DISTRIBUTED SYNCHRONIZATION UNDER DATA CHURN

A Dissertation

by

XIAOYONG LI

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Dmitri Loguinov
Committee Members,	Riccardo Bettati
	James Caverlee
	Narasimha Reddy
Head of Department,	Dilma Da Silva

May 2016

Major Subject: Computer Science

ABSTRACT

Nowadays an increasing number of applications need to maintain local copies of remote data sources to provide services to their users. Because of the dynamic nature of the sources, an application has to synchronize its copies with remote sources constantly to provide reliable services. Instead of push-based synchronization, we focus on pull-based strategy because it doesn't require source cooperation and has been widely adopted by existing systems.

The scalability of the pull-based synchronization comes at the expense of increased inconsistency of the copied content. We model this system under non-Poisson update/refresh processes and obtain sample-path averages of various metrics of staleness cost, generalizing previous results and studying its statistical properties.

Computing staleness requires knowledge of the inter-update distribution at the source, which can only be estimated through blind sampling – periodic downloads and comparison against previous copies. We show that all previous approaches are biased unless the observation rate tends to infinity or the update process is Poisson. To overcome these issues, we propose four new algorithms that achieve various levels of consistency, which depend on the amount of temporal information revealed by the source and capabilities of the download process

Then we focus on applying freshness to P2P replication systems. We extend our results to several more difficult algorithms – cascaded replication, cooperative caching, and redundant querying from the clients. Surprisingly, we discover that optimal cooperation involves just a single peer and that redundant querying can hurt the ability of the system to handle load (i.e., may lead to lower scalability).

DEDICATION

my family

ACKNOWLEDGEMENTS

I sincerely devote my utmost appreciation to my advisor Professor Dmitri Loguinov, without whom this dissertation is not possible. With his fully supporting and and invaluable guidance, I learn the skills to produce solid and meaningful results, write technical papers, and give impressive presentations. I am always surprised by his creativity and inspired by his insightful discussion. His great passion for solving problem, strong demand for excellence, and persistent attention for detail will make a numerous impact on me for the rest of career life.

I would like to thank Dr. Riccardo Bettati, Dr. James Caverlee, and Dr. Narasimha Reddy for serving on my committee and providing insightful feedback on my research. In addition, I am also indebted to Professor Daren B.H. Cline for his efforts to make our results more rigorous and solid. I receive enormous help from anonymous reviewers of IEEE INFOCOM, IEEE/ACM Transactions on Networking. I extend my thanks to them for assisting me to improve this work.

Furthermore, I sincerely appreciate my friends and fellow students here who make my life here more enjoyable. I owe gratitude to Xiaoming, Zhongmei, Tanzir, Zain, Yi and all other former and current IRL members, who provide help when it was needed.

Last, but not least, I gratefully acknowledge my family members for their constant support and encouragement. I express my special gratitude to my parents, who take care of my son Ethan during hard times, and my wife Yi, who has been my best supporter and helper.

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION

1.1 Overview

Nowadays an increasing number of applications need to maintain local copies of remote data sources to provide services to their users. Because of the *dynamic* nature of the sources, an application has to *synchronize* its copies with remote sources constantly to provide reliable services. The synchronization strategy can be broadly classified as *push-based* policy and *pull-based* policy. In push-based synchronization, the source is allowed to inform its copies whenever it detects important information changes. While appropriate for certain systems with a few machines, this strategy has difficulty to deal with large scale applications in real life, due either either to source noncooperation or the source difficulty of maintaining large number of copies. Instead, pull-based synchronization strategy, which is known to improve both scalability and availability, has been widely adopted in the current Internet (e.g., HTTP, DNS, network monitoring, web caching, web crawling, RSS feeds, stock-ticker aggregators, certain types of CDNs, sensor networks).

The scalability of the pull-based synchronization comes at the expense of increased inconsistency of the copied content [52, 108]. From the perspective of system designer, the objective is to minimize the system inconsistency cost given limited network resources (or minimize the resource usage given fixed inconsistency cost). To achieve the objective, we need to formally define inconsistency cost, which leads to our first work.

Our first work is to propose a general framework to define the inconsistency cost. We first consider a system with a single source and a single replica. In existing studies, the update of a source was usually modeled as a Poisson process [7, 8, 15,

14, 20, 29, 31, 34, 40, 49, 53, 58, 61, 65, 66, 84, 95, 106] and the refresh events were renewal process with either constant or exponential cycle length. While certain measurement results [15, 66] verified the correctness of the model, other measurement studies [67, 7, 37, 63] showed that the Poisson model is not universally applicable. Thus existing models and results are no longer applicable for general update and refresh patterns.

Our second work is to measure the update distribution of a source, which is a prerequisite to compute inconsistency cost. Again, existing works [7, 17, 44, 61] all assume Poisson update, which means that the inter-update delays $\{U_i\}_{i=1}^{\infty}$ follow exponential distribution $F_U(x) = 1 - e^{-\mu x}$. Under this assumption, the only variable unknown is the rate μ, which makes the estimation easier. For general update process, the problem becomes challenging because the measurement need to know the distribution of $\{U_i\}_{i=1}^{\infty}$ in addition to its mean value.

Our last work is to apply the knowledge obtained above to study the performance of distributed systems such as P2P replication and DNS. To deal with various complicated applications in real world, we need to develop basic modules, which is tractable and easily understandable. Then we can extend the proposed modules to more complicated scenarios.

1.2 Contributions

The contribution of this work can be classified to three folds.

1.2.1 Staleness Modeling

Consider a single source driven by an update process N_U and a single replica with the corresponding download process N_D, which is independent of N_U. We propose a general framework for modeling staleness under arbitrary stochastic processes (N_U, N_D), in contrast to prior work that has only considered Poisson cases

[7], [8], [14], [15], [17], [20], [31], [34], [40], [49], [53], [59], [61], [65], [66], [84], [95], [106]. Since staleness age and various penalties derived from it are usually defined in terms of sample-path averages [14], [15], [17], [20], [31], [34], [40], [59], [65], [66], [84], [95], [106], questions arise about their existence and possible variation across multiple realizations of the system. We address this issue by identifying the weakest set of conditions for which the distribution of staleness age exists and converges to a deterministic limit.

Armed with these results, we model interaction between the age processes of (N_U, N_D). For staleness to be a function of inter-update delays, we discover that sample-path ages of both processes, examined at the same random time Q_T, must be asymptotically *independent*. Interestingly, this condition does not automatically follow from independence of N_U and N_D, their stationarity, ergodicity, or even all three constraints combined. Instead, we show that it translates into a form of ASTA (Arrivals See Time Averages) [62], where the download process N_D must observe the sample-path distribution of update age.

Under the condition of age-independence, we next derive the distribution of time by which the replica trails the source, the fraction of consumers that encounter a stale copy, the average number of missing updates from the replica at query time, and the general staleness cost under all suitable penalty functions $w(x)$. Our results involve simple closed-form expressions that are functions of limiting age distributions of both processes.

We also analyze conditions under which N_D produces provably optimal penalty for a given download rate. We show that penalty reduces if and only if inter-refresh delays become stochastically larger in second order. This leads to constant synchronization delays being optimal under all N_U and $w(x)$. This, however, presents problems in satisfying ASTA and creates a possibility of worst-case (i.e., 100%) stal-

eness due to phase-lock between the source and the replica. To this end, we discuss broad requirements for ensuring that N_D avoids these drawbacks while remaining optimal.

Finally, we consider the practical aspects of staleness, including experimentation with Wikipedia page updates, error analysis of previous Poisson models, estimation of search-engine bandwidth requirements, and generalization to multiple sources/replicas.

1.2.2 Update Measurement

We formalize blind update sampling using a framework in which both N_U and N_S are general point processes. We then consider a simplified problem where the source provides last-modification timestamps for each download. We develop the necessary tools for tackling the more interesting cases that follow, build general intuition, consider conditions under which provably consistent estimation is possible, and explain the pitfalls of existing methods under non-Poisson updates.

Armed with these results, we next relax the availability of last-modified timestamps at the source. For situations where constant S_i is acceptable, we show that unbiased estimators developed earlier in the paper can be easily adapted to this environment and then suggest avenues for reducing the amount of numerical computation in the model, all of which forms our third contribution. We finish the paper by considering random S_i and arrive at our last contribution, which is a novel method that can accurately reconstruct the update distribution under arbitrary $F_U(x)$ and mildly constrained $F_S(x)$.

At last, we evaluate the Poisson update assumption and compare our methods on the Wikipedia dataset. We show that Poisson updates assumption hardly holds because it requires not only the inter-update delays to be exponentially distributed

but also independency between delays. Then we compare the accuracy of our methods use the top 10 most frequently modified articles in our dataset, from which we conclude the optimal methods to use in different scenarios.

1.2.3 P2P Application

Based on the results on the freshness probability p, we analyze *cascaded* synchronization, where replicas receive content from other replicas along a fixed multi-hop path from the source. A common arrangement covered by this model is a b-way replication tree, which limits the source to b concurrent downloads, but keeps client-scalability arbitrarily high depending on the depth of the tree. Assuming independent operation among the replicas, we derive a recursive model that provides freshness at each level i. Our results show that in certain cases p decays exponentially fast as a function of the depth, suggesting an interesting coupling between system size and staleness.

Next we propose a model of *cooperative caching*, in which m replicas form a single layer, in which each participant can synchronize not only with the source, but also k other replicas. For a target p, the goal is to determine the optimal (m, k) that maximizes the service rate of the entire system. The main caveat of this model is that it takes into account bandwidth constraints at the source and each replica. We show that making k or m too large is detrimental to performance; instead, *each parameter has a unique optimal value that achieves the highest service rate*, which can be $2 - 7$ times larger than under non-cooperative replication.

Finally we examine a scenario called *redundant querying*, in which consumers have access to multiple independent caches. Issuing parallel queries to k replicas out of the m available, the hope is to improve freshness by selecting the most up-to-date copy of the source. We first show that freshness in this case can be computed using

the original update process and a superposition of download processes from each of the contacted replicas. However, taking bandwidth into account, this analysis also leads to a surprising conclusion that *redundant querying with* $k \geq 2$ *sometimes produces lower performance than non-redundant.*

2. ON SAMPLE-PATH STALENESS IN LAZY DATA REPLICATION*

2.1 Introduction

With the massive growth of the Internet and deployment of large-scale distributed applications, mankind faces new challenges in acquiring, processing, and maintaining vast amounts of data. In response to this flood of information, companies deploy cloud-based solutions designed to provide replicated and distributed support to the skyrocketing storage and processing demand of their users.

One interesting problem in these applications is the *highly-dynamic* nature of content, especially when perfect synchronization of sources, replicas, intermediate caches, and various computation is impossible. In fact, many large-scale distributed systems (e.g., airline reservations, online banking, web search engines, social networks) operate under constant data churn and may never see consistent snapshots of the entire network. As a result, these applications may hold and/or manipulate a mixture of objects that existed at the source at different times t in the past. This leads to questions about staleness, synchronization costs, and techniques for deciding optimal refresh policies.

In traditional databases, the source opens outbound communication with the replicas whenever it detects important changes. This enables *push-based* operation that actively expires stale content and broadcasts notifications into the system. Even under multi-hop replication, staleness lags in these systems are described by simple models that can be reduced to convolutions of single-hop notification delays. In

*Reprinted with permission from "On Sample-Path Staleness in Lazy Data Replication" by Xiaoyong Li, Daren Cline, and Dmitri Loguinov, 2016, IEEE/ACM Transactions on Networking, Copyright by IEEE 2016.

other cases, however, scalability and administrative autonomy require that sources operate independently and provide information only based on explicit request, especially when they are unable to track their replicas or adopt modifications to existing protocols.

This *pull-based* replication (also called *optimistic* or *lazy*) improves both scalability of the service and availability of the data, but at the expense of increased age of manipulated content [52], [108]. This model of operation has enjoyed ubiquitous deployment in the current Internet (e.g., HTTP, DNS, network monitoring, web caching, RSS feeds, stock-ticker aggregators, certain types of CDNs, sensor networks); however, it still poses many fundamental modeling challenges. Our goal is to study them in this paper.

2.1.1 Motivation and Objectives

Suppose a *replica* is a system whose goal is to synchronize against *information sources*, apply certain processing to downloaded content, and serve results to *data consumers*. One challenge of this architecture is that sources not only require pull-based operation, but also lack the ability to predict future updates, which makes real-time estimates of remaining object lifetime (i.e., TTL) unavailable to the replica.

As information evolves at the source, which we call *data churn*, the replica may become stale and provide responses to that do not reflect the true state of the system. In such cases, we assume that user satisfaction and system performance are directly rated to the amount of time by which the replica is lagging behind the source. To convert time units into cost, suppose the application applies some weight function $w(x)$ to the age of stale content to determine the *penalty* associated with a particular refresh policy and data-churn process. Then, the goal of the system is to optimize the expectation of penalty observed by a stream of arriving customers.

This problem has been considered in the context of web systems [7], [8], [14], [15], [17], [20], [31], [34], [40], [49], [53], [59], [61], [65], [66], [84], [95], [106]; however, analytical results have predominantly assumed a Poisson update process at the source, with function $w(x)$ limited to either 1 or x. However, real systems driven by human behavior often require more complex families of processes (e.g., with heavy-tailed inter-update distributions, non-stationary dynamics, and slowly decaying correlation). Similarly, user sensitivity to outdated material may experience rapid increases for small x and eventual saturation for larger x, in which case other weight functions might be more appropriate. Since application performance under general update processes and wider classes of $w(x)$ is currently open, we aim to fill this void below.

2.1.2 Contributions

Consider a single source driven by an update process N_U and a single replica with the corresponding download process N_D, which is independent of N_U. Our first contribution is to propose a general framework for modeling staleness under arbitrary stochastic processes (N_U, N_D). Since staleness age and various penalties derived from it are usually defined in terms of sample-path averages [14], [15], [17], [20], [31], [34], [40], [59], [65], [66], [84], [95], [106], questions arise about their existence and possible variation across multiple realizations of the system. We address this issue by identifying the weakest set of conditions for which the distribution of staleness age exists and converges to a deterministic limit.

Armed with these results, our second contribution is to model interaction between the age processes of (N_U, N_D). For staleness to be a function of inter-update delays, we discover that sample-path ages of both processes, examined at the same random time Q_T, must be asymptotically *independent*. Interestingly, this condition does not automatically follow from independence of N_U and N_D, their stationarity, ergodicity,

or even all three constraints combined. Instead, we show that it translates into a form of ASTA (Arrivals See Time Averages) [62], where the download process N_D must observe the sample-path distribution of update age.

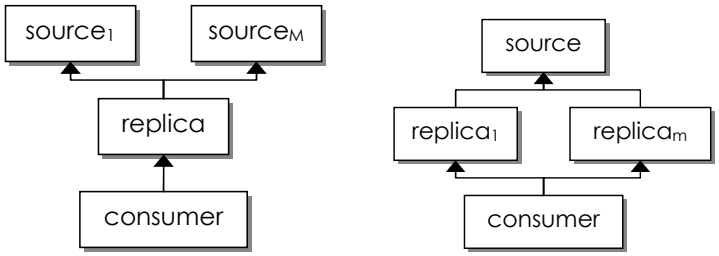
Under the condition of age-independence, our third contribution is to derive the distribution of time by which the replica trails the source, the fraction of consumers that encounter a stale copy, the average number of missing updates from the replica at query time, and the general staleness cost under all suitable penalty functions $w(x)$. Our results involve simple closed-form expressions that are functions of limiting age distributions of both processes.

Our fourth contribution is to analyze conditions under which N_D produces provably optimal penalty for a given download rate. We show that penalty reduces if and only if inter-refresh delays become stochastically larger in second order. This leads to constant synchronization delays being optimal under all N_U and $w(x)$. This, however, presents problems in satisfying ASTA and creates a possibility of worst-case (i.e., 100%) staleness due to phase-lock between the source and the replica. To this end, we discuss broad requirements for ensuring that N_D avoids these drawbacks while remaining optimal.

We finish the paper with our last contribution that considers the practical aspects of staleness, including experimentation with Wikipedia page updates, error analysis of previous Poisson models, estimation of search-engine bandwidth requirements, and generalization to multiple sources/replicas.

2.2 Staleness Formulation

We start by explaining the underlying assumptions on the system, defining the various processes that determine information flow, and specifying the metrics of interest.



(a) aggregation (b) backup/load-balancing

Figure 2.1: System model (arrows signify the direction of information requests).

2.2.1 System Operation

We assume a model of distributed data generation, replication, and consumption shown in Fig. 2.1. During normal system operation, sources sustain random updates in response to external action (e.g., new posts on Facebook, traffic congestion in Google maps) or possibly some internal computation (e.g., MapReduce [28] indexing with periodic writes to disk). In either case, each update represents certain non-negligible information that manipulates the current state of the source.

Replicas operate independently of the sources and perform one of the two general functions shown in the figure – many-to-one aggregation in part (a) and one-to-many replication in (b). The former case arises when the replica executes certain processing on multiple objects to provide the consumer with results that cannot be obtained otherwise. These applications include search engines, data-centric computing, and various web front-ends that cache queries against back-end databases. The purpose of the latter case is to handle failover during source crashes and/or ensure scalable load distribution under heavy customer demand. Applications in this category include CDNs, large websites, data centers (e.g., Amazon EC2), and various distributed file systems.

The final element of Fig. 2.1 is the consumer, which sends a stream of *requests*

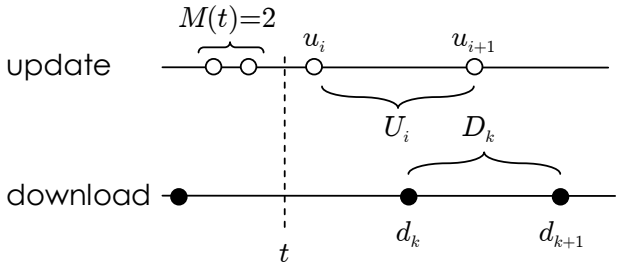


Figure 2.2: Process notation.

that represent either queries for information or attempts to recover the most-recent state of the source after it has crashed.

2.2.2 Updates and Synchronization

We next model interaction between a single source and a single replica, which is a prerequisite to understanding system performance. Suppose the source undergoes updates at random times $0 = u_1 < u_2 < \ldots$ and define $N_U(t) = \max\{i : u_i \leq t\}$ to be a stochastic process that counts the number of updates in $[0, t]$. When referring to the entire process, rather than its value at some point, we omit t and write simply N_U.

For the replica, denote its random download (synchronization) instances by $0 = d_1 < d_2 < \ldots$ and the corresponding point process by $N_D(t) = \max\{k : d_k \leq t\}$. This formulation neglects processing delays and treats all events as instantaneous. We additionally assume that both processes are simple and independent. Now, suppose the inter-update delays of N_U are given by a random process $\{U_i\}_{i=1}^{\infty}$ and inter-download cycles of N_D by $\{D_k\}_{k=1}^{\infty}$, which are illustrated in Fig. 2.2. Each of these sequences may be of fairly general nature, e.g., correlated and/or non-stationary.

2.2.3 Cost of Staleness

To understand the penalty of outdated content, suppose $M(t) = N_U(t) - N_U(d_{N_D(t)})$ counts the number of updates *missing* from the replica at time t (e.g., in Fig. 2.2, $M(t) = 2$). This is a discrete-state process that increments for each update and resets to zero for each synchronization.

Definition 1. *A replica is called* stale *at time t if $M(t) > 0$. Otherwise, it is called* fresh*.*

From the consumer's perspective, stale material reduces user satisfaction and lowers system performance, which needs to be translated into a cost metric that can be expressed via some known parameters of the system. The most basic penalty is the probability that the replica is stale at the time of request, i.e., $P(M(t) > 0)$, which determines how often users see outdated information and/or fail to fully restore a crashed source. The second obvious metric is the expected number of missing updates $E[M(t)]$, which measures the amount of lost information during a crash and estimates the difficulty in recreating it from the most recent checkpoint. This penalty is also important for Internet archiving applications that aim to capture every snapshot of the source [50] and situations when larger $M(t)$ may imply higher information divergence between the replica and the source.

More sophisticated cases are also possible. Suppose the source runs some computation, with updates representing certain intermediate states that are written to disk. A crash at time t requires computation to be restarted, which means that the penalty is determined not by $M(t)$, but rather by the *duration* of the computation that was lost due to staleness. Services that charge per CPU time-unit (e.g., Amazon EC2) may want to optimize against this metric rather than $E[M(t)]$. Furthermore, if the difficulty of recovering each update from other storage is proportional to the

delay since the update was made, then staleness cost may be based on the *combined* lag of all missing updates at time t.

Definition 2. *For a stale replica at time t, define lags $L_1(t) > L_2(t) > \ldots > L_{M(t)}(t)$ to be backward delays to each unseen update, i.e, $L_i(t) = t - u_{N_U(t)-M(t)+i}$.*

This concept is illustrated in Fig. 2.3(a) for the first two lags. To keep the model general and cover the various options already seen in the literature [7], [8], [14], [15], we assume that the consumer is sensitive to either just lag $L_1(t)$, i.e., how long the *source* has been stale at time t, or the entire collection of lags $\{L_1(t), \ldots, L_{M(t)}(t)\}$, i.e., how long each uncaptured *update* has been stale. Since it is usually difficult to predict the value of information freshness to each customer, one requires a mapping from staleness lags to actual cost, which we assume is given by some non-negative weight function $w(x)$.

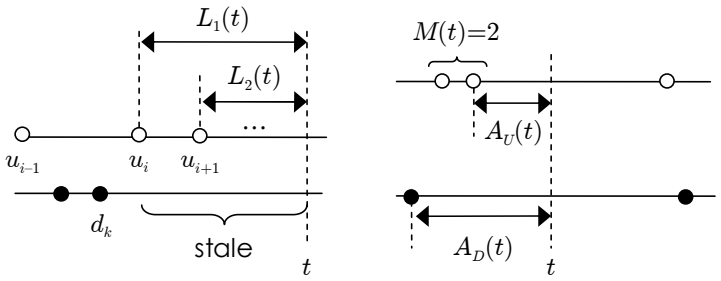
Definition 3. *At time t, the* source penalty *is given by the weight of the delay since the replica was fresh last time:*

$$\eta(t) = \begin{cases} w(L_1(t)) & M(t) > 0 \\ 0 & \textit{otherwise} \end{cases}, \tag{2.1}$$

while the update penalty *is given by the aggregate weight of all staleness lags:*

$$\rho(t) = \begin{cases} \sum_{i=1}^{M(t)} w(L_i(t)) & M(t) > 0 \\ 0 & \textit{otherwise} \end{cases}. \tag{2.2}$$

For example, $w(x) = 1$ produces the first two metrics discussed above, i.e., $P(M(t) > 0)$ via $E[\eta(t)]$ and $E[M(t)]$ via $E[\rho(t)]$. Both (2.1) and (2.2) are random variables, which suggests that system performance should be assessed by their



(a) staleness lags (b) age

Figure 2.3: Penalty lags and process age.

average values. But as neither N_U nor N_D is assumed to be stationary, the expected penalty requires additional elaboration. Instead of considering $E[\eta(t)]$ and $E[\rho(t)]$, which may depend on time t, it is more natural to replace them with sample-path averages [15]:

$$\bar{\eta} = \lim_{T\to\infty} \frac{1}{T} \int_0^T \eta(t)dt \quad \text{and} \quad \bar{\rho} = \lim_{T\to\infty} \frac{1}{T} \int_0^T \rho(t)dt, \tag{2.3}$$

where consumers are modeled as being equally likely to query the replica at any time in $[0, \infty)$.

2.2.4 Relationship to Prior Work

The majority of the literature on source penalty $\bar{\eta}$ is limited to Poisson N_U, either constant or exponential D, and $w(x) = 1$ or x [10], [14], [15], [20], [65], [97], [101], [106]. There has been only one attempt to model $\bar{\eta}$ under a renewal process N_U, in which [101] assumed $w(x) = 1$ and the entire sequence of refresh instances $\{d_1, d_2, \ldots\}$ was known. While appropriate in some cases, this model is difficult to evaluate in practice when N_D is given by its statistical properties.

Update penalty $\bar{\rho}$ has received less exposure, with almost all papers considering Poisson updates and just constant D. This includes $w(x) = 1$, where $\bar{\rho}$ is usually

called *divergence* [49] or *blur* [29], with analysis available in [31], [34], [84], and $w(x) = x$, where $\bar{\rho}$ is known as *additive age* [58], *aggregated age* [59], *delay* [84], or simply *cost* [31]. Finally, $\bar{\rho}$ with a general $w(x)$ was called *obsolescence cost* in [34] and analyzed under a non-stationary Poisson N_U, but no closed-form results were obtained.

The Poisson assumption on N_U allows easy computation of the various metrics of interest. Outside these special cases, superposition of non-memoryless processes produces much more complex behavior.

2.3 Age Model

While (2.3) is convenient, it is unclear whether these limits exist, if they are finite, and under what conditions they are deterministic. We investigate these issues next.

2.3.1 Main Framework

We start by performing a convenient transformation of (2.3) to remove the integrals. Define Q_T to be a uniform random variable in $[0, T]$, which models the random query time of consumers. Suppose Q_T is independent of N_U and N_D, in which case (2.3) is the limit of $E[\eta(Q_T)|N_U, N_D]$ and $E[\rho(Q_T)|N_U, N_D]$ as $T \to \infty$. To keep formulas manageable, we sometimes omit explicit conditioning on processes (N_U, N_D); however, it should be noted that all expectations and probabilities are still computed within each sample path (i.e., with respect to Q_T only).

At time t, suppose age processes $A_U(t)$ and $A_D(t)$, shown in Fig. 2.3(b), specify delays to the previous update and synchronization event, respectively. Using this notation and observing that $M(t) > 0$ is equivalent to $A_U(t) < A_D(t)$, define an

ON/OFF staleness process:

$$S(t) = \begin{cases} 1 & A_U(t) < A_D(t) \\ 0 & \text{otherwise} \end{cases}, \tag{2.4}$$

whose properties at random time Q_T determine whether the consumer sees outdated information or not.

To analyze (2.4), our next topic is the behavior of $A_U(Q_T)$ and $A_D(Q_T)$ as $T \to \infty$, including existence of these limits and their relationship to $\{U_i\}_{i=1}^{\infty}$ and $\{D_k\}_{k=1}^{\infty}$.

2.3.2 Assumptions

We next aim to establish a minimal set of conditions under which analysis of staleness admits closed-form results. Consider a point process N with cycle lengths $\{X_i\}_{i=1}^{\infty}$, where each $X_i \sim F_i(x)$ is a random variable. In order for the age $A(Q_T)$ of this process to have a usable limiting distribution as $T \to \infty$, one must impose three constraints on N, which we discuss informally and motivate next, followed by a more rigorous definition.

The first restriction is that collection $\{X_i\}_{i=1}^{\infty}$ within each sample-path have some limiting distribution $F(x)$. If this fails to hold, staleness in (2.3) does not exist either. The second prerequisite is that $F(x)$ not be a random limit. This condition ensures that almost all sample-paths produce the same result. Finally, the third condition is that an $o(1)$ fraction of cycles must consume an $o(1)$ fraction of length as $n \to \infty$. Allowing otherwise would be a problem because $F(x)$, being a limiting distribution, does not capture these intervals, but Q_T still lands there with a non-diminishing probability as $T \to \infty$ (we discuss an example demonstrating this effect shortly).

Let $\mathbf{1}_A$ be an indicator variable of A and $\bar{F}(x) = 1 - F(x)$ the complementary CDF (cumulative distribution function) of $F(x)$. We are now ready to summarize

our discussion.

Definition 4. *A process N is called* age-measurable *if:*

1. *For all $x \geq 0$, except possibly points of discontinuity of the limit, sample-path distribution $H_n(x)$ of variables $\{X_1, \ldots, X_n\}$ converges in probability as $n \to \infty$:*

$$H_n(x) := \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}_{X_i \leq x} \xrightarrow{\mathrm{P}} F(x); \tag{2.5}$$

2. *Function $F(x)$ is deterministic with mean $0 < \delta < \infty$;*

3. *The average cycle length converges to δ in probability as $n \to \infty$:*

$$Z_n := \frac{1}{n} \sum_{i=1}^{n} X_i \xrightarrow{\mathrm{P}} \delta = \int_0^\infty \bar{F}(x)dx. \tag{2.6}$$

Note that any renewal process $\{X_i\}_{i=1}^{\infty}$ satisfies this definition since all $F_i(x)$ are the same, which from the weak law of large numbers trivially leads to $F(x) = F_i(x)$ and $Z_n \to \delta$. Furthermore, condition (2.6) resembles mean-ergodicity, which is normally stated with a stronger type of convergence (e.g., mean-square or almost-sure) and only for *stationary* processes. For other cases, the fact that indicator variables are uniformly bounded allows application of the Dominated Convergence Theorem (DCT) [74] to show that $F(x)$ is the limiting average of individual distributions:

$$E[H_n(x)] = \frac{1}{n} \sum_{i=1}^{n} F_i(x) \to F(x). \tag{2.7}$$

It is pretty clear that (2.5) is not implied by (2.6). If $\{X_i\}_{i=1}^{\infty}$ are uniformly bounded, the reverse can be inferred (i.e., (2.6) follows from (2.5)); however, this

does not hold universally. In fact, many random variables used in practice (e.g., exponential and Pareto) are not bounded and thus require an explicit assumption that convergence in (2.6) take place. Additionally, even if this limit exists, it does not generally equal δ, which is why we require that as well.

