Quantifying Differences Between Batch and Streaming Detection of Internet Outages

Erica Stutz
Swarthmore College
Swarthmore, PA, USA

John Heidemann
USC/Information Sciences Institute
and USC/Thomas Lord Dept. of CS
Marina del Rey and Los Angeles, CA, USA

Yuri Pradkin
USC/Information Sciences Institute
Marina del Rey, CA, USA

***Abstract*—A number of different systems today detect outages in the IPv4 Internet, often using active probing and algorithms based on Trinocular's Bayesian inference. Outage detection methods have evolved, both to provide results in near-real-time, and adding algorithms to account for important but less common cases that might otherwise be misinterpreted. We compare two implementations of active outage detection to see how choices to optimize for near-real-time results with *streaming* compare to designs that use long-term information to maximize accuracy using *batch* processing. Examining 8 days of data, starting on 2021-02-26, we show that the two similar systems agree most of the time, more than 84%. We show that only *0.2% of the time the algorithms disagree*, and 15% of the time only one reports. We show these differences occur due to streaming's requirement for rapid decisions, precluding algorithms that consider long-term data (days or weeks). These results are important to understand the trade-offs that occur when balancing timely results with accuracy. Beyond the two systems we compare, our results suggest the role that algorithmic differences can have in similar but different systems, such as the several implementations of Trinocular-like active probing today.**

I. Introduction

Businesses, schools, and governments rely on the Internet for daily operations and communications [30], [26], [33]. Today the Internet is very reliable, in general, but outages do occur, due to natural disasters such as hurricanes [19], [18], human disasters such as wars [22], and everyday human error at ISPs [29], [28], [8], [9].

Several groups measure Internet outages to understand Internet reliability. Outage detection began with weather-focused measurements [24] and today includes active measurement of millions of networks in the IPv4 Internet [19], passive analysis of darknet data [11], [12], and analysis of CDN data [23], often processing in *batches* of an hour, days, or a few months. Comparison of these different methods has shown strong agreement of the core results [25], [23], but also pointed conditions where outages are misreported [1], [23], prompting algorithmic improvements that provided confidence to increase coverage [4]. Often, resolving incorrect cases requires observations over longer periods of time (a few hours or days) and from different vantage points, as more information can clarify conditions that would otherwise be ambiguous.

Streaming outage detection can help assess disaster response, often reporting data in near-real-time with updates every ten minutes or so. As data is generated, streaming systems quickly output estimates of the most recent state, often in as little as 20 to 60 minutes. Today, websites report country-wide [10] and geographic [3] outages in near-real-time. However, rapid (near-real-time) responses are incompatible with algorithms that employ observations over hours or days.

Contributions: This paper's goal is to quantify differences between batch-processed and streaming outage detection. Our first contribution is to directly compare batch and streaming systems using the same observations and similar but not identical algorithms. We find that *batch and streaming agree a great majority of the time (84%, Table III)*, suggesting streaming results are generally trustworthy.

Our second contribution is to examine the causes of the 16% of differences. In particular, we want to understand when differences are concentrated in specific types of networks, or by which specific algorithmic choices. We show that actual disagreements account for a very small fraction of time: *only about in 0.2% of overall duration do streaming and batch outage report results differ*. The remaining 15% of time only one system reports, while the other reports it is uncertain about network status. We show that most of these differences are because streaming omits several algorithms that require long-term data. These differences show the "cost" of near-real-time reporting is a few false outages in specific blocks.

While our results are specific to comparing batch and streaming results in two versions of Trinocular, we suggest they provide insight into other outage detection systems as well. They confirm the importance of specialized algorithms to correct for conditions that occur in a few blocks that would otherwise result in noticeable numbers of false outages, confirming prior observations [23], [4]. They also suggest that alternative implementations of Trinocular's active probing likely also will exhibit small but noticeable differences in reporting. Such differences motivate careful comparison of systems to build confidence in core results through independent methodology, and to understand the conditions where any algorithm (or omission of an algorithm) yields false results.

Data availability and ethics: All batch-processed data used here is available at no cost [2], and we will provide the streaming data for the period we analyze as well. Our work poses no ethical concerns. We re-analyze existing data, and since outage data begins with pings to specific IP addresses

978-3-903176-74-4

(a potential privacy concern), we evaluate the data only at the block-level and thus, have no information about individuals.

II. Review of Outage Detection

Our goal is to compare batch and streaming, near-real-time (NRT) reports of Internet outage detection, to understand the "cost" of running in NRT. For concreteness, we examine data from Trinocular [19], an outage detection system operating continuously since Oct. 2014. We next summarize Trinocular, the specifics of its batch and NRT implementations, and how these results apply to other outage detection systems.

II-A Trinocular's Core Algorithms

Trinocular detects outages in millions of address blocks, each a /24 IPv4 prefix, when active probes (ICMP echo requests, or "pings") no longer reach addresses in the block [19]. It checks the reachability of each of the approximately 5M /24 blocks every 11 minutes, a Trinocular probing *round*.

Each Trinocular observer minimizes the number of queries it sends to each block each round based on Bayesian inference assessing block reachability based on query responses and its knowledge of long-term block responsiveness. Reducing traffic to each block is designed to avoid any possible stress on the target network—when published, each Trinocular instance added only 1% to typical background traffic (and today, there is much more background traffic on the Internet). Minimizing traffic is also important to reduce time and traffic spent measuring, so that one computer can easily reach 5M blocks every 11 minutes. Trinocular sends between 1 and 16 queries to each block each round, stopping when it is confident in its assessment of block reachability. Typically one positive response confirms the block is up, and requires several negative responses to conclude the block is not reachable.

A complete system runs Trinocular observers from multiple, physically distributed locations, then integrates the results. By combining observations of the same target blocks from multiple locations, it becomes possible to distinguish network problems near the target block from problems near a prober.

In addition to the collection-time Bayesian algorithm, Trinocular uses several algorithms (Table I) that integrate information over time and space, sometimes changing specific states.

II-B Algorithms In Batched Outage Detection

Trinocular began as a batch-processing system: each observer would be started at a different site, where it would run for days or months and saving its results locally [19]. Trinocular then fetches results from each observer and centrally processes observations from all sites with all Trinocular algorithms (Figure 1). Batching data retrieval and processing is efficient, and batch analysis allows algorithms to consider three months of observations at the same time.

