ABSTRACT

Title of thesis: ANALYZING THE WIKISPHERE: TOOLS AND METHODS FOR WIKI RESEARCH

Jeffrey Stuckman, Master of Science, 2010

Thesis directed by: Professor James Purtilo
Department of Computer Science

We present tools and techniques that facilitate wiki research and an analysis of wikis found on the internet. We developed WikiCrawler, a tool that downloads and analyzes wikis. With this tool, we built a corpus of 151 Mediawiki wikis. We also developed a wiki analysis toolkit in R, which, among other tasks, fits probability distributions to discrete data, and uses a Monte Carlo method to test the fit.

From the corpus we determined that, like Wikipedia, most wikis were authored collaboratively, but users contributed at unequal rates. We proposed a distribution-based method for measuring wiki inequality and compared it to the Gini coefficient. We also analyzed distributions of edits across pages and users, producing data which can motivate or verify future mathematical models of behavior on wikis. Future research could also analyze user behavior and establish measurement baselines, facilitating evaluation, or generalize Wikipedia research by testing hypotheses across many wikis.

ANALYZING THE WIKISPHERE: TOOLS AND METHODS FOR WIKI RESEARCH

by

Jeffrey Stuckman

Thesis submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Master of Science
2010

Advisory Committee:
Professor James Purtilo, Chair/Advisor
Professor Amol Deshpande
Professor Adam Porter

Acknowledgments

I thank everyone who inspired this thesis and motivated my interest in wikis and measurement. First and foremost, I would like to thank Jim Purtilo, my advisor, for giving me the freedom to pursue my interests and for clarifying the oft-mystifying practices and traditions of academia. Next, I thank Jack Callahan for opening my eyes to the collaborative possibilities of wikis, and Sean Fahey for emphasizing the importance of measurement and evaluation. I would also like to thank our graduate coordinator, Jennifer Story, for helping me navigate the university's policies and procedures.

I also thank the United States Office of Naval Research, which partially supported this research under contract N000140710329.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

PMF	Probability Mass Function
CDF	Cumulative Distribution Function
K-S Statistic	Kolmogorov - Smirnov Statistic
K-S Test	Kolmogorov - Smirnov Test

Chapter 1

Introduction

Wikis are web-based repositories that allow for the collaborative editing of content. By making it easy to edit existing content and allowing editors to immediately see the results of their changes, wikis facilitate the collaborative construction of online resources. The original wiki software, WikiWikiWeb, was developed by Ward Cunningham [39] in 1995 and was used to construct an online knowledge base of software design patterns[1]. The popularity of this wiki and others led to the development of Wikipedia, a collaboratively constructed encyclopedia. The success and massive growth of Wikipedia led to interest in the academic community, which sought to describe the collective behavior of Wikipedia users, study ways that Wikipedia could be improved, or harness Wikipedia as a tool to build ontologies or train machine learning algorithms. Between April 2004 and February 2010, 322 papers were published to the ACM Digital Library which mentioned Wikipedia in the abstract, showing robust interest in Wikipedia research. Despite the growing maturity of the Wikipedia project, research interest continues to grow (Table 1.1).

Although Wikipedia is a very popular wiki, wikis other than Wikipedia are also useful. Corporations deploy wikis to facilitate employee collaboration, and thousands of public wikis on the internet are used to build user-created knowledge bases on a wide variety of topics. Despite this broad applicability of the wiki concept, we are only aware of three published papers (excluding individual organizational case

[1] `http://c2.com/cgi/wiki`

Publication year	2004	2005	2006	2007	2008	2009
Number of papers	1	7	23	55	71	155

Table 1.1: Number of papers in the ACM digital library with Wikipedia in the abstract

studies) [29][30][13] that analyzed wikis other than Wikipedia. The result is that a growing body of knowledge describes the behavior of Wikipedia users and suggests technological or policy improvements that could improve it, but it is unclear that this knowledge could be generalized to the broader population of wikis.

The lack of non-Wikipedia wiki research is partly because data on other wikis is hard to obtain, and tools to analyze this data do not exist. Wikipedia releases copies of its database (database dumps) to the general public, so anyone can study and exploit its content. Tools such as WikiXRay [24] can process these dumps and generate statistics from them with little additional work. In contrast, few wikis other than Wikipedia release Wikipedia-style database dumps to researchers, and therefore Wikipedia-specific tools cannot be used to analyze them.

For these reasons, we developed WikiCrawler, a tool that collects data from public wikis through their ordinary web interfaces. We then assembled a collection (or corpus) of 151 popular wikis, to allow wiki researchers to easily perform observational experiments on many wikis at once. Finally, we developed a framework in the R programming language to analyze these wikis, and did a preliminary analysis to suggest ways that this data could be mined in the future.

Because we were interested in fitting probability distributions in order to discover trends in large wiki databases, we needed a tool in the R programming language (the language that we used to import and visualize wiki data) that could fit probability distributions and test the goodness of the fit. Because we did not find such a tool for R that satisfied our needs, we developed one. In Chapter 2, we describe this tool, and we explain how past Wikipedia research could have benefited from the additional rigor that this kind of inferential statistics provides. We then developed the WikiCrawler, a Java-based tool that is designed to download and tabulate data from wikis, cleanly scaling up to hundreds of wikis and hundreds of thousands of individual documents. This tool is described in Chapter 3.

We then ran the WikiCrawler for several weeks in order to compile a large dataset for analysis. Chapter 4 describes the process that we performed to obtain this data and insure its integrity. More specifically, we demonstrated that the wikis in our corpus were created through an organic collaborative process, rather than having been automatically generated by reformatting another dataset. Although we believe that this dataset will be valuable in the future, by allowing researchers to test analytics and metrics on many wikis as a time, we also performed a preliminary analysis, described in Chapter 5, to demonstrate the ways that the data could be used. In this analysis, we discovered that, as in Wikipedia, users contribute to wikis at highly unequal rates, with some users contributing heavy amounts of content and other users contributing little. We also discovered that fitting power law tails to rates of user contribution is an effective way of measuring this inequality, as an alternative to the Gini coefficient which past Wikipedia research has utilized. Finally, to suggest how models may be built that describe the behavior of wiki users, we fit quantities within wikis to various probability distributions, with the aim of motivating the development of models that fit these distributions. In Chapter 6, we describe additional applications and future research that is facilitated by this work.

Chapter 2

Fitting probability distributions to wiki data

Probability distributions are often a component of computer science research [32], including wiki research, due to their ability to summarize datasets and give insight to the processes that were used to generate the data.

Previous research has noted that many quantities in Wikipedia, such as the number of distinct authors of an article [39], the in-degree of links from other pages [5], and the number of times that an article has been edited [5] are distributed according to power-law distributions (sometimes known as "long-tailed distributions"

or "Pareto curves"). Simple descriptive statistics (such as the median) are misleading when applied to such datasets [32], making it inappropriate to measure quantities such as the median number of revisions of wiki articles. Outside of wiki research, probability distributions have been used to describe the link structure of web pages [33], the distribution of file sizes [32], and the structure of certain networks of the internet. Other research has noted how other discrete quantities, such as the relative popularity of books [4] and the population of cities [32], conform to the power law.