2.3.3 Distribution

Define the sample-path distribution of age $A(Q_T)$, in points Q_T uniformly placed in $[0, T]$, to be:

$$G(x, T) := P(A(Q_T) \leq x | N). \tag{2.8}$$

For an age-measurable process N, suppose the residual (age) distribution of its $F(x)$ is given by:

$$G(x) := \frac{1}{\delta} \int_0^x \bar{F}(y) dy. \tag{2.9}$$

It is well-known that a renewal [103] or regenerative [85] assumption on N yields $G(x, T) \to G(x)$ as $T \to \infty$. Our next result produces a condition that is both sufficient and necessary for this to hold.

Theorem 1. *Process N is age-measurable if and only if $N(T)/T$ is almost surely bounded and $G(x, T)$ converges in probability to $G(x)$.*

Proof. We start with the forward (sufficiency) proof. Consider an age-measurable N and let $S_k = \sum_{i=1}^{k} X_i$ be the k-th arrival point of this process. Note that almost-sure boundedness of $N(T)/T$ immediately follows from (2.6) and the fact that $\delta > 0$. The rest of the proof deals with convergence of $G(x, T)$.

Assume some constant $x \geq 0$. Then, event $A(Q_T) \leq x$ is equivalent to the existence of some $k \geq 1$ such that Q_T belongs to the k-th interval $[S_k, S_{k+1})$, under

the condition that starting point $S_k \leq T$ and age $Q_T - S_k \leq x$. Defining

$$W_k = \min((T - S_k)^+, X_k, x), \tag{2.10}$$

where $(x)^+ = \max(x, 0)$, we get:

$$G(x, T) = \sum_{k=1}^{\infty} P(S_k \leq Q_T < S_k + W_k | N). \tag{2.11}$$

Since Q_T is uniform in $[0, T]$, the probability that it falls into an interval of length W_k is simply W_k/T:

$$G(x, T) = \sum_{k=1}^{N(T)} \frac{\min(T - S_k, X_k, x)}{T}, \tag{2.12}$$

where the upper limit is reduced from ∞ to $N(T)$ since $(T - S_k)^+ = 0$ for $k > N(T)$. Recalling that all probabilities and expectations are dependent on the sample path, it follows that $G(x, T)$ is a random variable. Our goal below is to show it converges to a constant as $T \to \infty$. To this end, first observe that it can be bounded as:

$$\frac{\sum_{k=1}^{N(T)-1} \min(X_k, x)}{\sum_{k=1}^{N(T)} X_k} \leq G(x, T) \leq \frac{\sum_{k=1}^{N(T)} \min(X_k, x)}{\sum_{k=1}^{N(T)-1} X_k}, \tag{2.13}$$

where we use the fact that $T - S_k \geq X_k$ for all $k \leq N(T)-1$ and $T \in [\sum_{k=1}^{N(T)-1} X_k, \sum_{k=1}^{N(T)} X_k]$.

Next, notice that (2.5) implies that for all bounded, continuous functions $f(x)$ [74]:

$$\frac{1}{n} \sum_{i=1}^{n} f(X_i) \xrightarrow{\text{P}} \int_0^{\infty} f(x) dF(x), \tag{2.14}$$

which leads to:

$$\frac{1}{N(T)} \sum_{k=1}^{N(T)-1} \min(X_k, x) \xrightarrow{\text{P}} \int_0^\infty \min(y, x) dF(y). \tag{2.15}$$

Using (2.6), we also have:

$$\lim_{T\to\infty} \frac{1}{N(T)} \sum_{k=1}^{N(T)-1} X_k = \lim_{n\to\infty} \frac{1}{n} \sum_{k=1}^{n-1} X_k = \delta. \tag{2.16}$$

Since both bounds in (2.13) have the same limit, $G(x, T)$ converges in probability[1] to the ratio of (2.15) to (2.16):

$$\frac{1}{\delta} \int_0^\infty \min(y, x) dF(y) = \frac{1}{\delta} \int_0^x \bar{F}(y) dy, \tag{2.17}$$

where the second integral follows from expanding the min function and integrating by parts.

We now present the reverse (necessity) proof. Assume that $G(x, T) \to G(x)$ for some $G(x)$ and $N(T)/T$ is bounded. Our goal is to show convergence of (2.5) and (2.6) to such deterministic limits that satisfy (2.9). From the Bolzano-Weierstrass theorem [38], every subsequence n_k contains a further subsequence $n'_k \to \infty$ such that

$$Z_{n'_k} \xrightarrow{\text{P}} \delta_{\{n'_k\}}, \tag{2.18}$$

where $\delta_{\{n'_k\}} > 0$ with probability 1 from $N(T)/T$ being bounded. Note that this limit may depend on the subsequence. To show finiteness of $\delta_{\{n'_k\}}$, notice that (2.13)

[1]Moreover, since $G(x, T)$ monotone and bounded by 1, convergence is uniform in x [4].

and $G(x,T) \to G(x)$ implies that for all $x \geq 0$:

$$\frac{\sum_{i=1}^{n} \min(X_i, x)}{\sum_{i=1}^{n} X_i} \xrightarrow{\text{P}} G(x). \tag{2.19}$$

The left side of (2.19) is concave in $[0, \infty]$, which means that $G(x)$ must be either concave or degenerate at $x = 0$. From (2.18) and (2.19), we know that:

$$x \geq \frac{1}{n'_k} \sum_{i=1}^{n'_k} \min(X_i, x) \xrightarrow{\text{P}} \delta_{\{n'_k\}} G(x), \tag{2.20}$$

where letting $x \to 0$ establishes that the latter case is impossible. Therefore, $G(x)$ must be concave, $G(x) > 0$ for $x > 0$, and finally $\delta_{\{n'_k\}} < \infty$.

From Helly's selection theorem [4], there exists a further subsequence $n''_k \to \infty$ of n'_k along which (2.18) holds and:

$$H_{n''_k}(x) \xrightarrow{\text{P}} F_{\{n''_k\}}(x), \tag{2.21}$$

where the limit is a proper CDF from Prohorov's theorem [4]. What remains to prove is that limits (2.18), (2.21) are independent of the subsequence and establish their relationship to $G(x)$. Using an analog of (2.14) for subsequences and applying (2.15):

$$\frac{\sum_{i=1}^{n''_k} \min(X_i, x)}{\sum_{i=1}^{n''_k} X_i} \xrightarrow{\text{P}} \frac{1}{\delta_{\{n''_k\}}} \int_0^x \bar{F}_{\{n''_k\}}(y) dy. \tag{2.22}$$

Invoking (2.19), this limit equals $G(x)$. Assuming $F(x)$ is some CDF, a function can be represented in the form of (2.9) using a *unique* pair $(\delta, F(x))$. Therefore, $F(x)$ must be $F_{\{n''_k\}}(x)$, which shows that for every subsequence n_k there exists a further subsequence n''_k such that $H_{n''_k}(x) \to F(x)$ and $Z_{n''_k} \to \delta$. But this means that the full sequence $H_n(x) \to F(x)$ and $Z_n \to \delta \in (0, \infty)$, i.e., (2.5), (2.6), and (2.9)

hold. □

To elaborate on this result, consider independent variables:

$$X_i = \begin{cases} 1 & \text{wp } 1 - 1/\sqrt{i} \\ \sqrt{i} & \text{wp } 1/\sqrt{i} \end{cases}, \tag{2.23}$$

whose limiting distribution in (2.5) is a constant with $\delta = 1$. However, Z_n in (2.6) converges to 2. Consequently, the distribution of age $A(Q_T)$ cannot be determined based on $F(x)$. Even worse, (2.9) suggests the age is uniform in $[0, 1]$, while $A(Q_T)$ is asymptotically finite only with probability $1/2$.

2.3.4 Expectation

While Theorem 1 establishes when $A(Q_T)$ has a limiting distribution, convergence of expectation $E[A(Q_T)|N]$ or suitability of $G(x)$ for computing it are not guaranteed. Furthermore, given that consumers may apply generic weights $w(x)$ to the various age-related metrics, it is important to identify when $E[w(A(Q_T))|N]$ exists as $T \to \infty$.

To build intuition for the next result, assume $X \sim F(x)$ is a non-negative variable and define its age A to be a random variable with CDF $G(x)$ in (2.9). Then, we are interested in the relationship between $E[w(A)]$ and X. To this end, suppose for any locally integrable function $w(x)$, we set $w_1(x) = w(x)$ and then recursively integrate the result $n - 1$ times to define:

$$w_n(x) := \int_0^x w_{n-1}(y)dy. \tag{2.24}$$

Using integration by parts in Lebesgue-Stieltjes integrals and keeping in mind

that $w_{n+1}(0) = 0$ for $n \geq 1$:

$$E[w_{n+1}(X)] = \int_0^\infty w_n(x)\bar{F}(x)dx = E[w_n(A)]E[X]. \tag{2.25}$$

Therefore, in order for $E[w(A)]$ to exist, one must ensure that both $E[w_2(X)]$ and $E[X]$ do. Note that the latter does so by (2.6), but the former requires an additional constraint.

Definition 5. *A point process N is called* age-measurable by weight function $w(x)$ *if it is age-measurable and*

$$\frac{1}{n}\sum_{i=1}^{n} w_2(X_i) \xrightarrow{P} \int_0^\infty w_2(x)dF(x) < \infty. \tag{2.26}$$

Note that age-measurable by a constant is equivalent to simply age-measurable since in that case (2.26) becomes (2.6). We omit the proof of the next result as it follows that of Theorem 1 pretty closely.

Theorem 2. *For a process N that is age-measurable by $w(x)$, the sample-path expectation of $w(A(Q_T))$ converges in probability as $T \to \infty$:*

$$\lim_{T\to\infty} E[w(A(Q_T))|N] = \int_0^\infty w(x)dG(x). \tag{2.27}$$

To understand this better, consider another counter-example:

$$X_i = \begin{cases} 1 & \text{wp } 1 - 1/i \\ 7\sqrt{i} & \text{wp } 1/i \end{cases} \tag{2.28}$$

The limiting distribution in (2.5) is again a constant equal to 1, but this time (2.6) converges to $\delta = 1$, which makes this process age-measurable. However, for

$w(x) = x$, the sum in (2.26) oscillates between 25 and 30 as n increases, while the corresponding integral is $1/2$. As a result, N is not measurable by $w(x)$ and $E[A(Q_T)]$ does not converge as $T \to \infty$.

From this point on, we omit explicit conditioning on the sample-path since results do not depend on N for age-measurable processes. However, we keep in mind that all probabilities and expectations involving Q_T are still taken in the sample-path sense.

2.4 Staleness Cost

This section models the probability of staleness and expected cost under both penalty metrics defined earlier.

2.4.1 Age Independence

We now return to examining (2.4). In order to determine when the replica is stale, one requires comparison of $A_U(Q_T)$ with $A_D(Q_T)$, which may not be independent random variables, even if N_U and N_D are. To prevent such cases, which are called *phase-lock* [5], conditions known as ASTA (Arrivals See Time Averages) [62] must apply to the age of one process when sampled by the arrival points of the other. This issue is delayed until a later section, but now we define more clearly what independence of $A_U(Q_T)$ and $A_D(Q_T)$ means.

Specifically, suppose (N_U, N_D) are age-measurable. Then, let $F_U(x)$ and $F_D(x)$ be respectively the limiting CDF functions of interval lengths defined in (2.5), with the corresponding average rates μ and λ, i.e.,

$$\frac{1}{\mu} = \int_0^\infty \bar{F}_U(x)dx \quad \text{and} \quad \frac{1}{\lambda} = \int_0^\infty \bar{F}_D(x)dx. \tag{2.29}$$

Further, let $U \sim F_U(x)$ and $D \sim F_D(x)$ be random update and download cycle

lengths. Similarly, suppose $G_U(x)$ and $G_D(x)$ are the limiting CDFs of age from (2.9), with lower-case functions $g_U(x)$ and $g_D(x)$ representing the corresponding PDFs. When the necessary limits exist, let $A_U \sim G_U(x)$ and $A_D \sim G_D(x)$ denote the two random ages as $T \to \infty$.

Definition 6. *Two age-measurable point processes N_U and N_D are called* age-independent *if for all $x, y \geq 0$:*

$$\lim_{T\to\infty} P(A_D(Q_T) < x | A_U(Q_T) = y) = G_D(x). \tag{2.30}$$

If either N_U or N_D is Poisson, (2.30) is guaranteed from PASTA (Poisson Arrivals See Time Averages) [102], which explains why prior work did not encounter these nuances. To shed more light on this condition, consider independent stationary renewal processes N_U and N_D with lattice[2] inter-arrival distributions, each with integer span. Due to stationarity, the first cycles are $U_1 \sim G_U(x)$ and $D_1 \sim G_D(x)$. Both $A_U(Q_T)$ and $A_D(Q_T)$ have continuous distributions; however, conditioning on the pair of sample paths, $A_U(Q_T) - A_D(Q_T) - U_1 + D_1$ can only take integer values. Consequently, age-independence (2.30) cannot hold, not even asymptotically.

[2]A random variable X is called *lattice* if there exists a constant c such that X/c is always an integer.

Unconditionally, however, the ages are independent:

$$\begin{aligned} P(A_U(Q_T) \leq x, A_D(Q_T) \leq y) & \\ &= E[E[\mathbf{1}_{A_U(t)\leq x}\mathbf{1}_{A_D(t)\leq y}|Q_T]] \\ &= \frac{1}{T}\int_0^T E[\mathbf{1}_{A_U(t)\leq x}\mathbf{1}_{A_D(t)\leq y}]dt \\ &= \frac{1}{T}\int_0^T P(A_U(t) \leq x)P(A_D(t) \leq y)dt \\ &= G_U(x)\, G_D(y), \end{aligned} \tag{2.31}$$

although this has no bearing on staleness.

To give a more concrete meaning to the conditional probability in (2.30), we have the next result.

Theorem 3. *Age-independence implies that $A_D(t)$ sampled in update points of N_U produces a sequence of random variables that converges in distribution to $G_D(x)$:*

$$\lim_{T\to\infty} \frac{1}{N_U(T)} \sum_{i=1}^{N_U(T)} \mathbf{1}_{A_D(u_i)<x} = G_D(x). \tag{2.32}$$

Proof. First define $d(y,T)$ to be the number of points in $[0,T]$ where $A_U(t) = y$ occurs. Recalling that $u_1 = 0$, this can be expressed as:

$$d(y,T) = \sum_{i=1}^{N_U(T)} \mathbf{1}_{U_i\geq y, u_i+y\leq T} = \sum_{i=1}^{N_U(T)} \mathbf{1}_{U_i\geq y} \cdot \mathbf{1}_{u_i+y\leq T}.$$

Next, let $c(y,T)$ be the number of these points in which the download age A_D is

smaller than x:

$$c(y,T) = \sum_{i=1}^{N_U(T)} \mathbf{1}_{U_i \geq y, u_i + y \leq T, A_D(u_i+y)<x}$$
$$= \sum_{i=1}^{N_U(T)} \mathbf{1}_{U_i \geq y} \cdot \mathbf{1}_{u_i+y \leq T} \cdot \mathbf{1}_{A_D(u_i+y)<x}. \tag{2.33}$$

Noticing that

$$P(A_D(Q_T) \leq x | A_U(Q_T) = y) = \frac{c(y,T)}{d(y,T)} \tag{2.34}$$

and applying Theorem 1, we get using (2.30):

$$G_D(x) = \lim_{T\to\infty} \lim_{y \to 0} \frac{c(y,T)}{d(y,T)}$$
$$= \lim_{T\to\infty} \frac{1}{N_U(T)} \sum_{i=1}^{N_U(T)} \mathbf{1}_{A_D(u_i)<x}, \tag{2.35}$$

which is (2.32) with the two sides swapped. □

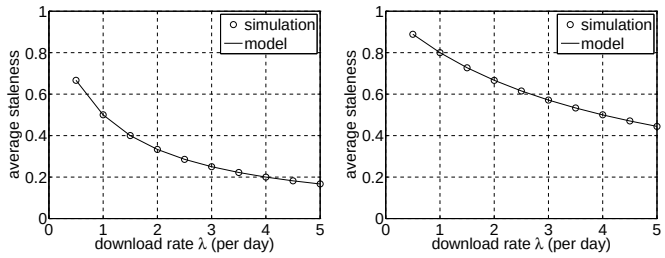
2.4.2 Preliminaries

Our first objective is to derive the probability of staleness.

Theorem 4. *Assuming that N_U and N_D are age-independent, the probability of staleness at time Q_T converges in probability as $T \to \infty$ to:*

$$P(S(Q_T) = 1) \to p := \mu \int_0^\infty \bar{F}_U(y)\bar{G}_D(y)dy. \tag{2.36}$$

Proof. Due to the existence and independence of $A_U(Q_T)$ and $A_D(Q_T)$ in the limit,



(a) Pareto U, constant D (b) constant U, Pareto D

Figure 2.4: Examination of (2.36) under $\mu = 2$.

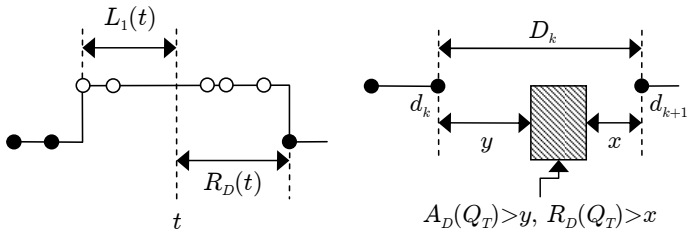
we immediately obtain:

$$p = P(A_D > A_U) = \int_0^\infty \bar{G}_D(y) dG_U(y). \tag{2.37}$$

Expanding $dG_U(y) = \mu \bar{F}_U(y) dy$ leads to the result. □

To perform a self-check against prior results with Poisson N_U, observe that (2.36) simplifies to $p = 1-\lambda(1-e^{-\mu/\lambda})/\mu$ under constant D and $\mu/(\mu+\lambda)$ under exponential D, which are consistent with [15], [20]. Simulations in Fig. 2.4 examine model accuracy in more interesting cases of general renewal processes. We use Pareto CDF $1 - (1 + x/\beta)^{-\alpha}$ with $\alpha = 3$ and mean $\beta/(\alpha - 1) = \beta/2$. Observe in the figure that the model matches simulations very well, with constant download intervals performing significantly better against Pareto update cycles in (a) than the other way around in (b). For example, synchronizing pages at their update rate (i.e., $\lambda = \mu = 2$) serves stale copies with probability 33% in the former case and 66% in the latter. Furthermore, for the same p, the scenario in (a) requires roughly 4 times less bandwidth than in (b).

The next intermediate result is the distribution of the first lag $L_1(Q_T)$, which relies on p in (2.36).



(a) staleness process $S(t)$ (b) reward of each cycle

Figure 2.5: Visualizing the proof of Theorem 5.

Theorem 5. *If N_U and N_D are age-independent, the CDF of $L_1(Q_T)$ converges in probability as $T \to \infty$ to:*

$$F_L(x) = 1 - \frac{\mu}{p} \int_0^\infty \bar{F}_U(y)\bar{G}_D(x+y)dy. \tag{2.38}$$

Proof. Consider the ON/OFF staleness process in Fig. 2.5(a) and suppose the query time t falls in the ON period. Then, since t is uniformly random within this cycle, the backward delay $L_1(t)$ is symmetrical to the forward (residual) delay $R_D(t)$, meaning they have the same distribution. Note that it is important to condition on $A_D(t) > A_U(t)$ since residual $R_D(t)$ depends on age $A_D(t)$, i.e.,

$$P(L_1(t) > x) = P(R_D(t) > x | A_D(t) > A_U(t)). \tag{2.39}$$

Since N_U and N_D are age-independent, we can condition on $A_U(Q_T) = y$ without impacting the distribution of $A_D(Q_T)$ or $R_D(Q_T)$. Following the proof of Theorem 1, define $L_k = d_k + y$ and $M_k = \min(T, d_{k+1} - x)$ to be the lower/upper boundaries within synchronization interval k such that if $Q_T \in [L_k, M_k]$, then $R_D(Q_T) > x$ and $A_D(Q_T) > y$. See Fig. 2.5(b) for an illustration.

Define $C_T = P(L_1(Q_T) > x | A_U(Q_T) = y)$ and observe that it converges as

$T \to \infty$:

$$\begin{aligned} C_T &= \frac{1}{pT} \sum_{k=1}^{\infty} P(L_k \leq Q_T \leq M_k) \\ &= \frac{1}{pT} \sum_{k=1}^{N(T)} (D_k - (x+y))^+ \\ &\to \frac{\lambda}{p} \int_0^{\infty} \max(z-(x+y), 0) dF_D(z) \\ &= -\frac{\lambda}{p} \int_{x+y}^{\infty} (z-(x+y)) d\bar{F}_D(z) = \frac{\lambda}{p} \int_{x+y}^{\infty} \bar{F}_D(z) dz \\ &= \frac{1}{p} \int_{x+y}^{\infty} g_D(z) dz = \frac{\bar{G}_D(x+y)}{p}. \end{aligned} \tag{2.40}$$

Unconditioning $A_U(Q_T)$ and keeping in mind that its distribution is well-defined as $T \to \infty$, we get (2.38). □

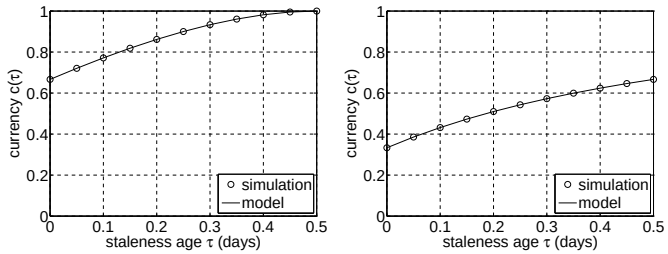
Theorem 5 allows a simple expression for the fraction of requests $c(\tau)$ that observe content outdated by less than τ time units, which was called β-currency in [7] and Δ-consistency in [97]. This can be expressed as:

$$c(\tau) = 1 - \bar{F}_L(\tau)p = \int_0^{\infty} g_U(y) G_D(\tau + y)) dy, \tag{2.41}$$

which conveniently simplifies to $P(A_D - A_U < \tau)$, where $A_D - A_U$ is the generalized lag between the replica and the source, i.e., non-positive values mean fresh states. Fig. 2.6 compares (2.41) to simulations using $\lambda = \mu$. As seen in the figure, this page retrieved at a random time is stale by less than $\tau = 0.4$ days (9.6 hours) with probability $c(\tau) = 98\%$ in the first case and 62% in the second.

2.4.3 Source Penalty

We are now ready to derive a general formula for $\bar{\eta}$.



(a) Pareto U, constant D (b) constant U, Pareto D

Figure 2.6: Examination of (2.41) under $\lambda = \mu = 2$.

Theorem 6. *If N_U and N_D are age-independent, while N_D is age-measurable by $w(x)$, the source penalty converges in probability to:*

$$\bar{\eta} = \lambda\mu \int_0^\infty \bar{F}_U(y) \int_0^\infty w(x)\bar{F}_D(x+y)dxdy. \tag{2.42}$$

Proof. First, observe that

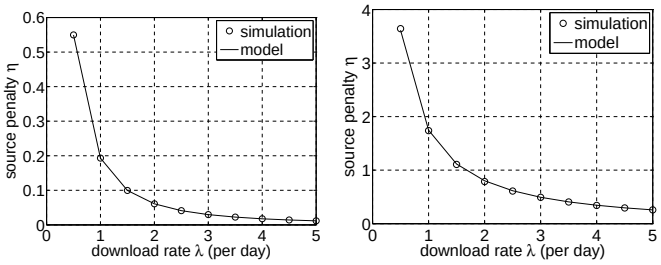
$$\bar{\eta} = E[w(L_1)]P(A_D > A_U), \tag{2.43}$$

where $L_1 \sim F_L(x)$. Working back from (2.38), the tail CDF of L_1 can be written more compactly as:

$$P(L_1 > x) = P(A_D - A_U > x | A_D > A_U). \tag{2.44}$$

Since $L_1 = A_D - A_U > 0$, conditioned on $A_D > A_U$, it suffices that only N_D be measurable by $w(x)$. In that case:

$$\bar{\eta} = E[w(L_1)]P(A_D > A_U) = p \int_0^\infty w(x)dF_L(x), \tag{2.45}$$



(a) Pareto U, constant D (b) constant U, Pareto D

Figure 2.7: Examination of (2.42) under $w(x) = x, \mu = 2$.

or equivalently:

$$\bar{\eta} = \int_0^\infty g_U(y) \int_0^\infty w(x) g_D(x+y) dx dy. \tag{2.46}$$

which immediately leads to (2.42) after expansion of $g_U(x) = \mu \bar{F}_U(x)$ and $g_D(x+y) = \lambda \bar{F}_D(x+y)$. □

With $w(x) = 1$, (2.42) reduces to staleness probability p already discussed above. For the other case $w(x) = x$ seen in the literature, we obtain the expected *staleness age* $\bar{\eta} = E[L_1(t)]p$ by which the replica trails the source. Under Poisson N_U and constant D, we get from (2.42):

$$\bar{\eta} = \frac{1}{2\lambda} - \frac{1}{\mu} + \frac{\lambda(1 - e^{-\mu/\lambda})}{\mu^2}, \tag{2.47}$$

and when both distributions are exponential:

$$\bar{\eta} = \frac{\mu}{\lambda(\lambda + \mu)}. \tag{2.48}$$

These special cases are consistent with [14]. Simulations in Fig. 2.7 additionally confirm that (2.42) is accurate under general renewal processes. Also observe in the

figure that the combination in (b) continues to offer inferior performance to that in (a); however, the difference between the two scenarios is now more pronounced. For example, using the same $\lambda = \mu$ considered earlier, search-engine clients encounter indexing results outdated on average by 0.06 days (1.5 hours) in the left subfigure and by 0.8 days (19 hours) in the right. This example shows how drastically the cost changes based on the *shape* of $F_U(x)$ and $F_D(x)$, which emphasizes the importance of utilizing models that can accurately handle any underlying processes (N_U, N_D).

We now offer a more intuitive look at source penalty. Modifying $w(x)$ to be zero for negative x, we can rewrite (2.42) in a more compact form:

$$\bar{\eta} = E[w(A_D - A_U)] = \lambda E[w_2(D - A_U)]. \tag{2.49}$$

This result shows that $\bar{\eta}$ is determined by the *positive* deviation of the generalized lag $A_D - A_U$ from zero, or equivalently by that of $D - A_U$, where the weight applied to each deviation is given respectively by $w(x)$ and $w_2(x)$. The only caveat is that simplification (2.49) requires weight functions that can explicitly handle negative arguments, e.g., a constant penalty would be $w(x) = \mathbf{1}_{x \geq 0}$ rather than just $w(x) = 1$. Throughout the rest of the paper, we avoid the extra notation dealing with $x < 0$, but keep this in mind.

2.4.4 Update Penalty

Unlike the previous section, we next show that $\bar{\rho}$ admits a much simpler result that depends only on the mean update rate μ rather than the entire distribution $F_U(x)$. This was first observed through simulations in [31] for constant D, but no explanation or extension to other cases was offered.

Theorem 7. *Assuming N_U and N_D are age-independent, while N_D is age-measurable*

by $w_2(x)$, the update penalty converges in probability to:

$$\bar{\rho} = \mu E[w_2(A_D)] = \lambda\mu E[w_3(D)]. \tag{2.50}$$

Proof. Using Lebesgue-Stieltjes integrals and treating point processes as random measures, we can re-write (2.2) as:

$$\rho(t) = \int_{t-A_D(t)}^{t} w(t-s)dN_U(s). \tag{2.51}$$

Taking the expectation along each sample path:

$$\begin{aligned}
\bar{\rho} &= \lim_{T\to\infty} E\Big[\int_{Q_T - A_D(Q_T)}^{Q_T} w(Q_T - s)dN_U(s)\Big] \\
&= \lim_{T\to\infty} \frac{1}{T}\int_0^T \int_{t-A_D(t)}^{t} w(t-s)dN_U(s)dt \\
&= \lim_{T\to\infty} \frac{1}{T}\int_0^T w_2(A_D(t))dN_U(t) \\
&= \lim_{T\to\infty} \frac{1}{T}\sum_{i=1}^{N_U(T)} w_2(A_D(u_i)).
\end{aligned} \tag{2.52}$$

Applying (2.32), the sequence $\{A_D(u_1), A_D(u_2), \ldots\}$ sampled in update points $\{u_i\}$ converges in distribution to that of $A_D(Q_T)$ as $T \to \infty$. Then, (2.52) becomes:

$$\bar{\rho} = \lim_{T\to\infty} \mu E[w_2(A_D(Q_T))]. \tag{2.53}$$

Since N_D is $w_2(x)$-measurable, (2.27) shows that this expectation converges and its limit equals $\mu E[w_2(A_D)]$. By (2.25), this is also $\lambda\mu E[w_3(D)]$. □

To compare against prior results, consider Poisson N_U and constant D. Then, (2.50) produces $\bar{\rho} = \mu/(2\lambda)$ for $w(x) = 1$ and $\mu/(6\lambda^2)$ for $w(x) = x$, both of which

match previous analysis of these special cases [58], [84], [106]. Generalizing to exponential D, we obtain from (2.50) respectively μ/λ and μ/λ^2. Interestingly, this shows that switching downloads from constant intervals to exponential *doubles* the number of missing updates and *sextuples* their combined age.