Several of the Trinocular algorithms take advantage of this long-term perspective to make the best decision about network status for each instant. Sometimes these decisions consider only a small time window: Hole-Filling resolves uncertainty in one round by looking at the prior and next results. Other

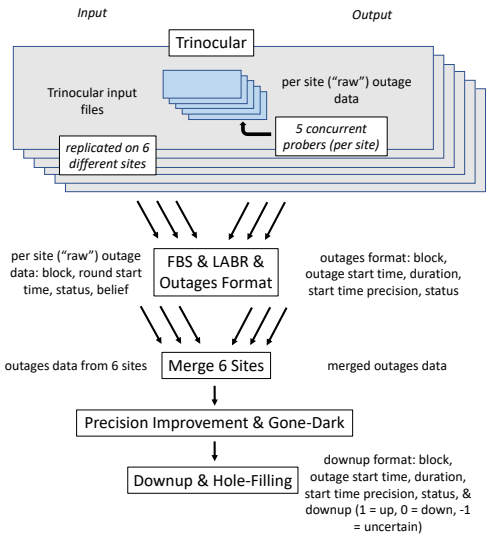

Fig. 1: Flow chart detailing batch post-processing steps.

decisions require more time: Full-Block Scanning evaluates data from several rounds (enough to evaluate all measured addresses in the block) before confirming an outage. Other times, decisions require days or weeks of data to conclude: the Gone-Dark algorithm replaces a week-long non-responsive period with unmeasurable, meaning it requires at least a week of measurements.

When processing a full quarter of data in a batch, algorithms can easily consider as much time as needed to reach the best possible conclusion. We consider batch-processed Trinocular data as our best estimate of Internet status.

II-C NRT Streaming Outage Detection

When using Internet outage detection to assess an *ongoing* event like a hurricane, one cannot wait three months for a batch of results. We therefore implemented *near-real-time streaming* outage detection (or just "streaming").

Streaming outage detection uses the same Trinocular observer and observations, but rather than writing results to local disk, data is sent to the central site in near-real-time. Streaming has several steps: a *compressor* runs with the observer, but since most results are "no change", it supresses duplicate results for up to 90 minutes. Any change in block status is sent immediately, plus one result every 6 h to confirm that measurement of that block is ongoing.

At the central site, we reproduce the usual batch-processed algorithms of precision improvement and multi-site resolution, but with Kafka [16], a streaming framework. We then join the results with geolocation to group networks in each physical location, a 0.5, 1, or 2-degree latitude/longitude "grid cell". Updated results are written to a database, and a custom website makes it easy to examine outage results, as shown in Figure 2. The website updates in real time as new results arrive, with updates appearing representing about 30 to 60 minutes in the past (due to collection, forwarding, and processing times).

Algorithm	Description	Batch	Streaming	State Initial	State Post
Insufficient Site Detection ([19])	Mark blocks with fewer than three sites as unknown	✓	✓	up/down	unknown
Gone-Dark ([19] §4.4)	Mark long-term unresponsive blocks as unknown	✓	*§IV-C	down	unknown
Precision Improvement ([19] §4.5)	Resolve outage start and end times that differ between observer in favor of the earlier time	✓	✓	slightly adjusts change start and end time	
Multi-Site Resolution ([19] §4.5; [4])	Resolve conflicting observers in favor of majority	✓	✓	up/down	majority
Hole-Filling ([14] §3.5)	For blocks with lost replies, retroactively mark block up if periods preceding and succeeding it are up	✓		down	up
Full Block Scanning (FBS) ([4] §3.1)	Retroactively decide block status for sparsely responsive blocks once all IP addresses are scanned	✓		down	up
Lone-Address-Block Recovery (LABR) ([4] §3.2)	For blocks with a single active IP address, outages become unmeasurable (treat it as a host problem)	✓		down	unmeasurable

TABLE I: Post-processing algorithms and their results.

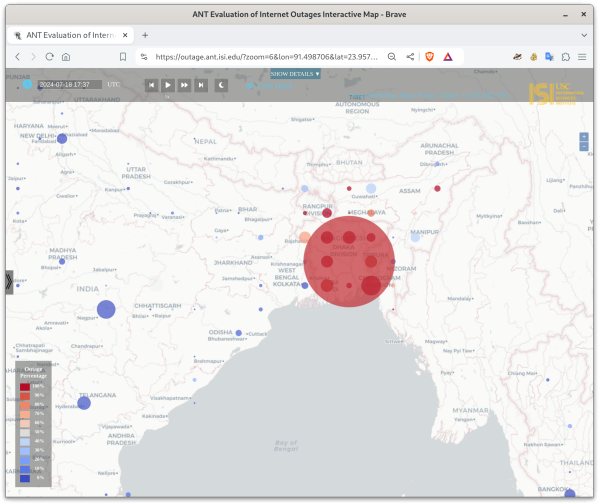

Fig. 2: Our website https://outage.ant.isi.edu showing the onset of the 2024-07-18 country-wide outage in Bangladesh [13].

Because streaming emphasizes *near-real-time* reporting, streaming omits algorithms that require examining hours or months of data. In particular, streaming does not use Hole-Filling, FBS, LABR, and Gone-Dark algorithms (Table I).

The absence of these post-processing algorithms are the subject of our study: our goal is to see how much each algorithm contributes to differences which can be seen in the batch and streaming datasets. We examined Hole-Filling and found it rarely activates, so we do not consider it further.

II-D Relevance to other active outage detection systems

The direct results of our work are specific to comparing Trinocular's batch and streaming implementations. There are no other existing works on NRT outage detection and thus, we focus on similar outage detection systems. However, our general approach of comparing similar algorithms can also be used to evaluate other Trinocular-like implementations of outage detection using active probing.

We compare batch and streaming versions of Trinocular. They are implemented by the same research group, and are intended to provide near-identical results, although within the limitation of near-real-time reporting for streaming. Our comparison of batch and streaming show that small algorithmic differences change the outcomes, particularly for some blocks.

We are aware of at least two independent implementations of Trinocular-like active outage detection; one is in IODA [10]. IODA uses Trinocular-inspired algorithms for active probing [7], but it reports results every 10 minutes. It is unclear if IODA integrates results from multiple locations, or if it uses all of the algorithms in Trinocular (see Table I). Given its emphasis on near-real-time reporting, it seems unlikely that IODA identifies Gone-Dark blocks or includes algorithms like FBS and LABR added later to improve accuracy for sparse blocks [5].

Although we have not compared to alternate implementations, the differences we observe here suggest that independent implementations done from published papers are likely to diverge in even larger ways. They suggest the importance of directly comparing outcomes with the goal of converging on the most accurate estimate of the true condition of networks.

These results also suggest the need to understand where *different measurement systems reach different conclusions*, possibly indicating one system is wrong, *or* indicating legitimate ambiguity in what can be observed. We explore this ambiguity in our examination of partial outages [5].

III. Comparison

We next compare the accuracy of batch and streaming Trinocular and evaluate the effects of algorithmic differences.

III-A Datasets and Metrics

We evaluate existing batch and streaming data over 8 days, from 2021-02-26 to 2021-03-06 using publicly available data [31], [32]. This timeframe was selected because it was the most recent streaming data available at the time. Additionally, we believe this is a representative sample as it lasts longer than a week and outages are relatively rare. Figure 1 shows the steps of batch processing, and Table I shows the algorithms that batch and streaming each employ. We compare the accuracy of streaming relative to batch as our best measurement of truth.