Aside from existing wiki research which used methodologies involving probability distributions, we observed that other wiki research could have benefited from the involvement of probability distributions as well. Much quantitative wiki research has focused on measuring the *concentration of effort* found in the wiki. It has long been observed that user activity in wikis is concentrated unequally, where a small number of users is responsible for a large amount of the content, or a small amount of pages attracts a disproportionate amount of attention. Therefore, measuring concentration of effort is important because it provides context for interpreting simple descriptive statistics, such as the number of users or articles in the wiki. The extent that effort in Wikipedia is concentrated is debated: Kittur *et al.* [12] tracked words and revisions of articles to suggest that occasional users are responsible for an increasing amount of Wikipedia's content, while Ortega *et al.* [25] concluded that few users are responsible for a bulk of the activity. While that research used the Gini coefficient (see Section 5.3) to measure inequality, fitting a long-tailed distribution to the data would have provided a scale-invariant way of measuring inequality in a dataset. (Small datasets tend to have smaller Gini coefficients, even if the distribution of the data is the same [8]. In contrast, random subsets of a dataset should be distributed in the same way that the entire dataset was.) We evaluate the use of power-law distributions to measure inequality in Section 5.5.

Aside from describing data or determining if it is distributed unequally, prob-

ability distributions are also useful for motivating or verifying generative models that predict user behavior in wikis. For example, Wilkinson and Huberman [46] proposed a stochastic model that describes how Wikipedia articles grow over time, and then they verified this model by confirming that the measured lengths of articles with a given age was log-normally distributed as expected. While rarely done in wiki research, this kind of model development is more common when studying other computing-related phenomena, as well as in economics, actuarial science, and other fields. For example, the discovery of power-law fits led to the proposal of generative models that explained numerous distributions on the World Wide Web [17], and fitting other probability distributions motivated a model that predicts how file sizes will be distributed on a system [19]. We believe that similar models could describe phenomena seen in wikis, such as the emergence of frequent and infrequent wiki contributers. Along with the other ways that probability distributions are used in wiki research, this motivates our discussion of distribution fitting methodology and our development of distribution fitting tools.

2.1 Approaches to fitting probability distributions

While some Wikipedia research has claimed that various quantities are distributed according to the power law, not all such research fitted and tested distributions in a statistically rigorous manner. Some wiki research used a visual method to fit a power-law distribution, by observing that the probability distribution appears as a straight line on a log-log scale. However, this method can yield misleading results [32] and it cannot reliably determine the equation of the distribution, which is useful both as a summary statistic and as a means of determining the parameters of the model from which the long tailed distribution arose. Other wiki research used statistically sound methods to fit the distributions and test the fits, but they implemented their distribution fitting and data analysis algorithms from scratch each

time, making reproducibility more difficult. These one-off implementations also tend to be specific to a certain probability distribution, making it difficult to make a fair comparison between probability distributions to assess which one best fits the data.

While looking for a statistically sound and reproducible way to fit distributions in our research, we discovered that the R programming language [26], a language widely used for data analysis, lacked a consistent way to fit and test discrete probability distributions (as discussed in Section 2.3). Therefore, to facilitate our wiki research, we have developed a framework that allows users to fit discrete (count) data to various probability distributions, and statistically test the plausibility of the fit in a distribution-neutral way. Aside from supporting the current research, this framework will serve as a useful tool for future research into wikis or other collaborative systems, and in any other place where the probability distributions of discrete (count) data must be known. Additional probability distributions can be defined by the user by entering a few formulas, further increasing the flexibility of our framework.

2.2 Some terminology

From this point forward, we will use the following terminology when discussing our framework.

- *Probability mass function* (PMF): A function defined over the natural numbers which gives the probability of encountering each possible value.
- *Density function*: Often called *probability density function*: A function defined over the real numbers (typically the positive real numbers in our case) which has the properties generally associated with a probability density function.
- *Cumulative distribution function* (CDF): A function defined over the natural numbers which gives the probability of encountering any value less than or

equal to the given value.

- *Empirical distribution*: A probability distribution computed by counting the relative frequency of values in the user's dataset (or *empirical data*).

- *Distribution family*: A term used in our framework to refer to parameterized probability distributions without an associated parameter assignment. Parameters are assigned to a distribution family during the fitting process. The term "family" is used in our framework to distinguish these from fitted distributions.

- *Fitted distribution*: The PMF and associated parameter assignment derived from maximum-likelihood fitting of the user's dataset.

2.3 A framework for fitting probability distribution in the R programming language

The R programming language has many available packages that fit or do goodness-of-fit tests on particular distributions [28], such as the Normal Distribution (`ADGofTest`), and the Pareto Distribution (`gPdtest`). The `fitdistrplus` package fits several univariate distributions and performs a chi-squared goodness of fit test, while the `distr` package provides a number of distributions but does not provide general methods to fit them. Some built-in functions provide PDFs and CDFs for common distributions but do not fit them, while R's built-in optimization functions can be used for maximum likelihood estimation but require the user to supply the appropriate function to optimize.

The model for our framework was the `plfit` [32] package, which fits power-law (continuous or discrete Pareto) distributions and performs a Kolmogorov-Smirnov goodness-of-test fit, in a single integrated workflow. Our methods are based on the methods of this package, generalizing them to many other distributions, and im-

proving them so the methods used to fit the Pareto distribution (which only has a single parameter) could be used to fit distributions with multiple parameters. So the researcher does not need to implement or install new code for every experiment and every distribution that is being fit, we created an integrated framework that allows for probability distribution fits, goodness-of-fit tests, and graphical visualizations, while making it easy to extend the system by adding new distribution families. Unlike other probability distribution frameworks, we optimized the system for working with discrete (count) data, because computer science research in collaboration and graph theory normally ends up with such data.

An earlier version of the framework described in this section was developed as a part of our previously published research [34].

2.3.1 A discussion of methods for fitting probability distributions

Any effort to describe data with a probability distribution must begin with fitting the distribution. Fitting a probability distribution involves picking a parameterized distribution and then setting its parameters such that the fitted probability distribution best matches the empirical distribution (what was observed in the data.)

One of the simplest ways to fit a probability distribution is the moment-matching method [31], which entails choosing the parameters of the distribution such that the moments of the distribution match the observed moments in the data. The parameters of the distribution are easy to compute under this method, but the method must be customized for each family of distribution being fitted because the moments are computed differently for each one (and for many distributions, the moments do not exist.) Also, in general, the moment matching method is not a maximum-likelihood method because the probability of the data is not guaranteed to be at its maximum with the chosen parameters (although there are some exceptions, such as the normal distribution [44]).

Another method to estimate parameters of probability distributions is maximum likelihood estimation. Maximum likelihood estimation entails choosing the parameters such that the probability of the data is maximized under the selected parameters. For some distributions, the maximum likelihood parameter assignment can be computed directly, for example, in the case of the normal distribution (because the moment matching method gives a maximum likelihood estimate) [44] or the Pareto distribution (because calculus can be used to directly maximize the likelihood function [32]). However, for most distributions, this cannot be done directly. A more general solution is to use "hill-climbing" maximization optimization methods, using the likelihood of the data as the parameter to be maximized, and the parameters of the distribution as the parameters to optimize over [37]. As with all optimization, the algorithm could converge on a local maximum and return a parameter estimate that does not maximize the probability of the data.

2.3.2 A generalized fitting procedure

We implemented a general optimization procedure to assign parameters to a probability distribution. Although algorithms tuned to a particular probability distribution family may produce better fits and more powerfully test the fit, we sought to develop an algorithm that could be used for any distribution that the researcher needs to work with.