For $w(x) = 1$, a simple closed-form expression is possible for all D:

$$E[M(t)] = \frac{\lambda\mu E[D^2]}{2} = \frac{\mu}{2\lambda}\Big(1 + \lambda^2 Var[D]\Big). \tag{2.54}$$

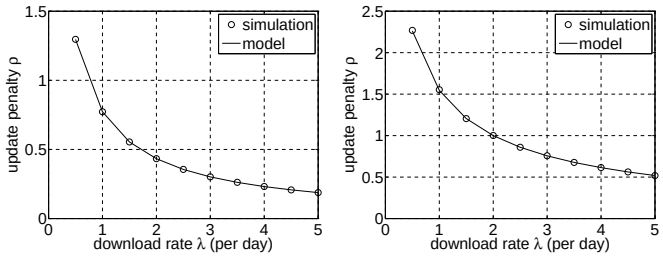
For example, Pareto D produces in (2.54):

$$E[M(t)] = \frac{\mu(\alpha - 1)}{\lambda(\alpha - 2)}, \tag{2.55}$$

which for $\alpha = 3$ is quadruple that of constant D and double that of exponential D. Another peculiar case is $\alpha \to 2$, where $E[M(t)]$ tends to infinity regardless of N_U. In fact, the update process itself may exhibit $Var[U] = \infty$, but the expected number of updates by which the replica falls behind will still become unbounded as α approaches 2.

Since source penalty $\bar{\rho}$ sums up the ages of all missing updates, it allows usage of *decaying* functions $w(x)$ such that their integral is increasing. We demonstrate this effect using $w(x) = 1/(1 + x)$, for which $w_2(x) = \log(1 + x)$. This cost function increases rapidly for small x, but then becomes less sensitive to staleness as the age of replicated content grows. Since $w_3(x) = (1 + x)\log(1 + x) - x$, constant D yields:

$$\bar{\rho} = \mu[(\lambda + 1)\log(1 + 1/\lambda) - 1]. \tag{2.56}$$



(a) Pareto U, constant D (b) constant U, Pareto D

Figure 2.8: Examination of (2.56) and (2.57) under $\mu = 2$.

For $D \sim \text{Pareto}(\alpha, \beta)$ with $\alpha = 3$ and $\beta = 2/\lambda$, we get:

$$\bar{\rho} = 2\mu \begin{cases} \frac{2\log(2/\lambda) - 2 + \lambda}{(\lambda-2)^2} & \lambda \neq 2 \\ 0.25 & \lambda = 2 \end{cases}. \tag{2.57}$$

Fig. 2.8 confirms that both models are accurate, with constant D enjoying a 60% lower penalty compared to Pareto.

2.5 Optimality

Motivated by (2.54) and consistently worse performance of Pareto D, the goal of this section is to understand the impact, if any, of $Var[D]$ on penalty and determine whether there exists an optimal distribution $F_D(x)$ that, for a fixed download budget λ, provably results in the lowest cost for all N_U and all suitable functions $w(x)$.

2.5.1 Stochastic Dominance

We start with general concepts from economics and game theory that are useful for understanding optimality. For two non-negative random variables $X \sim F_X(x)$ and $Y \sim F_Y(x)$, let their CDF difference be:

$$H(x) = F_Y(x) - F_X(x), \tag{2.58}$$

whose generalization $H_n(x)$ is given by (2.24). Then, we have the following definition.

Definition 7. *Variable X is said to stochastically dominate Y in n-th order, which we write as $X \geq_{st}^{n} Y$, if $H_n(x) \geq 0$ for all $x \in \mathbb{R}$.*

This concept is important because desirable characteristics of D can be inferred from those of A_D, as shown next.

Lemma 1. *Assume $E[X] = E[Y]$ and $n \geq 2$. Then, X stochastically dominates Y in n-th order, i.e., $X \geq_{st}^{n} Y$, iff the age of Y stochastically dominates the age of X in $(n-1)$-st order, i.e., $A_Y \geq_{st}^{n-1} A_X$.*

Proof. Let $G_X(x)$ and $G_Y(x)$ be the CDF of A_X and A_Y, respectively. Define:

$$J(x) = G_Y(x) - G_X(x), \tag{2.59}$$

which can be expressed using $H_2(x)$ as:

$$\begin{aligned} J(x) = & \quad \frac{\int_0^x (1 - F_Y(x))dy}{E[Y]} - \frac{\int_0^x (1 - F_X(y))dy}{E[X]} \\ = & \quad \frac{\int_0^x (F_X(y) - F_Y(y))dy}{E[X]} = -\frac{1}{E[X]} H_2(x). \end{aligned} \tag{2.60}$$

Integrating both sides $n-2$ additional times leads to:

$$J_{n-1}(x) = -\frac{1}{E[X]} H_n(x). \tag{2.61}$$

From this and Definition 7, it follows that $X \geq_{st}^{n} Y$ implies $A_Y \geq_{st}^{n-1} A_X$ and vice versa. □

As given by the next lemma, first-order stochastic dominance allows one to determine the relationship between expected utilities $E[w(X)]$ and $E[w(Y)]$. While it is

possible to establish a more general version of this result using n-th order dominance, it would restrict $w(x)$ to a narrower class of functions and thus would be less useful in practice.

Lemma 2. *Condition $X \geq_{st}^{1} Y$ holds iff for all non-decreasing functions $w(x)$ it follows that $E[w(X)] \geq E[w(Y)]$.*

2.5.2 Penalty Analysis

Returning to the topic of information staleness, our goal is to determine the condition under which both types of penalty can be reduced without changing the refresh rate. Define $\bar{\eta}(D_1)$ and $\bar{\eta}(D_2)$ to be the source penalties corresponding to random synchronization intervals D_1 and D_2, both with mean $1/\lambda$. For the opposite problem, i.e., finding the worst update distribution, define $\bar{\eta}(U_1)$ and $\bar{\eta}(U_2)$ to be the penalties that correspond to update intervals U_1 and U_2 under a fixed μ.

The next result shows that stochastic (rather than variance) ordering is needed to improve staleness penalty. Define $w(x)$ to be a *measure* if it is non-negative, non-decreasing, and right-continuous with $w(x) = 0$ for $x < 0$.

Theorem 8. *Assume the conditions of Theorem 6. For a given N_U and fixed download rate λ, $D_1 \geq_{st}^{2} D_2$ iff $\bar{\eta}(D_1) \leq \bar{\eta}(D_2)$ for all measures $w(x)$. Similarly, with a given N_D and fixed μ, $U_1 \geq_{st}^{2} U_2$ iff $\bar{\eta}(U_1) \geq \bar{\eta}(U_2)$ for all measures $w(x)$.*

Proof. Using (2.49), observe that $\bar{\eta} = E[w(A_D - A_U)]$ is fully determined by the properties of variable $X = A_D - A_U$. For a fixed A_U, it is not difficult to show that X becomes stochastically smaller in first order iff A_D does. Applying Lemmas 1-2, this means that penalty $\bar{\eta}$ gets smaller iff D increases stochastically in second order.

Similarly, for a fixed A_D, X gets stochastically larger in first order iff A_U becomes stochastically smaller. Again applying Lemmas 1-2, penalty $\bar{\eta}$ increases iff U becomes stochastically larger in second order. □

A similar result holds under update penalty $\bar{\rho}$. Note that all $F_U(x)$ with the same μ are equivalent here, which is why we state only half of Theorem 8, and $w(x)$ is less restricted.

Theorem 9. *Assume the conditions of Theorem 7. For a given N_U and fixed λ, $D_1 \geq_{st}^2 D_2$ iff $\bar{\rho}(D_1) \leq \bar{\rho}(D_2)$ for all non-negative $w(x)$.*

Proof. Since $\bar{\rho} = \mu E[w_2(A_D)]$, where $w_2(x)$ is a measure for all non-negative $w(x)$, Lemmas 1-2 yield that $\bar{\rho}$ decreases iff D gets stochastically larger in second order. □

The preceding results set up motivation to ask the question of whether there exists a distribution that dominates all others in second order. We answer this next.

Lemma 3. *For a given mean, a constant stochastically dominates all other random variables in second order.*

Proof. Suppose l is the fixed mean of all distributions under consideration. Let $F_X(x) = \mathbf{1}_{x>l}$ be the CDF of a constant and $F_Y(x)$ be the CDF of another random variable Y such that $E[Y] = l$. Our goal is to show that $H_2(x) \geq 0$.

When $x \leq l$, we have trivially:

$$H_2(x) = \int_0^x (F_Y(y) - F_X(y))dy = \int_0^x F_Y(y)dy \geq 0. \tag{2.62}$$

For $x > l$, we get:

$$\begin{aligned} H_2(x) &= \int_0^l F_Y(y)dy + \int_l^x (F_Y(y) - 1)dy \\ &= l + \int_0^x F_Y(y)dy - x = l - \int_0^x (1 - F_Y(y))dy \\ &\geq l - \int_0^\infty (1 - F_Y(y))dy = 0, \end{aligned} \tag{2.63}$$

where we use the fact that $l = \int_0^\infty (1 - F_Y(y))dy$. □

This leads to the main result of this section.

Theorem 10. *When the conditions of Theorems 8-26 hold, constant inter-synchronization delays are optimal under the corresponding staleness metric.*

This allow us to resolve the relationship between the variance of D and penalty. If $E[D_1] = E[D_2]$, then $D_1 \geq_{st}^2 D_2$ implies $Var[D_1] \leq Var[D_2]$, but the opposite is not true. This shows that for a given download rate, just reducing the variance of refresh intervals, without enforcing $D_1 \geq_{st}^2 D_2$, is *insufficient* to improve the penalty across *all* functions $w(x)$. As an example, recall the special case of $\bar{\rho}$ with $w(x) = 1$ in (2.54), where the penalty was reduced iff the variance of D was; however, no such causality exists for $w(x) = x$ or $\log(1 + x)$. On the other hand, if reduction in penalty holds for all measures $w(x)$, then stochastic ordering between D_1 and D_2 follows and thus variance has to decrease (i.e., ordering of variances is *necessary*, but not sufficient).

2.5.3 Phase-Lock

Even though constant D is optimal from the staleness perspective, it unfortunately fails to guarantee age-independence (2.30) against *all* underlying N_U. We now deal with principles related to ASTA (Arrivals See Time Averages) [62], placing them in our context. In general, ASTA can be viewed as a condition that allows discrete and continuous sample-path averages of a process $X(t)$ to be equal almost surely:

$$\lim_{n\to\infty} \frac{1}{n} \sum_{k=1}^{n} X(t_k) = \lim_{T\to\infty} \frac{1}{T} \int_0^T X(t)dt. \tag{2.64}$$

Let $X(t) = \mathbf{1}_{A_D(t)<x}$ and $t_k = u_k$. Then, if (2.64) holds for all x, it follows that the distribution of refresh age $A_D(t)$ sampled in update points u_k equals that

sampled in uniformly random instances Q_T, which in turn is equivalent to our earlier formulation (2.30). While we have given conditions for the right side of (2.64) to exist and equal a constant almost surely, existence of the left side or its equality to the integral is not guaranteed. ASTA analysis focuses on the properties of points $\{t_k\}$ and their relationship to $X(t)$ that allow (2.64) to hold; however, this normally requires conditions that are difficult to verify in practice (e.g., LAA, WLAA, LBA [62]). We therefore discuss guidelines for ensuring that (2.64) is satisfied, without becoming engrossed in unnecessary rigor.

While sampling constant update cycles with constant synchronization intervals sometimes leads to phase-lock, we next discuss how to achieve asymptotic age-independence in such cases. To build intuition, suppose $U_i = 5$ and $D_k = \pi$ for all $k \geq 1$. Then, from the equidistribution theorem, $A_U(d_k) = k\pi \pmod 5$ is a uniformly random variable in $[0, 5]$, meaning that $A_U(d_k)$ has the same distribution as $A_U(Q_T)$. The key observation is to ensure that N_D puts its download points uniformly across the cycles of N_U.

In general, sequence $a_k = k\xi \pmod T$, where T is rational, ξ is irrational, and $k \in \mathbb{N}$, is uniformly distributed in $[0, T]$, in which case for any Riemann-integrable function, an ASTA-like condition automatically holds:

$$\lim_{n\to\infty} \frac{1}{n} \sum_{k=1}^{n} f(a_k) = \frac{1}{T} \int_0^T f(x)dx. \tag{2.65}$$

While using $D_k = \pi$ to sample $U_i = 5$ works well, there is a possibility that U_i itself happens to be a multiple of π. To preclude these cases, N_D must exhibit enough randomness to prevent D_k/U_i from becoming deterministically an integer. One option for doing so is to require that either process employ non-lattice cycle lengths. Recall that non-lattice distributions may be entirely continuous, including

the classical PASTA (Poisson Arrivals See Time Averages) [102] and the uniform distribution often suggested for network measurement [5]. However, they can also be entirely discrete. In such cases, cycle lengths must distribute mass across at least two values (a, b), where a/b is irrational, e.g., pairs $(\pi, 1)$ or $(e, \sqrt{2})$. By bringing spread $|b - a|$ closer to zero, it is possible to obtain a variety of approximations to the optimal (constant) synchronization delay with mean $l = (a + b)/2$.

Note that a non-lattice distribution may still enter phase-lock if its cycle lengths follow a deterministic pattern, e.g., both updates and downloads strictly alternate between 1 and π. To rule out these cases, it is sufficient to require that the non-lattice process randomize its delays, leaving the other one general. This produces the following.

Theorem 11. *If either N_U or N_D uses iid, non-lattice cycle lengths, age-independence holds.*

Proof. The result follows from the equidistribution theorem and the iid nature of delays. □

2.6 Applications

We now examine the presence of Poisson updates in real data sources and show how to apply the developed models to solve several classes of multi-source/replica problems.

2.6.1 Real-Life Update Processes

We first discuss possible reasons for the frequent use of memoryless source-update processes in the literature. If indeed this is universal, extensions to non-Poisson dynamics may be unnecessary. While modeling convenience is one possible explanation [20], there is certain belief in the field that updates to individual web pages can be

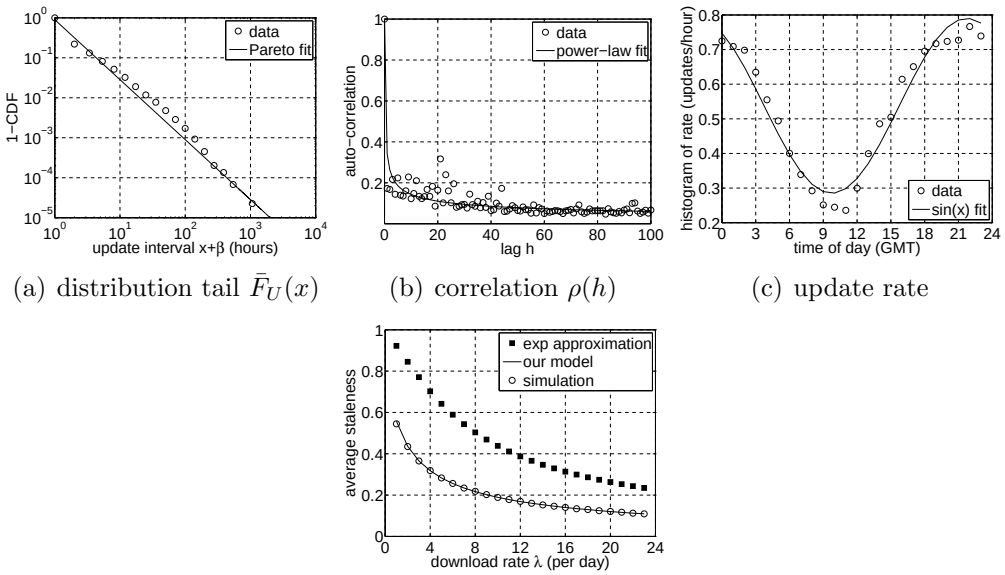
accurately described by a Poisson process, which has fueled this line of modeling for over a decade [7], [8], [14], [15], [17], [31], [34], [40], [49], [53], [59], [61], [65], [66], [84], [95], [106].

Intuitively, there is no fundamental reason why a single source should exhibit Poisson dynamics, especially when modified by humans. A more likely scenario would be heavy-tailed behavior observed in many areas of computer networks [25], [54], [72] and user-driven distributed systems [9], [79], [92]. Another intuitively reasonable inter-update distribution is constant, where certain information is injected into the system periodically by design or is obtained from an ON/OFF source (e.g., sensors trying to conserve energy).

Closer examination of the origin [15] of the Poisson conclusion reveals several limitations. First, the distribution of page inter-update intervals was sampled using *incomplete observation*, meaning that some of the updates went unnoticed. As a result, bias could have been introduced in the measurements. Second, the exponential distribution was fitted to updates of *multiple* pages rather than a single page. Poisson dynamics have been known to emerge when aggregating arrival processes [2] and summing up variables [3], which does not tell us much about the individual distributions being combined. Finally, to conclude that N_U is Poisson, it is insufficient to observe an exponential distribution in $\{U_i\}_{i=1}^{\infty}$; instead, one must also show stationary independent increments [103].

2.6.2 Wikipedia

Even though certain measurement studies [7, 18, 45, 63] have found non-Poisson updates among web pages, they also lack ground truth. These pitfalls can be avoided if model verification is performed over sources that expose information about *each* update. One particularly interesting source with public traces of all modification



(a) distribution tail $\bar{F}_U(x)$ (b) correlation $\rho(h)$ (c) update rate

(d) model accuracy

Figure 2.9: George W. Bush page dynamics.

timestamps is Wikipedia [100]. From a search-engine perspective, this website represents a realistic example of data churn stemming from user interaction with each other (e.g., edits from other people), flash crowds in response to external events, and diurnal activity patterns of the human lifecycle. Wikipedia is also well-suited for purposes of model validation and discussion.

To shed light on the complexity of real $F_U(x)$, we plot in Fig. 2.9(a) the tail CDF of inter-update delay for the most frequently modified article – "George W. Bush" with $44,296$ updates in 10 years (mean delay $E[U] = 1.86$ hours). The figure is a close match to Pareto tail $(1 + x/\beta)^{-\alpha}$ with $\alpha = 1.4$ and $\beta = 0.93$. In Fig. 2.9(b), we show the corresponding auto-correlation function $\rho(h)$ with a power-law fit $h^{-0.37}$, which suggests long-range dependence (LRD) with Hurst parameter 0.81. Of course, LRD effects might be caused and/or compounded by non-stationarity. To address this question, Fig. 2.9(c) shows the update rate throughout the day, clearly

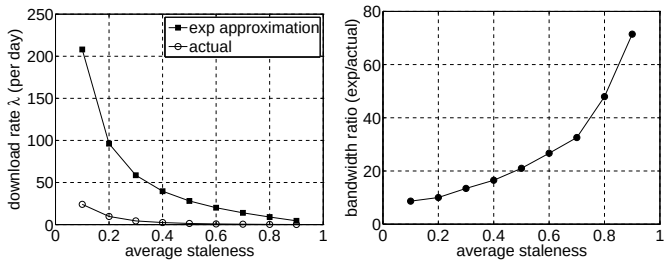
indicating non-stationary dynamics.

This example underscores the need to keep the model general and not limit results to renewal or even stationary cases, which was our goal with assumptions (2.5)-(2.6). Approximating $F_U(x)$ as non-lattice and using constant D, we next compute the probability of staleness for this page by supplying (2.42) with George W. Bush' empirically computed distribution $F_U(x)$. We contrast the result against the closest Poisson formula $1 - \lambda(1 - e^{-\mu/\lambda})/\mu$ from [15]. Fig. 2.9(d) shows that (2.42) is accurate, but the Poisson approximation suffers over 100% relative error for much of the examined range.

What is more important is the performance of the model in providing an accurate assessment of the download bandwidth needed to achieve a given p. We invert the formulas to solve for λ as a function of p and plot the result in Fig. 2.10(a). These results show a much more dramatic difference. For example, 20% staleness requires 95 downloads/day according to previous Poisson models, while in reality this can be achieved with just 8. To illustrate this better, we show the ratio of these two curves in Fig. 2.10(b), where the amount of Poisson overestimation varies from one to almost two orders of magnitude depending on the desired p.

2.6.3 Aggregation (Many-to-One)

When a single replica tracks M sources, as in Fig. 2.1(a), performance is assessed by its ability to provide usable aggregate information to the consumer. If sources are independent, many results are relatively easy to obtain. For example, consider a system that selects a replica and loads it with a MapReduce job that has to execute over the data of all sources. A computation may be considered successful if at least one source is fresh at the time of job request. Then, the fraction of successful attempts is $1 - \prod_{i=1}^{M} p_i$, where p_i in (2.36) is the probability of staleness for source i.



(a) crawl rate (b) ratio

Figure 2.10: Application of staleness models to the update process of George W. Bush.

Alternatively, application consistency may require that *all* sources be simultaneously fresh, which leads to the probability of success via $\prod_{i=1}^{M}(1-p_i)$.

A more interesting problem is optimal allocation of download rates to different sources. Suppose q_i is the probability that an incoming query requests data from source i and μ_i is its update rate. Then, the goal is to allocate refresh rates λ_i so as to optimize the expected staleness cost $C(\lambda_i, \mu_i)$ for a given bandwidth budget Λ:

$$\min \sum_{i=1}^{M} q_i C(\lambda_i, \mu_i) \qquad \text{subject to} \qquad \sum_{i=1}^{M} \lambda_i \leq \Lambda, \tag{2.66}$$

where $C(\lambda_i, \mu_i)$ refers to either $\bar{\eta}$ or $\bar{\rho}$.

For $\Lambda \ll \sum_{i=1}^{M} \mu_i$ and certain choices of $w(x)$, solutions to (4.38) using cost $\bar{\eta}$ are known to completely starve frequently modified sources in favor of those that are updating slowly [15]. Since (4.38) does not have a closed-form solution under $\bar{\eta}$ even in the simplest cases, specific conditions for starvation are not clear. Complete loss of synchronization for sources whose μ_i is above some (typically unknown) threshold may be an unwelcome surprise for many applications. This naturally leads to the question of whether $\bar{\rho}$ suffers from the same drawback. We address this next.

Theorem 12. *Assume $q_i\mu_i > q_j\mu_j > 0$ and let refresh delays be optimal (i.e.,*

constant). Then, the solution to (4.38) *using* $\bar{\rho}$ *guarantees that* $\lambda_i > \lambda_j > 0$.

Proof. Using Lagrange multipliers, we get that all partial derivatives of $q_i C(\mu_i, \lambda_i)$ must equal some constant κ:

$$\kappa = -\frac{\partial[q_i C(\mu_i, \lambda_i)]}{\partial \lambda_i} = -q_i \mu_i \frac{\partial[\lambda_i w_3(1/\lambda_i)]}{\partial \lambda_i}, \tag{2.67}$$

which follows from (2.50) and $D_i = 1/\lambda_i$. Expanding, we get $\kappa = q_i \mu_i f(\lambda_i)$, where

$$f(x) = \frac{w_2(1/x)}{x} - w_3(1/x) \tag{2.68}$$

is a monotonically non-increasing function:

$$f'(x) = -\frac{w_2(1/x)}{x^2} - \frac{w(1/x)}{x^3} + \frac{w_2(1/x)}{x^2} = -\frac{w(1/x)}{x^3}. \tag{2.69}$$

Notice that $f(\lambda_i) \geq 0$ for all λ_i since w_3 is an integral of $w_2(x)$ from 0 to $1/\lambda_i$. Therefore, $\kappa \geq 0$ and the relationship between μ_i and λ_i is determined by:

$$\lambda_i = f^{-1}\Big(\frac{\kappa}{q_i \mu_i}\Big). \tag{2.70}$$

If since f is non-increasing, larger $q_i \mu_i$ implies larger λ_i. Finally, since $f(x) > 0$ for all $x > 0$, it follows that its inverse f^{-1} has the same properly and thus no positive $q_i \mu_i > 0$ can achieve $\lambda_i = 0$. This means the optimal allocated rate must be strictly positive (i.e., no starvation). □

To explain how optimization with $\bar{\rho}$ can be used, we assume constant D and $w(x) = 1$, with the goal to maximize $\sum_{i=1}^{M} q_i E[M_i(t)]$. Solving (4.38), the optimal

download rate of each page is proportional to the square root of $q_i\mu_i$:

$$\lambda_i = \Lambda \frac{\sqrt{q_i\mu_i}}{\sum_{j=1}^{M}\sqrt{q_j\mu_j}}. \tag{2.71}$$

The optimal penalty is then:

$$\sum_{i=1}^{M} q_i E[M_i(t)] = \sum_{i=1}^{M} \frac{q_i\mu_i}{2\lambda_i} = \frac{(\sum_{j=1}^{M}\sqrt{q_j\mu_j})^2}{2\Lambda}. \tag{2.72}$$

Define random variable μ to have the same distribution as $\{\mu_1, \ldots, \mu_M\}$. Then, for the most basic scenario where all pages are equally popular, i.e., $q_i = 1/M$, we get:

$$\sum_{i=1}^{M} q_i E[M_i(t)] = M\frac{E[\sqrt{\mu}]^2}{2\Lambda}. \tag{2.73}$$

For the other extreme, where pages are searched for in proportion to their modification rate, i.e., $q_i \sim \mu_i$, we have:

$$\sum_{i=1}^{M} q_i E[M_i(t)] = M\frac{E[\mu]}{2\Lambda}. \tag{2.74}$$

To put these models in perspective, we use Wikipedia's distribution of μ, which happens to be quite heavy-tailed (i.e., Zipf shape $\alpha = 0.6$). The average update rate across all pages is $E[\mu] = 8$ updates/day; however, 98% of them exhibit μ_i less than 1/day, 90% less than 1/week, and 50% below 8/year. Using this distribution in (2.73) and (2.74) shows that optimizing staleness of the entire Wikipedia under uniform page access $q_i = 1/M$ requires 46 times less bandwidth Λ than under Zipf. This can be explained by the fact that keeping frequently modified pages fresh costs

more bandwidth. This effect is related to the variance of $\sqrt{\mu}$:

$$\frac{E[\mu]}{E[\sqrt{\mu}]^2} = \frac{Var[\sqrt{\mu}] + E[\sqrt{\mu}]^2}{E[\sqrt{\mu}]^2} = \frac{Var[\sqrt{\mu}]}{E[\sqrt{\mu}]^2} + 1. \tag{2.75}$$

Consider extrapolating these results to $M = 100$B sources and keeping the expected consumer lag $\sum_{i=1}^{M} q_i E[M_i(t)]$ below ω updates. We use the two models above as lower/upper bounds on the actual search-engine crawl rate. The first case requires download capability $\Lambda_1 = M \cdot E[\sqrt{\mu}]^2/2\omega = 99/\omega$ thousand pages per second (pps), while the second one $\Lambda_2 = M \cdot E[\mu]/2\omega = 4.6/\omega$ million pps. For $\omega = 10$ and 25 KB per page, these translate into 2 and 92 Gbps, respectively. Results can be easily adjusted to non-Wikipedia situations as long as $E[\sqrt{\mu}]$ and $E[\mu]$ are known.

2.6.4 Load-Balancing (One-to-Many)

The issue of redundant replication from a single source, as in Fig. 2.1(b), to m nodes is quite different from the opposite case considered in the previous subsection. When the source fails, suppose the goal is to deduce the expected penalty afforded by the freshest member of the entire collection of m replicas. The issue at stake is how this $1 \times m$ case compares to a single replica with some refresh rate λ and optimal D. To keep comparison fair, assume that each of the m replicas is allowed budget λ/m in synchronization with the source. Decentralized operation leads to much better robustness under failure, but is it possible that this causes reduced freshness? If so, what is the amount of extra download bandwidth needed to keep both scenarios equally stale?

The main caveat in solving this problem is that staleness at different replicas is no longer independent. This happens because updates at the source simultaneously make all copies outdated, which means that reliability does not benefit exponentially with increased m. To overcome this issue, let $N_D^1, \ldots, N_D^m$ be the download

processes used by the individual replicas. Then, observe that the entire collection can be replaced by a single replica that implements a refresh pattern N_D^*, which is a *superposition* of all point processes $\{N_D^i\}_{i=1}^m$. Therefore, the source can be recovered during the crash with a probability determined solely by N_D^*.