Table II shows possible outcomes of comparison. A block's status is always in exactly one of three states: up, down, or unknown. In addition to *agreement* and *disagreement*, we

		streaming report		
		up	unknown	down
batch report	up	agreement	singleton	disagreement
	unknown	singleton	uncovered	singleton
	down	disagreement	singleton	agreement

TABLE II: Possible outcomes of comparison.

identify *singleton* and *uncovered* as outcomes when one or neither can confirm the block's status.

Metrics: When comparing accuracy, one can compare outage durations or specific events. We generally prefer to compare outage durations rather than events, because minor differences in algorithms can easily create many differing events, magnifying small differences.

We normalize outage durations in terms of *block-seconds per block-day* (bs/bd). Block-seconds captures the duration a block is in each of the states listed in Table II: agreement, disagreement, singleton, or uncovered. Block-days capture the total possible time: 5,233,827 unique blocks by 8 days. We normalize block-seconds by block-days to compute bs/bd.

By normalizing, bs/bd captures the mean number of seconds of difference, a number that has physical meaning independent of how many blocks we measure (per block), and reduced to a meaningful scale (per day). With an 86,400 s day, we know that 10 bs/bd is a small amount, and 3,600 bs/bd is one hour of disagreement. An alternative to bs/bd is to count any differences as a different event. We do not count events, because it weighs a trivial, one-second difference as much as a day- or week-long difference.

We also evaluate the number of unique blocks that show any differences, to identify if differences are pervasive or rare.

III-B Visualizing an Example Disagreement

Before we quantify differences between batch and streaming, we first visualize a small example. We visualized all outages in our study period, then selected outages for two networks, each affecting more than 20 blocks, in Figure 3.

In the visualization (Figure 3), the top left and middle images show outages that batch and streaming each report. In the images, each horizontal line is a /24 block, colored when it is unreachable, and white when it is reachable. Blocks are grouped by similarity of timing of all batch outages seen over the entire quarter [15]. Outages are colored by geolocation of the blocks that are unreachable, mapping hue to longitude and brightness (see our library [2] and a description [20]). The result is that shades of light green are in east Asia, while South America is different pinks. Block geolocation is from MaxMind GeoLite [17], and organizations are from Whois.

We first consider the 23 pink blocks in the middle of the batch and streaming figures, where two 5-hour outages are separated by a 5.5 h period of reachability. (Specific addresses are given in §A.) The first pink outage, marked (br-1), starts at 2021-03-02t07:00Z, and the second, (br-2), at about t18:00Z. This outage occurs in AS263015 G7 Telecom Ltda, in Bahia, Brazil and affects 23 /24 blocks in 5 different /16.

The second example outage is in green, marked (kr), with an outage starting at 2021-03-02t08:00Z and lasting for 240 minutes. This outage occurs in AS17858 LG POWERCOMM, in Seoul, South Korea, affecting 27 /24 blocks in 5 different /16 (listed in the appendix). (We chose these examples arbitrarily after looking at many; they are similar to others.)

Comparing batch and streaming: Visually, the top left and middle images in Figure 3 look similar, but the streaming side has a number of long outages (horizontal lines at (long-1)) that do not occur in batch.

We highlight smaller differences in the bottom images. On the bottom left, we show times when batch is down but streaming is up, and on the bottom right, the opposite. In both of the bottom graphs, colored areas show outages occurring in the above dataset that do not appear in the other. The primary difference we see in the bottom difference-graphs are that the beginnings of batch outages are earlier than those in streaming, for both the Brazilian and Korean outages, and the ends of the Korean outage last longer in streaming. We also see the short Brazilian outage around hour 38, labeled (br-3), occurs only in streaming and not in batch. Finally, although the long horizontal (labeled (long-1)) outages appear in streaming and not batch, they do *not* appear in the batch-up/streaming-down difference. We explain how these differences occur due to algorithmic choices in batch and streaming, next.

III-C How different are batch and streaming?

We expect that differences in the algorithms applied in Table I will produce differences in the outages reported by batch and streaming. Because streaming implements a subset of the algorithms, we expect it will be less accurate—we want to know how much the accuracy "cost" of timely results is: *how* different are batch and streaming? Does that difference disproportionately affect certain types of blocks?

Another perspective is that we have added algorithms to detect outages over time to address problems as we [1] and others [23] discovered them. Since streaming implements only the core Trinocular algorithms and omits these additions, we expect differences to address the specific cases they identified. This also implies that we expect batch and streaming will agree for the majority of time.

III-D Overall Agreement Rate

III-D1 Overall Similarity: We find that **batch and streaming are in agreement for 72,829.28 bs/bd, or 84% of each day** (fraction: 0.8449, Table III, Cell a, Cell b). This data confirms our hypothesis that, regardless of differing algorithms, batch and streaming agree on block status for the majority of time. This agreement shows that both batch and streaming are similar for the majority of blocks and conditions, even with just core algorithms. We explore agreement further in §III-D2.

We find that **batch and streaming produce differing up/down results only 0.2% of the time** (about 178.50 bs/bd Table III, Cell f), specifically 0.002071 fraction of the time (Table III, Cell g). This type of disagreement occurs in 0.05682 (Table III, Cell h) fraction of unique blocks, indicating that only a small number of blocks are affected by disagreements. We will explore these disagreements further in §III-D3.

