M_1 in the weaker of the senses R_1 and R_2 (although it might also realize M_1 in a stronger sense).

Conversely, if M_2 realizes M_1 in sense R_1 but M_3 cannot realize M_1 in sense R_2, and if R_1 is at least as strong as R_2, then M_3 also cannot realize M_2 in sense R_2. (Otherwise,

the composition of the realization of M_1 in M_2 and the realization of M_2 in M_3 would produce a realization of M_1 in M_3 in sense R_2.) Similarly, if M_3 realizes M_1 in sense R_1 but M_3 cannot realize M_2 in sense R_2, and if R_1 is at least as strong as R_2, then M_1 also cannot realize M_2 in sense R_2. (Again, transitivity would lead to a realization of M_2 in M_3 in sense R_2.)

Cor. 3.14 is a detailed example of application of these arguments to Prop. 3.11. Section 3.5 presents all such corollaries in a condensed form.

Corollary 3.14. *For every y, y', z with $z \neq \mathsf{O}$, $\mathrm{R}yz$ cannot be realized with repetition in $\mathrm{R}y'\mathsf{O}$.*

Proof: By Thm. 3.5, R1O realizes RMO (and thus, by Prop. 3.3, REO) with repetition. As this is at least as strong as the sense of non-realization in Prop. 3.11, for every $y' \in \{1, \mathsf{M}, \mathsf{E}\}$, $\mathrm{R}y'\mathsf{O}$ cannot realize REA with repetition. Repeated applications of Prop. 3.3 show that for $y \in \{\mathsf{M}, \mathsf{E}\}$ and $z \in \{\mathsf{S}, \mathsf{F}, \mathsf{A}\}$, $\mathrm{R}yz$ realizes REA exactly; by Thm. 3.5, we then have that $\mathrm{R1}z$ realizes REA with repetition for $z \in \{\mathsf{S}, \mathsf{F}, \mathsf{A}\}$. The arguments above and the first observation imply that for every $y, y' \in \{1, \mathsf{M}, \mathsf{E}\}$ and $z \in \{\mathsf{S}, \mathsf{F}, \mathsf{A}\}$, $\mathrm{R}y'\mathsf{O}$ cannot realize $\mathrm{R}yz$ with repetition. □

3.5. Discussion

Table 1 summarizes the results stated in Sec. 3.2–3.3 along with various corollaries obtained from these by the transitivity arguments above. The entry whose row is labeled with model A and whose column is labeled with model B indicates what is known about B's ability to realize A. If the entry is 4, then B exactly realizes A; if the entry is 3, then B realizes A with repetition; if the entry is 2, then B realizes A as a subsequence. A $\geq$ symbol indicates that the value is a lower bound and the relationship could be stronger, while a $\leq$ symbol indicates an upper bound, so the relationship could be weaker. In cases where both upper and lower bounds are known, we list all of the possible relationships. If the entry is -1, then B does not preserve the oscillations of A. We note that the because $\mathrm{R}yz$ is exactly realized by $\mathrm{U}yz$, the entry in row $\mathrm{U}yz$ and column $xy'z'$ is at most the entry in row $\mathrm{R}yz$ and column $xy'z'$. Blank entries indicate pairs for which the relationship is unknown.

It is important to note how strong our results show the queueing models to be. RMS is able to realize all reliable channel models exactly and all unreliable channel models either with repetition or exactly. UMS is able to exactly realize all models, reliable and unreliable. Furthermore, among the reliable channel models, R1O, RMO, R1S, RMS, RES, R1F, and RMF are all able to capture all of the oscillations of all other models in our taxonomy. In contrast, REO, REF, R1A, RMA, and REA are provably unable to capture some oscillations that may occur when using other models, so conditions that guarantee convergence in these models may not do so in general.