Our fitting algorithm and test were motivated by the procedure developed by Clauset *et al* [32] for power-law distributions. In this procedure, the parameters of the distribution are estimated by maximizing the likelihood function, and the fit is evaluated by computing the K-S statistic. It is not appropriate to use the likelihood as a goodness-of-fit statistic because it does not have a consistent scale – varying widely with the size of the dataset and the parameters of the distribution – and also because Clauset *et al* observed that, by using the K-S statistic to evaluate the

fit, the choice of inappropriately small subsets (see Section 2.6.1) of data chosen for fitting is avoided.

2.3.3 Defining a probability distribution family

Instead of hardcoding the supported probability distribution families into our framework, we allow users of our framework to define their own probability distribution families, so users are not limited to specific distributions. Probability distributions are defined by an R data structure that contains several required equations and settings. At a minimum, a PMF must be provided. The PMF should return the natural logarithm of the probability. It is encouraged that the internal computations for the PMF are also computed in logarithms to improve accuracy. A CDF may also be provided, which is used to compute the K-S statistic. If this function is not specified, the framework will manually compute the CDF by taking the cumulative sum of the probability mass function. When the CDF can be computed without iteration, it should be explicitly defined to improve efficiency. The provided CDF need not match the cumulative sum of the PMF, because inaccuracies due to performing continuity corrections (see Section 2.3.4) may prevent this. However, if the CDF is not provided, the cumulative sum of the PMF must approach 1 as the terms approach infinity.

The probability distribution family defines the desired optimization method (see Section 2.3.5) which will be used for fitting, along with a list of variables that should be set during optimization. If the BFGS optimization method is chosen, the partial derivatives of the log PMF (otherwise known as the gradient function) must also be supplied, in order to improve the performance and accuracy of the likelihood maximization algorithm.

In cases where an equation for the CDF is not known, but taking the cumulative sum of the probability mass function is not appropriate due to continuity

corrections, the probability distribution family may define the CDF using R's built-in numerical integration function. Experimentally, we found that this resulted in the cumulative distribution function having occasional "glitches", where one value of the CDF is lower than the previous value due to numerical integration inaccuracies. Our framework makes an attempt to detect and compensate for these glitches (see Section 2.4.1).

Because our framework has been designed to process discrete count data, any distributions should be designed with the assumption that it is impossible to encounter a value less than 1. The framework will assume that the entire probability mass is in the range $[1, \infty]$ and distributions should be defined with this in mind. Distributions can also define a higher minimum value, and exclude smaller data points from the fit using the aforementioned data subset capabilities. Also, all functions defined should accept vectors of data points as well as single data points, because many internal functions of our framework are vectorized.

2.3.4 Continuity correction

Although our framework is designed to fit data to discrete probability distributions, we allow users to fit discrete data to continuous probability distributions. This may be done to make computations more tractable (such as when the normal distribution is used to approximate the binomial distribution[41], or when the Pareto distribution is used to approximate the Zipf distribution[32]) or to utilize a continuous distribution that has no discrete analog. A naive way to accomplish this is to obtain the discrete PMF and CDF by evaluating the probability density function and the continuous CDF at the integers. However, for the K-S statistic to be computed properly, the cumulative sum of the resulting PMF should be reasonably close to the corresponding value of the CDF. This means that applying an appropriate continuity correction is necessary when defining a continuous probabil-

ity distribution in our framework.

Consider the case where the PMF and CDF are evaluated at 1. At this point, the values of the PMF and CDF should be as close as possible. A naive solution is to evaluate the density function at 1 to obtain the PMF, but this typically results in a probability mass that is too high, because CDF(1) equals $\int_0^1 \text{PDF}(x)\,dx$, which is the mean value of the density function between 0 and 1. Therefore, for the value of the CDF to be close to the value of the PMF, the density function should be evaluated at 0.5, which will be closer to the CDF (assuming that the density function strictly increases or decreases during this interval.) Generalizing this reasoning, we then let the PMF at x equal the density function at $x - 0.5$. Applying this continuity correction makes the behavior of the CDF more consistent with the PMF.

Note that after applying the continuity correction, the resulting distribution is no longer the same distribution as the original distribution. Also, the resulting discrete distribution lacks properties that probability distributions are expected to have – namely, the probabilities of all frequencies do not sum to 1, and the cumulative sum of the PMF does not equal the cumulative distribution function. However, the new distribution does have the two properties that are important to us – the CDF approaches 1 as x approaches infinity, and the difference between successive values of the CDF is close to the corresponding value of the PMF.

This procedure must be modified when the minimum allowable data point in the discrete distribution should be greater than 1. The corresponding continuous distributions typically define the CDF such that, if the minimum allowable data point is x_{min}, $\text{CDF}(x_{min}) = 0$. If x_{min} is set to the minimum allowable data point in the discrete dataset, then for the differences between successive values of the CDF to be similar to the corresponding values of the PMF, the CDF must be evaluated at $x + 1$ and the probability density function must be evaluated at $x + .5$.

2.3.5 Estimating parameters through optimization

Our framework fits probability distributions to discrete data by finding a parameter assignment that maximizes the likelihood of the observed data. Therefore, this problem can be framed as an optimization (maximization) problem, allowing us to use existing optimization techniques to fit arbitrary probability distributions. The function to be optimized is the sum (over all data points) of the log of the probability mass function, evaluating the probability mass function at each data point with the parameter assignments being made by the optimization algorithm. This results in a log-likelihood function, given the assumption that the samples in the empirical dataset are independent.[37]

Two direct-search optimization methods [45] are available when using the framework – the BFGS method and the Nelder-Mead method. These methods were chosen because mature implementations of them are already built into the R language. The Nelder-Mead method only requires the function being optimized to be defined, while the BFGS method is a gradient-based method that also utilizes the partial derivatives of the function with respect to each one of the function's parameters. This is satisfied by requiring users to supply the partial derivatives of the log PMF, as described in Section 2.3.3.

One frequently encountered problem when optimizing is encountering a solution that is far from optimal due to terminating at a local minimum [18]. To help counter this, our framework runs the optimization algorithm with multiple stating points supplied, and only the best solution is used. Each probability distribution family defines a grid of possible starting points, which represents the range of practical parameter values for each dimension. The spacing of points on the grid can be defined with either linear or logarithmic spacing (it is expected that logarithmic spacing will be used for parameters proportional to the points in the dataset, such as the center of a normal distribution.) The starting points can also be computed

based on the dataset, by supplying a user-defined function to compute them.

Because the grid of possible starting points contains many starting points where the data has extremely low likelihood, the likelihood of the data is first computed at each starting point, and 30% of the starting points where the data had the lowest likelihood are discarded. This was performed to avoid starting the search in areas of the solution space where a solution is extremely unlikely (these areas of the solution space could still be explored, unless the cell-constrained optimization described later was also being used.) Any starting point where the mean log likelihood of data points was less than -10 was also discarded for containing non-feasible solutions. The optimization function is then applied once for every remaining starting point, and the solution that maximizes the likelihood of the data is chosen.

To further improve the optimization algorithm, we added the option of possibility of using the grid of starting points to actually constrain the values that the optimization algorithm can try. Instead of only using the grid to select a set of starting points, we also used the grid to partition the parameter space into cells, where the points to the left and the right of the starting point become the left and right boundaries of the cell (repeated for every dimension). The optimization is then constrained to only explore values within the starting point's cell. The cell boundaries are computed before discarding starting points, so whenever a starting point is discarded by the aforementioned rule, any cells that the starting point touches are no longer considered during optimization.