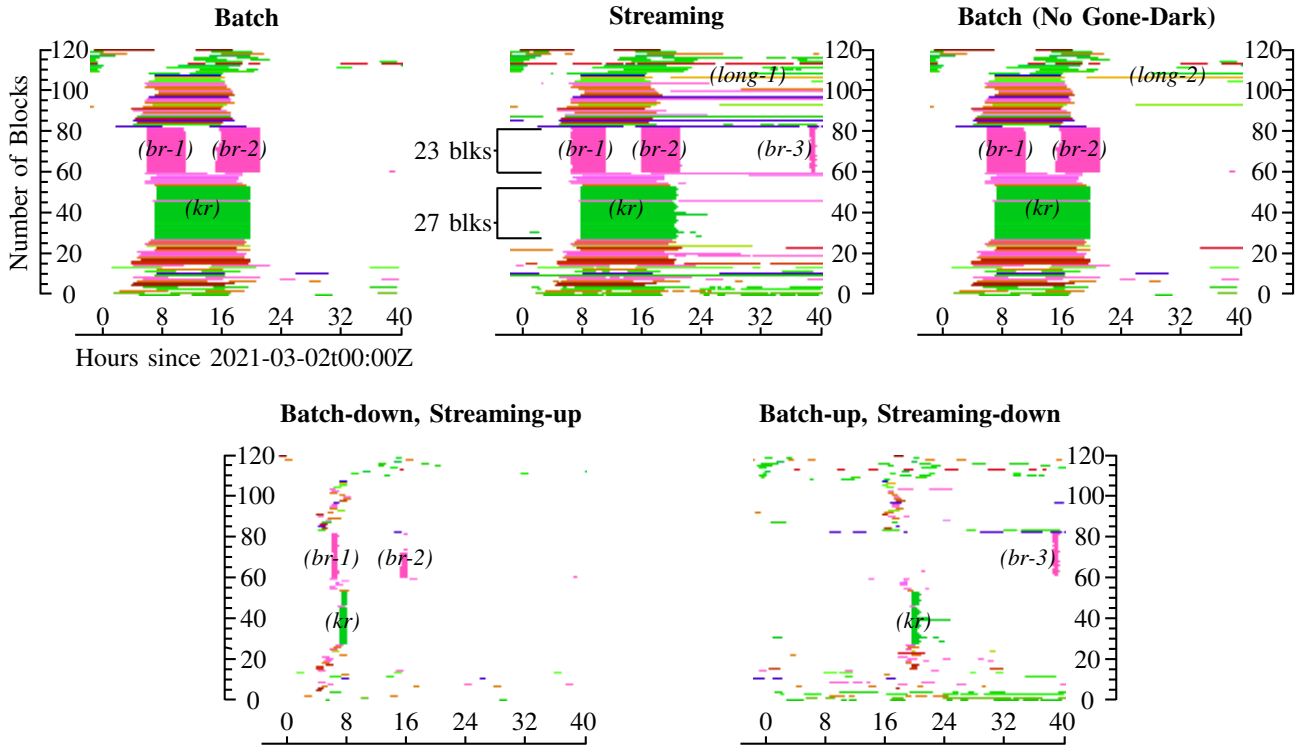

Fig. 3: Visual representation of outages from 2021-03-01T22:00Z to 2021-03-03T20:00Z from batch and streaming datasets.

Block Status	Events	Total Time (Block-Seconds)	Unique Blocks	Normalized Time (bs/bd)	Fraction of Time	Fraction of Unique Blocks	Fraction of Agreement Time	Non-Transient Disagr.	Root Cause
Total Duration		3,852,661,900,000							
Covered Duration	19,464,390	3,609,321,639,731	5,233,827	86,201.78	1.000	1.000			
Agreement	6,516,996	3,049,406,768,513	4,668,538	72,829.28[a]	0.8449[b]	0.8920	1.000		
Batch-up, Streaming-up	6,207,295	3,038,033,977,803	4,666,558	72,557.66	0.8417[c]	0.8916	0.9963		
Batch-down, Streaming-down	309,701	11,372,790,710	139,739	271.62[d]	0.003151	0.02670	0.003730[e]		
Disagreement	1,704,019	7,473,715,613	297,373	178.50[f]	0.002071[g]	0.05682[h]			
Transient	251,184[i]	81,682,109	128,454	1.95[j]	0.00002262[k]	0.02454[l]			
Batch-up, Streaming-down	187,963	61,884,831	111,849	1.48	0.00001717	0.02137			
Batch-down, Streaming-up	63,221	19,797,278	50,900	0.47	0.000005452	0.009725			
Non-transient	1,452,835	7,392,033,504	244,419	176.55[m]	0.002048[n]	0.04670[o]		1.000	
Batch-up, Streaming-down	1,215,907	6,160,159,217	210,812	147.12[p]	0.001707	0.04028		0.8333[q]	FBS
Batch-down, Streaming-up	236,928	1,231,874,287	113,237	29.42[r]	0.0003413	0.02164		0.1666[s]	
Singleton	11,204,260	552,149,122,098	5,233,827	13,187.03[t]	0.1530[u]	1.000			
Batch-up	9,568,835	121,933,463,217	4,629,220	2,912.15[v]	0.03378[w]	0.8845[x]			
Batch-down	55,162	521,384,397	47,319	12.45[y]	0.0001444[z]	0.009041[aa]			
Streaming-up	408,819	16,228,286,159	166,942	387.58	0.004496	0.03190			
Streaming-down	1,171,444	413,465,988,325	700,588	9874.85	0.1146	0.1339			LABR/G.D.
Uncovered	39,115	292,033,507	32,515	6.98[ab]	0.00008097	0.006213			

TABLE III: Fraction of total time in seconds of block decision status with original post-processing algorithms applied.

In addition to disagreements where batch and streaming reach different conclusions, there are two other cases where only one has a conclusion: *singletons* occur because either batch or streaming does not report when the other does, and *uncovered* time arises when neither batch nor streaming report. Together, singletons and uncovered time are responsible for the remainder (about 15.8%) of the 16% of non-agreement time not due to disagreements.

Specifically, singletons account for 3.7 h per day, approximately 74× the bs/bd of actual disagreements. This indicates that reporting differences are the primary source of most non-agreement time. We show singletons are a direct result of the application of LABR and Gone-Dark algorithms in §IV-B and §IV-C. Uncovered time, on the other hand, contributes much less, approximately 0.1 minutes per block-day (Table III, Cell ab). This relatively small daily fraction is likely a result of startup differences between batch and streaming.

III-D2 Are there differences in agreements?: The majority of time (84%) both batch and streaming agree that the block is up or down. We expect that the majority of these cases report a block as reachable since the Internet is generally reliable §I. A large quantity of down agreements would indicate that there is widespread presence of outages during this period (such as during the Covid-19 pandemic [27]).

Batch and streaming report that a block is up for 0.9963 (Table III, Cell c)fraction of agreement time, and report the block down for 0.003730 (Table III, Cell e) fraction of agreement time. This data validates our hypothesis and confirms prior results that outages are relatively rare occurrences, occurring much less than 1% of the time (§I, and [19], [23]).

We also observe that agreement that the block is down is not very common, accounting for 271.62 bs/bd (Table III, Cell d), which makes sense as outages are quite rare.

III-D3 How often and why do batch and streaming disagree?: Disagreements account for a tiny percentage of time (about 0.2%), yet they can provide valuable information on the role the algorithms that comprise outage detection play.

There are two types of disagreements: *transient* and *non-transient*. Transient disagreements are defined as those lasting less than one round (11 minutes), because such brief disagreements occur due to different probing times when measuring outage start and end. Transient disagreements are fairly common (251,184 events (Table III, Cell i)), but account for little time because they are, by definition transient (short-lasting). Additionally, we see transient disagreements in about 2.5% of unique blocks (Table III, Cell l), consistent with prior observations that only a few blocks fail the main Trinocular algorithms [4] although those few blocks produce many incorrect events [23]. Such measurement outages are expected in any measurement system; we discuss their causes in §III-F.

We classify disagreements longer than one Trinocular round as non-transient. Non-transient disagreements must indicate differences in algorithmic post-processing; we expect them to occur and study them to understand the roles the different Trinocular algorithms play. We expect non-transient disagreements to occur in relatively few blocks because both batch and streaming use the same Trinocular algorithms for common outage cases. We confirm this hypothesis, as non-transient disagreements occur in only 4.6% of blocks (Table III, Cell o).