The BGP specification [14] outlines technical details of interdomain route computation, but leaves some room for interpretation. As an example, the amount of time nodes wait before sending route announcements can vary. In some cases, longer wait times may slow BGP convergence because nodes' discovery of potential routes is delayed; in other cases, longer wait times may hasten convergence because nodes do not waste resources on spurious or transient announcements. Our work studies these types of effects, and some of the identified models represent various interpretations of the BGP specification.

The *Route Refresh Capability* [2], which is an optional extension to BGP, can be used to learn, immediately and on-demand, a node's current route choice. Although intended to prevent information loss after a policy change, nodes can use this capability to poll nodes for the most current update. This permits implementation of the polling models, and the BGP specification alludes to this possibility. Our results imply that certain types of nonconvergence cannot be realized in these polling models.

4. Related Work

Nodes' ranking functions and permitted paths are independently configured inputs, and previous work showed that BGP may not converge on certain inputs [16]. Attempting to characterize these inputs has given some necessary conditions [4] and some sufficient conditions [6], [8], [9], [15] on the inputs for convergence, but not a complete characterization. These results were proved in differing communication models. For example, Feamster, Johari, and Balakrishnan [4] used a polling model. In contrast, other work [6], [8], [9], [15] assumed a queueing or message-passing model. In addition to these results about policy constraints, some hardness results are also known. Using a queueing model, Griffin, Shepherd, and Wilfong [9] showed the decision problem of whether an SPP instance has a stable, consistent path assignment is NP-complete. Using a polling model, Fabrikant and Papadimitriou [3] showed that determining whether BGP converges on a succinctly-represented problem instance is PSPACE-complete. Finally, previous work on game-theoretic models of BGP (*e.g.*, [7], [12]) have assumed polling models. Our results relating various models, including those used in previous work, suggest which models are best used for continuing this line of work. For example, our results imply that although the sufficient condition in [9] was shown in a queueing model, the condition also guarantees convergence in a polling model.

Many general results in distributed computing [13] consider assumptions about network communication, including atomicity and synchrony. In contrast to those general results,

	R1O	RMO	REO	R1S	RMS	RES	R1F	RMF	REF	R1A	RMA	REA
R1O	–	4	-1	4	4	4	4	4	-1	-1	-1	-1
RMO	3	–	-1	3	4	4	3	4	-1	-1	-1	-1
REO	3	4	–	3	4	4	3	4	4	-1	-1	-1
R1S	2	2	-1	–	4	4	$\geq$2	$\geq$2	-1	-1	-1	-1
RMS	2	2	-1	3	–	4	2,3	$\geq$2	-1	-1	-1	-1
RES	2	2	-1	3	4	–	2,3	$\geq$2	-1	-1	-1	-1
R1F	2	2	-1	4	4	4	–	4	-1	-1	-1	-1
RMF	2	2	-1	3	4	4	3	–	-1	-1	-1	-1
REF	2	2	$\leq$2	3	4	4	3	4	–	-1	-1	-1
R1A	2	2	$\leq$2	4	4	4	4	4		–	4	
RMA	2	2	$\leq$2	3	4	4	3	4		3	–	
REA	2	2	$\leq$2	3	4	4	3	4	4	3	4	–
U1O	$\geq$2	$\geq$2	-1	4	4	4	$\geq$2	$\geq$2	-1	-1	-1	-1
UMO	2,3	$\geq$2	-1	3	$\geq$3	$\geq$3	2,3	$\geq$2	-1	-1	-1	-1
UEO	2,3	$\geq$2		3	$\geq$3	$\geq$3	2,3	$\geq$2		-1	-1	-1
U1S	2	2	-1	$\geq$3	$\geq$3	$\geq$3	$\geq$2	$\geq$2	-1	-1	-1	-1
UMS	2	2	-1	3	$\geq$3	$\geq$3	2,3	$\geq$2	-1	-1	-1	-1
UES	2	2	-1	3	$\geq$3	$\geq$3	2,3	$\geq$2	-1	-1	-1	-1
U1F	2	2	-1	$\geq$3	$\geq$3	$\geq$3	$\geq$2	$\geq$2	-1	-1	-1	-1
UMF	2	2	-1	3	$\geq$3	$\geq$3	2,3	$\geq$2	-1	-1	-1	-1
UEF	2	2	$\leq$2	3	$\geq$3	$\geq$3	2,3	$\geq$2		-1	-1	-1
U1A	2	2	$\leq$2	$\geq$3	$\geq$3	$\geq$3	$\geq$2	$\geq$2				
UMA	2	2	$\leq$2	3	$\geq$3	$\geq$3	2,3	$\geq$2		$\leq$3		
UEA	2	2	$\leq$2	3	$\geq$3	$\geq$3	2,3	$\geq$2		$\leq$3		