If we assume centralized scheduling between the replicas, then it is possible to run the system optimally (i.e., using a perfectly spaced out round-robin) and thus keep the overall penalty exactly the same as with a single replica. Under fully decentralized (i.e., independent) replica operation and $m \to \infty$, each rate $\lambda/m \to 0$ and thus N_D^* likely converges in distribution to a Poisson process with rate λ (Palm-Khintchine theorem [46]). This creates a problem, however, because exponential D requires noticeably more overhead than constant D to achieve the same staleness penalty. For example, using our model for $\bar{\rho}$ and discussion after (2.54), this difference is by a factor of 2 for $w(x) = 1$ and by a factor of 6 for $w(x) = x$, which shows that a distributed cluster of replicas may need to consume $100 - 500\%$ more bandwidth than a centralized solution for a given level of QoS (quality-of-service).

2.6.5 Many-to-Many

We conclude the paper by noting that Internet applications often combine the last two scenarios, i.e., $M \times 1$ and $1 \times m$ replication, into a single framework. However, these problems are usually separable into subproblems that can be reduced to the analysis above. For example, suppose we are interested in the probability that a query to a random subset of j replicas finds at least one of the k sources fresh. First, we compute the staleness probability for each source based on the aggregate synchronization process N_D^* from j replicas. Second, since each source is independent, we multiply these probabilities to deduce the likelihood that all k sources are stale. Taking the complement of the result, we get the desired probability.

3. TEMPORAL UPDATE DYNAMICS UNDER BLIND SAMPLING

3.1 Introduction

Many distributed systems in the current Internet manipulate objects that experience periodic modification in response to user actions, real-time events, data-centric computation, or some combination thereof. In these cases, each source (e.g., a webpage, DNS record, stock price) can be viewed as a stochastic process N_U that undergoes *updates* (i.e., certain tangible changes) after random delays $U_1, U_2, \ldots$

Consistent estimation of inter-update distribution $F_U(x)$ is an important problem, whose solution yields not only better caching, replication [56], and allocation of download budgets [55], but also more accurate modeling and characterization of complex Internet systems [15, 17, 21, 23, 29, 53, 66, 69, 70, 71, 84, 90, 101, 106]. Similar issues arise in lifetime measurement, where U_i represents the duration of online presence for object or user i [9, 79, 92, 99].

The first challenge with measuring update-interval dynamics is to infer their distribution using *blind* sampling, where variables $U_1, U_2, \ldots$ are hidden from the observer. This scenario arises when the source can only be queried over the network using some process N_S whose inter-download delays $S_1, S_2, \ldots$ are bounded in expectation from below (e.g., due to bandwidth and/or CPU restrictions). Unlike censored observations in statistics, which have access to truncated values of *each* U_i, the sampling process here has a tendency to miss entire update cycles and land in larger-than-average intervals.

The second challenge in blind sampling is to reconstruct the distribution of U_i from severely limited amounts of information available from each download. Specifically, the observer can only compare the two most-recent copies of the source and

obtain indicator variables Q_{ij} of a change occurring between downloads i and j, for all $i < j$. This constraint is necessary because the source generally has no ability to determine object modification timestamps (e.g., dynamic webpages served by scripts are considered new on each download). Furthermore, even for static pages, object updates are very application-specific (e.g., search engines may remove ad banners and other superfluous information before indexing), which makes variables $U_1, U_2, \ldots$ *hidden not just from the observer, but also the source.*

Existing studies on this topic [7, 17, 43, 44, 61] use Poisson N_U and constant S_i. Due to the memoryless assumption on $F_U(x)$, the problem reduces to estimating just rate $\mu = 1/E[U_i]$, rather than an entire distribution, and many complex interactions between N_S and N_U are avoided in the analysis. However, more interesting cases arise in practice, where non-Poisson updates are quite common [7, 18, 45, 63]. Furthermore, guaranteeing constant S_i is impossible in certain applications where the return delay to the same object is computed in real-time and is governed by the properties of trillions of other sources (e.g., in search engines). Thus, new analytical techniques are required to handle such cases.

3.1.1 Contributions

Our first contribution is to formalize blind update sampling using a framework in which both N_U and N_S are general point processes. We then consider a simplified problem where the source provides last-modification timestamps for each download. Our contribution here is to develop the necessary tools for tackling the more interesting cases that follow, build general intuition, consider conditions under which provably consistent estimation is possible, and explain the pitfalls of existing methods under non-Poisson updates.

Armed with these results, we next relax the availability of last-modified times-

tamps at the source. For situations where constant S_i is acceptable, we show that unbiased estimators developed earlier in the paper can be easily adapted to this environment and then suggest avenues for reducing the amount of numerical computation in the model, all of which forms our third contribution. We finish the paper by considering random S_i and arrive at our last contribution, which is a novel method that can accurately reconstruct the update distribution under arbitrary $F_U(x)$ and mildly constrained $F_S(x)$.

Our last contribution is to evaluate the Poisson update assumption and compare our methods on the Wikipedia dataset. We show that Poisson updates assumption hardly holds because it requires not only the inter-update delays to be exponentially distributed but also independency between delays. Then we compare the accuracy of our methods use the top 10 most frequently modified articles in our dataset, from which we conclude the optimal methods to use in different scenarios.

3.2 Related Work

Analytical studies on estimating the update distribution under blind sampling have all assumed N_U was Poisson and focused on determining its average rate, i.e., μ for stationary cases [7, 17, 43, 44, 61] and $\mu(t)$ for non-stationary [89]. Extension to general processes was achieved by [61] under the assumption that sampling intervals S_i were infinitely small; however, the problem in these scenarios is trivial since every U_i is available to the observer with perfect accuracy.

In measurement literature, the majority of effort was spent on the behavior of web pages, including analysis of server logs [67], page-modification frequency during crawling [7, 14, 45, 63], RSS feed dynamics [84], and content change between consecutive observations [1, 36, 66]. Problems related to estimation of $F_U(x)$ have also emerged in prediction of future updates [15, 16, 35, 47, 73, 95], with a good survey

in [64], and user lifetime measurement in decentralized P2P networks [9, 79, 92, 99].

3.3 Overview

This section introduces notation, formulates objectives, and lays down a roadmap of the studied methods.

3.3.1 Notation and Assumptions

Let u_i be the time of the i-th update at the source. Define $N_U(t) = \max\{i : u_i \leq t\}$ to be the number of updates in the time interval $[0, t]$ and suppose $U_i = u_{i+1} - u_i$ represents the inter-update delay. Similarly, denote by s_j the j-th sampling time. Let $N_S(t) = \max\{j : s_j \leq t\}$ be the number of samples in $[0, t]$ and $S_j = s_{j+1} - s_j$ be the inter-sample delay. At time t, define *age* $A_U(t) = t - u_{N_U(t)}$ and *residual* $R_U(t) = u_{N_U(t)+1} - t$ as the backward/forward delays to the nearest update. These are illustrated in Fig. 3.1. Note that interval U_i in the figure cannot be seen or measured by the observer, which is why we called it "hidden" earlier.

Since most real applications have only one sample path, we adopt the sample path approach in [55] to model both processes, which needs the following assumption.

Assumption 1. *Both N_U and N_D are age-measurable.*

Age measurable assumption guarantees the existence and convergence of the inter-update delay distribution. This allows us to define random variables $U \sim F_U(x)$ and $S \sim F_S(x)$ to represent the lengths of update/sample cycles, respectively [55]. Furthermore, denote by $\mu = 1/E[U]$ and $\lambda = 1/E[S]$ the corresponding rates.

Suppose A_U and R_U are the equilibrium versions of $A_U(t)$ and $R_U(t)$, respectively, as $t \to \infty$. From previous results in [55], they have the same CDF:

$$G_U(x) := \mu \int_0^x (1 - F_U(y))dy, \tag{3.1}$$

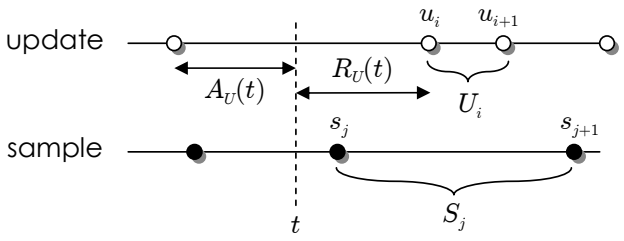


Figure 3.1: Update/sample process notation.

whose density is $g_U(x) := G'_U(x) = \mu(1 - F_U(x))$. We set the goal of the sampling process to determine the distribution $F_U(x)$ based on observations at times $s_1, s_2, \ldots$, i.e., using a single realization of the system.

3.3.2 Applications

Knowledge of $F_U(x)$ enables performance analysis in many fields that employ lazy (i.e., pull-based) data replication. For example, search engines implement a sampling process N_S using crawlers that periodically revisit web content and merge updates into backend databases. These companies are often concerned with staleness of pages in their index and the probability that users encounter outdated results. In order to determine the download frequency needed to maintain staleness below a certain threshold, the expected number of updates by which the index is trailing the source, or the amount of bandwidth needed for a collection of pages, accurate knowledge of source dynamics is required [55].

In another example, suppose a data center replicates a quickly changing database (driven by some update process N_U) among multiple nodes for scalability and fault-tolerance reasons. Because of the highly dynamic nature of the source, individual replicas may not stay fresh for long periods of time, but their collection may offer much better performance as a whole. In such cases, questions arise about the number of replicas k that should be queried by clients to obtain results consistent with the

source [56] and/or the probability that a cluster of n replicas can recover the most-recent copy of the source when it crashes [55]. Similar problems appear in multi-hop replication and cooperative caching, where service capacity of the caching network is studied as well [56].

Finally, accurate measurement of $F_U(x)$ enables better characterization of Internet systems, their update patterns in response to external traffic, and even user behavior. While it is possible to use the exponential distribution to approximate any $F_U(x)$, as typically done in the literature [7, 17, 43, 44, 61], this can lead to significant errors in the analysis. As shown in [55] using the search-engine example and Wikipedia's update process N_U, the exponential assumption may produce errors in the download bandwidth that are two orders of magnitude. In more complicated settings, such as cascaded and cooperative systems [56], the impact of inaccurate $F_U(x)$ may be even higher.

3.3.3 Caveats

The sample-path approach, in general, leads to a possibility of *phase-lock* where the distance of download points from the last update, i.e., $\{A_U(s_j)\}_{j\geq 1}$, is not a mixing process. For example, consider $U_i = 1$ for $i \geq 1$ and $S_j = 2$ for $j \geq 1$, in which case update ages observed at $\{s_j\}_{j=1}^{\infty}$ are all equal to zero. Since this case cannot be distinguished from $U_i = 0.5$ or $U_i = 2$, it is easy to see how phase-lock precludes consistent estimation of $F_U(x)$. The problem can be avoided by requiring that the considered cycle lengths exhibit certain mixing properties. This leads to our next definition.

Definition 8. *A random variable X is called* lattice *if there exists a constant c such that X/c is always an integer, i.e., $\sum_{i=1}^{\infty} P(X/c = i) = 1$.*

Lattice distributions are undesirable in our context as they produce phase-lock.

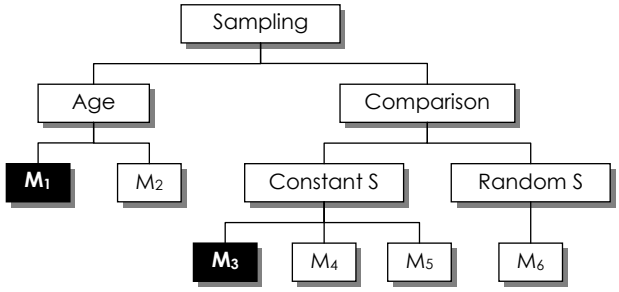


Figure 3.2: Method taxonomy (shaded boxes indicate Poisson-only techniques).

Before introducing the condition to avoid phase-lock, we need the following definition.

Definition 9. *A random process is called* age-mixing *if it is renewal and its inter-delay distribution $F(x)$ is non-lattice.*

Now we are ready to present our conditions to prevent phase-lock [55].

Assumption 2. *At least one of U and S is age-mixing.*

This condition is easy to satisfy with any continuous random variable, including exponential U in previous work. A more esoteric example would be a discrete variable placing mass on two numbers whose ratio is irrational, e.g., $(\pi, 3)$ or $(e, \sqrt{2})$.

3.3.4 Roadmap

As illustrated in Fig. 3.2, we partition the various approaches into two broad categories. In *age* sampling, the observer has access to the last-modified timestamp $u_{N_U(s_j)}$ at each download point s_j, or equivalently, the update age $A_U(s_j)$. Although now rare, this information can still be sometimes obtained from the HTTP headers, timestamps within the downloaded HTML, or sitemaps [64]. As shown in the figure, we call the two studied methods in this category M_1 and M_2. They operate by deriving $F_U(x)$ from the collected age samples, where M_1 has been proposed in previous work [17, 61] for Poisson-only cases and M_2 is novel.

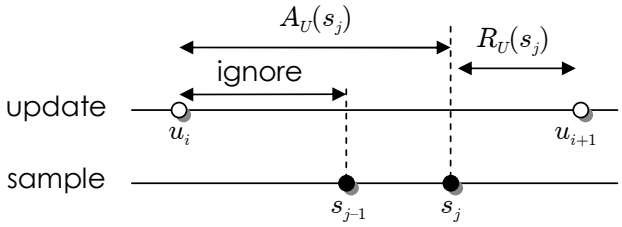


Figure 3.3: Illustration of M_1.

In *comparison* sampling, we assume that the observer retains the most recent copy of the object or a fingerprint of its relevant portions (e.g., after removing ads and repeated keywords). Define Q_{ij} to be an update-indicator process:

$$Q_{ij} = \begin{cases} 1 & \text{update occurs between } s_i \text{ and } s_j \\ 0 & \text{otherwise} \end{cases}. \tag{3.2}$$

Unlike the previous scenario, estimation of $F_U(x)$ here must use only binary values $\{Q_{ij}\}$. Going back to Fig. 3.2, we study comparison sampling under two strategies. For *constant* S, we first analyze two methods we call M_3 and M_4, which are discrete versions of M_1 and M_2, respectively. We then propose a novel method M_5 that is both consistent and computationally efficient. For *random* S, we introduce our final approach M_6 that is unbiased under the most general conditions.

3.4 Age Sampling

This section is a prerequisite for the results that follow. It starts with understanding state of the art in this field and its pitfalls. It then shows that a simple modification allows prior work to become unbiased under non-Poisson updates.

3.4.1 Basics

In age sampling, the observer has a rich amount of information about the update cycles. This allows reconstruction of $F_U(x)$ in all points $x \geq 0$, which we set as our goal.

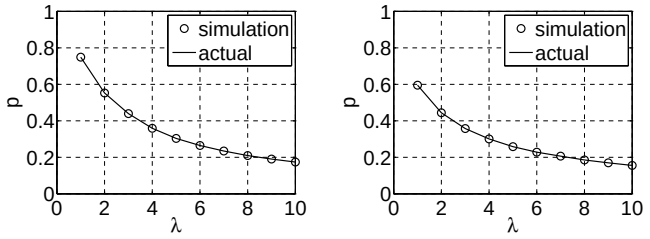
Definition 10. *Suppose $\tilde{F}(x,T)$ is a CDF estimator that uses observations in $[0,T]$. Then, we call it* consistent *with respect to distribution $F(x)$ if it converges in probability to $F(x)$ as the sampling window becomes large:*

$$\lim_{T\to\infty} \tilde{F}(x,T) = F(x), \quad x \geq 0. \tag{3.3}$$

Note that consistent estimation of $F_U(x)$ is equivalent to that of $G_U(x)$ since there is a one-to-one mapping (3.1) between the two functions. Specifically, knowledge of $G_U(x)$ allows numerical differentiation and/or kernel density estimators to obtain $g_U(x) = G'_U(x)$, from which $F_U(x) = 1 - g_U(x)/g_U(0)$ follows. Furthermore, the update rate $\mu = 1/E[U]$ is also readily available as $g_U(0)$. Under Poisson N_U, the memoryless property ensures that $F_U(x) = G_U(x)$; however, in more general cases, this distinction is important.

3.4.2 Modeling M_1

To estimate the mean μ of a Poisson update process, prior studies [17, 61] proposed that only a subset of age samples $\{A_U(s_j)\}_{j\geq 1}$ be retained by the observer. Specifically, when multiple sample points land in the same update interval, only the one with the largest age is kept, while the others are discarded. As shown in Fig. 3.3, points s_{j-1} and s_j hit the same update cycle $[u_{i-1}, u_i]$, in which case only $A_U(s_j)$ is used in the measurement and $A_U(s_{j-1})$ is ignored. It was perceived in [17, 61] that doing otherwise would create a bias and lead to incorrect estimation, but no



(a) constant S (b) exponential S

Figure 3.4: Verification of (3.5) under Pareto U ($\mu = 2$).

proof was offered. Compared to the previous studies [17, 61] which mainly focus on constant S, we consider S as a general random variable. We call this method M_1 and study its performance next.

From Fig. 3.3, notice that M_1 collects ages $A_U(s_j)$ at such points s_j that satisfy $R_U(s_j) < S_j$, or equivalently $Q_{j,j+1} = 1$. All other age measurements are ignored. Defining $\mathbf{1}_A$ as the indicator variable of event A, the fraction of age samples retained by M_1 in $[0, T]$ is given by:

$$p(T) := \frac{1}{N_S(T)} \sum_{j=1}^{N_S(T)} \mathbf{1}_{R_U(s_j)<S_j}, \tag{3.4}$$

which is an important metric that determines the overhead of M_1 and its bias later in the section. Expansion of (3.4) in the next result follows from Assumption 2, the equilibrium residual equation for non-lattice intervals, and the law of large numbers [103].

Theorem 13. *As $T \to \infty$, $p(T)$ converges in probability to:*

$$p := \lim_{T\to\infty} p(T) = P(R_U < S) = E[G_U(S)]. \tag{3.5}$$

This result shows that p is affected not just by the update distribution $F_U(x)$, but

also the sample distribution $F_S(x)$. To see this effect in simulations, we use constant and exponential S to sample Pareto $F_U(x) = 1 - (1 + x/\beta)^{-\alpha}$, where $\alpha = 3$ and $\beta = 1$ throughout the paper. Fig. 3.4 confirms a good match between the model and simulations. As expected, p decreases as the sampling rate $\lambda = 1/E[S]$ increases, which is caused by an increased density of points landing within each update interval and thus a higher discard rate. The figure also shows that constant S samples more points than the exponential case. In fact, it is possible to prove a more general result – constant S exhibits the largest p (i.e., highest overhead) for a given λ.

Let $K(x,T)$ be the number of samples that M_1 obtains in $[0,T]$ with values no larger than x:

$$K(x,T) := \sum_{j=1}^{N_S(T)} \mathbf{1}_{R_U(s_j)<S_j} \mathbf{1}_{A_U(s_j)\leq x}. \tag{3.6}$$

Then, it produces a distribution in $[0,T]$ given by:

$$G_1(x,T) := \frac{K(x,T)}{K(\infty,T)}. \tag{3.7}$$

Theorem 14. *Denoting by $\bar{F}(x) = 1 - F(x)$ the complement of function $F(x)$ and letting $T \to \infty$, the tail distribution of the samples collected by M_1 converges in probability to:*

$$\bar{G}_1(x) := \lim_{T\to\infty} \bar{G}_1(x,T) = \frac{E[G_U(x+S) - G_U(x)]}{E[G_U(S)]}. \tag{3.8}$$

Proof. Under Assumption 2 and $T \to \infty$, $A_U(s_j)$ and $R_U(s_j)$ converge to their

equilibrium versions A_U and R_U, respectively. Therefore:

$$\lim_{T\to\infty} \frac{K(x,T)}{N_S(T)} = P(A_U \le x, R_U < S). \tag{3.9}$$

From Theorem 13, we know that:

$$\lim_{T\to\infty} \frac{K(\infty,T)}{N_S(T)} = p = E[G_U(S)]. \tag{3.10}$$

Dividing (3.9) by (3.10) yields:

$$G_1(x) = \lim_{T\to\infty} G_1(x,T) = \frac{P(A_U \le x, R_U < S)}{E[G_U(S)]}, \tag{3.11}$$

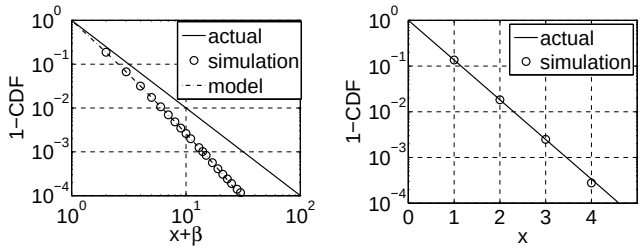
where $E[G_U(S)] > 0$ is guaranteed for all cases except S being zero with probability 1. To derive the numerator of (3.11), condition on R_U and S:

$$\begin{aligned} &P(A_U \le x, R_U < S) \\ &= \int_0^\infty \Big[\int_0^z P(A_U \le x | R_U = y) g_U(y) dy\Big] dF_S(z), \end{aligned} \tag{3.12}$$

Expanding the probability of event $A_U \le x$ given a fixed residual $R_U = y$ leads to:

$$\begin{aligned} P(A_U \le x | R_U = y) &= \frac{P(y < U \le x + y)}{P(U > y)} \\ &= \frac{F_U(x+y) - F_U(y)}{1 - F_U(y)}. \end{aligned} \tag{3.13}$$

Recalling that $g_U(y) = \mu(1 - F_U(y))$ is the residual density and applying (3.13),



(a) Pareto U (b) exponential U

Figure 3.5: Simulation results of M_1 under exponential S ($\lambda = 1, \mu = 2$).

the inside integral of (3.12) becomes:

$$\begin{aligned}
&\int_0^z P(A_U \leq x | R_U = y) g_U(y) dy \\
&= \mu \int_0^z (F_U(x+y) - F_U(y)) dy \\
&= \mu \int_0^z \bar{F}_U(y) dy - \mu \int_x^{z+x} \bar{F}_U(w) dw \\
&= G_U(z) + G_U(x) - G_U(x+z). \qquad (3.14)
\end{aligned}$$

This transforms (3.11) to:

$$\begin{aligned}
G_1(x) &= \frac{\int_0^\infty \left(G_U(z) + G_U(x) - G_U(x+z)\right) dF_S(z)}{E[G_U(S)]} \\
&= \frac{E[G_U(S) - G_U(x+S)] + G_U(x)}{E[G_U(S)]}, \qquad (3.15)
\end{aligned}$$

which is the complement of the tail in (3.8). □

Observe from (3.8) that M_1 generally measures neither the update distribution $F_U(x)$ nor the age distribution $G_U(x)$. To see the extent of this bias, Fig. 3.5(a) plots simulation results for exponential S and Pareto U in comparison to (3.8). Observe in the figure that our model closely tracks the simulated tail $\bar{G}_1(x)$, which remains

heavy-tailed, albeit different from that of the target distribution $F_U(x)$. Fig. 3.5(b) shows that M_1 is indeed unbiased for exponential $F_U(x)$. We next investigate other conditions under which this approach may work well.

3.4.3 Quantifying Bias in M_1

Suppose $D_1 \sim G_1(x)$ is the random variable observed by M_1 over an infinitely long measurement period. Our goal in this subsection is to determine the relationship between D_1, U, and A_U under different sampling strategies and update distributions. We first re-write (3.8) in a more convenient form.

Theorem 15. *The tail distribution measured by M_1 can be expressed in two alternative forms:*

$$\bar{G}_1(x) = \bar{G}_U(x)\frac{P(A_U < x + S | A_U > x)}{P(A_U < S)} \tag{3.16}$$

$$= \bar{F}_U(x)\frac{E[\int_0^S P(U > x + y | U > x)dy]}{E[\int_0^S P(U > y)dy]}. \tag{3.17}$$

Proof. We first show (3.16). Recalling that $G_U(x) = P(A_U < x)$ yields:

$$\bar{G}_1(x) = \frac{P(A_U < x + S) - P(A_U < x)}{P(A_U < S)}$$

$$= \bar{G}_U(x)\frac{P(x < A_U < x + S)}{P(A_U < S)P(A_U > x)}. \tag{3.18}$$

From the definition of conditional probability, we get:

$$\frac{P(x < A_U < x + S)}{P(A_U > x)} = P(A_U < x + S | A_U > x). \tag{3.19}$$

Substituting (3.19) into (3.18), we get (3.16).

To establish (3.17), rewrite (3.18) as:

$$\bar{G}_1(x) = \bar{F}_U(x)\frac{P(x < A_U < x + S)}{P(A_U < S)P(U > x)}, \tag{3.20}$$

whose numerator can be transformed to:

$$\begin{aligned} P(x < A_U < x + S) &= \mu E\Big[\int_x^{x+S} \bar{F}_U(y)dy\Big] \\ &= \mu E\Big[\int_0^{S} \bar{F}_U(x + y)dy\Big], \end{aligned} \tag{3.21}$$

where we use the fact that $g_U(x) = \mu\bar{F}(x)$. Dividing (3.21) by $\bar{F}_U(x)$ produces:

$$\frac{P(x < A_U < x + S)}{P(U > x)} = \mu E\Big[\int_0^{S} P(U > x + y|U > x)dy\Big].$$

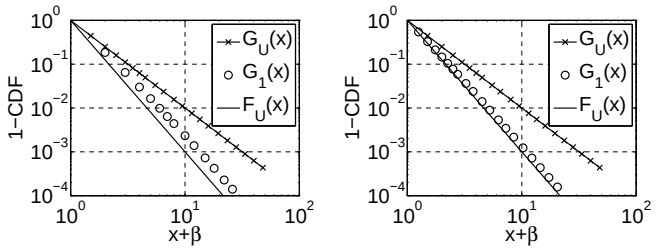
Similarly, we can expand:

$$P(A_U < S) = \mu E\Big[\int_0^{S} P(U > x)dy\Big]. \tag{3.22}$$

Substituting the last two equations into (3.20), we obtain the desired result in (3.17). □

Theorem 15 suggests that the tail of D_1 may indeed have some relationship to those of A_U and U. In order to establish this formally, we need to define three classes of variables.

Definition 11. *Variable X is said to be NWU (new worse than used) if $P(X > x + y|X > y) > P(X > x)$ for all $x, y \geq 0$. If this inequality is reversed, X is said to be NBU (new better than used). Finally, if $P(X > x + y|X > y) = P(X > x)$ for all $x, y \geq 0$, the variable is called* memoryless.



(a) $\lambda = 1$ (b) $\lambda = 4$

Figure 3.6: Tail sandwich of M_1 under Pareto updates and constant S ($\mu = 2$).

Note that NWU distributions are usually heavy-tailed, with two common representatives being Pareto and Weibull. Conditioning on U's survival to some age y, its residual length $U - y$ is stochastically larger than U itself. NBU are typically light-tailed distributions, exemplified by uniform and constant. Finally, the memoryless class consists of only exponential distributions, where past knowledge has no effect on the future.

When both U and A_U are NWU, as is the case with Pareto distributions, Theorem 15 shows that $\bar{G}_1(x)$ is "sandwiched" between the other two tails, i.e., $\bar{F}_U(x)$ serves as a lower bound and $\bar{G}_U(x)$ as an upper. This means that D_1 is stochastically smaller than A_U, but stochastically larger than U. Fig. 3.6 shows an example confirming this, where the faster sampling rate in (b) moves the curve closer to $\bar{F}_U(x)$. The relationship among the tails is reversed if U and A_U are NBU. For exponential update distributions, all three tails are equal, which leads to the following result:

Corollary 1. *Exponential is the only update distribution that allow M_1 to be consistent with $F_U(x)$ for all S.*

We examine a few other cases next.

3.4.4 Achieving Consistency in M_1

Now that we know that $\bar{G}_1(x)$ is contained between the tails of update and age distributions, there are two intuitive ways how bias can be removed. First, we could tighten the distance between tails $\bar{F}_U(x)$ and $\bar{G}_U(x)$; however, this can only be achieved by forcing the source to undergo updates with U that is "closer" to exponential. As this is usually impractical, the second technique is to adjust the sampling distribution $F_S(x)$ such that the distance of $\bar{G}_1(x)$ to one of U's tails shrinks to zero. To this end, our next result demonstrates that D_1 "leans" towards U or A_U solely based on the fraction of retained samples p.

Theorem 16. *For $p \to 1$, variable D_1 sampled by M_1 converges in distribution to A_U. For $p \to 0$ and mild conditions on S, variable D_1 converges in distribution to U.*

Proof. Recall that $p = E[G_U(S)]$. When $E[G_U(S)] \to 1$, so does $E[G_U(S+x)]$. Therefore:

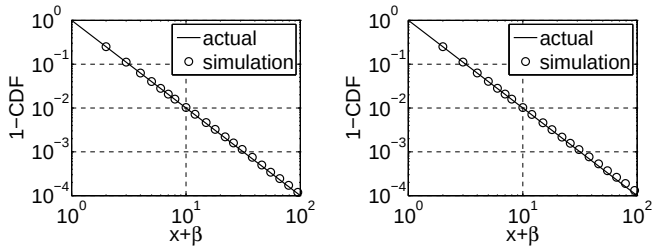
$$\bar{G}_1(x) = \frac{E[G_U(S+x)] - G_U(x)}{E[G_U(S)]} \to \bar{G}_U(x). \tag{3.23}$$

To prove the second part, assume that $S/E[S]$ converges to a random variable with mean 1. Since $p \to 0$ implies that $S \to 0$ almost surely, we get:

$$\frac{G_U(S)}{E[S]} = \frac{\int_0^S \bar{F}_U(y)dy}{E[U]E[S]} = \frac{S\int_0^1 \bar{F}_U(Sy)dy}{E[U]E[S]} \to \mu, \tag{3.24}$$

where we use the fact that $\bar{F}_U(Sy) \to 1$ for all fixed y.