When examining non-transients, we expect that non-transient disagreements will only account for a small fraction of total time §III-C. We find that batch and streaming experience non-transient disagreements for 176.55 bs/bd or 0.002048 fraction of time (Table III, Cell m, Cell n). This validates our hypothesis and demonstrates that non-transient disagreements are likely due to algorithm activation on edge cases §III-C.

III-E Batch is more responsive than streaming

Overall block up-time is larger for batch than for streaming. Obviously, the time that they agree is the same, but when we look at non-transient disagreements, batch-up/streaming-down is $5\times$ larger than batch-down/streaming-up (Table III, Cell p and Table III, Cell r). In addition, 3% of overall time is batch-up singletons (more than the small 0.0045 fraction of time that is streaming-up singletons), and large fraction (11%) of overall time is streaming-down singletons (Table III, Cell t).

The trend that batch is more responsive than streaming is because batch-only processing uses three algorithms (FBS, LABR, and Gone-Dark) to correct mistaken down-time to either up or non-responsiveness [4]. Of these algorithms, FBS changes a down report to up, resulting in a batch-up/streaming-down disagreement. LABR and Gone-Dark change down to unknown, making it a streaming singleton. Since these algorithms change down to up or unknown and none change to down, we would expect batch to have a greater number of up decisions than streaming. Therefore, we hypothesize that the majority of non-transient disagreements will be where batch reports a block as up, while streaming reports a block as down.

To put these values into context of non-transient disagreements, we now only consider non-transient disagreement values. We find that batch and streaming report that a block is up for 0.8333 (Table III, Cell q) fraction of non-transient disagreement time, while they report that a block is down for 0.1666 (Table III, Cell s) fraction of non-transient disagreement time. Thus, we were correct that the majority of non-transient disagreements are batch-up, streaming-down. This skew can be seen in Figure 3 where there are far more lines in the batch-up/streaming-down image. It indicates that a large percentage of non-transient disagreements are explained by algorithms correcting block decisions to up. This result is positive, showing the algorithms correct a relatively small number of edge cases.

We explore the sources of different disagreements in §IV.

III-F Accounting for Timing

Trinocular actively probes about 5M IPv4 networks every 11 minutes from six independent sites [19], and combining these results can lead to *timing differences* that are typically smaller than one round. Trinocular makes no attempt to synchronize probing between sites, and although each has the same target list, they start at different times and run at different rates because sites typically see different responses. Both batch and streaming merge observations from different sites, and apply the *precision improvement* algorithm to maximize sensitivity, reexamining responses that occur in the same round by assuming the last positive response and the first positive response are the most accurate timing bounding an outage. Each separate observer also "votes" to resolve when some sites consider the block reachable or not, but batch can retroactively adjust the outage start time, while streaming cannot.

These algorithmic differences in batch and streaming result in short-term timing differences in outage start and end time—we call these *transient differences*. Such differences are always less than 11 minutes, since all sites should complete probing sometime in each 11-minute window. We next look at the effects of these transient differences to isolate significant, long-term disagreements between batch and streaming. We expect transient differences to be present, but not very large.

We identify transient disagreements in Table III, and see that they account for 1.95 bs/bd and 0.002% of overall time (Table III, Cell j, Cell k).

This data confirms our hypothesis that transient differences are small. They reflect the cost of the finite precision present in any real-world measurement system. We therefore ignore these small differences and focus on larger algorithmic differences in the next section.

Block Status	Unique Blocks	Evts. w/Diff. Count	Fract.	Non-transients	Single-tons
Samples	50				
Difference Event	50	488[ac]	1.0		
Transient	23	43	0.1		
Batch-up, Streaming-down	21	33			
Batch-down, Streaming-up	8	9			
Non-transient	40	249	0.5[ad]	1.0	
Batch-up, Streaming-down	32[ae]	200		0.80[af]	
Batch-down, Streaming-up	23	49		0.20	
Singleton	50[ag]	196	0.4[ah]		1.0
Batch Singleton	46[ai]	86			
Batch-up	44	77			0.39
Batch-down	8	9			0.05
Streaming Singleton	31	110			
Streaming-up	10	31			0.16
Streaming-down	21	79			0.40
LABR	18	75[aj]			
Gone-Dark	4	4[ak]			

TABLE IV: Event differences in 50 sample blocks.

IV. Effects of Batch-Specific Algorithms

While batch and streaming agree the majority of the time (about 84%, Table III, Cell b), we see disagreements about 0.2% (Table III, Cell g) of the time, even after we account for minor timing differences (§III-F). These differences are due to additional algorithms that run only in the batch system.

To understand where these algorithms are used and why, we next examine a random selection of 50 blocks in batch and streaming, and look at specific events where that block shows differences between them. Table IV classifies the 488 differences in these 50 blocks (Table IV, Cell ac). (We look at specific events rather than blocks because a single block can be affected by different algorithms at different times.) This is why the number of unique blocks at each block status does not sum to 50, as various types of difference events may be present in the same blocks at different times.

IV-A Full-Block Scanning: Handling Sparse Blocks

We first look to events where batch is up and streaming is down in Table IV. These differences are due to the Full-Block Scanning (FBS) algorithm.

FBS is designed to correct false outages in low-activity blocks [4]. False outages occur in a low-activity block because it is easy to probe several unoccupied (or non-responsive) addresses and conclude the block appears unreachable, a false outage due to an incorrect conclusion from Bayesian inference. FBS addresses this condition by detecting low-activity blocks based on estimates of long-term activity, then subjecting such blocks to a second requirement before concluding they are out. For such blocks, even if basic Trinocular concludes the block appears down, FBS withholds judgement until *all* addresses (the "full block") have been scanned with a negative result before concluding an outage is legitimate. Since Trinocular limits how many addresses are scanned per round (up to 16), the FBS check can take several rounds after the block is tentatively declared as down. In batch mode, if any of those rounds show activity, the block is retroactively marked as reachable and the apparent outage is never reported. Streaming model currently lacks the ability to go back in time to fix a false outage, so it does not run the FBS algorithm. For this reason, FBS runs only in batch mode, and FBS-detected blocks can result in batch-up/streaming-down disagreements.

FBS accounts for all batch-up/streaming-down disagreements because it is the only algorithm that converts down results to up (LABR and Gone-Dark convert down to unknown). In Table IV we see that about 40% (Table IV, Cell ad) of the events are non-transient batch-up/streaming-down. These events occur in 32 blocks (Table IV, Cell ae). We confirmed that 34 of the 50 sample blocks with errors, and 30 of these 32 blocks with this error, have long-term low-availability. (We measure availability as the probability any expected responsive address responds, the A value [19]. These blocks have A $<$ 0.2.) This observation affirms that FBS accounts for 80% of the non-transient difference events that we see (Table IV, Cell af). Furthermore, of the differing algorithms, FBS can be attributed as the largest source of decision reversals.