	U1O	UMO	UEO	U1S	UMS	UES	U1F	UMF	UEF	U1A	UMA	UEA
R1O	4	4		4	4	4	4	4				
RMO	3	4		$\geq$3	4	4	$\geq$3	4				
REO	3	4	4	$\geq$3	4	4	$\geq$3	4	4			
R1S	$\geq$3	$\geq$3		4	4	4	$\geq$3	$\geq$3				
RMS	3	$\geq$3		$\geq$3	4	4	$\geq$3	$\geq$3				
RES	3	$\geq$3		$\geq$3	4	4	$\geq$3	$\geq$3				
R1F	$\geq$3	$\geq$3		4	4	4	4	4				
RMF	3	$\geq$3		$\geq$3	4	4	$\geq$3	4				
REF	3	$\geq$3		$\geq$3	4	4	$\geq$3	4	4			
R1A	$\geq$3	$\geq$3		4	4	4	4	4		4	4	
RMA	3	$\geq$3		$\geq$3	4	4	$\geq$3	4		$\geq$3	4	
REA	3	$\geq$3		$\geq$3	4	4	$\geq$3	4	4	$\geq$3	4	4
U1O	–	4		4	4	4	4	4				
UMO	3	–		$\geq$3	4	4	$\geq$3	4				
UEO	3	4	–	$\geq$3	4	4	$\geq$3	4	4			
U1S	$\geq$3	$\geq$3		–	4	4	$\geq$3	$\geq$3				
UMS	3	$\geq$3		$\geq$3	–	4	$\geq$3	$\geq$3				
UES	3	$\geq$3		$\geq$3	4	–	$\geq$3	$\geq$3				
U1F	$\geq$3	$\geq$3		4	4	4	–	4				
UMF	3	$\geq$3		$\geq$3	4	4	$\geq$3	–				
UEF	3	$\geq$3		$\geq$3	4	4	$\geq$3	4	–			
U1A	$\geq$3	$\geq$3		4	4	4	4	4		–	4	
UMA	3	$\geq$3		$\geq$3	4	4	$\geq$3	4		$\geq$3	–	
UEA	3	$\geq$3		$\geq$3	4	4	$\geq$3	4	4	$\geq$3	4	–

Table 1. Summary of realization results. Section 2.2 describes the meaning of the models (*i.e.*, the row and column headings). Section 3.5 describes the meaning of the entries.

we do not attempt to simulate an algorithm in one model by a modified algorithm in another. Instead, we are interested in properties of the executions of the *same* algorithm in a variety of models. Rather than general simulation, we define the more specialized notion of a realization relationship (Sec. 3) and apply it to routing-algorithm convergence. Recently, Chambart and Schnoebelen [1] studied the impact of channel reliability on the ability of an algorithm execution to transition between two states; their results do not directly apply to our context because state transitions do not preclude the possibility of nonconvergence.

5. Conclusions and Future Work

In this paper, we have systematically defined a taxonomy of communication models that play an important, but generally overlooked, role in the convergence behavior of distributed autonomous routing algorithms. We have defined new realization relationships and given an extensive characterization of the relationship between different pairs of these models. We have shown that queueing models are very strong, in that they can capture the algorithm behavior seen in all the other models. As such, they are useful for proving guarantees of convergence that transcend model differences. In contrast, polling models are strictly weaker than other models in that they do not capture all the algorithm behavior that may be seen using other models (and in the real world). These models are useful for providing examples of divergence, but not for proving generalizable convergence guarantees.