One purpose of dividing the parameter space into cells is to prevent evaluating extremely unlikely values that cause an overflow in the log-likelihood function. Our framework begins any optimization by evaluating points on the grid, and noting parameter assignments that result in an overflow. When cell-constrained optimization is enabled, any cell that has an overflow point at its boundary is excluded from optimization, even if the starting point (at the center of the cell) was not partic-

ularly unlikely. We found that this feature prevented the optimization algorithm from crashing when the parameter space had numerous regions that would cause an overflow and crash the optimizer (see Section 2.6.2).

We also discovered that optimizing with the cell boundaries also resulted in better solutions being found in some cases. This is because we often encountered a situation where the optimization would repeatedly converge on the same local minimum, regardless of the choice of starting point. By restricting the boundaries of the optimization, the algorithm can only converge on any given local minimum when optimizing within the cell where the minimum is contained. The disadvantage of using the cell-bounds optimization is that the optimization function must be run numerous times to find a result, instead of being run just once.

2.4 Testing the goodness of the fit

Above, we described we the algorithm that we implemented to find the set of parameters that maximizes the likelihood that a specific probability distribution would generate the observed data. If we were already convinced that the data was distributed according to a specific probability distribution family, this would be good enough, and we would be finished. However, we also want to consider the possibility that the data is not distributed according to the given distribution family at all.

Some previous wiki research (along with other computer science research [32]) has alleged the validity of a power-law fit by overlaying the a distribution line over the empirical data on a graph. However, this technique for confirming a fit is misleading [32] because it is difficult to visually determine that the inevitable differences between the empirical distribution and the fitted distribution can be attributed to natural variation or chance. Although the likelihood function gives the probability that the observed data would be produced if a random dataset was drawn from the fitted distribution, this cannot directly be translated into a probability that the

observed data was generated by such a process.

Rather than visually assessing the goodness of a probability distribution fit, it is preferable to compute a goodness-of-fit statistic that gives an objective measurement of how well the fitting process was able to make the fitted distribution conform to the empirical data. Intuitively, a good result increases the researcher's confidence that the data was generated by the proposed distribution. One such statistic is the Kolmogorov-Smirnov statistic (also known as the K-S statistic) [16]. If $F_0(x)$ is the CDF of the fitted distribution and $S_N(x)$ is the cumulative step-function of the empirical data, then the K-S statistic is [16]:

$$\max_x |F_0(x) - S_N(x)|$$

This statistic measures the farthest distance between the fitted CDF and the empirical CDF, and can be used to gauge the goodness of a distribution fit [16]. However, a goodness-of-fit statistic alone isn't sufficient for the researcher to judge if a fit should be accepted or not. For this, a goodness-of-fit *test* is required. A goodness-of-fit test is a statistical test that rejects a fit if the fitted distribution does not sufficiently conform with the empirical distribution [16].

One common goodness-of-fit test is the chi-squared goodness-of-fit test [41]. In this test, the range of data that could be generated by the probability distribution is divided into bins, and the fitted distribution is used to compute the number of data points that would be expected in each bin. If the actual number of data points that falls in each bin is too different from the expected number, the fit is rejected. Although this test is computationally efficient, its results depend on the way that the range of possible values was divided into bins. Depending on how the datapoints are binned, the test may arrive at different outcomes, complicating reproducibility of the test results. Also, deviations from the fitted distribution will not be detected

if they don't cause any data points to move from one bin to another. For these reasons, we instead implemented a goodness-of-fit test based on one designed by Clauset *et al* [32] , who used a simple Monte Carlo process to determine if the K-S statistic of the fitted distribution is consistent with the assumption that the empirical data was drawn from the fitted distribution.

Note that goodness-of-fit tests cannot determine with any certainty that a dataset was generated by an underlying process conforming to a given probability distribution (or that the dataset was "drawn" from that distribution). Even if the observed data was indistinguishable from a typical dataset drawn from the fitted distribution, it could have come from another distribution which is only imperceptibly different. What a goodness-of-fit test *can* do is *reject* that data came from its fitted distribution (subject to a predetermined false-rejection rate). The goodness-of-fit test is tuned so random datasets which are drawn from the distribution in question are rejected at this predetermined false rejection rate, with the assumption that anything not drawn from its fitted distribution should be rejected more often than this.

2.4.1 A goodness-of-fit test

We implemented a goodness-of-fit test that compares the K-S statistic of the fit with a threshold that is computed by a Monte Carlo process. Therefore, to perform this test, we start by computing this statistic. The K-S statistic describes the difference between the CDF of the fitted distribution and the empirical CDF (or cumulative sum) of the data. If the fitted distribution perfectly fits the data, then the two cumulative distributions will be identical, resulting in a K-S statistic of 0.

Our framework computes this statistic by first computing the empirical CDF and then evaluating the fitted CDF at each point within the range of the observed data. This is straightforward, except in cases where the CDF as defined by the prob-

ability distribution family is not well-behaved. Some probability distribution families use the numerical integration capabilities built into R to compute their CDFs, because the formula to directly compute the CDF may not be known. However, R's numerical integration algorithm does not return perfect results, and occasional "glitches" may be encountered, where the value returned from the integration is higher or lower than it should be. Even in cases where R's numerical integration function is not being used, the CDF can still decrease slightly in cases where an internal loss of precision occurs. To compensate for these problems, our framework detects when a point on the CDF is higher than the one that follows (see Figure 2.1), and the erroneous points are replaced with ones interpolated from the adjacent values. The end result is a non-decreasing CDF, which prevents errors from occurring when the CDF is being used.

2.4.2 Performing the Kolmogorov-Smirnov test

Because the K-S statistic acts as a measure of how well a fitted distribution conforms to the empirical data, we can use it to test if the data came from the distribution being fitted. Consider the case where the *empirical data* was drawn from a certain, *known distribution*. Because of random variation in the selection process, it is unlikely that the empirical distribution that results from this process will perfectly match the fitted distribution (which would result in a K-S statistic of zero). Instead, the K-S statistic in this hypothetical case will have a distribution of its own, which will become apparent if many empirical distributions are drawn from this known distribution and their K-S statistics computed. This motivates the Kolmogorov-Smirnov test. In the same way that a T-test can use the *Student's t distribution* to reject the null hypothesis of two means being equal, the K-S test can use the *distribution of the K-S statistic* to reject the null hypothesis that a dataset was drawn from a known distribution. [16]