Noticing that $G_U(S)/E[S]$ is upper bounded by random variable $\mu S/E[S]$, the latter of which has a finite mean, and applying the dominated convergence theorem



(a) exponential S (b) constant S

Figure 3.7: Verification of (3.30) under Pareto updates and $\lambda = 1$.

(DCT), we get:

$$\lim_{p\to 0} \frac{E[G_U(S)]}{E[S]} = \mu. \tag{3.25}$$

Similarly, we obtain:

$$\begin{aligned} \frac{G_U(S+x) - G_U(x)}{E[S]} &= \frac{\int_x^{S+x} \bar{F}_U(y)dy}{E[U]E[S]} \\ &= \frac{S\int_0^1 \bar{F}_U(Sy+x)dy}{E[U]E[S]}, \end{aligned} \tag{3.26}$$

which converges to $\mu\bar{F}_U(x)$. Applying the DCT again, we get:

$$\lim_{p\to 0} \frac{E[G_U(S+x) - G_U(x)]}{E[S]} = \mu\bar{F}_U(x). \tag{3.27}$$

Combining (3.25) and (3.27) produces:

$$\lim_{p\to 0} \frac{E[G_U(S+x) - G_U(x)]}{E[G_U(S)]} = \bar{F}_U(x), \tag{3.28}$$

which is what we intended to prove. □

To understand this result, we discuss several examples. In order to converge p to

1, method M_1 has to sample with sufficiently large S to achieve $P(S > R_U) = 1$. For general $F_U(x)$, this can be guaranteed only if S converges to infinity, in which case the measurement process will be quite slow. If an upper bound on U is known, then setting S to be always larger can also produce $p = 1$. In these scenarios, however, M_1 will sample $G_U(x)$ and additional steps to recover $F_U(x)$ must be undertaken.

To achieve $p = 0$, M_1 has to use high sampling rates such that each update interval contains an infinite number of samples, i.e., S must converge to zero. In this case, the method may consume exorbitant network resources and additionally create undesirable load conditions at the source.

3.4.5 Method M_2

Instead of using the largest age sample for each detected update, a more sound option is to use *all* available ages. While extremely simple, this method has not been proposed before. We call this strategy M_2 and define $G_2(x, T)$ to be the fraction of its samples with values smaller than or equal to x in $[0, T]$:

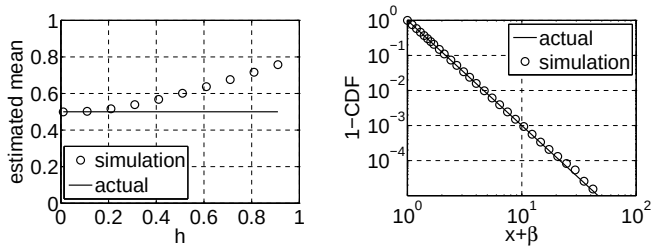
$$G_2(x, T) := \frac{1}{N_S(T)} \sum_{j=1}^{N_S(T)} \mathbf{1}_{A_U(s_j) \leq x}. \tag{3.29}$$

The next result follows from Assumption 2 and the equilibrium residual equation [55].

Theorem 17. *Method M_2 is consistent with respect to the age distribution:*

$$G_2(x) := \lim_{T \to \infty} G_2(x, T) = G_U(x). \tag{3.30}$$

Next we use simulations to verify the usefulness of (3.30). From Fig. 3.7, observe that the sampled distribution of M_2 does in fact equal $G_U(x)$. To obtain



(a) $E[U]$ (b) $F_U(x)$ $(h = 0.1)$

Figure 3.8: Performance of M_2 under Pareto U and constant S $(\mu = 2, \lambda = 100)$.

$F_U(x) = 1 - g_U(x)/g_U(0)$ from an empirical CDF $G_U(x)$, we adopt numerical differentiation from [98]. This method uses bins of size h and k-point derivatives, bounding Taylor-expansion errors to $O(h^k/k!)$. For the estimator to work, it must first accurately determine $g_U(0) = 1/E[U]$. Using $k = 5$ and non-symmetric (i.e., one-sided) derivatives around $x = 0$, Fig. 3.8(a) demonstrates that the estimated $E[U]$ monotonically decreases in h and eventually stabilizes at the true value. Since h is a user-defined parameter independent of (N_U, N_S), it can be arbitrarily small. Thus, a binary search on h to find the flat region in $E[U]$ can always determine its value with high accuracy. Applying this technique, the update distribution estimated by M_2 is shown in Fig. 3.8(b) in comparison to $F_U(x)$. Notice that the two curves are indistinguishable.

3.4.6 Discussion

Although M_1 has fewer samples, its network traffic remains the same as that of M_2, because they both have to contact the source $N_S(t)$ times in $[0, t]$. However, the smaller number of *retained* values in M_1 may lead to lower computational cost and better RAM usage in density-estimation techniques that utilize all available samples (e.g., kernel estimators). For the route we have taken, i.e., differentiation of $G_2(x)$, the two methods exhibit the same overhead.

We now focus on the performance of M_2 in finite observation windows $[0, T]$. One potential issue is the redundancy (and high dependency) of samples that it collects (i.e., all ages within the same update interval are deterministically predictable), which is what M_1 tried to avoid. While necessary, can this redundancy lead to slower convergence? For a given T, would it be better to collect fewer samples that are spaced further apart?

Define

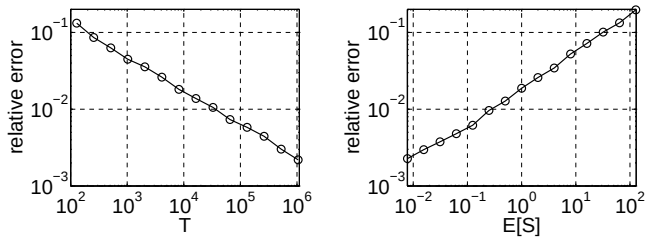
$$\zeta(T) := \frac{1}{N_S(T)} \sum_{j=1}^{N_S(T)} A_U(s_j) \tag{3.31}$$

to be the average age observed by M_2 in $[0, T]$ using one realization of the system. We now use deviation of $\zeta(T)$ from $E[A_U] = \mu E[U^2]/2$ as indication of error. Specifically, let

$$\epsilon(T) := E\Big[|1 - \frac{\zeta(T)}{E[A_U]}|\Big]. \tag{3.32}$$

be the expected relative error computed over m sample-paths.

First, we fix the sampling rate $\lambda = 1$ and change T from 100 to 10M time units. As expected, $\epsilon(T)$ in Fig. 3.9(a) monotonically decreases as the observation window gets larger, confirming asymptotic convergence of M_2 discussed throughout this section. Next, we keep T constant at 10K and vary $E[S]$. As shown in Fig. 3.9(b), the error drops with $E[S]$, but then stabilizes. This means that having more samples, regardless of how redundant, improves performance only up to a certain point.



(a) impact of T ($\lambda = 1$) (b) impact of S ($T = 10$K)

Figure 3.9: Average relative error of $\zeta(T)$ of M_2 under Pareto U and exponential S ($\mu = 2, m = 1000$).

3.5 Comparison Sampling: Constant Intervals

In contrast to the previous section, the remaining methods do not have access to age; instead, they must work with binary observations Q_{ij}, which indicate whether an update occurred between two sampling points s_i and s_j. This section deals with constant inter-download delay. This special case of comparison sampling is not just simple to implement and the only one considered in the literature, but also maximally polite (i.e., least bursty) for a given download rate λ.

3.5.1 Basics

Assume constant inter-sample delays $S = \Delta$ and notice that all observations related to update intervals must be multiples of Δ. It is therefore impossible to reconstruct $F_U(x)$, or even $G_U(x)$, in every point x. This requires an adjustment in objectives.

Definition 12. *An estimator $\tilde{F}(x, T)$ is* Δ-consistent *with respect to distribution $F(x)$ if it can correctly reproduce it in all discrete points $x_n = n\Delta$ as $T \to \infty$:*

$$\lim_{T\to\infty} \tilde{F}(x_n, T) = F(x_n), \quad n = 1, 2, \ldots \tag{3.33}$$

As we discuss above and confirm below, none of the methods can measure $F_U(x)$ directly (unless the sampling rate is infinite or U is exponential). As a result, all algorithms generally face the issue of recovering $F_U(x)$ from $G_U(x)$. The main caveat of this section is that knowledge of the age distribution in discrete points is generally insufficient for Δ-consistent estimation of $F_U(x)$. This occurs because the estimated $G_U(x)$ lacks data in every interval (x_n, x_{n+1}), which precludes differentiation and leaves $g_U(x)$ unobtainable.

Depending on the smoothness of $G_U(x)$ and/or prior knowledge about the target distribution, one can use interpolation between the known points $G_U(x_n)$. In such cases, $F_U(x)$ may be reconstructed with high accuracy using kernel density-estimation techniques; however, the result is application-specific. We thus do not dwell on numerical methods needed to perform these manipulations and instead focus on Δ-consistency in regard to $G_U(x)$.

3.5.2 Method M_3

Prior work in several fields [9, 17, 61, 79, 92] has suggested an estimator, which we call M_3, that rounds the distance between each adjacent pair of detected updates to the nearest multiple of Δ, from which it builds a distribution $G_3(x)$. This technique was used in [17, 61] to track webpage updates, in [76] to estimate lifetimes of storage objects, and in [9, 79, 92] to sample user lifetimes in P2P networks. In the OS/networking literature, the approach is known as *Create-Based Method* (CBM) because it tracks each object from its creation, as opposed to other methods that track deletions.

Define r_k to be the number of downloads after which the k-th update is detected,

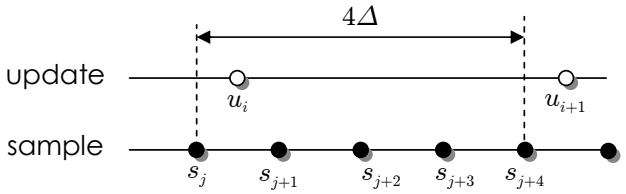


Figure 3.10: Comparison sampling in M_3 with constant intervals of size Δ.

i.e.,

$$r_k := \min\Big\{m \geq 1 : \sum_{j=1}^{m} Q_{j,j+1} = k\Big\}. \tag{3.34}$$

Then, the samples collected by M_3 are $(r_{k+1} - r_k)\Delta$ for $k = 1, 2, \ldots$ To understand this better, Fig. 3.10 shows an example where updates are detected after downloads j and $j + 4$, which produces $r_{k+1} - r_k = 4$ and a single sample 4Δ. Based on the description in prior work, this technique serves the purpose of directly measuring U_i by counting full intervals of size Δ that fit in $[u_i, u_{i+1}]$. As a result, the output of M_3 is usually expected to produce the update distribution $F_U(x)$.

While this makes sense for the case in Fig. 3.10, the method becomes grossly inaccurate when multiple updates occur within Δ time units of each other, which brings us back to the issue of hidden variables U_i. Consider Fig. 3.11, where 2/3 of the visible update durations are less than Δ. Since M_3 in this scenario produces one sample 4Δ, it skews the mass of the distribution to much higher values than needed.

We now model the performance of M_3 under general U and obtain the limiting distribution of its samples. Define $G_3(x, T)$ to be the CDF of observed durations in $[0, T]$:

$$G_3(x, T) := \frac{\sum_{k=1}^{\infty} \mathbf{1}_{r_k \leq T} \mathbf{1}_{(r_{k+1} - r_k)\Delta \leq x}}{\sum_{k=1}^{\infty} \mathbf{1}_{r_k \leq T}}. \tag{3.35}$$

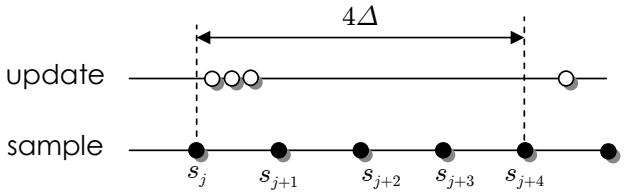


Figure 3.11: Pitfalls of M_3.

Then, we have the following result.

Theorem 18. *The tail distribution of M_3 is a step-function:*

$$\bar{G}_3(x_n) := \lim_{T\to\infty} \bar{G}_3(x_n, T) = \frac{G_U(x_{n+1}) - G_U(x_n)}{G_U(\Delta)}. \tag{3.36}$$

Proof. Notice from Fig. 3.11 that age samples collected by M_3 can be viewed as discrete versions of those in M_1. Indeed, define $x^+ = \Delta\lceil x/\Delta\rceil$ to be x rounded-up to the nearest multiple of Δ. Then, the sample obtained by M_3 at download instance s_j is $A_U^+(s_j)$. Since condition $A_U^+(s_j) < x_n$ is equivalent to $A_U(s_j) < x_n$ for $x_n = n\Delta$, we obtain:

$$G_3(x_n, T) = \frac{\sum_{j=1}^{N_S(T)} \mathbf{1}_{R_U(s_j)<S_j} \mathbf{1}_{A_U(s_j)\le x_n}}{\sum_{j=1}^{N_S(T)} \mathbf{1}_{R_U(s_j)<S_j}}, \tag{3.37}$$

which is exactly the same as $G_1(x_n, T)$ in (3.7). Therefore, the tail of $G_3(x_n, T)$ converges to the result in (3.8), with S replaced by Δ. Doing so produces (3.36). Since $G_3(x)$ has no information between discrete points x_n, it must be constant in each interval $[x_n, x_{n+1})$, which means it is a step-function. □

Define a random variable $D_3 \sim G_3(x)$. With the result above, its average becomes readily available.

Theorem 19. *The expectation of D_3 is given by:*

$$E[D_3] = \frac{\Delta}{G_U(\Delta)}. \tag{3.38}$$

Proof. It is well-known that the mean of a non-negative lattice random variable can be obtained by summing up its tail distribution:

$$E[D_3] = \Delta \sum_{n=0}^{\infty} \bar{G}_3(x_n). \tag{3.39}$$

Expanding $\bar{G}_3(x_n)$ using (3.36) and canceling all but two remaining terms leads to the desired result. □

Similar to M_1, method M_3 is consistent when $F_U(x)$ is exponential, which is shown as follows:

Corollary 2. *Exponential is the only update distribution that allow M_3 to be Δ-consistent with $F_U(x)$ for all Δ.*

However, in broader NWU/NBU settings, its distribution lies between $F_U(x)$ and $G_U(x)$. As sampling interval $\Delta \to \infty$, which corresponds to $p \to 1$, variable D_3 converges in distribution towards A_U. When $\Delta \to 0$, which reflects $p \to 0$, D_3 tends to U. Unfortunately, neither scenario is usable in practice, which makes the method generally biased.

3.5.3 Method M_4

Using the rationale behind M_2, we now propose another new method, which we call M_4. At each sampling point s_j, the obtained value is:

$$D_4(s_j) := \begin{cases} \Delta & Q_{j-1,j} = 1 \\ D_4(s_{j-1}) + \Delta & \text{otherwise} \end{cases}. \tag{3.40}$$

For the example in Fig. 3.10, this method collects four samples – Δ, 2Δ, 3Δ and 4Δ. Denote by $G_4(x,T)$ the distribution generated by M_4 in $[0,T]$. Then, we have the following result.

Theorem 20. *Method M_4 is Δ-consistent with respect to the age distribution:*

$$G_4(x_n) := \lim_{T\to\infty} G_4(x_n,T) = G_U(x_n). \tag{3.41}$$

Proof. It is not difficult to see that M_4 collects samples $A_U^+(s_j)$ in all points s_j. Therefore,

$$G_3(x_n,T) = \frac{\sum_{j=1}^{N_S(T)} \mathbf{1}_{A_U^+(s_j)\le x_n},}{N_S(T)} \tag{3.42}$$

where $x^+ = \Delta\lceil x/\Delta\rceil$ as before. Since the CDF is computed only in discrete points x_n, the above can be written as:

$$G_3(x_n,T) = \frac{\sum_{j=1}^{N_S(T)} \mathbf{1}_{A_U(s_j)\le x_n}}{N_S(T)} = G_2(x_n,T), \tag{3.43}$$

which converges to $G_U(x)$ using (3.30). □

Define a random variable $D_4 \sim G_4(x)$, where $G_4(x)$ is a continuous step-function

taking jumps at each x_n. Interestingly, even though M_3 keeps the largest age sample in each detected update interval $[u_i, u_{i+1}]$, the mean of its values $E[D_3]$ is not necessarily larger than that of D_4. For example, with Pareto updates and $\Delta = 1$, we get $E[D_4] = 1.63$ and $E[D_3] = 1.33$. This can be explained by our previous discussion showing that under NWU update intervals the tail $\bar{G}_3(x)$ is upper-bounded by $\bar{G}_4(x)$, which implies $E[D_4] \geq E[D_3]$. Note that if U is NBU, this relationship is again reversed.

3.5.4 Undoing Bias in M_3

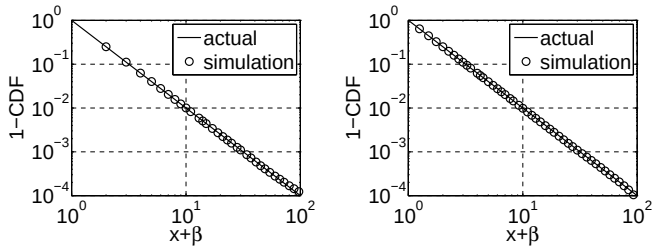
From the last two subsections, we learned that M_4 is always Δ-consistent with respect to $G_U(x)$, while M_3 is biased unless U is exponential or Δ is infinitely small. One advantage that M_3 may have is that it operates with significantly fewer samples. As mentioned in the previous section, M_3 is inconsistent but widely used by researchers to measure lifetimes in storage objects and P2P systems. This raises the question of whether one can achieve Δ-consistency using exact same samples as M_3.

To this end, let $x^+ = \Delta\lceil x/\Delta \rceil$ be x rounded-up to the nearest multiple of Δ and define:

$$G_5(x_n, T) := \frac{1}{T} \sum_{j=1}^{T/\Delta} \min(x_n, A_U^+(s_j)) Q_{j,j+1} \tag{3.44}$$

to be an estimator that takes samples of M_3, passes them through the min function, and normalizes the resulting sum by window size T. Note that the number of terms in the summation is $K(\infty, T)$, i.e., the number of detected updates.

Theorem 21. *Estimator M_5 is Δ-consistent with respect to the age distribution:*



(a) $\lambda = 1$ (b) $\lambda = 4$

Figure 3.12: Verification of (3.45) under Pareto U ($\mu = 2$).

$$G_5(x_n) := \lim_{T\to\infty} G_5(x_n, T) = G_U(x_n). \tag{3.45}$$

Proof. We start with an auxiliary result:

$$\begin{aligned}\sum_{k=0}^{n-1} \mathbf{1}_{A_U(s_j)>x_k} &= \sum_{k=0}^{n-1} \mathbf{1}_{A_U^+(s_j)>x_k} = \sum_{k=0}^{n-1} \mathbf{1}_{\lceil A_U(s_j)\Delta\rceil>k} \\ &= \min(n, \lceil A_U(s_j)\Delta\rceil) = \frac{\min(x_n, A_U^+(s_j))}{\Delta}. \end{aligned} \tag{3.46}$$

Next, applying this to expansion of (3.44):

$$\begin{aligned} G_5(x_n, T) &= \frac{\Delta}{T} \sum_{j=1}^{T/\Delta} Q_{j,j+1} \sum_{k=0}^{n-1} \mathbf{1}_{A_U(s_j)>x_n} \\ &= \frac{\Delta}{T} \sum_{k=0}^{n-1} \sum_{j=1}^{T/\Delta} \mathbf{1}_{A_U(s_j)>x_n} Q_{j,j+1} \\ &= \frac{K(\infty, T)}{N_S(T)} \sum_{k=0}^{n-1} \bar{G}_3(x_n, T), \end{aligned} \tag{3.47}$$

where $K(x, T)$ is given by (3.6) and $\bar{G}_3(x_n, T)$ by (3.37). Since $K(\infty, T)/N_S(T)$

converges to p, we get after applying (3.36) to the expansion of $\bar{G}_3(x_n, T)$:

$$G_5(x_n) = p\frac{G_U(x_n)}{G_U(\Delta)} = G_U(x_n),$$

where we use the fact that $p = G_U(\Delta)$. □

Fig. 3.12 shows that M_5 accurately obtains the tail of $G_U(x)$, even for Δ bounded away from zero. We next compare M_5 with M_4 to see if the reduction in the number of samples has a noticeable impact on accuracy. The first metric under consideration is the Weighted Mean Relative Difference (WMRD), often used in networking [33]. Assuming $H(x, T)$ is some empirical CDF computed in $[0, T]$, then the WMRD between $H(x, T)$ and $G_U(x)$ is:

$$w(T) := \frac{\sum_n |H(x_n, T) - G_U(x_n)|}{\sum_n (H(x_n, T) + G_U(x_n))/2}. \tag{3.48}$$

The second metric is the Kolmogorov-Smirnov (KS) statistic, which is the maximum distance between two distributions:

$$\kappa(T) := \sup_x |H(x, T) - G_U(x)|. \tag{3.49}$$

Simulations results are shown in Table 3.1. Observe that M_4 performs slightly better for $T \leq 10^3$, but then the two methods become identical and their error decays as $1/\sqrt{T}$. Even if T is small, the minor loss of accuracy in M_5 may well be worth a 20% reduction in the number of samples. As given in Fig. 3.4(a), larger λ leads to even higher savings, e.g., 80% for $\lambda = 10$.

Table 3.1: Convergence of Both Δ-Consistent Methods under Pareto U ($\mu = 2, \lambda = 1$)

T	M_4		M_5	
	$w(T)$	$\kappa(T)$	$w(T)$	$\kappa(T)$
10^2	3.5×10^{-2}	6.4×10^{-2}	3.7×10^{-2}	6.7×10^{-2}
10^3	1.4×10^{-2}	2.2×10^{-2}	1.4×10^{-2}	2.2×10^{-2}
10^4	4.7×10^{-3}	7.2×10^{-3}	4.7×10^{-3}	7.3×10^{-3}
10^5	1.5×10^{-3}	2.4×10^{-3}	1.5×10^{-3}	2.4×10^{-3}
10^6	4.1×10^{-4}	5.8×10^{-4}	4.1×10^{-4}	5.8×10^{-4}
10^7	2.2×10^{-4}	2.6×10^{-4}	2.2×10^{-4}	2.6×10^{-4}

3.6 Comparison Sampling: Random Intervals

Although M_4 and M_5 are consistent estimators of $G_U(x)$, they do not generally guarantee recovery of $F_U(x)$. Furthermore, constant S may not always be achievable in practice. For instance, search engine juggle trillions of pages, whose download rate is dynamically adjusted based on real-time ranking and budgeting. It may thus be difficult to ensure constant return delays to each page. Additional problems stem from lattice update processes, where constant S fails to satisfy Assumption 2, rendering measurements arbitrarily inaccurate.

In this section, we consider comparison sampling with random intervals. We first show that extending M_4 to this scenario delivers surprisingly biased results. Then, we present our new method M_6 and verify its correctness using simulations.

3.6.1 Straightforward Approaches

Our first attempt is to generalize M_4 to random S, which we call G-M_4. For a given s_j, define the most-recent sample point after which an update has been detected

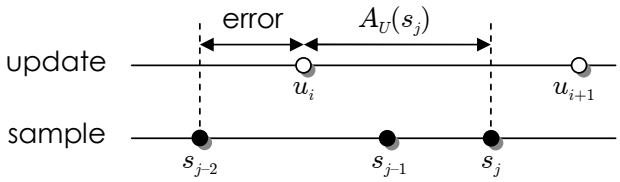


Figure 3.13: Illustration of G-M_4.

as:

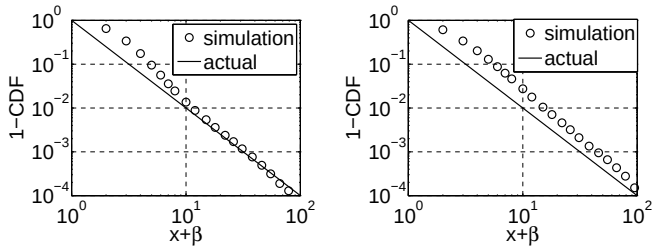
$$s_j^* := \max_{i<j}\{s_i : Q_{ij} = 1\}. \tag{3.50}$$

Then, G-M_4 rounds age $A_U(s_j)$ up to $s_j - s_j^*$. An example is shown in Fig. 3.13, where the measured value is $s_{j+2} - s_j$. For constant S, this method is identical to M_4, which we know is consistent. The main difference with random S is that the amount of round-off error in G-M_4 varies from interval to interval. This issues has a profound impact on the result, as shown in Fig. 3.14. Observe that the exponential case becomes somewhat consistent only for $x_n \gg 0$ and the Pareto case produces a tail that is completely different from the actual $\bar{G}_U(x)$. This motivates us to search for another approach.

3.6.2 Method M_6

Our rationale for this technique stems from the fact that $Q_{ij} = 1$ if and only if $A_U(s_j) < s_j - s_i$. Therefore, counting the fraction of pairs (i, j) that sustain an update may lead to $G_U(x)$. Define $y^\circ = h\lceil y/h \rceil$ to be the rounded-up value of y with respect to a user-defined constant h. Let $y_n = nh$ and:

$$W_{ij}(y_n) := \begin{cases} 1 & (s_j - s_i)^\circ = y_n \\ 0 & \text{otherwise} \end{cases}. \tag{3.51}$$



(a) exponential S (b) Pareto S

Figure 3.14: Bias of G-M_4 with Pareto updates ($\mu = 2, \lambda = 1$).

Then, the number of inter-sample distances $s_j - s_i$ in $[0, T]$ that round up to y_n is given by:

$$W(y_n, T) := \sum_{i=1}^{N_S(T)} \sum_{j=i+1}^{N_S(T)} W_{ij}(y_n) \tag{3.52}$$

and the number of them with an update is:

$$Z(y_n, T) := \sum_{i=1}^{N_S(T)} \sum_{j=i+1}^{N_S(T)} Q_{ij} W_{ij}(y_n). \tag{3.53}$$

We can now define estimator M_6 by its CDF:

$$G_6(y_n, T) := \frac{Z(y_n, T)}{W(y_n, T)} \tag{3.54}$$

For a given λ, method M_6 has the same network overhead as the other methods; however, it utilizes $\Theta(n^2)$ pairwise comparisons, significantly more than the other methods, which are all linear in n. Despite a higher computational cost, M_6 gains significant accuracy advantages when distances $s_i - s_j$ are allowed to sweep all possible points $x \geq 0$. Combining this with bins of sufficiently small size creates a continuous CDF, which allows recovery of not only $G_U(x)$, but also $F_U(x)$.

Theorem 22. *Assume $h \to 0$, N_S is age-mixing, and $F_S(x) > 0$ for all $x > 0$. Then, method M_6 is consistent with respect to the age distribution:*

$$G_6(y) := \lim_{T\to\infty} G_6(y,T) = G_U(y). \tag{3.55}$$

Proof. First, it helps to observe that:

$$Q_{ij} = \mathbf{1}_{R_U(s_i)\le(s_j-s_i)}. \tag{3.56}$$

Since the download process is renewal, it follows that:

$$s_j - s_i \sim F_S^{*(j-i)}, \tag{3.57}$$

where $F^{*k}(x)$ denotes a k-fold convolution of distribution $F(x)$. Furthermore, the renewal nature of N_S implies that variable $s_j - s_i$ is independent of s_i. Now, let

$$Y_k \sim F_S^{*k}(x) \tag{3.58}$$

be a random variable with the same distribution as $S_1 + \ldots + S_k$ and define the renewal function driven by $F_S(x)$ as [103]:

$$M_S(t) = 1 + \sum_{k=1}^{\infty} F_S^{*k}(t). \tag{3.59}$$

Then, renewal theory shows for $x > h$ and $n \to \infty$ that:

$$\frac{1}{n}\sum_{i=1}^{n}\sum_{j=i+1}^{n} \mathbf{1}_{R_U(s_i)\le(s_j-s_i)}\,\mathbf{1}_{s_j-s_i\in(x-h,x]} \tag{3.60}$$

converges to

$$\sum_{k=1}^{\infty} P(R_U \leq Y_k, Y_k \in (x-h, x])$$
$$= \sum_{k=1}^{\infty} \int_{x-h}^{x} G_U(y)\, dF_S^{*k}(y) = \int_{x-h}^{x} G_U(y)\, dM_S(y). \tag{3.61}$$

Let $n = N_S(T)$ and assume that $h(T) = T^{-\delta}$, where $\delta \in (0, 1)$ ensures that h diminishes to zero at some appropriate rate[1]. Since $G_U(x)$ is continuous, it follows that:

$$\lim_{T\to\infty} \frac{Z(y_n, T)}{W(y_n, T)} = \lim_{h\to 0} \frac{\int_{x-h}^{x} G_U(y)\, dM_S(y)}{\int_{x-h}^{x} dM_S(y)} = G_U(x) \tag{3.62}$$

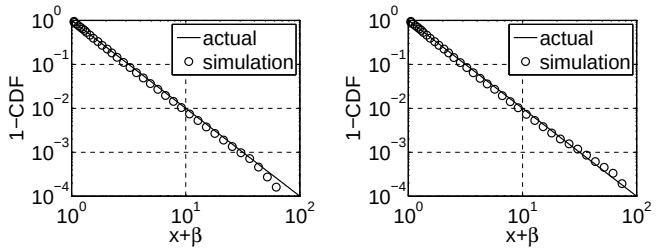
for each $x > 0$. □

The assumption that $F_S(x)$ contains non-zero mass in the vicinity of zero is necessary for accurate estimation of $g_U(x)$ at $x = 0$, which then leads to $F_U(x)$. This can be accomplished by a number of continuous distributions, e.g., $\exp(\lambda)$ or uniform in $[0, 2\lambda]$. It should also be noted that M_6 works for lattice S, but in that case it offers no benefits over M_4. Fig. 3.15 compares the M_6 estimator against $G_U(x)$ under two sampling distributions $F_S(x)$, both satisfying Theorem 22. Compared to Fig. 3.14, this result is overwhelmingly better.