There are important questions remaining in this line of work. We have not extensively studied models in which multiple nodes are activated simultaneously. Polling in that case is strictly stronger than the polling models we consider here; we do not expect this to be true of the other models. The question of algorithm behavior in the context of mixed channels may also be interesting. Although unreliable channels can model reliable channels—so our results for unreliable channels apply to a mixture of reliable and unreliable channels—we do not have results when, for example, some nodes poll and others act on messages. Unreliable delivery in the case of out-of-order message also remains open. Our current work also does not comprehensively consider expected convergence when there is a probability distribution on message loss, rather concentrating on worst-case analysis. Finally, the question of behavior in the presence of misbehaving nodes is important. We assume that nodes use the most recent information they have to execute the algorithm when activated; however, in adversarial settings, nodes may use the other messages read to act in different ways that may affect algorithm convergence.

References

[1] P. Chambart and P. Schnoebelen. Mixing Lossy and Perfect FIFO Channels. In *Proc. CONCUR*, pp. 340–355, 2008.

[2] E. Chen. Route Refresh Capability for BGP–4. RFC 2918, Sep. 2000.

[3] A. Fabrikant and C. H. Papadimitriou. The Complexity of Game Dynamics: BGP Oscillations, Sink Equilibria, and Beyond. In *Proc. SODA*, pp. 844–853, Jan. 2008.

[4] N. Feamster, R. Johari, and H. Balakrishnan. Implications of Autonomy for the Expressiveness of Policy Routing. *IEEE/ACM Trans. Net.*, 15(6):1266–1279, Dec. 2007.

[5] L. Gao, T. Griffin, and J. Rexford. Inherently Safe Backup Routing with BGP. In *INFOCOM*, pp. 547–556, Aug. 2001.

[6] L. Gao and J. Rexford. Stable Internet Routing Without Global Coordination. *IEEE/ACM Trans. Net.*, 9(6):681–692, December 2001.

[7] S. Goldberg, S. Halevi, A. D. Jaggard, V. Ramachandran, and R. N. Wright. Rationality and Traffic Attraction: Incentives for Honest Path Announcements in BGP. In *Proc. ACM SIGCOMM*, pp. 267–278, Aug. 2008.

[8] T. G. Griffin, A. D. Jaggard, and V. Ramachandran. Design Principles of Policy Languages for Path Vector Protocols. In *Proc. ACM SIGCOMM*, pp. 61–72, Aug. 2003.

[9] T. G. Griffin, F. B. Shepherd, and G. Wilfong. The Stable Paths Problem and Interdomain Routing. *IEEE/ACM Trans. Net.*, 10(2):232–243, Apr. 2002.

[10] T. G. Griffin and J. L. Sobrinho. Metarouting. In *Proc. ACM SIGCOMM'05*, pp. 1–12, Aug. 2005.

[11] A. D. Jaggard, V. Ramachandran, and R. N. Wright. The Impact of Communication Models on Routing-Algorithm Convergence. DIMACS Tech. Rep. 2008–06, Nov. 2008.

[12] H. Levin, M. Schapira, and A. Zohar. Interdomain Routing and Games. In *Proc. STOC*, pp. 57–66, May 2008.

[13] N. A. Lynch. *Distributed Algorithms*. Morgan Kaufmann Publishers, 1996.

[14] Y. Rekhter, T. Li, and S. Hares. A Border Gateway Protocol 4 (BGP–4). RFC 4271, Jan. 2006.

[15] J. L. Sobrinho. An Algebraic Theory of Dynamic Network Routing. *IEEE/ACM Trans. Net.*, 13(5):1160–1173, Oct. 2005.

[16] K. Varadhan, R. Govindan, and D. Estrin. Persistent route oscillations in inter-domain routing. *Comp. Networks*, 32(1):1–16, Jan. 2000.