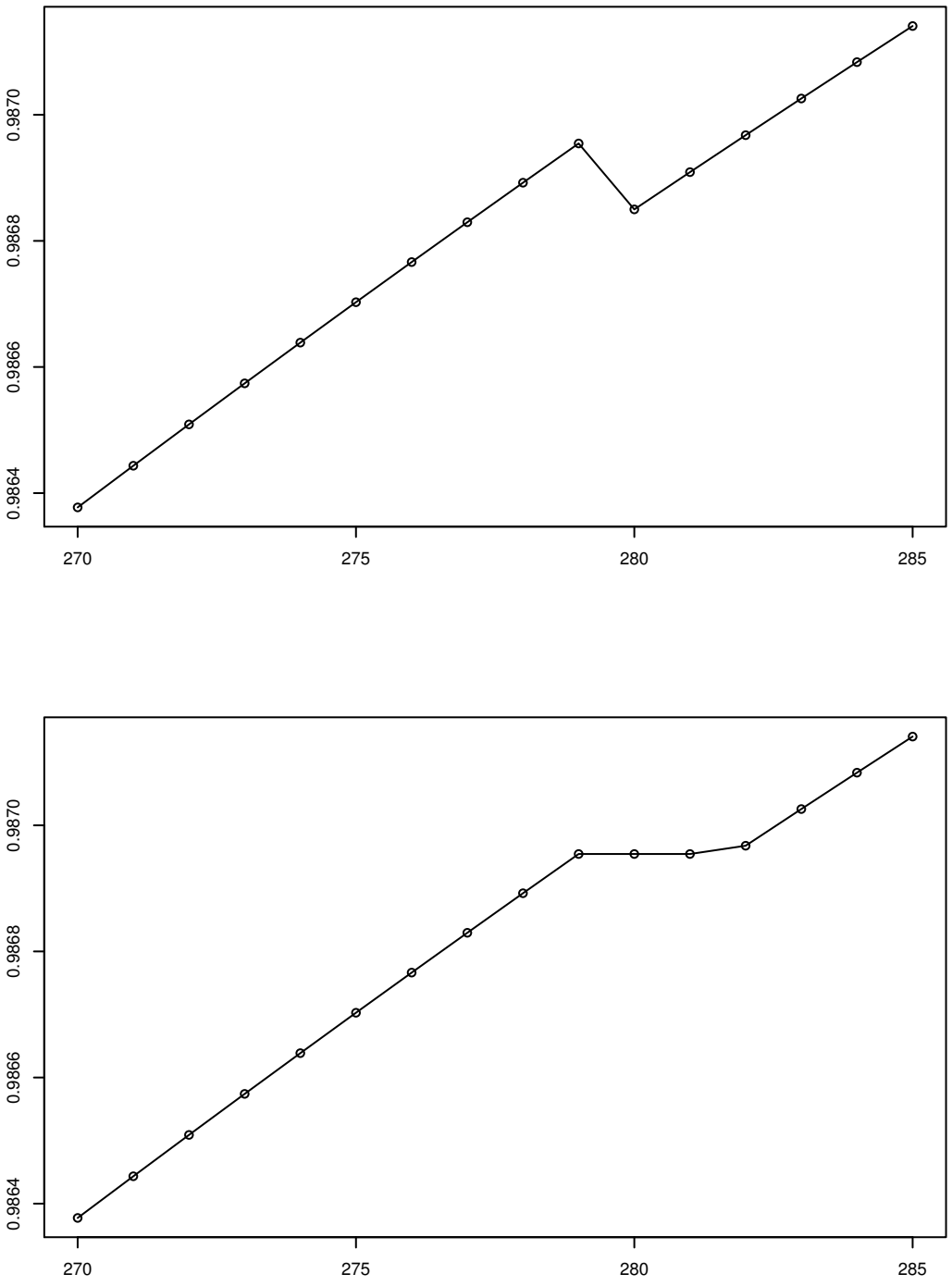

Figure 2.1: Example of how cumulative distribution functions are adjusted in order to make them non-decreasing. In this case, the CDF of a Double Pareto Lognormal distribution fit to one of our datasets was adjusted.

Accordingly, we implement the K-S test as a Monte Carlo process where we must draw enough synthetic datasets to determine if the K-S statistic of our empirical data lies within the critical region of the test. To accomplish this, first, the empirical data is fit to the desired probability distribution family (resulting in one *primary fit*), and the Kolmogorov-Smirnov statistic is computed. Next, datasets are drawn from the fitted distribution and fitted to the same probability distribution family (resulting in many *secondary fits*). The number of data points drawn into one of these *synthetic datasets* must be equal to the number of data points in the empirical dataset. To draw these data points from the fitted distribution, points from the CDF of the fitted distribution must be computed and explicitly stored in memory before drawing any datasets. This allows data points to be drawn by choosing a random value from the uniform distribution and then performing a binary search on the points from the CDF. The K-S statistic of the secondary fits are then compared with the K-S statistic of the primary fit. The p-value for the test is computed as the proportion of the secondary fits with a K-S statistic greater than that of the primary fit. In this way, the distribution of the Kolmogorov-Smirnov statistic is implicitly computed while computing the p-value.

The secondary datasets are compared with the secondary fits, as opposed to the primary fit, to compensate for the fact that the parameters of the distribution being tested came from the data that it is being tested against. When performing the chi-squared goodness-of-fit test, the experimenter compensates for the unknown parameters by increasing the number of degrees of freedom of the test.[16] [28] In our test, we use the methodology in [32] to compensate for it by re-fitting each dataset drawn from the fitted distribution (which will be referred to as *synthetic datasets*). The distribution of K-S statistics is computed by comparing each synthetic dataset to its corresponding secondary fit.

Note that our framework allows for probability distribution families to selec-

tively omit data points from the fit (see Section 2.6.1), which is intended to be done for tail-only fits, where a probability distribution describes the distribution of large data values but not small ones. The omitted points are also ignored when computing the K-S statistic. This poses a problem when drawing the secondary datasets, because the fitted distribution may not generate the small-valued data points that the primary dataset had. To compensate for this, we apply the same procedure used in [32], by randomly including the omitted points from the empirical dataset in the synthetic dataset (along with points drawn from the fitted distribution). For example, if 10% of the data points were omitted from the empirical dataset when taking the primary fit, then there is a 10% chance that a data point being added to a synthetic dataset will actually be drawn from these omitted points rather than drawing a value from the primary fit.

2.4.3 Interpreting the results of the Kolmogorov-Smirnov test

The Kolmogorov-Smirnov test will conclude by returning a p-value. This p-value should be interpreted as the probability that a dataset drawn from the fitted distribution (a synthetic dataset) will have a Kolmogorov-Smirnov statistic that is greater than the Kolmogorov-Smirnov statistic of the original dataset. Small p-values are more likely if the fitted distribution poorly fits the empirical data, while large p-values are more likely if the fit is good.

This motivates a more rigorous definition of the test. Define the null hypothesis as the case where the dataset was drawn from the fitted distribution (in other words, the null hypothesis is that the fitted distribution is the correct distribution.) We now demonstrate that rejecting the null hypothesis when $p < \alpha$ for some threshold α will result in a Type-I error rate of α. To compute the Type I error rate, consider the case where the null hypothesis is true and the dataset has been drawn from the fitted distribution. In this case, the empirical dataset and the synthetic datasets