We see the impact of FBS in Figure 3. The outage labeled (br-3) appears in the streaming (top middle) and batch-up/streaming-down (bottom right) images, but not in the batch image (top left), demonstrating that streaming originally found the block as down but FBS reverted it to up.

IV-B LABR: From False Outages to Singletons

Next, we will examine the streaming-down singleton category and identify which and how many are due to LABR.

The LABR algorithm handles blocks with only one or two active addresses [4]. In a block with only one active address, a computer reboot or packet loss make the block appear to be inactive. Originally, Trinocular refused to report statistics for blocks with 15 or fewer active addresses. LABR proposes accepting such blocks as up when they respond, but mapping a non-response of a block with a single active address to unknown. (When the block has multiple active addresses, it can be handled without LABR.)

LABR runs only in batch Trinocular, where it maps lone-address blocks from outages to unknown. In streaming, these events are recorded as outages. When comparing batch and streaming, "unknown" is neither up nor down, so LABR creates *singletons*, with batch not reporting and streaming down.

Singletons have a large effect: accounting for 13,187.03 bs/bd (Table III, Cell t), 15% of overall time (Table III, Cell u), and 40% (Table IV, Cell ah) of events. Additionally, they are present in all 50 sample blocks (Table IV, Cell ag). The examples in Figure 3 labeled (long-1) show how such a large amount of block-time can accumulate in this category: these lone-address blocks are often non-responsive for a long time.

To identify LABR, we studied streaming-down singletons and record where there was 1 or less active IP addresses. We find that LABR is responsible for 75 of the 79 streaming-down events (Table IV, Cell aj). This observation affirms that LABR accounts for the majority of streaming-down singleton events that we see.

LABR was designed to smooth over short-term uncertainties, however, while analyzing these blocks, we noticed relatively large durations, often lasting days. It is worth noting

Block Status	Events	Total Time (Block-Seconds)	Unique Blocks	Normalized Time (bs/bd)	Fraction of Time	Fraction of Unique Blocks	Fraction of Agreement Time	Non-Transient Disagr.	Root Cause
Total Duration		3,852,661,900,000							
Covered Duration	20,399,975	3,726,365,651,765	5,233,827	88,997.15	1.000	1.000			
Agreement	7,019,912	3,358,727,610,933	5,117,036	80,216.82	0.9013	0.9777	1.000		
Batch-up, Streaming-up	6,207,295	3,038,033,977,803	4,666,558	72,557.66	0.8153	0.8916	0.9045		
Batch-down, Streaming-down	812,617	320,693,633,130	616,403	7,659.16	0.08606	0.1178	0.09548		
Disagreement	1,729,930	8,531,482,027	311,422	203.76	0.002290	0.05951			
Transient	253,961	82,590,945	130,889	1.97	0.00002214	0.02501			
Batch-up, Streaming-down	187,963	61,884,831	111,849	1.48	0.00001663	0.02137			
Batch-down, Streaming-up	65,998	20,706,114	53,549	0.50	0.000005618	0.01023			
Non-transient	1,475,969	8,448,891,082	256,924	201.79	0.002267	0.04909		1.000	
Batch-up, Streaming-down	1,215,907	6,160,159,217	210,812	147.12	0.001653	0.04028		0.7291	FBS
Batch-down, Streaming-up	260,062	2,288,731,865	128,782	54.66	0.0006142	128,782		0.2709	
Singleton	11,623,398	254,739,432,142	5,233,827	6,083.97	0.06836	1.000			
Batch-up	9,568,835	121,933,463,217	4,629,220	2,912.14	0.03272	0.8845			
Batch-down	1,000,802	13,490,303,275	529,059	322.19	0.003620	0.1011			
Streaming-up	384,213	15,170,519,745	148,604	362.32	0.004071	0.02839			
Streaming-down	669,548[al]	104,145,145,905	224,122	2,487.31	0.02795	0.04282			LABR *(long-2)*
Uncovered	26,735	221,980,758	22,011	5.30	0.00005955	0.004206			

TABLE V: Fraction of total time in seconds of block decision status with retroactive Gone-Dark applied to streaming.

that this does appear to be a bug in LABR, but does not drastically impact our results.

We believe that LABR accounts for most of the differences labeled (long-1) in Figure 3—these long-term outages in streaming (top middle) are not outages in batch (top left), and do not appear as differences in batch-up/streaming-down because batch cannot confirm they are up.

IV-C Gone-Dark: Long-Term Outages

Finally, we will explain why the remaining streaming-down singletons are due to application of Gone-Dark [1].

The Gone-Dark algorithm serves to handle blocks with long-term unresponsiveness. Trinocular would originally map these blocks to non-responsive, or down, wasting valuable time and resources sending probes to these blocks. Instead, Gone-Dark maps these blocks to unknown and only begins probing again when the census denotes that it is active (every 2–3 months). We expect this algorithm only to affect a small number of blocks, since long term outages are uncommon.

Similar to LABR, Gone-Dark turns batch-down events to unknown. As a result, we further examine streaming-down singletons and the duration of the events to identify application of Gone-Dark. Table IV shows that Gone-Dark is applied to 4 out of 50 sample blocks, and is responsible for 4 of 79 streaming-down events (Table IV, Cell ak).

As described in §IV-B, when looking at Figure 3, we expect the majority of outages labeled (long-1) to be a result of LABR rather than Gone-Dark. This is because Gone-Dark is only applied when an outage lasts longer than one week. However, there are a few outages that started before or ended after the time period displayed in these images, making it possible that a few are due to Gone-Dark.

To verify this hypothesis, we reversed the application of the Gone-Dark algorithm on batch by changing blocks labeled as unknown to batch-down. We expect this to cause an increase in the number of batch-down/streaming-down agreement events and a decrease in the number of streaming-down singleton events. As shown in Table V, our hypothesis is correct: the number of batch-down/streaming-down agreement events increased by 38% and the number of streaming-down singleton events decreased by 43% (Table V, Cell al). This suggests that while Gone-Dark did contribute to the occurrence of streaming-down singleton events, LABR, the only other algorithm that produces streaming-down singletons, is the source of more than half of them (the remaining 57%).

We choose to visualize this difference in batch and compare them to the original batch and streaming images in Figure 3. We can see at (long-2) that there are additional long outages that appear in the batch image where Gone-Dark is not applied (top right) that are not present in the original batch image where Gone-Dark is applied (top left). Not all long outages are "brought back", such as the magenta outage below (long-1) in the streaming image. This is because their absence is due to LABR, which is not remedied when we "unapply" Gone-Dark.

We can also verify that Gone-Dark was correctly "unapplied" as each of the newly appeared long outages touch either the right- or left-hand border of the image. By touching the border of the image, these outages likely last longer than the time period shown, and in the case of Gone-Dark, they last longer than one week.

IV-D What is the cause of batch singletons?