3.7 Discussion

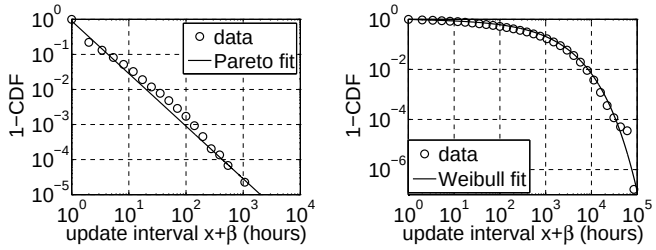
We have finished establishing our methods in previous sections. In this section, we first introduce our Wikipedia dataset and show that Poisson update assumption is not universally applicable, especially on our dataset. Then, we compare the performance

[1]From well-known results in non-parametric function estimation, $\delta = 1/5$ should be optimal for the mean squared error of the estimate.



(a) exponential S (b) Pareto S

Figure 3.15: Simulations of M_6 under Pareto updates ($h = 0.05, \mu = 2, \lambda = 1$).



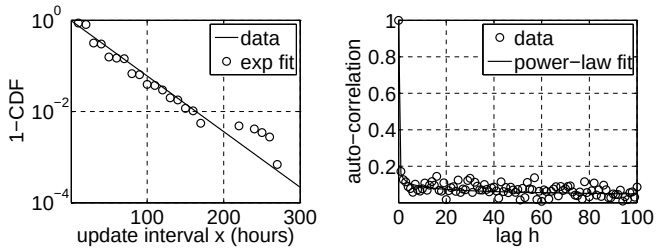
(a) George W. Bush CCDF (b) All articles CCDF

Figure 3.16: Inter-update delay distribution.

of our methods and propose the optimal choice of our methods in different scenarios.

3.7.1 Dataset and Poisson Assumption

Wikipedia [100] is a collaborative editing website which keeps the all the revision histories of its pages. This provides us modification timestamps of all pages, from which we can obtain the inter-update delays by subtracting two consecutive timestamps. We downloaded 3.6 million English Wikipedia articles and computed the update intervals of all pages. The average number of updates among all pages is 71.4. Because all sampling methods require sufficiently large amount of samples, our first focus is the most frequently modified article "George W. Bush", whose CCDF is shown in Fig. 3.16(a). Observe that it fits Pareto tail $(1 + x/\beta)^{-\alpha}$ with $\alpha = 1.4$



(a) exp updates distribution

(b) exp updates $\rho(h)$

Figure 3.17: Pages with exponential updates.

and $\beta = 0.93$ well. Fig. 3.16(b) displays the inter-update delay distribution of all articles. Curve fitting on the data shows that update delay distribution matches Weibull tail $e^{-(x/\nu)^k}$ with $\nu = 400$ and $k = 0.5$. Results in both figures show that inter-update delay distributions are not exponential.

Even when the inter-update delay distribution is exponential, a Poisson process requires the independency between delays. To find articles with exponential inter-update delays, we compute coefficient of variation of all pages with non-trivial modification history. It can be shown that for exponential X, the coefficient of variation $v = \sqrt{Var[X]}/E[X]$ is 1. Then we scan for articles with $|v - 1| \leq 0.1$ among $444K$ pages with at least 100 updates and obtain exactly 202 results. As shown in Fig. 3.17(a), the joint distribution of selected articles fits exponential pretty well. To measure the dependency between delays, we use the auto-correlation function $\rho(h)$. Note that iid processes should exhibit $\rho(h) = 0$ for all lags $h = 1$. Observe from Fig. 3.17(b) that the auto-correlation was heavy-tailed, which matches a power-law function $h^{-\gamma}$ with $\gamma = 0.25$. This suggests long-range dependence (LRD) with Hurst parameter $H = 0.87$. This prevents the applicability of previous models [17, 61] because of the Poisson update assumption. With our new model and proofs, M_1 and M_3 can now be applied to the correlated case as long as $F_U(x)$ is exponential.

Table 3.2: Top 10 most frequently modified Wikipedia articles

Page	updates	$E[U]$ (hrs)	duration (days)
George W. Bush	44,296	1.86	3,433
LWWE *	33,744	1.46	2,058
Wikipedia	30,407	2.70	3,420
United States	28,771	2.90	3,482
Michael Jackson	25,558	3.22	3,433
Jesus	23,027	3.82	3,670
Britney Spears	21,549	3.85	3,454
World War II	21,424	3.83	3,418
Wii	21,023	3.00	2,630
Adolf Hitler	20,964	3.91	3,413

*LWWE: List of World Wrestling Entertainment personnel

3.7.2 Method Comparison

Now we start comparing the performance of our methods. In order to get enough samples, we choose the top 10 most frequently modified Wikipedia articles as our test data. This is listed in Table 3.2 with the first page "George W. Bush" experiencing more than two times updates than the last "Adolf Hitler". The average inter-update delays $E[U]$ spans from 1 to 4 hours. Majority of pages have duration between 9 to 10 years. More frequently modified page does not always gives lower $E[U]$ because pages have different duration intervals.

Since all our methods compute $G_U(x)$ first and apply numerical derivative to obtain $F_U(x)$, we focus on comparing the difference between $G_U(x)$. For the top 10 articles shown in Table 3.2, we compute $G_U(x)$ using (3.1) as the ground truth. We simulate all our methods with time $T = 100,000$ hours to test their performance. We repeat the update traces when the update sample duration is less than T. The performance metrics in consideration are WMRD in (3.48) and KS in (3.49). We do not consider methods M_1 and M_3 because they both are unbiased only when the

Table 3.3: Method comparison using $\kappa(T)$ ($\lambda = 0.5$, $T = 10^5$)

Page	Exponential S		Constant S			
	M_2	M_6	M_2	M_4	M_5	M_6
1	0.19%	1.19%	0.06%	1.07%	1.07%	1.07%
2	0.13%	1.26%	0.08%	1.40%	1.40%	1.40%
3	0.25%	1.34%	0.08%	0.56%	1.00%	0.56%
4	0.19%	1.33%	0.09%	0.73%	0.73%	0.73%
5	0.63%	1.51%	0.07%	0.62%	0.62%	0.62%
6	0.32%	1.50%	0.06%	0.54%	0.54%	0.54%
7	0.11%	1.30%	0.06%	0.41%	0.41%	0.41%
8	0.31%	1.23%	0.06%	0.47%	0.47%	0.47%
9	0.22%	1.13%	0.09%	1.08%	1.08%	1.08%
10	0.25%	1.34%	0.08%	0.46%	0.46%	0.46%

inter-updated delay distribution is exponential, which is not the case for the top 10 Wikipedia articles.

We use both exponential S and constant S with the same mean $E[S] = 30$ minutes to sample the pages. The methods which work under exponential S include M2 and M6, both of which produce continuous CDF curves. We compute WMRD and KS at discrete points from 0 to 1000 hours with gap 3 minutes. Observe from Table 3.3 and 3.4 that M_2 performs much better than M_6 because it M_2 has the knowledge of the age of the update which M_6 does not have.

For constant $S = \Delta$, we consider age-based method M_2 and comparison based methods M_4, M_5, and M_6. Notice that all compassion based methods produce discrete CDF at points which are multiple of $\Delta = 0.5$ hours. To make a fair comparison, we compute their WMRD and KS at the same points as M_2 and M_6 under exponential S. Since the evaluation points have smaller granularity than Δ, we apply linear interpolation to get the values at all points. Table 3.3 and 3.4 shows that age-based method M_2 produces better results than comparison-based methods M_4, M_5 and M_6. When age information is not available, M_4 is better choice than M_6

Table 3.4: Method comparison using $w(T)$ ($\lambda = 0.5$, $T = 10^5$)

Page	Exponential S		Constant S			
	M_2	M_6	M_2	M_4	M_5	M_6
1	0.02%	0.13%	0.01%	0.01%	0.02%	0.01%
2	0.002%	0.10%	0.001%	0.001%	0.005%	0.01%
3	0.03%	0.11%	0.01%	0.01%	0.04%	0.01%
4	0.07%	0.19%	0.02%	0.02%	0.04%	0.04%
5	0.41%	0.39%	0.02%	0.02%	0.02%	0.07%
6	0.18%	0.33%	0.04%	0.04%	0.43%	0.10%
7	0.04%	0.21%	0.02%	0.02%	0.02%	0.05%
8	0.04%	0.15%	0.01%	0.01%	0.06%	0.03%
9	0.02%	0.13%	0.03%	0.03%	0.02%	0.01%
10	0.12%	0.25%	0.03%	0.03%	0.03%	0.06%

because its lower complexity and higher accuracy. However, this does not preclude the usefulness of M_5 because existing methods such as CBM already measure lots of traces using M_3, which is biased but can be corrected by applying M_5. Furthermore, M_5 needs less number of samples which is useful to balance the accuracy and system capacity, especially in system with limited space.

According to our discussion above, we propose our optimal choice under different scenarios, as shown in Fig. 3.18. When age is available, method M_2 is always preferable. For comparison-based methods under constant S, M_4 should be used if one wants to start a new measurement, while M_5 should be used to correct the bias of existing measurement samples collected by M_3. Finally, when age is not available and S is random, M_6 should be adopted since it is the only option.

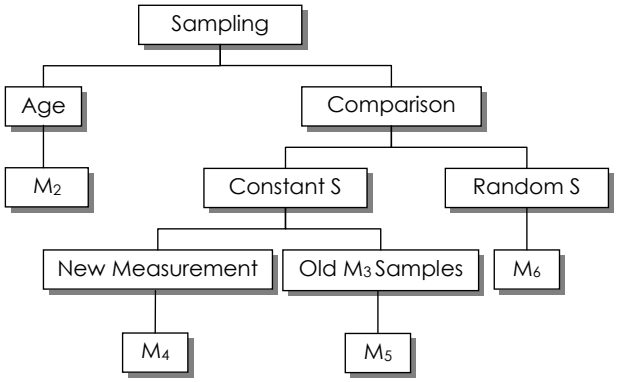


Figure 3.18: Optimal choice of methods.

4. STOCHASTIC MODELS OF PULL-BASED DATA REPLICATION IN P2P SYSTEMS

4.1 Introduction

P2P file sharing systems have received tremendous interest in recent years among both Internet users and computer networking professionals. In the study of these networks, one fundamental problem is to handle peer overload that may arise due to the highly popular nature of certain content and/or temporal fluctuations in demand (e.g., flash crowds). A common solution for *static* files is to replicate them from each source to multiple peers, which distributes the load and thus improves file-query efficiency. Examples include protocols that replicate content close to the owner [26, 78, 91], near the requester [42], and along query paths [24, 77, 107].

With real-time operation of certain P2P applications, such as online auctions [39], decentralized collaboration [109], web caching [51], and online games [105], replication faces new challenges related to *data churn* (i.e., periodic content updates at the source). To provide accurate and reliable query results in these systems, replicated material must be continuously synchronized with that at the source. Without an effective consistency-maintenance strategy, these applications may suffer from degraded performance and lower user participation.

The majority of existing P2P synchronization methods are *push-based*. To allow the source easy discovery of replica location, these networks often employ structured P2P networks to establish a mapping from file IDs to nodes where they can be found [11, 13, 48, 57, 77, 104, 107]. Since the source must track the status and location of each replica, as well as reconfigure the distribution tree, these methods may suffer from high maintenance overhead, especially when the network structure is volatile

(i.e., under high user churn).

In unstructured P2P networks, management of replicas is usually achieved by message spreading [19, 27, 41, 60], which may generate large amounts of redundant traffic and even lead to network collapse when search rates become sufficiently high. To address this problem, several studies [60, 81, 82, 83, 96] propose *pull-based* consistency control, which allows replicas to self-manage their membership in replication paths and decide when to download content from the source. Pull-based techniques have also been used in pub/sub systems [12, 32] and hybrid push/pull methods found application in decentralized online social networks [88].

In databases, it is well-known [52], [108] that pull-based synchronization improves both scalability and availability of the data, but at the expense of increased age of the content served to clients. As the source evolves, replicas in these networks go through periods of staleness, during which they provide outdated responses that do not reflect the true condition of the source. To measure system performance, it is generally accepted that the *probability of freshness* (i.e., likelihood that the most-recent version is available to consumers) accurately reflects the quality of a replication strategy. Although this metric has been considered by researchers in web-based systems [10, 15, 20, 71, 69, 101], it has never been explored in the context of P2P systems. We aim to fill this void below.

4.1.1 Contributions

We start by considering a single source driven by an update process N_U and a single replica with the corresponding download process N_D, which is independent of N_U. Our first contribution is to propose a general framework for modeling freshness under arbitrary renewal processes (N_U, N_D). This allows us to derive the freshness probability p in closed-form as a function of inter-update distribution $F_U(x)$ and

inter-download distribution $F_D(x)$. Our formula for p generalizes all previous analytical results in the literature [7], [8], [14], [15], [17], [20], [31], [34], [40], [49], [53], [59], [61], [65], [66], [84], [95], [106], which were predominantly limited to Poisson N_U and two simple cases of $F_D(x)$.

Given a fixed download rate λ, our second contribution is to obtain a condition that allows comparison of freshness achieved by different download strategies $F_D(x)$. We show that freshness improves if the inter-synchronization interval becomes *stochastically larger in second order*. This allows us to prove that constant delays are optimal against *all* N_U. For the same p, they require 33% less bandwidth than exponential inter-download delays and 50% less than Pareto.

Based on these results, our third contribution is to analyze *cascaded* synchronization, where replicas receive content from other replicas along a fixed multi-hop path from the source. A common arrangement covered by this model is a b-way replication tree, which limits the source to b concurrent downloads, but keeps client-scalability arbitrarily high depending on the depth of the tree. Assuming independent operation among the replicas, we derive a recursive model that provides freshness at each level i. Our results show that in certain cases p decays exponentially fast as a function of the depth, suggesting an interesting coupling between system size and staleness.

Our fourth contribution is to propose a model of *cooperative caching*, in which m replicas form a single layer, in which each participant can synchronize not only with the source, but also k other replicas. For a target p, the goal is to determine the optimal (m, k) that maximizes the service rate of the entire system. The main caveat of this model is that it takes into account bandwidth constraints at the source and each replica. We show that making k or m too large is detrimental to performance; instead, *each parameter has a unique optimal value that achieves the highest service rate*, which can be $2-7$ times larger than under non-cooperative replication.

Our last contribution is to examine a scenario we call *redundant querying*, in which consumers have access to multiple independent caches. Issuing parallel queries to k replicas out of the m available, the hope is to improve freshness by selecting the most up-to-date copy of the source. We first show that freshness in this case can be computed using the original update process and a superposition of download processes from each of the contacted replicas. However, taking bandwidth into account, this analysis also leads to a surprising conclusion that *redundant querying with* $k \geq 2$ *sometimes produces lower performance than non-redundant.*

4.2 Related Work

Consistency maintenance in existing P2P networks can be classified into *push-based* and *pull-based*. In the former, the source is responsible for sending updates to replicas whenever it deems necessary. To achieve this, the source has to know the location of all of its replicas either by utilizing a rigid network structure that maps files to nodes [11, 13, 77, 104, 107] or randomly flooding the graph [19, 27, 41, 60]. In pull-based methods, nodes become replicas, discontinue being such, and adjust their download policy independently of the source. Existing work in this direction [81, 82, 83] focuses on determine the polling frequency using a family of linear-increase multiplicative-decrease (LIMD) algorithms.

In other fields, pull-based data synchronization has also been studied. In the context of web systems [10, 15, 20, 101], sources are typically HTML pages modified by their owners. Replicas can be search engines that use web-crawlers to periodically reload content and refresh their indexes; however, additional applications are possible as well – online monitoring systems [69, 71] of highly dynamic web streams (e.g., stock market, traffic), traditional web caching [21, 22, 23], RSS feed aggregation [84], and many others.

Figure 4.1: System model. Arrows represent pull-based requests for information.

In the analytical literature, freshness p was first proposed by Coffman *et al.* [20] and later used by much of the follow-up work [10, 15, 14, 65, 101]. Other ways to capture staleness include information divergence between the source and its replica [66], age of the content served to clients [15], the number of missing updates at the replica [29], [49], and their combined age [58, 59, 84]. In all of these cases, models are derived under the assumption that the update process N_U is Poisson. There has been only one attempt [101] to relax this constraint, but it required deterministic knowledge all download instances. While appropriate in some cases, this model is difficult to evaluate in practice when inter-download delays are random and given by their distribution $F_D(x)$.

4.3 Single-Hop Replication

In this section, we consider a single source and one of its replicas. We first introduce our assumptions and notation, define freshness p, and derive its closed-form model. We then use simulations to highlight several examples.

4.3.1 System Model and Notation

We assume a model of data generation, replication, and consumption in Fig. 4.1. During system operation, the source experiences random updates in response to either external events (e.g., price bids in e-auctions, status changes in online games) or some internal computation (e.g., indexing, MapReduce output). In either case, each update represents certain tangible information that manipulates the current state of the source. The replica has no direct knowledge of these updates and must infer

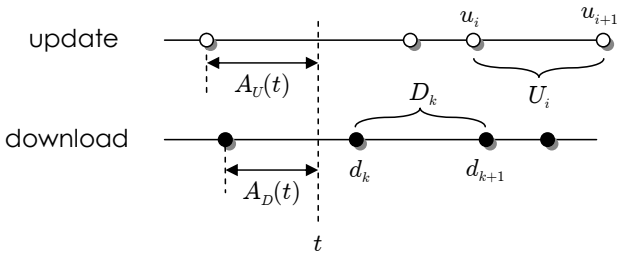


Figure 4.2: Illustration of age and other variables.

their occurrence only through periodic downloads. Its goal is to provide consumers with up-to-date responses to various types of queries.

Let u_i denote the time when the i-th update occurs at the source. Define $N_U(t) = \max\{i : u_i \leq t\}$ to be the number of updates in $[0, t]$ and $U_i = u_{i+1} - u_i$ as the i-th inter-update delay. Similarly, let d_k be the k-th download instance, $N_D(t) = \max\{k : d_k \leq t\}$ the number of such points in $[0, t]$, and $D_k = d_{k+1} - d_k$ the k-th inter-synchronization delay. To keep the system tractable, assume that N_D and N_U are independent renewal processes. This means that update intervals $\{U_i\}_{i=1}^{\infty}$ and synchronization delays $\{D_i\}_{i=1}^{\infty}$ are two sets of independent and identically distributed (iid) variables. This allows us to replace them with random variables $U \sim F_U(x)$ and $D \sim F_D(x)$, respectively.

4.3.2 Performance Measure

Observe that a local copy at the replica is *fresh* if and only if the last update time $u_{N_U(t)}$ is smaller than the last download time $d_{N_D(t)}$. Freshness of the data can thus be modeled by an alternating (ON/OFF) process:

$$\phi(t) = \begin{cases} 1 & u_{N_U(t)} < d_{N_D(t)} \\ 0 & \text{otherwise} \end{cases}, \tag{4.1}$$

where t represents the time of a consumer's request to the replica. In practice, it is more convenient to express $\phi(t)$ as a function of age. Define the *download age* at t to be the time lag from the last synchronization to t:

$$A_D(t) = t - d_{N_D(t)} \tag{4.2}$$

and the *update age* to be that from the last update:

$$A_U(t) = t - u_{N_U(t)}. \tag{4.3}$$

These definitions are illustrated in Fig. 4.2, where the empty circles are update events and the solid ones are download instances. Using the figure, it is not difficult to see that a copy is fresh if and only if the download age is smaller than the update age, which means that:

$$\phi(t) = \begin{cases} 1 & A_U(t) > A_D(t) \\ 0 & \text{otherwise} \end{cases}. \tag{4.4}$$

We model the consumer as querying the replica at some large time t by which the system can be considered stationary. As $t \to \infty$, $A_U(t)$ and $A_D(t)$ converge to their equilibrium versions, which we call A_U and A_D, respectively. Define $\mu = 1/E[U]$ to be the update rate and let $\lambda = 1/E[D]$ be the download rate. Then, from renewal theory [103], the two ages have well-known distributions:

$$G_U(x) := P(A_U < x) = \mu \int_0^x (1 - F_U(y))dy \tag{4.5}$$

and

$$G_D(x) := P(A_D < x) = \lambda \int_0^x (1 - F_D(y))dy. \tag{4.6}$$

This allows us to formulate the main metric of system performance – the limiting probability that consumers encounter a fresh copy:

$$p := \lim_{t\to\infty} P(\phi(t) = 1) = P(A_U > A_D). \tag{4.7}$$

To keep notation simple and prevent unnecessary explanation, we generally use $F_X(x)$ to represent the distribution of variable X, $G_X(x)$ to denote its age distribution similar to (4.5)-(4.6), and lower-case functions $f_X(x)$ and $g_X(x)$ as the corresponding densities. We also use $\bar{F}_X(x) = 1 - F_X(x)$ as the CCDF (complementary cumulative distribution function) of variable X and replace $X(t)$ with its limiting variable X as $t \to \infty$, whenever doing so is appropriate.

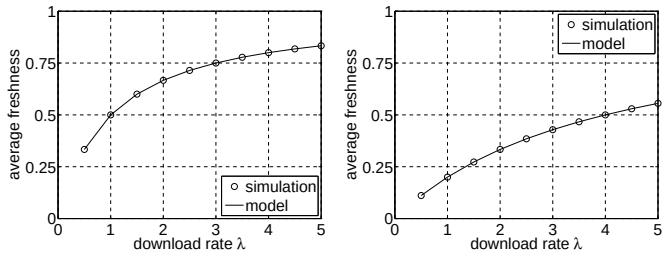
4.3.3 Freshness Probability

Our next result directly follows from (4.4).

Theorem 23. *Freshness experienced by consumers in steady-state is given by:*

$$p = E[\bar{G}_U(A_D)] = \int_0^\infty \bar{G}_U(x) g_D(x) dx. \tag{4.8}$$

To perform a self-check against prior results with Poisson N_U, observe that (4.8) simplifies to $p = \lambda(1 - e^{-\mu/\lambda})/\mu$ under constant D and $\lambda/(\lambda + \mu)$ under exponential D, which is in agreement with [15], [20]. We use simulations to examine model accuracy in more interesting cases of general renewal processes. Since U and D are non-negative random variables defined on $(0, \infty)$, our Pareto CDF is $1-(1+x/\beta)^{-\alpha}$ for $\alpha > 1$ and $\beta > 0$. The mean of this distribution is $\beta/(\alpha - 1)$, where α is kept



(a) Pareto U, constant D (b) constant U, Pareto D

Figure 4.3: Simulation results of (4.8) under $\mu = 2$.

at 3 throughout the paper. We simulate each process to large enough t to reach stationarity of the underlying processes. This typically requires a few hundred units of time.

Observe in Fig. 4.3 that the model matches simulations very well, with constant download intervals performing significantly better against Pareto update cycles in (a) than the other way around in (b). For instance, using download rate $\lambda = 3$, which is 50% faster than the update rate $\mu = 2$ in the figure, part (a) achieves 75% freshness, while part (b) only 43%. It is unclear, however, whether constant D is always better than Pareto and what impact $F_U(x)$ has on the resulting p. We address this question next.

4.4 Best Download Strategy

In this section, we study conditions under which one combination of processes (N_U, N_D) performs better than another.

4.4.1 Basics

Noticing from (4.6) that $g_D(x) = G'_D(x) = \lambda(1 - F_D(x))$, the result in (4.8) shows that expected freshness p is impacted by not only the product of update and download rates $\mu\lambda$, but also the entire functions $F_D(x)$ and $F_U(x)$. To establish order

between distributions, we need the next definition commonly used in game theory and statistics.

Definition 13. *Variable X is said to be* stochastically larger than Y*, which is written as $X \geq_{st} Y$, if $\bar{F}_X(x) \geq \bar{F}_Y(x)$ for all $x \geq 0$. Variable X is* stochastically larger than Y in second order*, which is written as $X \geq^2_{st} Y$, if $\int_0^x \bar{F}_X(y)dy \geq \int_0^x \bar{F}_Y(y)dy$ for all $x \geq 0$.*

Since $\bar{G}_U(x)$ is a monotonically non-increasing function, it is easy to show that (4.8) produces the following result.

Lemma 4. *For two download strategies driven by inter-synchronization delays X and Y, freshness $p_X \geq p_Y$ when the age of Y stochastically dominates that of X, i.e., $A_Y \geq_{st} A_X$.*

To make this result useful, we next translate stochastic dominance between ages to that between the corresponding variables.

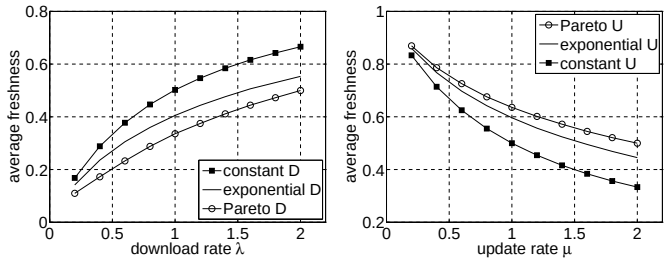
Lemma 5. *Assuming $E[X] = E[Y] > 0$, variable X is stochastically larger than Y in second order, i.e., $X \geq^2_{st} Y$, if and only if the age of Y is stochastically larger than that of X, i.e., $A_Y \geq_{st} A_X$.*

Proof. Let $G_X(x)$ and $G_Y(x)$ be the CDFs of A_X and A_Y, respectively. Define:

$$J(x) = \bar{G}_Y(x) - \bar{G}_X(x) \tag{4.9}$$

and expressed it as:

$$J(x) = \frac{\int_0^x \bar{F}_X(y)dy}{E[X]} - \frac{\int_0^x \bar{F}_Y(y)dy}{E[Y]}. \tag{4.10}$$



(a) Pareto U (b) Pareto D

Figure 4.4: Ordering of freshness under different families of distributions.

Since both means are positive and equal to each other, we get that $A_Y \geq_{st} A_X$ iff $J(x) \geq 0$ for all $x \geq 0$, which holds iff $X \geq^2_{st} Y$. □

Combining these observations, we obtain the following.

Theorem 24. *For a given N_U and two download distributions $F_X(x)$ and $F_Y(x)$ with the same rate λ, freshness $p_X \geq p_Y$ when X stochastically dominates Y in second order. Similarly, for a given N_D and two update distributions $F_W(x)$ and $F_Z(x)$ with the same rate μ, freshness $p_W \geq p_Z$ when Z stochastically dominates W in second order.*

4.4.2 Examples

Our last result shows that freshness is improved when D becomes stochastically larger in second order or U becomes the opposite, i.e., smaller. To put this in perspective, consider three classes of distributions often observed in practice. The first one is NWU (new worse than used), which means that conditioned on the fact that an interval is at least y time units, its surplus length beyond y is stochastically larger than the original interval size, i.e., $P(X > x + y|X > y) \geq P(X > x)$ for all $x, y \geq 0$. While many heavy-tailed distributions belong to NWU, two most common examples are Pareto and Weibull. If the inequality is reversed, we obtain what is

known as NBU (new better than used). Examples include uniform and constant. Finally, if $P(X > x+y|X > y) = P(X > x)$, the distributions are called *memoryless* (i.e., exponential).

Suppose X is NWU, Y is memoryless, and Z is NBU such that $E[X] = E[Y] = E[Z]$. It then follows [30] that $Z \geq_{st}^{2} Y \geq_{st}^{2} X$. Applying this observation to Fig. 4.4(a), it is no wonder than Pareto D produces worse freshness than exponential, which in turn is worse than constant. For a fixed probability $p = 0.5$, the optimal case requires $33-50\%$ less bandwidth than the other two distributions. This relationship is reversed in application to U in Fig. 4.4(b) – Pareto is the best, while constant is the worst.