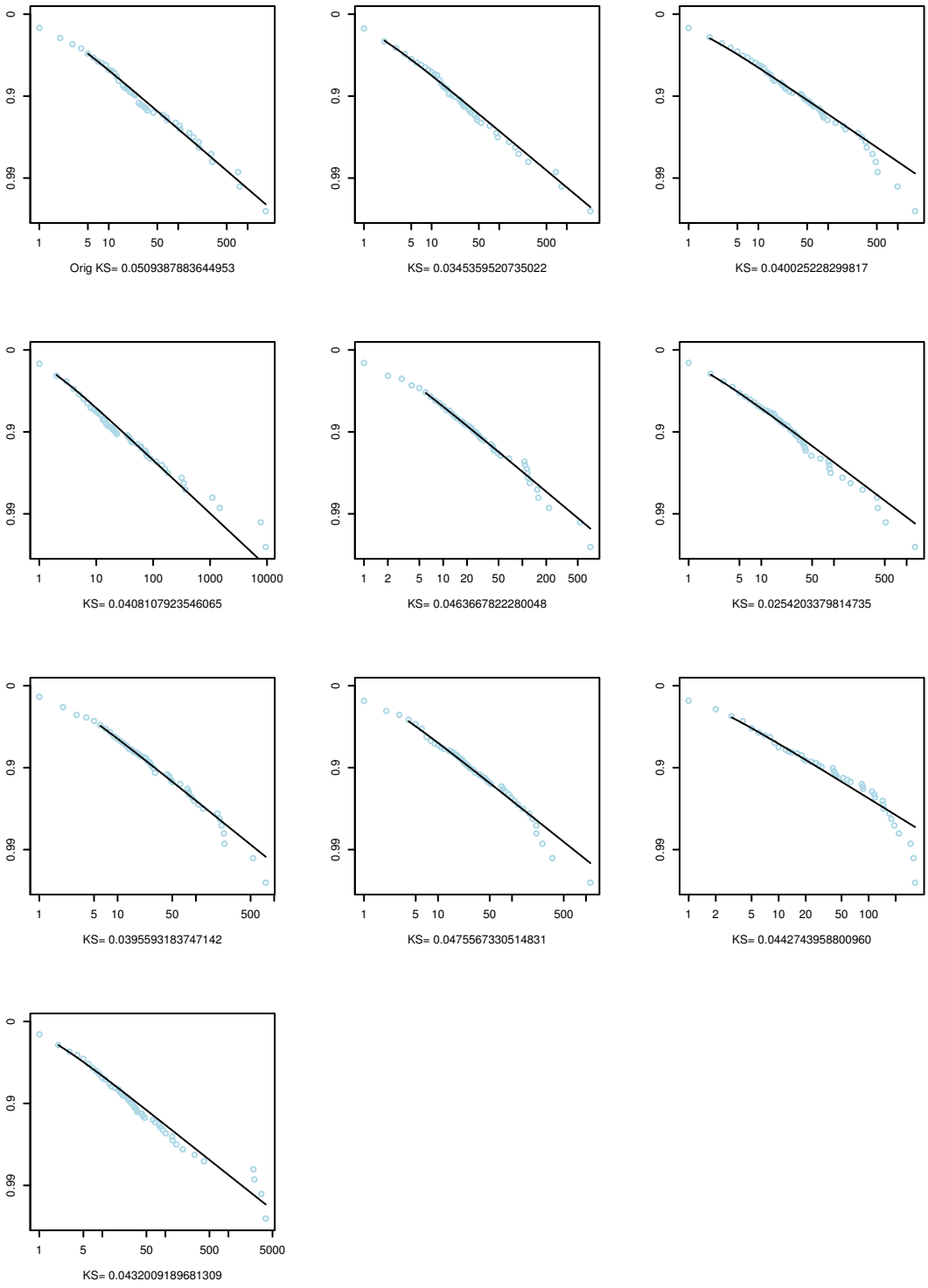

Figure 2.2: Depiction of the K-S test being performed. In this case, the original dataset and 9 synthetic datasets are fit to a Zeta distribution. The K-S statistic of the original dataset was lower than the K-S statistic of 5 out of the 9 empirical datasets; therefore, the fit was accepted.

were generated by exactly the exact same process. Therefore, the K-S statistics for the empirical dataset and the synthetic datasets are effectively drawn from the same distribution. For any given K-S test, this has the effect of drawing the resulting p-value from a uniform distribution between 0 and 1 (assuming that the p-value can only take on a finite number of values – this assumption is explored in Section 2.4.4 and tested experimentally in Section 2.7.2). Therefore, if α is set to some value, the null hypothesis will be rejected with that probability. This means that the p-value of this Kolmogorov-Smirnov test can be interpreted in the same way that one would ordinarily interpret p-values.

Note that this test cannot prove that a dataset has been drawn from a specific distribution, nor can it give the probability that it was. For example, it is impossible to state with certainty that a dataset "has a power-law distribution" or "does not have a power-law distribution" Generally, it is impossible to make such statements because one could construct a distribution that is infinitesimally different from the fitted distribution, and argue that the empirical dataset was drawn from that one instead. This test can only conclude that it is implausible that the dataset conforms to the fitted distribution. Increasing the α threshold will make it more likely that a fit will be properly rejected when the data does not conform to it, while also making it more likely that a fit will be improperly rejected when the data actually does conform to it. Finally, it is impossible to state with certainty whether a dataset was drawn from a given distribution or not, because an "unlucky" or "lucky" draw can make this impossible to determine, regardless of the procedure used or the amount of computational power available.

2.4.4 Choosing the number of iterations

Note that in the above definition of the test, the number of synthetic datasets to generate was not specified. Because the p-value is determined by the proportion

of the K-S statistics of the synthetic datasets that are lower than the K-S statistic of the empirical dataset, doing additional iterations (and hence computing additional Kolmogorov-Smirnov statistics) can alter the results of the test.

Note that each iteration is an independent, random process, and the K-S statistic of the empirical dataset is computed at the beginning of the algorithm and does not change. Furthermore, the exact value of the K-S statistic for each iteration does not matter for the purposes of computing the p-value – it only matters if the statistic is above or below that of the empirical dataset. Therefore, we can consider each iteration of the algorithm to be a Bernoulli trial, where the possible outcomes are accepting and rejecting the probability distribution. A Bernoulli trial is defined by specifying a probability (which we will call q) which determines the success rate of the trial. [41] In our case, the probability q is a function of the empirical data and its fitted probability distribution – so q is the unobservable probability that any given K-S statistic will be higher than the empirical one.

It is impossible to directly determine the value of q because the distribution of the Kolmogorov-Smirnov statistic cannot be directly computed in our case; however, we can estimate this value by generating multiple synthetic datasets and hence performing multiple Bernoulli trials. Note that the p-value that results from the test will approach q as the number of trials approaches infinity. Also note that the user must choose a threshold α to perform the Kolmogorov-Smirnov test. Therefore, if we run the algorithm for an infinite number of iterations, and then then accept or reject the fit per the procedure described in the previous section, we would be effectively comparing the values of q and α, rejecting if $q < \alpha$ and accepting otherwise. Observe that across all null hypothesis cases for a particular probability distribution, q will be uniformly distributed between 0 and 1, meaning that the Type-I error rate is properly controlled in this case.

This formalization of the K-S test also allows us to decide how many iterations

of the algorithm should be run. If, based on the Bernoulli trials run so far, it is implausible that $q < \alpha$ or it is implausible that $q > \alpha$, then the algorithm can terminate because it must distinguish these two cases and it ruled one out. To declare that a statement $q < \alpha$ is "implausible", we must (1) obtain an estimate of q, which will be called q_{obs}, by performing Bernoulli trials, (2) observe that $q_{obs} > \alpha$, and (3) determine that q_{obs} and α are "different enough", establishing that the observation that $q_{obs} > \alpha$ wasn't a fluke. (Without loss of generality, the same procedure could also determine that $q > \alpha$ is implausible.)