We finally consider batch singletons. Batch singletons indicate that the streaming record was originally marked as unknown, yet there is some algorithmic difference that leads batch to have enough information to report a decision. This could happen if multiple sites are behind in reporting either because it is slow or has a large burst of changes. Since streaming's purpose is to run in near-real-time, there is a limit on how much time it can wait on any site which has fallen behind. In streaming Trinocular, this limit is set to 3 probing intervals or 33 minutes. If any sites are still behind after this wait, the decision is made on the data available from the sites that are current. Batch does not experience this problem as it does not have to sync streaming in real time. Thus, batch has all data available for all the sites from the start to generate either a batch-down or batch-up decision.

We find that batch singletons account for 3% of overall time and 2,924.60 bs/bd (Table III, Cell v, Cell y, Cell z, Cell w).

These findings indicate that sites become out-of-sync for a small portion of time, although often, as they occur in 89% of unique blocks and account for almost half of all events (Table III, Cell x, Cell aa).

V. Implications and Future Directions

The goal of our paper was to compare two outage detection systems with different goals: NRT streaming strives to provide reasonable answers quickly, with batch processing using months of data to provide the most accurate possible answers. Even though both systems are implemented by the same group and use the same core algorithms, the ability to see months of data allows batch processing to implement additional algorithms (see Table I).

While a comparison seems straightforward, it is complicated for several reasons. First, both systems are careful to report results when confident, while omitting results when observations are uncertain, making "no comparison possible" (singleton reporting) a possible outcome alongside agree and disagree (see Table II). Second, alignment of observations and processing results in slightly different timing (§III-F), and failing to account for these differences produces transient differences (differences that reflect measurement imprecision) rather than meaningful differences.

Overall Implications: Our overall conclusion is that **both systems are quite similar with a tiny 0.2% overall disagreement** (177 s per day) when both systems report. This extremely small difference is a reasonable tradeoff to provide both near-real-time and highest quality results.

The work also emphasizes where we should expect differences. Both batch and streaming report "unknown" when they cannot reach a conclusion with confidence, and *about 15% of time only one system is confident* (nearly 3.7 hours per day!). For most of this time (2.7 hours), only streaming reports down, while another 0.8 hours batch reports up, so **streaming slightly over-reports outages in cases where batch confirms reachability or lack of enough information**. The implication is that streaming results need to be used with care, and batch results should be employed where accuracy is critical. For example, streaming is ideal for reporting what is happening now, but batch should be used to assess improvements in overall reliability.

Implications for batch: Our study supports the **need to continue batch processing for highest accuracy**. Although the batch-up/streaming-down case is tiny across the whole Internet (only 147 bs/bd, about a 2% error), this error is significant *for specific blocks*. This problem was highlighted when one considers disagreements by event rather than by time, and the initial analysis [23] prompted the additional algorithms currently in batch [4]. It also highlights the challenge of distinguishing outages due to failures from long-term ISP changes, something the Gone-Dark algorithm, and more recent ISP Availability Sensing [6] do.

Implications for NRT streaming: This work also shows the **need for NRT streaming for rapid results** to support operational uses that cannot wait. We confirm that streaming is quite close to batch, at least for the overall network. Our work also shows where one should be skeptical of streaming's claims.

An important implication for streaming is the need to support post-facto result correction. We have considered reporting initial results in streaming, then later going back and correcting those results as we gain a longer perspective. Such a change requires the ability to "undo" claims in the database storing streaming results.

Implications for other systems: Our result confirms the **need to carefully validate independent implementations, even those that use the same conceptual algorithms**. Minor differences in implementation will result in different outcomes, and if those differences are concentrated on certain blocks, one must understand the implications to be able to trust the results of new systems. This work suggests an ongoing need to validate outage detection systems against public data. As one example, to our knowledge, while IODA's active probing uses an independent re-implementation of Trinocular's core algorithms, without the results of a quantitative comparison we cannot know how much effect any implementation differences have on IODA accuracy.

Implications for the evolution of outage detection: Finally, our work shows the results of a decade of work on outage detection [19], [21], [1], [4], [6] by multiple groups [19], [25], [23], [12]. As the approach of active outage detection has matured, we have added algorithms to improve handling of additional cases (Table I). To the extent that streaming represents the core algorithms from 2013 and batch captures the algorithms as of 2020, our comparison confirms the role those additions play in handling corner cases. Our analysis here does not consider the most recent proposals from 2024 [6], but shows the need to track the evolving state-of-the-art to provide the best possible accuracy. It also motivates the need to back-port new algorithms to streaming systems.

VI. Conclusion

Several outage detection systems exist today. We compared two with a common heritage, one providing near-real-time streaming results, and the other using batch processing with algorithms to improve accuracy. We found broad agreement (about 84%) between the two. Explicit disagreements are quite rare, less than 0.2% of the observation time, but 15% of the time only one system reports because the other cannot confidently state a conclusion. Our results show that streaming can provide mostly correct responses quickly, but batch processing with all algorithms is important to provide the highest accuracy. Our work shows the many details that must be considered to emphasize meaningful differences in conclusion rather than measurement details, showing the importance of careful validation and tracking of state-of-the-art approaches.

Acknowledgments

This work is partially supported by the project "CNS Core: Small: Event Identification and Evaluation of Internet Outages (EIEIO)" (CNS-2007106) through the U.S. National Science

Foundation, and by an REU supplement to that project. Erica Stutz began this work at Swarthmore College, working remotely for the University of Southern California; her current affiliation is Yale University.

References

[1] Alwabel, A., Healy, J., Heidemann, J., Luu, B., Pradkin, Y., and Safavian., R. Evaluating externally visible outages. Tech. Rep. ISI-TR-2015-701, USC/Information Sciences Institute, Aug. 2015.

[2] ANT Project. Ant project outage datasets. https://ant.isi.edu/datasets/outage/, Apr. 2013.

[3] ANT Project. ANT Internet outages interactive map. https://outage.ant.isi.edu/ and https://ant.isi.edu/blog/?p=1141, Dec. 2017.

[4] Baltra, G., and Heidemann, J. Improving coverage of internet outage detection in sparse blocks. In *Proceedings of the Passive and Active Measurement Workshop* (Eugene, Oregon, USA, Mar. 2020), Springer.

[5] Baltra, G., and Heidemann, J. What is the Internet? partial connectivity of the Internet core. Tech. Rep. arXiv:2107.11439v3, USC/Information Sciences Institute, Mar. 2023.

[6] Baltra, G., Song, X., and Heidemann, J. Ebb and flow: Implications of ISP address dynamics. In *Proceedings of the Passive and Active Measurement Conference* (Virtual Location, Mar. 2024), Springer.

[7] Bischof, Z. S., Pitcher, K., Carisimo, E., Meng, A., Nunes, R. B., Padmanabhan, R., Roberts, M. E., Snoeren, A. C., and Dainotti, A. Destination unreachable: Characterizing internet outages and shutdowns. In *Proceedings of the ACM SIGCOMM Conference* (New York, NY, USA, Sept. 2023), ACM, pp. 608–621.