4.4.3 Optimality

From the discussion above, NBU download delays are better than NWU and exponential; however, it is unclear which NBU distribution is the best and whether other classes may be better than NBU. We are now ready to seek answers to these questions.

Lemma 6. *For a given mean, a constant stochastically dominates all other random variables in second order.*

Proof. Suppose l is the fixed mean of all distributions under consideration. Let $F_X(x) = \mathbf{1}_{x>l}$ be the CDF of a constant and $F_Y(x)$ be the CDF of another random variable Y such that $E[Y] = l$. Define:

$$\delta(x) := \int_0^x (\bar{F}_X(y) - \bar{F}_Y(y))dy \tag{4.11}$$

and observe that it suffices to prove that $\delta(x) \geq 0$ for all $x \geq 0$. For $x \leq l$, notice

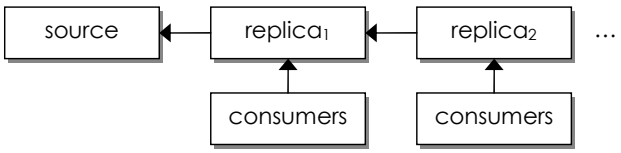


Figure 4.5: Cascaded replication at depth two.

that $\bar{F}_X(x) = 1$ and thus:

$$\delta(x) = \int_0^x F_Y(y)dy \geq 0. \tag{4.12}$$

For $x > l$:

$$\begin{aligned}\delta(x) &= \int_0^x \bar{F}_X(y)dy - \int_0^x \bar{F}_Y(y)dy \\ &= l - \int_0^x \bar{F}_Y(y)dy \geq l - \int_0^\infty \bar{F}_Y(y)dy = 0,\end{aligned}$$

where we use the fact that $\int_0^\infty \bar{F}_Y(y)dy$ equals the mean $E[Y]$ of a non-negative random variable Y. □

This leads to the main result of this section.

Theorem 25. *For a fixed download rate λ, constant inter-synchronization delays are optimal under all N_U.*

4.5 Cascaded Replication

We begin this section by introducing our cascaded model and show that freshness at each level can be determined if we know the residual distribution of ON cycles of $\phi(t)$. We then derive a recursive formula for the freshness at each layer i and finish this section with simulations and discussion.

4.5.1 Objectives

Motivated by the fact that replication trees are a common mechanism to scale the system to a large number of clients, we next consider a *cascaded model* in which caches at level i download content from those at level $i-1$. As before, replicas operate independently of the source and each other. As illustrated in Fig. 4.5, it suffices to consider a single branch of the tree that starts from the source (at level 0) and traverses towards the leaves.

We assume that each level i of the tree operates using inter-download delays $D^{(i)} \sim F_D^{(i)}(x)$ and has its own freshness process $\phi_i(t)$. This allows nodes near the root to synchronize faster or slower than those near the leaves (e.g., due to bandwidth constraints of the source or other reasons). Our task is to derive the average freshness p_i for queries directed towards replicas at depth i. Unlike (4.7), which is a simple function of two ages, there is no obvious way (yet) to express how p_i depends on the parameters of the system. The next subsection builds enough results to perform just that.

4.5.2 Freshness Residuals

For now, assume the single-layer case. Given the freshness process $\phi(t)$ in Fig. 4.6, define the ON durations (i.e., periods when $\phi(t) = 1$) to be given by some variable V. Note that $\phi(t)$ transitions from OFF to ON upon the first download following an update. It similarly goes from ON to OFF at the first update following a download. At time t, define the age $A_V(t)$ and residual $R_V(t)$ as the backward and forward delays, respectively, to the end of the ON segment.

Our later sections will require the distribution of $R_V(t)$, which is our focus here. Let the residual of the update process at time t be the interval from t to the next

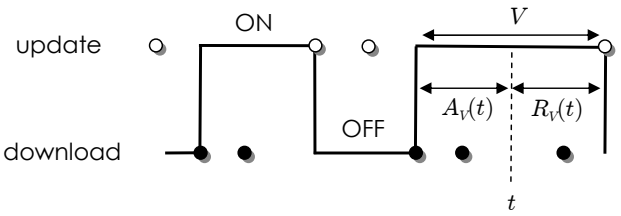


Figure 4.6: The age and residual of V in process $\phi(t)$.

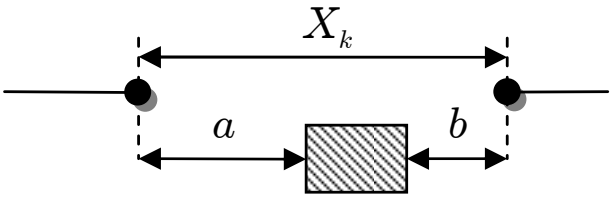


Figure 4.7: Visualizing the proof of Lemma 7.

update event:

$$R_U(t) = u_{N_U(t)+1} - t. \tag{4.13}$$

When t lands in an ON cycle, Fig. 4.6 shows that $R_V(t)$ is the same with $R_U(t)$, which yields:

$$\begin{aligned} P(R_V(t) < x) &= P(R_U(t) < x | \phi(t) = 1) \\ &= P(R_U(t) < x | A_U(t) > A_D(t)). \end{aligned} \tag{4.14}$$

This is a subtle point, but $R_V(t)$ and $R_U(t)$ have different distributions, unless U is exponential. Conditioning on the ON state of $\phi(t)$ introduces bias into $R_V(t)$, which in certain cases makes it stochastically larger than update residuals $R_U(t)$ and at other times smaller (see below). In order to simplify (4.14), we need the following lemma.

Lemma 7. *Consider a renewal process with interval lengths $X \sim F_X(x)$. As $t \to \infty$, the probability that the age of this process at time t is greater than a and simultaneously the residual is greater than b is:*

$$\lim_{t\to\infty} P(A_X(t) > a, R_X(t) > b) = \bar{G}_X(a+b). \tag{4.15}$$

Proof. Without loss of generality, assume that t lands on the k-th renewal cycle. As shown in Fig. 4.7, we can assign a random reward to this cycle equal to $S_k = \max\{X_k - a - b, 0\}$ (i.e., length of the shaded box) and use the renewal-reward theorem to obtain:

$$\lim_{t\to\infty} P(A_X(t) > y, R_X(t) > x) = \frac{E[S_k]}{E[X]}. \tag{4.16}$$

Using integration by parts:

$$\begin{aligned} E[S_k] &= \int_0^\infty \max\{z-a-b,0\} dF_X(z) \\ &= -\int_{a+b}^\infty (z-a-b) d\bar{F}_X(z) = \int_{a+b}^\infty \bar{F}_X(z) dz \\ &= E[X]\bar{G}_X(a+b), \end{aligned} \tag{4.17}$$

which immediately produces the desired result. □

Armed with Lemma 7, we are ready to obtain the residual distribution of ON durations.

Theorem 26. *The residual distribution of V is given by:*

$$G_V(x) := P(R_V < x) = 1 - \frac{E[\bar{G}_U(A_D + x)]}{E[\bar{G}_U(A_D)]}. \tag{4.18}$$

Proof. Re-writing (4.14) and applying Lemma 7:

$$\begin{aligned}
\bar{G}_V(x) &= \lim_{t\to\infty} P(R_U(t) > x | A_U(t) > A_D(t)) \\
&= \frac{1}{p} \lim_{t\to\infty} P(R_U(t) > x, A_U(t) > A_D(t)) \\
&= \frac{1}{p} \int_0^\infty g_D(y) \lim_{t\to\infty} P(R_U(t) > x, A_U(t) > y) dy \\
&= \frac{1}{p} \int_0^\infty g_D(y) \bar{G}_U(x+y) dy. \qquad (4.19)
\end{aligned}$$

Recalling (4.8) and collapsing the integral, we get (4.18). □

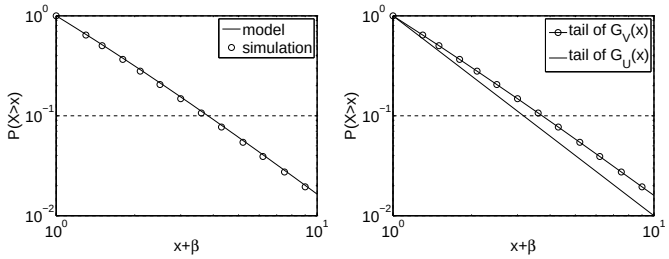
Fig. 4.8(a) shows that the model matches simulations very accurately under Pareto U. As mentioned earlier, when the update distribution is exponential, i.e., $F_U(x) = G_U(x) = 1 - e^{-\mu x}$, from (4.18) we get:

$$G_V(x) = 1 - \frac{E[e^{-\mu(A_D+x)}]}{E[e^{-\mu A_D}]} = 1 - e^{-\mu x}, \qquad (4.20)$$

which indicates that $G_V(x)$ remains exponential with the same rate μ. However, this is not true for non-exponential cases. To see this, we plot in Fig. 4.8(b) the tails of R_U and R_V for Pareto U. Observe in the figure that the latter is more heavy-tailed than the former. In fact, it can be shown that $R_V \geq_{st} R_U$ for NWU update distributions and the opposite for NBU.

Recalling (4.5)-(4.6), analysis above allows easy access to the CDF of ON durations:

$$F_V(x) := P(V < x) = 1 - \frac{g_V(x)}{g_V(0)} = 1 - \frac{E[g_U(A_D + x)]}{E[g_U(A_D)]}$$



(a) simulation of (4.18) (b) comparison of tails

Figure 4.8: Residual distribution $G_V(x)$ under Pareto U ($\alpha = 3, \beta = 1$) and exponential D ($\lambda = 2$).

and the average amount of time the replica stays fresh:

$$E[V] = \frac{1}{g_V(0)} = \frac{E[\bar{G}_U(A_D)]}{E[g_U(A_D)]}. \tag{4.21}$$

Denote by random variable W the OFF durations and let R_W be its residual. We can obtain similar results for the OFF durations.

Corollary 3. *The residual distribution of W given by:*

$$G_W(x) := P(R_W < x) = 1 - \frac{E[\bar{G}_D(A_U + x)]}{1 - p}. \tag{4.22}$$

This leads to the CDF of OFF durations:

$$F_W(x) : = P(W < x) = 1 - \frac{g_W(x)}{g_W(0)} = 1 - \frac{E[g_D(A_U + x)]}{E[g_D(A_U)]}$$

and its average:

$$E[W] = \frac{1}{g_W(0)} = \frac{E[\bar{G}_D(A_U)]}{E[g_D(A_U)]}. \tag{4.23}$$

Notice that the numerators in (4.21) and (4.23) can be simplified to:

$$E[\bar{G}_U(A_D)] = \int_0^\infty \bar{G}_U(x) g_D(x) dx$$
$$= P(A_U > A_D) = p \tag{4.24}$$

and

$$E[\bar{G}_D(A_U)] = \int_0^\infty \bar{G}_D(x) g_U(x) dx$$
$$= P(A_U < A_D) = 1 - p, \tag{4.25}$$

respectively.

The denominators in (4.21) and (4.23) have the same value:

$$E[g_U(A_D)] = E[g_D(A_U)]$$
$$= \int_0^\infty (1 - g_D(y))(1 - g_U(y)) dy. \tag{4.26}$$

Then we can get the time fraction that the replica is fresh:

$$\frac{E[V]}{E[V] + E[W]} = p. \tag{4.27}$$

This provides us another way to get freshness p by applying the renewal reward theorem [103] to the alternating renewal process $\phi(t)$, where the renewal cycle is $V + W$ and the reward at each cycle is V.

4.5.3 Cascaded Freshness

We now return to the main problem of this section and reactivate usage of sub/super-scripts i to denote the depth of the replica in the tree. As illustrated

in Fig. 4.9, the copy at level i is fresh at time t if and only if the copy at level $i-1$ is fresh and the download age $A_D^{(i)}(t)$ is smaller than the current age $A_V^{(i-1)}$ of the ON duration at level $i-1$:

$$\phi_i(t) = \begin{cases} 1 & A_V^{(i-1)}(t) > A_D^{(i)}(t), \phi_{i-1}(t) = 1 \\ 0 & \text{otherwise} \end{cases}. \tag{4.28}$$

Our first result allows p_i to be expressed as a function of the residual ON duration within the previous level $i-1$.

Lemma 8. *The expected freshness at depth i is given by:*

$$p_i = E[r_{i-1}(A_D^{(i)})], \tag{4.29}$$

where

$$r_i(y) = \lim_{t\to\infty} P(R_V^{(i)}(t) > y, \phi_i(t) = 1). \tag{4.30}$$

Proof. Using (4.28) and recalling from renewal theory that $A_V^{(i-1)}(t)$ has the same distribution as $R_V^{(i-1)}(t)$:

$$\begin{aligned} p_i &= \lim_{t\to\infty} P(\phi_i(t) = 1) \\ &= \lim_{t\to\infty} \int_0^\infty P(R_V^{(i-1)}(t) > y, \phi_{i-1}(t) = 1) g_D^{(i)}(y) dy \\ &= \int_0^\infty r_{i-1}(y) g_D^{(i)}(y) dy = E[r_{i-1}(A_D^{(i)})], \end{aligned} \tag{4.31}$$

where $r_i(y)$ was given earlier in (4.30). □

Our next step is to recursively expand $r_i(y)$.

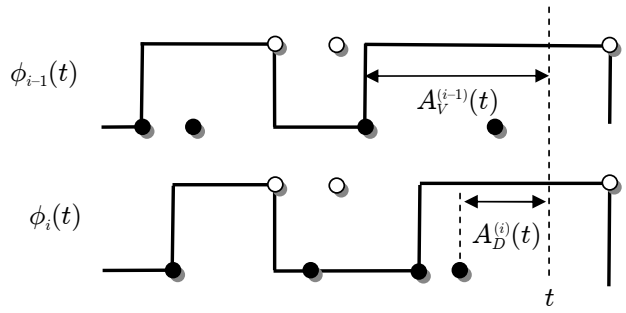


Figure 4.9: Processes $\phi_{i-1}(t)$ and $\phi_i(t)$ in cascaded replication.

Lemma 9. *For all $i \geq 1$:*

$$r_i(y) = E\Big[\bar{G}_U\Big(y + \sum_{k=1}^{i} A_D^{(k)}\Big)\Big]. \tag{4.32}$$

Proof. First, observe from Fig. 4.9 that residual ON durations at levels i and $i-1$ are the same as long as the i-th replica is fresh:

$$P(R_V^{(i)}(t) > y, \phi_i(t) = 1) = P(R_V^{(i-1)}(t) > y, \phi_i(t) = 1).$$

On the right-hand side of this result, expanding event $\phi_i(t) = 1$ using (4.28) and applying Lemma 7, we get:

$$\begin{aligned} &P(R_V^{i}(t) > y, \phi_i(t) = 1) \\ &= P(R_V^{(i-1)}(t) > y, A_V^{(i-1)}(t) > A_D^{(i)}(t), \phi_{i-1}(t) = 1) \\ &= P(R_V^{(i-1)}(t) > y + A_D^{(i)}(t), \phi_{i-1}(t) = 1). \end{aligned} \tag{4.33}$$

Letting $t \to \infty$ and closely examining the last equation, notice that it provides a

recursive formula on $r_i(y)$:

$$r_i(y) = r_{i-1}(y + A_D^{(i)}) = r_1\Big(y + \sum_{k=2}^{i} A_D^{(k)}\Big), \tag{4.34}$$

where the last result is obtained by repeatedly expanding $r_{i-1}(.)$. Since $r_1(y) = P(R_V > y)p$, we can combine (4.18) and (4.8) to obtain:

$$r_1(y) = E[\bar{G}_U(y + A_D^{(1)})]. \tag{4.35}$$

Merging (4.34)-(4.35), we immediately get (4.32). □

This leads to our main result.

Theorem 27. *The probability of freshness at depth i is:*

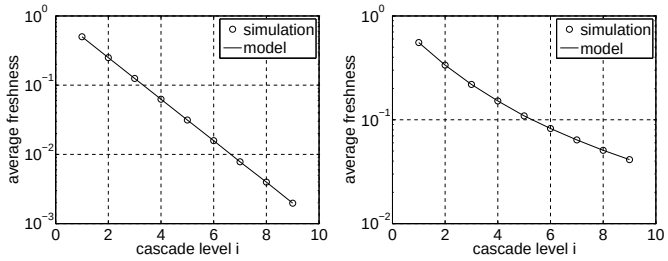
$$p_i = E[\bar{G}_U(Q_i)], \tag{4.36}$$

where $Q_i = A_D^{(1)} + A_D^{(2)} + \ldots + A_D^{(i)}$.

To perform a self-check, notice that p_1 in (4.36) reduces to p in (4.8). For $i \geq 2$, the model is also quite simple – it says that the freshness at level i is given by that of a single-layer system in which the download process N_D operates using intervals D whose age A_D is the summation of ages at levels $1, 2, \ldots, i$. For example, using constant $D^{(i)} = d$, each of the ages $A_D^{(i)}$ is uniform in $[0, d]$. For non-trivial i, their convolution Q_i is approximately Gaussian with mean $id/2$.

4.5.4 Discussion

Analyzing (4.36), first notice that it makes no difference in which order the replicas form the chain – freshness p_i only depends on the summation $Q_i = \sum_{k=1}^{i} A_D^{(k)}$.



(a) exponential U (b) Pareto U

Figure 4.10: Cascaded freshness with exponential D and $\lambda = \mu = 2$.

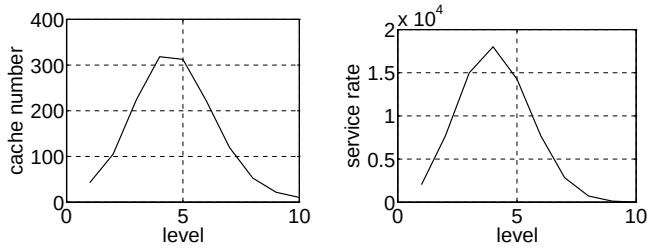
Therefore, placing high-rate download processes towards the top of the tree and slow towards the bottom produces exactly the same freshness as doing vice versa. Keeping source overload in mind, the best strategy may then be to design trees with high branching factors b (i.e., low depth) and slow λ near the root, placing faster processes towards the leaves.

When the update process at the source is Poisson, i.e., $G_U = 1 - e^{-\mu x}$, the tail CDF of U decomposes into a product:

$$p_i = \prod_{k=1}^{i} E\left[\bar{G}_U(A_D^{(k)})\right] = \prod_{k=1}^{i} p(A_D^{(i)}), \tag{4.37}$$

where $p(A_D^{(i)})$ is the freshness probability of $N_D^{(i)}$ working directly with the source. Therefore, if we know that replicas A and B separately achieve freshness p_A and p_B, their cascaded performance will produce freshness $p_A p_B$ at level 2. If additionally the download processes are all homogeneous, i.e., $F_D^{(i)}(x) = F_D^{(j)}(x)$ for all (i, j, x), the freshness value p_i is an exponentiation of the single-step model (4.8).

Note that multiplicative reduction in p_i as a function of i presents an interesting tradeoff – as the tree size scales exponentially up, freshness scales exponentially down. In order to prevent $p_i \to 0$, one must increase the download rate λ_i at depth i



(a) cache server number (b) service rate

Figure 4.11: Cache server number and service rate under cascaded caching (exponential U with $\mu = 2$ and $1 - \epsilon = 0.7$).

such that $\sum_{i=1}^{\infty} \log p(A_D^{(i)}) > -\infty$. One of the slowest growing functions that satisfies this condition is $p(A_D^{(i)}) = 1 - (i+1)^{-\rho}$, where ρ is slightly larger than 1. Using exponential D as an example, we get $p(A_D^{(i)}) = \lambda_i/(\lambda_i + \mu)$, which translates into $\lambda_i = \mu((i+1)^{\rho} - 1)$, showing that bandwidth requirements *must scale super-linearly, but at least not exponentially*, with i.

We next use simulations with homogeneous download delays to compare the decay rate in p_i for exponential and Pareto U. Fig. 4.10 shows that model (4.36) is quite accurate and that the Pareto case in (b) decreases much slower than the exponential in (a). This suggests that NWU distributions of U are easier to scale than either exponential or NBU. This can be confirmed by noticing that the heavier the tail $\bar{G}_U(x)$, the slower p_i decays in (4.36).

4.5.5 System

A real system needs to serve its clients under certain freshness level. Denote by $1 - \epsilon$ the freshness threshold that the system wants guarantee. Assume the all the servers in the network is homogeneous with same bandwidth B. Now we study the effects of cascades level i on the system service rate and the number caching servers.

From the discussion above, it seems beneficial to put all replicas in the first level.

However, the maximum number of replicas m under this scenario is limited, which in turn puts constrains on the service rate s. Specifically, to achieve certain freshness ratio $1-\epsilon$, each replica has to download from the source with rate λ, which can be computed by inverting (4.8). Given fixed sources bandwidth B, the number of replicas in the first level m is limited to B/λ, which gives the service rate to the client $s = m(B-\lambda)$.

4.6 Cooperative Caching

In this section, we consider a novel cooperative model in which all replicas are at level 1, but they are allowed to communicate with each other. We first introduce the details of this configuration and the corresponding notation. We follow that up with analyzing parameter selection and formulating optimality conditions that lead to best freshness.

4.6.1 Model and Notation

As shown in Fig. 4.12, suppose there are m replicas in the system that form a cluster. As before, $N_D^{(i)}$ is the download process between each replica i and the source; however, we now additionally assume that $N_C^{(i)}$ represents the *communication* process that each replica uses to poll other nodes within the cluster. At each point of $N_C^{(i)}$, the node contacts k other peers and selects the freshest response for download. All processes are renewal and independent of each other. Let $C \sim F_C(x)$ describe the length of cycles in each communication process and $\nu = 1/E[C]$ be its rate.

In order for a replica to cooperate with others, it has to know their location. One option is to require that the source maintain a replica list ordered by the most-recent download timestamp. This list can then be disseminated to replicas upon each contact with the source. In order to choose k peers among m, we consider two strategies: *random* and *recent*. The former selects k peers in m uniformly randomly

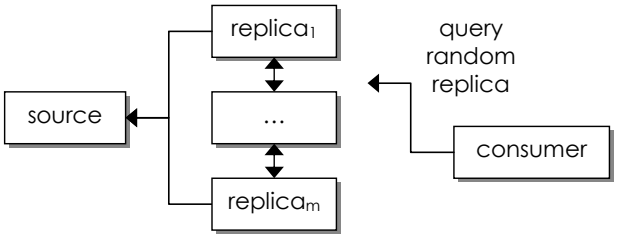


Figure 4.12: Cooperative replication.

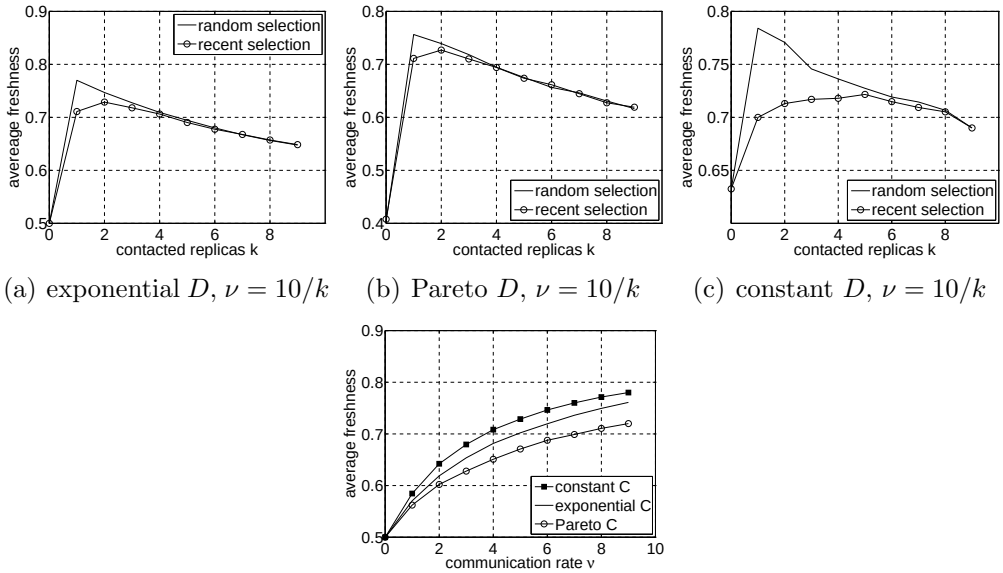
(assuming global knowledge or other mechanisms) and the latter selects those with the largest contact timestamps.

4.6.2 Simple Scenario

With a fixed per-node intra-cluster communication bandwidth $k\nu$, the first question is how to choose k and ν such that they provide the highest freshness. With cooperation, closed-form derivations are difficult because downloads follow random cascaded chains (i.e., $s \to x_1 \to x_2 \to \ldots$ where s signifies the source and x_k the k-th replica ID along this chain). We therefore use simulations to study this problem.

For random selection, observe from Fig. 4.13(a)-(c) that the expected freshness p achieves a global maximum at $k = 1$, which is noticeably higher than in non-cooperative cases (i.e., $k = 0$). Freshness then monotonically decreases as the number of contacted nodes k increases. This can be explained by the fact that the replicas' freshness processes $\{\phi_i(t)\}_{t\geq 0}$ are highly correlated, i.e., an update at time t makes all of them transition from ON to OFF. Thus, the benefit of contacting $k \geq 2$ peers at lower rate ν is smaller than contacting one peer with higher ν.

Interestingly, recent selection in Fig. 4.13(a)-(c) peaks at later points $k \geq 2$, but then succumbs to the same effect. In fact, as k gets larger, the most-recent list becomes essentially composed of random nodes and both methods converge. In all studied cases, random selection beats recent. The intuition is that the latter is biased



(a) exponential D, $\nu = 10/k$ (b) Pareto D, $\nu = 10/k$ (c) constant D, $\nu = 10/k$

(d) random selection $k = 1$

Figure 4.13: Effect of k and ν in cooperative replication ($\mu = \lambda = 1, m = 10$). Exponential U and C.

towards certain peers that were the most up-to-date at time d_k, but are no longer the freshest by the next download instance d_{k+1}. Instead, results show that a random peer has fresher information when we average over the entire interval $[d_k, d_{k+1}]$. This indicates that *the optimal strategy is to choose exactly one uniformly random peer (i.e., $k = 1$) and hence keep ν as large as possible.*

4.6.3 Convergence of Freshness

We now fix $k = 1$ and vary ν in Fig. 4.13(d) to analyze the effect of C and ν. As before, NBU synchronization performs the best, followed by exponential and NWU. Reasoning similar to that used in previous sections suggests that constant C remains optimal for cooperative caching. In the figure, p starts at 0.5 and gradually improves to $0.72 - 0.78$ depending on the distribution of C; however, what happens to freshness as intra-cluster bandwidth ν becomes very large? We address this next.

Theorem 28. *As $\nu \to \infty$, freshness under random selection equals that computed using the original update process N_U and a superposition N_D^* of m download processes $\{N_D^{(i)}\}_{i=1}^m$.*

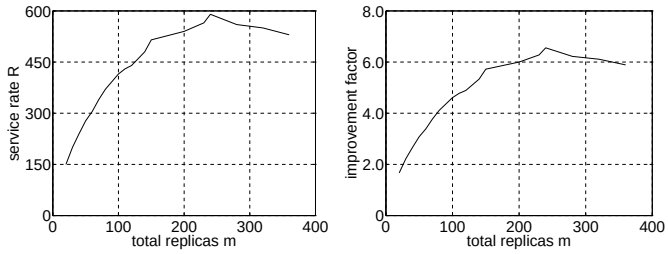
Note that this result holds for all $k \geq 1$. We are now able to compute the limiting freshness probability for the case in Fig. 4.13(d) with exponential C. Since a superposition of Poisson process remains Poisson, we get that $p_\infty = m\lambda/(m\lambda + \mu) = 0.91$. As an alternative to raising ν, freshness can be increased by allowing *bidirectional* communication between replicas. When A requests updates from B, it can offer its own content for upload to B. This effectively doubles rate ν in our model, but keeps it fully applicable to this technique as well.

4.6.4 Full System

We now consider a more realistic cooperative replication system that has two conflicting goals – serving clients and maintaining freshness. The tradeoff arises from bandwidth consumption – larger ν leads to higher freshness, but reduces the ability of each node to answer consumer queries.

Let s be the service rate that each replica can offer to its clients and suppose that all peers have the same bandwidth constraint B (including the source). The system is considered usable if the average freshness is no smaller than $1 - \epsilon$, where $\epsilon > 0$ is some design parameter. Define $p(\lambda, k, \nu)$ to be the freshness probability achieved by a cluster using the source download rate λ, k-way cooperation, and intra-cluster synchronization rate ν. Then, the objective is to maximize the combined service rate $R := ms = m(B - \lambda - k\nu)$ subject to:

$$\begin{cases} m\lambda \leq B \\ p(\lambda, k, \nu) \geq 1 - \epsilon \end{cases}. \tag{4.38}$$



(a) service rate R (b) service rate ratio

Figure 4.14: Effect of m on service rate R ($\epsilon = 0.5, \mu = 1, B = 10$). All distributions are exponential.