2.4.5 The binomial test

Note that, if $q_{obs} < \alpha$ holds and $q = \alpha$ is declared implausible, then $q > \alpha$ must also be declared implausible. (Without loss of generality, the same is true when $q_{obs} > \alpha$ holds.) Consequentially, instead of directly testing if $q > \alpha$ is implausible, we can instead test if $q = \alpha$ is implausible, allowing us to use a standard binomial test. The binomial test accepts or rejects the null hypothesis of $q = \alpha$ when Bernoulli trials have been performed [40]. If this null hypothesis is rejected, then we can conclude that $q > \alpha$ (or $q < \alpha$) is also implausible, and therefore we can stop doing iterations of the algorithm because only one plausible result remains. A second threshold α_2 (not to be confused with α) must be chosen for this intermediate hypothesis test. In the binomial distribution, the probabilities of values away from α decrease as n increases; therefore, choosing a smaller α_2 will normally increase the number of iterations that must be performed. This property of the binomial test exposes a trade-off inherent in our Monte Carlo fit-testing algorithm – performing more iterations will increase the certainty that the fit was correctly accepted or rejected.

We must perform a two-tailed binomial test to ensure that the Type I error rate does not climb above the indicated α_2, but care must be taken when computing

the critical region because the binomial distribution is a discrete distribution and is not generally symmetrical. To compute the critical region, we adopt the *method of small p-values* [1]. First, we compute the binomial distribution's PMF with a success probability of α and a n equal to the number of iterations performed. Then, we sum up the function's smallest values until the sum is greater to or equal than the desired α_2. The set of values summed define the critical region, which will be one or two contiguous regions at the extreme left and/or right of the distribution. Next, we count the number of iterations where the K-S statistic of the secondary fit was lower than the K-S statistic of the primary fit. If this number is in the critical region, then the K-S test can terminate by accepting or rejecting the fit (depending on which critical region the number was in). Otherwise, more iterations should be performed.

2.5 Visualizing the fitted data

While assessing the goodness of the fit is better done with our fit-testing algorithm than by inspecting a graph, it is still useful to graph the empirical data and the fitted distribution, in order to better understand the nature of the distribution, or why the fit was good or bad. Such graphs depict both the frequencies of empirical data values and a line representing the expected empirical data frequencies based on the fitted distribution. Our framework can automatically plot and visualize any fitted distribution on two types of graphs: a histogram graph and a complementary cumulative distribution graph.

2.5.1 Histogram graphs

Probability distributions are often visualized and explained by graphing their probability density functions. For example, the famous "bell curve" of the normal

distribution comes from the graph of its probability density function. [41] Overlaying the probability density function over the empirical data helps visualize the ways that the data deviated from its empirical distribution.

However, because we are working with discrete data only, probability mass functions are used instead of probability density functions. Also, the data may be sparse in the sense that not all possible values may be represented when the number of data points is not large enough. Binning the data and creating a bar-graph histogram allows the fitted probability mass function to be graphed over the original data while making the dataset neat enough to interpret. The x-axis represents the range of possible values, and histogram bars are drawn so their left and right edges represent the minimum and maximum value in each bin. The y-axis represents a probability, where the height of the histogram bars is equal to the proportion of values in the corresponding bin. A curve is drawn (with line segments between integer points) that represents the probability mass function of the fitted distribution.

An example can be seen in the second graph of Figure 4.1, where the empirical dataset was fitted to a power-law distribution. In this example, the axes were drawn with a logarithmic scale, in order to show more detail for low values and in order to visualize the fitted distribution as a straight line.

2.5.2 Complementary CDF graphs

Although the histogram graph allows the distribution of the empirical data to be visualized, it is not suitable for all purposes. Individual data points do not appear on the graph, and it misleadingly portrays the degree that the empirical data deviates from the fitted distribution because the Kolmogorov-Smirnov statistic is based on the cumulative distribution function. For this reason, we added a second visualization that shows the empirical data superimposed on the complementary CDF

of the fitted distribution, which is easier to interpret in such situations [32]. Such graphs have been used to visualize the complementary CDFs (known as "survival functions" in this case) of income distributions in economics research [2].

In this graph, the x-axis represents the range of possible values, while the y-axis represents the cumulative probability of the data point along with all preceding data points. As with the previous graph, the axes can be drawn with a logarithmic scale as well. Because a logarithmic scale cannot represent all values between 0 and 1, the complementary CDF is drawn on the y-axis, instead of the regular CDF. This results in a graph of the function $\log(1 - \mathrm{CDF}(x))$ instead of the function $\mathrm{CDF}(x)$, at a scale such that the maximum y-axis value is 0 and the minimum y-axis value determines the cutoff. The cutoff should be chosen so all data points but the last data point can be visualized. (The last data point is always omitted from the graph because its CDF will be 1 and hence an infinite value would be graphed.) For example, if there are 80 data points, then the CDF of the second to last data point will be .9875. Choosing a minimum y-axis value of 2 with a base-10 logarithm will result in a maximum CDF of .99, allowing all but one data point to be graphed.

An example can be seen in Figure 5.2 or in the first graph of Figure 4.1, where the empirical dataset is fitted to a power-law distribution.

2.6 Implementation challenges

In the sections above, we described our algorithms for fitting probability distributions and testing the goodness of fit. Below, we describe some of the difficulties we encountered when implementing these algorithms.

2.6.1 Partial distribution fits

Some probability distribution families may choose to include only a subset of the empirical data in the fit, excluding data points that lie outside this subset. Out of the distribution families that we implemented, the Pareto, Zeta, and Levy distribution families (described in Section 2.8) only consider a subset of the data points $x|x \geq x_{min}$. This effectively splits datasets into a *head* and a *tail* with x_{min} as the split point, and the resulting probability mass function only describes the relative frequency of values that fall into the tail section of the distribution.

Such probability distribution families will determine this split point as part of the fitting process itself, which complicates some steps of the fitting and testing process. For this reason, our framework allows probability distribution families to define a function indicating the subset of data points included in the fit. This will be a function of one or more parameters of the probability distribution. Because the likelihood of the data cannot be computed without the use of this function, such parameters must be determined before optimization. The user must mark such parameters as *non-optimizing* parameters, meaning that a brute-force process is performed where the optimization is performed once for every assignment of these parameters. (This methodology was used by [32] when determining x_{min} for the Pareto distribution.) Excluded data points are then omitted from the log-likelihood calculations, and are also omitted when computing the empirical CDF for the Kolmogorov-Smirnov statistic. (The probability distribution family's PMF and CDF must normalize the returned values so the included subset of values constitutes a valid probability distribution.)

Performing the Kolmogorov-Smirnov test becomes more complicated when using a subset of the data, because the synthetic datasets would lack values under x_{min} if they were only drawn from the fitted distribution. This is because the fitted distribution only describes the distribution of a certain subset of values, but the

original data may have contained values outside of this subset. We again use the procedure described by [32] to resolve this. First, the proportion of excluded values in the empirical dataset is determined. Then, when generating a random data point for a synthetic dataset, the framework will randomly (with the same probability as in the empirical dataset) decide that the data point being drawn will be an *excluded-subset* data point. Excluded-subset data points are drawn from the set of excluded data points in the empirical distribution, while non excluded-subset data points are drawn from the fitted distribution as usual. In other words, if the fit split the data into a head and a tail, data points in the synthetic distributions will be drawn from the original head or the fitted tail, at a probability equal to the proportions of empirical data points that were in the original head or tail.