[8] Brodkin, J. How malformed packets caused CenturyLink's 37-hour, nationwide outage. Ars Technica, Aug. 2019.

[9] Cimpanu, C. CenturyLink outage led to a 3.5% drop in global web traffic. *ZDNet* (Aug. 30 2020).

[10] Dainotti, A., kc claffy, King, A., Asturiano, V., Benson, K., Fomenkov, M., Huffaker, B., Hyun, Y., Keys, K., Koga, R., Ma, A., Orsini, C., and Polterock, J. IODA: Internet outage detection & analysis. Talk at CAIDA Active Internet Measurement Workshop (AIMS), Mar. 2017.

[11] Dainotti, A., Squarcella, C., Aben, E., Chiesa, M., Claffy, K. C., Russo, M., and Pescapé, A. Analysis of country-wide Internet outages caused by censorship. In *Proceedings of the ACM Internet Measurement Conference* (Berlin, Germany, Nov. 2011), ACM, pp. 1–18.

[12] Guillot, A., Fontugne, R., Winter, P., Merindol, P., King, A., Dainotti, A., and Pelsser, C. Chocolatine: Outage detection for Internet background radiation. In *Proceedings of the IFIP International Workshop on Traffic Monitoring and Analysis* (Paris, France, June 2019), IFIP.

[13] Heidemann, J. Major Internet outage in Bangladesh. blog https://ant.isi.edu/blog/?p=2094, July 2024.

[14] Heidemann, J., Pradkin, Y., Govindan, R., Papadopoulos, C., Bartlett, G., and Bannister, J. Census and survey of the visible Internet. In *Proceedings of the ACM Internet Measurement Conference* (Vouliagmeni, Greece, Oct. 2008), ACM, pp. 169–182.

[15] Heidemann, J., Pradkin, Y., and Nisar, A. Back out: End-to-end inference of common points-of-failure in the Internet (extended). Tech. Rep. ISI-TR-724, USC/Information Sciences Institute, Feb. 2018.

[16] Kreps, J., Narkhede, N., and Rao, J. Kafka: a distributed messaging system for log processing. In *Proceedings of the NetDB Workshop* (Athens, Greece, June 2011), ACM.

[17] Maxmind. Geolite city. Web page http://dev.maxmind.com/geoip/geolite, 2012.

[18] Padmanabhan, R., Schulman, A., Levin, D., and Spring, N. Residential links under the weather. In *Proceedings of the ACM SIGCOMM Conference* (Beijing, China, Aug. 2019), ACM, pp. 145–158.

[19] Quan, L., Heidemann, J., and Pradkin, Y. Trinocular: Understanding Internet reliability through adaptive probing. In *Proceedings of the ACM SIGCOMM Conference* (Hong Kong, China, Aug. 2013), ACM, pp. 255–266.

[20] Quan, L., Heidemann, J., and Pradkin, Y. Visualizing sparse Internet events: Network outages and route changes. *Computing 96*, 1 (Jan. 2014), 39–51.

[21] Quan, L., Heidemann, J., and Pradkin, Y. When the Internet sleeps: Correlating diurnal networks with external factors. In *Proceedings of the ACM Internet Measurement Conference* (Vancouver, BC, Canada, Nov. 2014), ACM, pp. 87–100.

[22] Ramesh, R., Raman, R. S., Virkud, A., Dirksen, A., Huremagic, A., Fifield, D., Rodenburg, D., Hynes, R., Madory, D., and Ensafi, R. Network responses to Russia's invasion of Ukraine in 2022: A cautionary tale for internet freedom. In *Proceedings of the 32nd USENIX Security Symposium* (Anaheim, CA, USA, Aug. 2023), USENIX, pp. 2581–2598.

[23] Richter, P., Padmanabhan, R., Spring, N., Berger, A., and Clark, D. Advancing the art of Internet edge outage detection. In *Proceedings of the ACM Internet Measurement Conference* (Boston, Massachusetts, USA, Oct. 2018), ACM, pp. 350–363.

[24] Schulman, A., and Spring, N. Pingin' in the rain. In *Proceedings of the ACM Internet Measurement Conference* (Berlin, Germany, Nov. 2011), ACM, pp. 19–25.

[25] Shah, A., Fontugne, R., Aben, E., Pelsser, C., and Bush, R. Disco: Fast, good, and cheap outage detection. In *Proceedings of the IEEE Network Traffic Monitoring and Analysis Conference* (Dublin, Ireland, June 2017), Springer, pp. 1–9.

[26] Smith, A. Government online. https://www.pewresearch.org/internet/2010/04/27/government-online/.

[27] Song, X., Baltra, G., and Heidemann, J. Inferring changes in daily human activity from internet response. In *Proceedings of the ACM Internet Measurement Conference* (Montreal, QC, Canada, Oct. 2023), ACM, pp. 627–644.

[28] Stelter, B. Time Warner Cable comes back from nationwide Internet outage. CNN Media Website, http://money.cnn.com/2014/08/27/media/time-warner-cable-outage/index.html, Aug. 2014.

[29] Time Warner Cable. This morning's outage. web http://www.twcableuntangled.com/2014/08/twc-identifies-cause-of-internet-outage/, Aug. 2014.

[30] UNCTAD. Unctad data highlights need to strengthen business ict statistics.

[31] USC/LANDER Project. Internet outage measurements. Daaset ID internet_outage_adaptive_a43all-20210101, Jan. 2021.

[32] USC/LANDER Project. Internet streaming. Daaset ID internet_outage_streaming_2021q1, Jan. 2021.

[33] West, M. *An ed-tech tragedy? Educational technologies and school closures in the time of COVID-19*. 2023.

Appendix

These are the IP addresses associated with the blocks visualized in Figure 3:

Brazilian Blocks (pink)
170.0.212.0
170.0.213.0
170.0.214.0
170.0.215.0
170.83.252.0
170.83.253.0
170.83.254.0
170.83.255.0
177.137.120.0
177.137.121.0
177.137.122.0
177.137.123.0
177.137.124.0
177.137.126.0
177.137.127.0
186.227.177.0
186.227.179.0
186.227.180.0
186.227.181.0
186.227.182.0
186.227.183.0
190.83.94.0
190.83.95.0

South Korean Blocks (green)	
115.139.76.0	122.47.10.0
116.44.62.0	122.47.68.0
116.44.158.0	124.58.176.0
116.45.80.0	124.58.200.0
122.36.44.0	124.58.224.0
122.44.210.0	179.186.185.0
122.44.224.0	122.45.170.0
122.44.240.0	122.45.202.0
122.44.246.0	122.45.234.0
122.45.224.0	122.45.244.0
122.45.230.0	122.46.20.0
122.46.36.0	122.47.32.0
122.46.82.0	122.47.40.0
122.47.4.0	