Since all peers are homogenous, the source bandwidth should be shared equally, which means that the first line of (4.38) reduces to $m = B/\lambda$. For non-cooperative replication, both k and ν are zero and thus $R = B^2/\lambda - B$. The only unknown parameter is λ, which can be determined from the freshness condition of (4.38) as $\lambda = q^{-1}(1-\epsilon)$, where $q(x) = p(x, 0, 0)$. For example, with exponential U and D:

$$q(x) = \frac{x}{x+\mu} = 1 - \epsilon, \tag{4.39}$$

from which we get:

$$\lambda = \frac{\mu(1-\epsilon)}{\epsilon} \quad \text{and} \quad R = \frac{B^2\epsilon}{\mu(1-\epsilon)} - B. \tag{4.40}$$

For cooperative replication, the previous subsection shows that the optimal case is $k = 1$. Now suppose we fix m. This allows us to determine the remaining parameters of the system using the model above. Indeed, $\lambda = B/m$ and $\nu = q_2^{-1}(1-\epsilon)$, where $q_2(x) = p(\lambda, 1, x)$. To understand why there is an inherent tradeoff in this system, notice that larger m affords the system more combined bandwidth $R = m(B - \lambda - k\nu)$; however, this also leads to lower source-synchronization rate $\lambda = B/m$, which

Table 4.1: Optimal Service Rate Comparison Between Cooperative and Non-Cooperative Replication

$1-\epsilon$	Non-cooperative R	Cooperative R	Ratio
0.4	270	1865	6.9
0.5	90	590	6.6
0.6	50	134	2.7
0.7	33	42	1.3

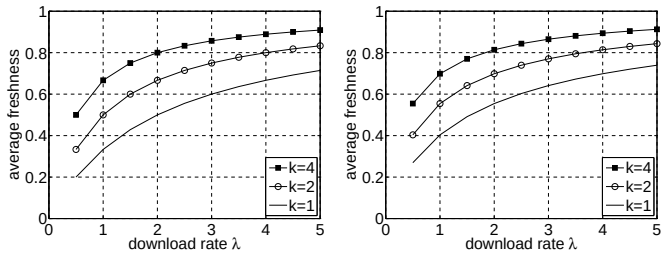
decreases freshness. To compensate for lower freshness, replicas must communicate with other peers at a higher rate ν, which in turn lowers R.

Thus, there must be an optimal m that maximizes the service rate for a given set (ϵ, B, N_U). To demonstrate this, we plot simulation results in Fig. 4.14(a), where R hits a peak at $m = 240$ and then drops monotonically. The improvement ratio between cooperative and non-cooperative R in Fig. 4.14(b) reaches 4 by $m = 90$ and 6 by $m = 200$. Table 4.1 shows additional scenarios. Observe that cooperative replication is more effective when the target freshness is smaller. The benefit decreases as $\epsilon \to 0$ since it becomes progressively more difficult to find peers with exceedingly fresh copies.

4.7 Redundant Querying

To obtain fresher results, existing work [6] suggests that consumers contact multiple replicas. As illustrated in Fig. 4.16, this model uses a single-layer non-cooperative caching with redundant queries to achieve higher robustness against staleness. Contacting $k \geq 2$ random replicas and retrieving the freshest copy, consumers effectively replace a single download process N_D with a superposition N_D^* of k processes $\{N_D^{(i)}\}_{i=1}^k$. This observation is similar to Theorem 28 except the number of superposed processes is k rather than m.

To understand this better, suppose the download processes in replicas are inde-



(a) exponential U (b) Pareto U

Figure 4.15: Redundant querying with exponential D.

pendent Poisson with the same rate λ. Then the superposition process N_D^* is also Poisson, but with rate $k\lambda$. Improvement is shown in Fig. 4.15, where growth in k from 1 to 4 yields an increase in p from 0.5 to 0.8 for exponential U and from 0.45 to 0.75 for Pareto U.

4.7.1 Analysis

As before, suppose $1 - \epsilon$ is the target freshness at the user and $p(\lambda)$ is that achieved by a non-redundant system (i.e., using $k = 1$). Then, recalling the previous section, there is a unique download rate $\lambda = p^{-1}(1 - \epsilon)$ that determines the optimal cluster size $m = B/\lambda$ and the combined service rate:

$$R = m(B - \lambda) = \frac{B^2}{\lambda} - B. \tag{4.41}$$

We now contrast this with a redundant configuration. Define λ' to be the download rate of each replica and $\lambda^* = k\lambda'$ to be that of the superposed process N_D^*. Since freshness is now determined by $\lambda^* = p_*^{-1}(1 - \epsilon)$, where $p_*(x)$ is the freshness probability under N_D^*, we can lower each λ' and hopefully increase system size $m^* = B/\lambda' = Bk/\lambda^*$ beyond m. However, the main caveat is that each replica now serves k times more traffic to the clients, which means that the best possible query

rate of the new system is:

$$R^* = \frac{m^*(B-\lambda')}{k} = \frac{B^2}{\lambda^*} - \frac{B}{k}. \tag{4.42}$$

4.7.2 Discussion

In the simplest case of exponential D, both N_D and N_D^* are Poisson, which immediately leads to $\lambda^* = \lambda$ since functions $p(x)$ and $p_*(x)$ are the same. This shows that the redundant system can support exactly $m^*/m = k$ times more servers, but the service-rate ratio R^*/R stays pretty close to 1, i.e., there is virtually no benefit. To tackle more complex cases, assume k is sufficiently large, in which case the superposition process N_D^* tends to Poisson from the Palm-Khintchine theorem [68]. From (4.40), we know that $\lambda^* = \mu(1-\epsilon)/\epsilon$, which leads to a closed-form service rate:

$$R^* = \frac{B^2\epsilon}{\mu(1-\epsilon)} - \frac{B}{k}. \tag{4.43}$$

If D has an NWU distribution, k-way aggregation makes D^* stochastically larger in second order and thus improves freshness (see the discussion in Section 4.4.2). Counter-intuitively, however, if D has an NBU distribution, transition to a Poisson N_D^* *makes the redundant case perform worse than the non-redundant.* These effects are shown in Table 4.2 for $k = 20$ and $B = 1000$. As target freshness increases, the Pareto case gradually improves and finishes with rates 110% above those in the non-redundant scenario. The opposite trend occurs with constant D, which gradually becomes worse and ends up losing 44% of service capacity in the last row.

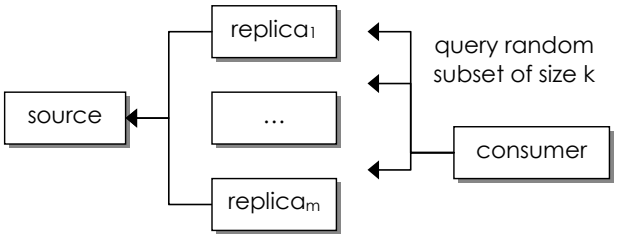


Figure 4.16: Redundant querying with $k = 3$.

Table 4.2: Improvement From Redundant Querying Compared to Non-Redundant

$1-\epsilon$	Pareto D		exponential D		constant D	
	m^*/m	R^*/R	m^*/m	R^*/R	m^*/m	R^*/R
0.1	21.8	1.1	20	1	18.0	0.90
0.3	26.6	1.3	20	1	14.6	0.73
0.5	32.8	1.6	20	1	12.5	0.63
0.7	42.3	2.1	20	1	11.2	0.56

5. SUMMARY AND FUTURE WORK

5.1 Summary

This work focus on studying staleness in pull-based data synchronization systems. Our work can be partitioned to four topics.

We first introduced a novel model of sampled age under general non-Poisson update/synchronization processes and applied it to obtain many useful metrics of staleness. We additionally established that constant inter-refresh intervals were optimal for all considered cases and provided guidelines for achieving ASTA even in those cases. We finally considered a family of related problems stemming from $1 \times m$ and $M \times 1$ replication, showing that they can be easily solved from the preceding analysis of the 1×1 case.

Second, we studied the problem of estimating the update distribution at a remote source under blind sampling. We analyzed prior approaches in this area, showed them to be biased under general conditions, introduced novel modeling techniques for handling these types of problems, and proposed several unbiased algorithms that tackled network sampling under a variety of assumptions on the information provided by the server and conditions at the observer.

Third, we considered the data staleness in the context of P2P replication networks. We extended our results to cascaded and cooperative replication, finding solutions to a number of optimization problems in those contexts. We also examined redundant querying and found cases when doing so was detrimental to system performance.

5.2 Future Work

We assume that download process is age-independent of the update process, which might not hold in certain systems. For example, some existing systems [60, 75, 80, 87, 86, 94, 93, 97] choose to increase the next download interval if an update is detected in the current download cycle and decrease otherwise. When should these adaptive strategy should be chosen and what's their performance?

There is another potential way to break the age-independence assumption and gain better performance. Specifically, the update process is non-stationary [84] and exhibit diurnal patterns while the download process can synchronize more at daytime than at night to capture more updates. Under this scenario, how to measure and model the update process? How much performance gain can one in this case compared to the constant download strategy.

Given age-independent assumption, the resource allocation problem can be solved in closed form. How to choose the adaptation strategy for each page given fixed download budget? Can we solve the problem using stochastic control theory? For non-stationary update process, how to solve the resource allocation problem?

We only consider that the update history is available in this work. However, we might have more information in certain applications. For example, pages in search engine have file type, file size, content and link to its neighbors. Can we use these information to improve the download performance?

REFERENCES

[1] Eytan Adar, Jaime Teevan, Susan T. Dumais, and Jonathan L. Elsas. The web changes everything: Understanding the dynamics of web content. In *WSDM*, pages 282–291, Jun. 2009.

[2] S. L. Albin. On poisson approximations for superposition arrival processes in queues. *Management Sci.*, 28(2):126–137, 1982.

[3] R. Arratia, L. Goldstein, and L. Gordon. Two moments suffice for poisson approximations: The chen-stein method. *The Annals of Probability*, 17(1):9–25, Jan. 1989.

[4] Robert B. Ash and Catherine A. Doleans-Dade. *Probability & Measure Theory*. Academic Press, 2 edition, 1999.

[5] F. Baccelli, S. Machiraju, D. Veitch, and J. Bolot. The Role of PASTA in Network Measurement. In *ACM SIGCOMM*, Sep. 2006.

[6] Peter Bailis, Shivaram Venkataraman, Michael J. Franklin, Joseph M. Hellerstein, and Ion Stoica. Probabilistically bounded staleness for practical partial quorums. *VLDB Endow.*, 5(8):776–787, Apr. 2012.

[7] Brian E. Brewington and George Cybenko. How dynamic is the web. *Computer Networks*, (1-6):257–276, Jun. 2000.

[8] Laura Bright, Avigdor Gal, and Louiqa Raschid. Adaptive pull-based policies for wide area data delivery. *ACM Trans. Database Syst.*, 31(2):631–671, Jun. 2006.

[9] F. E. Bustamante and Y. Qiao. Friendships that last: Peer lifespan and its role in p2p protocols. In *Web Content Caching and Distribution*, Sep. 2003.

[10] Donald Carney, Sangdon Lee, and Stan Zdonik. Scalable application-aware data freshening. In *IEEE ICDE*, pages 481–492, Mar. 2003.

[11] M. Castro, P. Druschel, A. M. Kermarrec, and A. I.T. Rowstron. Scribe: A large-scale and decentralized application-level multicast infrastructure. *IEEE JSAC*, 20(8):1489–1499, Sep. 2006.

[12] Badrish Chandramouli and Jun Yang. End-to-end support for joins in large-scale publish/subscribe systems. *VLDB Endow.*, 1(1):434–450, Aug. 2008.

[13] Xin Chen, Shansi Ren, Haining Wang, and Xiaodong Zhang. Scope: Scalable consistency maintenance in structured p2p systems. In *IEEE INFOCOM*, pages 1502–1513, Mar. 2005.

[14] Junghoo Cho and Hector Garcia-Molina. The evolution of the web and implications for an incremental crawler. In *VLDB*, pages 200–209, Sep. 2000.

[15] Junghoo Cho and Hector Garcia-Molina. Synchronizing a database to improve freshness. In *ACM SIGMOD*, pages 117–128, May 2000.

[16] Junghoo Cho and Hector Garcia-Molina. Effective page refresh policies for web crawlers. *ACM Transactions on Database Systems*, (4):1–36, Dec. 2003.

[17] Junghoo Cho and Hector Garcia-Molina. Estimating frequency of change. *ACM Trans. Internet Technol.*, 3(3):256–290, Aug. 2003.

[18] Giovanni Luca Ciampaglia and Alberto Vancheri. Empirical analysis of user participation in online communities: the case of wikipedia. In *ICWSM*, pages 219–222, Sep. 2010.

[19] Ian Clarke, Oskar Sandberg, Brandon Wiley, and Theodore W. Hong. Freenet: A distributed anonymous information storage and retrieval system. In *ICSI*

Workshop on Design Issues in Anonymity and Unobservability, pages 46–66, Jul. 2000.

[20] E. G. Coffman, Zhen Liu, and Richard R. Weber. Optimal robot scheduling for web search engines. *Journal of Scheduling*, (1):15–29, Jun. 1998.

[21] Edith Cohen and Haim Kaplan. The age penalty and its effect on cache performance. In *USENIX USITS*, pages 73–84, Mar. 2001.

[22] Edith Cohen and Haim Kaplan. Aging through cascaded caches: Performance issues in the distribution of web content. In *ACM SIGCOMM*, pages 41–53, Aug. 2001.

[23] Edith Cohen and Haim Kaplan. Refreshment policies for web content caches. In *IEEE INFOCOM*, pages 1398–1406, Apr. 2001.

[24] Russ Cox, Athicha Muthitacharoen, and Robert Morris. Serving dns using a peer-to-peer lookup service. In *IPTPS*, pages 155–165, Mar. 2002.

[25] M. E. Crovella and A. Bestavros. Self-similarity in world wide web traffic: Evidence and possible causes. 5(6):835–846, 1997.

[26] Frank Dabek, M. Frans Kaashoek, David Karger, Robert Morris, and Ion Stoica. Wide-area cooperative storage with cfs. In *ACM SOSP*, pages 188–201, Oct. 2001.

[27] Anwitaman Datta, Manfred Hauswirth, and Karl Aberer. Updates in highly unreliable, replicated peer-to-peer systems. In *IEEE ICDCS*, pages 76–85, May 2003.

[28] Jeff Dean and Sanjay Ghemawat. Mapreduce: Simplified data processing on large clusters. In *USENIX OSDI*, pages 137–150, Dec. 2004.

[29] Dimitar Denev, Arturas Mazeika, Marc Spaniol, and Gerhard Weikum. Sharc: Framework for quality-conscious web archiving. In *VLDB*, pages 183–207, Aug. 2009.

[30] Jayant V. Deshpande, Subhash C. Kochar, and Harshinder Singh. Aspects of positive ageing. *J. Applied Probability*, 23(3):748–758, Sep. 1986.

[31] Debabrata Dey, Zhongju Zhang, and Prabuddha De. Optimal synchronization policies for data warehouses. *INFORMS J. on Computing*, (2):229–242, Jan. 2006.

[32] Yanlei Diao, Shariq Rizvi, and Michael J. Franklin. Towards an internet-scale xml dissemination service. In *VLDB*, pages 612–623, Aug. 2004.

[33] N. Duffield, C. Lund, and M. Thorup. Estimating flow distributions from sampled flow statistics. In *ACM SIGCOMM*, pages 325–336, Aug. 2003.

[34] Jonathan Eckstein, Avigdor Gal, and Sarit Reiner. Monitoring an information source under a politeness constraint. *INFORMS J. on Computing*, (1):3–20, Jan. 2008.

[35] Jenny Edwards, Kevin McCurley, and John Tomlin. An adaptive model for optimizing performance of an incremental web crawler. In *WWW*, pages 106–113, May 2001.

[36] Dennis Fetterly, Mark Manasse, and Marc Najork. On the evolution of clusters of near-duplicate web pages. In *LA-WEB*, pages 37–45, Nov. 2003.

[37] Dennis Fetterly, Mark Manasse, Marc Najork, and Janet Wiener. A large-scale study of the evolution of web pages. In *WWW*, pages 669–678, May 2003.

[38] Patrick M. Fitzpatrick. *Advanced Calculus*. Thomson Brooks/Cole, 2 edition, 2006.

[39] Marcus Fontoura, Mihail Ionescu, and Naftaly Minsky. Law-governed peer-to-peer auctions. In *WWW*, pages 109–116, May 2002.

[40] Avigdor Gal and Jonathan Eckstein. Managing periodically updated data in relational databases: A stochastic modeling approach. *J. ACM*, (6):1141–1183, Nov. 2001.

[41] Gnutella.

[42] Vijay Gopalakrishnan, Bujor Silaghi, Bobby Bhattacharjee, and Pete Keleher. Peer-to-peer caching schemes to address flash crowds. In *IEEE ICDCS*, pages 360–369, Mar. 2004.

[43] Carrie Grimes and Daniel Ford. Estimation of web page change rates. In *JSM*, 2008.

[44] Carrie Grimes, Daniel Ford, and Eric Tassone. Keeping a search engine index fresh: Risk and optimality in estimating refresh rates for web pages. In *INTERFACE*, May 2008.

[45] D. Gruhl, L. Chavet, D. Gibson, J. Meyer, P. Pattanayak, A. Tomkins, and J. Zien. How to build a webfountain: An architecture for very large-scale text analytics. *IBM Systems Journal*, 43(1):64–77, 2004.

[46] D. Heyman and M. Sobel. *Stochastic Models in Operations Research, Volume 1.* McGraw-Hill, 1982.

[47] Roxana Horincar, Bernd Amann, and Thierry Artiëres. Online change estimation models for dynamic web resources. In *ICWE*, pages 395–410, Jul. 2012.

[48] Yi Hu, Min Feng, and Laxmi N. Bhuyan. A balanced consistency maintenance protocol for structured p2p systems. In *IEEE INFOCOM*, pages 286–290, Mar. 2010.

[49] Yixiu Huang, Robert H. Sloan, and Ouri Wolfson. Divergence caching in client-server architectures. In *IEEE PDIS*, pages 131–139, Sep. 1994.

[50] Internet Archive.

[51] Sitaram Iyer, Antony Rowstron, and Peter Druschel. Squirrel: A decentralized peer-to-peer web cache. In *IEEE PODC*, pages 213–222, Jul. 2002.

[52] Rivka Ladin, Barbara Liskov, Liuba Shrira, and Sanjay Ghemawat. Providing high availability using lazy replication. *ACM Trans. Comput. Syst.*, (4):360–391, Nov. 1992.

[53] Jeong-Joon Lee, Kyu-Young Whang, Byung Suk Lee, and Ji-Woong Chang. An update-risk based approach to ttl estimation in web caching. In *IEEE WISE*, pages 21–29, Dec. 2002.

[54] W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson. On the self-similar nature of ethernet traffic. In *ACM SIGCOMM*, pages 183–193.

[55] Xiaoyong Li, Daren B.H. Cline, and Dmitri Loguinov. On sample-path staleness in lazy data replication. In *IEEE INFOCOM*, pages 1104–1112, Apr. 2015.

[56] Xiaoyong Li and Dmitri Loguinov. Stochastic models of pull-based data replication in p2p systems. In *IEEE P2P*, pages 1–10, Sep. 2014.

[57] Zhenyu Li, Gaogang Xie, and Zhongcheng Li. Efficient and scalable consistency maintenance for heterogeneous peer-to-peer systems. *IEEE Trans. Parallel and Distributed Systems*, 19(12):1695–1708, Dec. 2008.

[58] Yibei Ling and Wai Chen. Measuring cache freshness by additive age. *SIGOPS Oper. Syst. Rev.*, 38:12–17, Jul. 2004.

[59] Yibei Ling and Jie Mi. An optimal trade-off between content freshness and refresh cost. *Applied Probability*, 41(3):721–734, Sep. 2004.

[60] Xiaotao Liu, Jiang Lan, Prashant Shenoy, and Krithi Ramaritham. Consistency maintenance in dynamic peer-to-peer overlay networks. *Comput. Netw.*, 50(6):859–876, Apr. 2006.

[61] Norman Matloff. Estimation of internet file-access/modification rates from indirect data. *ACM Trans. Model. Comput. Simul.*, 15:233–253, Jul. 2005.

[62] Benjamin Melamed and Ward Whitt. On arrivals that see time averages. *Operations Research*, 38(1):156–172, 1990.

[63] Alexandros Ntoulas, Junghoo Cho, and Christopher Olston. What's new on the web? the evolution of the web from a search engine perspective. In *WWW*, pages 1–12, May 2004.

[64] Marilena Oita and Pierre Senellart. Deriving dynamics of web pages: A survey. In *TWAW*, pages 25–32, Mar. 2011.

[65] Chris Olston and Jennifer Widom. Best-effort cache synchronization with source cooperation. In *ACM SIGMOD*, pages 73–84, May 2002.

[66] Christopher Olston and Sandeep Pandey. Recrawl scheduling based on information longevity. In *WWW*, pages 437–446, Apr. 2008.

[67] Venkata N. Padmanabhan and Lili Qiu. The content and access dynamics of a busy web site: Findings and implications. In *ACM SIGCOMM*, pages 111–123, Aug. 2000.

[68] K. Palm. Intensitätsschwankungen im fernsprechverkehr. *Ericsson Technics*, 44, 1943.

[69] Sandeep Pandey, Kedar Dhamdhere, and Christopher Olston. Wic: A general-purpose algorithm for monitoring web information sources. In *VLDB*, pages 360–371, Aug. 2004.

[70] Sandeep Pandey and Christopher Olston. User-centric web crawling. In *WWW*, pages 1–12, May 2005.

[71] Sandeep Pandey, Krithi Ramamritham, and Soumen Chakrabarti. Monitoring the dynamic web to respond to continuous queries. In *WWW*, pages 659–668, May 2003.

[72] Kihong Park, Gitae Kim, and Mark Crovella. On the relationship between file sizes, transport protocols, and self-similar network traffic. In *IEEE ICNP*, pages 171–180, Oct. 1996.

[73] Kira Radinsky and Paul N. Bennett. Predicting content change on the web. In *WSDM*, pages 415–424, Feb. 2013.

[74] S. Resnick. *A Probability Path*. Boston, MA: Birkhäuser, 1999.

[75] Haggai Roitman, David Carmel, and Elad Yom-Tov. Maintaining dynamic channel profiles on the web. *PVLDB*, 1(1):151–162, Aug. 2008.

[76] D. Roselli, J. R. Lorch, and T. E. Anderson. A comparison of file system workloads. In *USENIX Annual Technical Conference*, pages 41–54, Jun. 2000.

[77] Mema Roussopoulos and Mary Baker. Cup: Controlled update propagation in peer-to-peer networks. In *USENIX Annual Technical Conference*, pages 167–180, Apr. 2003.

[78] Antony Rowstron and Antony Rowstron. Storage management and caching in past, a large-scale, persistent peer-to-peer storage utility. In *ACM SOSP*, pages 188–201, Nov. 2001.

[79] S. Saroiu, P. K. Gummadi, and S. D. Gribble. A measurement study of peer-to-peer file sharing systems. In *SPIE/ACM Multimedia Computing and Networking*, volume 4673, pages 156–170, Jan. 2002.

[80] Haiying Shen. Irm: Integrated file replication and consistency maintenance in p2p systems. *IEEE Trans. Parallel Distrib. Syst.*, 21(1):100–113, Jan. 2010.

[81] Haiying Shen. Irm: Integrated file replication and consistency maintenance in p2p systems. *IEEE Trans. Parallel Distrib. Syst.*, 21(1):100–113, Jan. 2012.

[82] Haiying Shen and Guoxin Liu. A geographically aware poll-based distributed file consistency maintenance method for p2p systems. *IEEE Trans. Parallel Distrib. Syst.*, 24(11):2148–2159, Nov. 2013.

[83] Haiying Shen and Guoxin Liu. A lightweight and cooperative multifactor considered file replication method in structured p2p systems. *IEEE Trans. Comput.*, 62(11):2115–2130, Jan. 2013.

[84] Ka Cheung Sia and Junghoo Cho. Efficient monitoring algorithm for fast news alerts. *IEEE Trans. Knowl. Data Eng.*, (7):950–961, Jul. 2007.

[85] K. Sigman and R.W. Wolff. A review of regenerative processes. *SIAM Review*, 35(2):269–288, Jun. 1993.

[86] Kristinn Sigurosson. Adaptive revisiting with heritrix. Master's thesis, University of Iceland, May 2005.

[87] Kristinn Sigurosson. Incremental crawling with heritrix. In *IWAW*, Spet. 2005.

[88] Adam Silberstein, Jeff Terrace, Brian F. Cooper, and Raghu Ramakrishnan. Feeding frenzy: Selectively materializing users' event feeds. In *ACM SIGMOD*, pages 831–842, Jun. 2010.

[89] Sanasam Ranbir Singh. Estimation of web page change rates. In *International Joint Conferences on Artificial Intelligence*, pages 2874–2879, Jun. 2007.

[90] Marc Spaniol, Dimitar Denev, Arturas Mazeika, Gerhard Weikum, and Pierre Senellart. Data quality in web archiving. In *WICOW*, Apr. 2009.

[91] Tyron Stading, Petros Maniatis, and Mary Baker. Peer-to-peer caching schemes to address flash crowds. In *IPTPS*, pages 203–213, Mar. 2002.

[92] D. Stutzbach and R. Rejaie. Understanding churn in peer-to-peer networks. In *ACM IMC*, pages 189–202, Oct. 2006.

[93] R. Sundaresan, M. Lauria, T. Kurc, S. Parthasarathy, and Joel Saltz. Adaptive polling of grid resource monitors using a slacker coherence model. In *HDPC*, Jun. 2003.

[94] Radhakrishnan Sundaresan, Tahsin Kurc, Mario Lauria, Srinivasan Parthasarathy, and Joel Saltz. A slacker coherence rotocol for pull-based monitoring of on-line data sources. In *CCGRID*, Apr. 2003.

[95] Qingzhao Tan and Prasenjit Mitra. Clustering-based incremental web crawling. *ACM Transactions on Information Systems*, (4):1–27, Nov. 2010.

[96] Xueyan Tang, Jianliang. Xu, and Wang-Chien Lee. Analysis of ttl-based consistency in unstructured peer-to-peer networks. *IEEE Trans. Parallel Distrib. Syst.*, 19(12):1683–1694, Feb. 2008.

[97] Bhuvan Urgaonkar, Anoop George Ninan, Mohammad Salimullah, Raunakprashant Shenoy, and Krithi Ramamritham. Maintaining mutual consistency for cached web objects. In *IEEE ICDCS*, pages 371–380, Apr. 2001.

[98] C. Voglis, P.E. Hadjidoukas, I.E. Lagaris, and D.G. Papageorgiou. A numerical differentiation library exploiting parallel architectures. *Computer Physics Communications*, 180(8):1404–1415, Aug. 2009.

[99] X. Wang, Z. Yao, and D. Loguinov. Residual-based measurement of peer and link lifetimes in gnutella networks. In *IEEE INFOCOM*, pages 391–399.

[100] Wikipedia Dumps.

[101] Joel L. Wolf, Mark S. Squillante, Philip S. Yu, Jay Sethuraman, and L. Ozsen. Optimal crawling strategies for web search engines. In *WWW*, pages 136–147, May 2002.

[102] R. W. Wolff. Poisson arrivals see time averages. *Operations Research*, 30(2):223–231, 1982.

[103] R. W. Wolff. *Stochastic Modeling and the Theory of Queues*. Prentice Hall, 1989.

[104] Yinglian Xie, Fang Yu, Kannan Achan, Rina Panigrahy, Geoff Hulten, and Ivan Osipkov. Spamming botnets: Signatures and characteristics. In *ACM SIGCOMM*, pages 171–182.

[105] Amir Yahyavi and Bettina Kemme. Peer-to-peer architectures for massively multiplayer online games: A survey. *ACM Comput. Surv.*, 46(1):9:1–9:51, Oct. 2013.

[106] Mohan Yang, Haixun Wang, Lipyeow Lim, and Min Wang. Optimizing content freshness of relations extracted from the web using keyword search. In *ACM SIGMOD*, pages 819–830, Jun. 2010.

[107] Liangzhong Yin and Guohong Cao. Dup: Dynamic-tree based update propagation in peer-to-peer networks. In *IEEE ICDE*, pages 258–259, Apr. 2005.

[108] Haifeng Yu and Amin Vahdat. Design and evaluation of a continuous consistency model for replicated services. In *USENIX OSDI*, pages 305–318, Jun. 2000.

[109] Jun Zhu, Jianhua Gong, Weiguo Liu, Tao Song, and Jianqin Zhang. A collaborative virtual geographic environment based on p2p and grid technologies. *Inf. Sci.*, 177(21):4621–4633, Nov. 2007.