2.6.2 Floating-point math and legal parameter ranges

Because numerical optimization algorithms are used to maximize the likelihood of the data and fit the distribution, the functionality of the algorithm depends on being able to compute the likelihood of the data at any point in the parameter space. Recall that either the BFGS method or the Nelder-Mead method can be used for optimization. The BFGS method requires that both the partial derivatives of the likelihood and the likelihood itself resolve to real numbers with any parameter assignment [26]. Under the BFGS method, the range of legal values for each parameter is defined separately (making the region of legal values an n-dimensional rectangle in the parameter space.) The Nelder-Mead method does not require the derivative to be defined, and it is permissible for the likelihood to be undefined for portions of the search space, meaning that a non-rectangular region of legal parameter assignments can be constructed using the cell-bounds method described in Section 2.3.5. However, the algorithm may not fully explore such a non-rectangular region. Both methods require that the likelihood be not be erroneous when defined.

This means that inaccurate likelihood computations and overflow errors while computing the likelihood must be avoided. Because the optimization process will evaluate many points in the search space without regard to their plausibility, the PMF calculation may involve extremely unlikely values (nearing the smallest values that can be represented with an R floating-point number.) Although the formula used to compute the PMF may be sound in an algebraic sense, R's computation of it may result in internal overflow or underflow, because the end result of the computation is too small to represent accurately, or because calculations may work with extremely large intermediate results (for example, in the denominator of a fraction.)

To make it easier to mitigate these problems, our framework expects probability distribution families to define the logarithm of their probability density function, meaning that some computations can be done entirely in logarithms, reducing the impact of working with very large or very small numbers. However, this is still not enough to mitigate the problem in some cases (for example, we experienced this when working with the Double Pareto Lognormal distribution, which subtracts two nearly identical quantities to obtain a very small probability – a computation which cannot be transformed to logarithms.) In these cases, the only way to ensure accurate results is to prevent problematic parameter assignments from being computed in the first place.

This makes it important to carefully define a range of legal values for each parameter. This range can be a function of the data (in this case, our framework will compute one set of legal parameter values for each dataset.) Because the result is an n-dimensional rectangle of legal parameter assignments, this only works when all of the undesirable assignments lie outside the rectangle and nearly all of the likely assignments are inside the rectangle. This is the case in most distributions; however, for some complicated distributions, such a rectangle cannot be defined for

some datasets. In this case, a non-rectangular region of parameter assignments must be defined. This can be done by preceding each PMF computation with an overflow check. If the input data will cause an overflow, the function should return infinity. When paired with the cell-bounds optimization described in Section 2.3.5, this will allow a complex region of plausible parameter assignments to be approximated by a cellular boundary. (Attempting to optimize within a cell that contains a non-computable point at any of its corners will cause the cell to be skipped.)

2.6.3 Parallelization

Many computations in our framework have been vectorized, because the R language contains constructs that encourage this programming style. This introduces opportunities to parallelize computations when running on multiprocessor machines, which could speed up the distribution fitting. We use the `multicore` R package [36] to parallelize vectorized computations. This package introduces a function that allows vectorized computations to be split up among multiple UNIX processes, where the number of processes is equal to the number of installed processors.

Our first attempt at parallelization was to analyze data from multiple wikis simultaneously, without parallelizing anything within the analysis of a single wiki. This was easy to implement, but it resulted in low CPU utilization because some wikis took dramatically longer to analyze than others, resulting in long computations that could not be shared between CPUs. This led us to instead parallelize the iterations of the Kolmogorov-Smirnov test, which speeds up goodness-of-fit testing and allows interactive users of the software to take advantage of the parallelization.

Note that the worker processes use copy-on-write memory management to share common data structures and avoid duplicating data already loaded into memory. This is important because the datasets could consume large amounts of memory, and also because before performing the K-S test iterations, the CDF of the fitted

distribution is precomputed (in order to draw synthetic datasets from it.) This CDF may be very large, but only one copy needs to be stored at a time.

2.6.4 Handling large datasets

The current version of our framework has been designed to work with datasets that easily fit into main memory. All datasets that we processed with the framework had fewer than 30,000 elements. However, it may be desirable to fit larger datasets to probability distributions, such as those derived from analyzing data from large wikis such as Wikipedia. This section describes the modifications that would have to be performed to the framework to handle such datasets.

When fitting large datasets, several operations will consume double the memory that the entire dataset consumes. This can be a problem if the dataset is too large to fit in memory (or if double the memory required to load the dataset is not available):

- Initial sort – The framework begins by making a sorted copy of the dataset, and deleting the original dataset.
- Subset extraction – If the algorithm will only fit a subset of the original dataset (as described in Section 2.6.1), the subset is explicitly computed and stored along with the original dataset
- Computing the log likelihood function and gradient – The entire dataset is passed as a vector to the vectorized probability distribution and gradient functions. The list of resulting likelihoods is as long as the dataset itself. This list is then summed to obtain the overall log likelihood. If the list is very long, the summation must be done such that internal loss of precision is not a problem.

Finally, even if the dataset does not have a massive number of values, a large amount of memory may be consumed when computing the Kolmogorov-Smirnov

statistic, because arrays must be allocated for both the empirical and fitted cumulative distributions, with one element for every integer that lies between the smallest and largest fitted data point. Because the CDF is increasing, points may be omitted from the cumulative distribution, because the maximum difference between the fitted CDF and the empirical CDF will occur either on a data point (the only points where the empirical CDF increases) or on the point immediately before a data point. Therefore, computing the CDF on and immediately before data points will capture both the maximum amount that the fitted cumulative distribution exceeds the empirical cumulative distribution, and vice versa. However, this process cannot be used when computing the large CDF used to draw synthetic datasets. To avoid running out of memory here, the maximum values in synthetic datasets could be constrained (see Section 2.7.1) or values could be drawn into synthetic datasets without pre-computing this large CDF (resulting in performance penalties.)

2.7 Threats to validity

Because the statistical test performed by our framework can be used to draw scientific conclusions from the data, it is important to ensure that the test is valid and well-understood. However, there are several pitfalls that could affect the results of the test if not considered, such as properly configuring user-definable tuning parameters.

2.7.1 Maximum values in synthetic datasets

As described above, our framework creates a number of synthetic datasets to perform the Kolmogorov-Smirnov test, by mapping a uniform random variable to the inverse function of the CDF of the fitted distribution. However, such a process could result in arbitrarily large values being chosen, because nearly all probability

distributions assign non-zero probabilities to arbitrarily large values. We discovered that choosing extremely large and unlikely values can rarely cause crashes or severe performance problems when computing the K-S statistic, due to the large cumulative distribution that must be computed.

Eliminating such values will have a minimal impact on the likelihood function (because they are extremely unlikely and are dominated by the other numbers being summed) or the K-S statistic (because both cumulative distributions converge to 1 at extremely large values, minimizing the potential for a large absolute difference between the two cumulative distributions to exist at these large values.) Therefore, we prevent extremely large values from being drawn into synthetic datasets. The authors of [32] prevented such values from being drawn by excluding data points from the synthetic dataset that are greater than $x_{max} \cdot 20$, where x_{max} is the largest data point in the empirical dataset. We take a similar approach, but we increase the set of allowable points by also allowing any point where the CDF evaluates to .995 or less, as long as such values are less than 100000. This ensures that sensible synthetic datasets are created in cases where the fitted distribution has a high probability of generating large values, while the empirical dataset contains no such values. These thresholds must be modified when analyzing datasets where values over 100000 could plausibly appear.

2.7.2 Choosing significance levels for statistical tests

The two statistical tests that we perform (described in Section 2.4.2 and Section 2.4.5) require choosing significance levels (previously defined as α and α_2) which control the Type-I error rate of the tests. A low significance level will normally reduce the Type-I error rate while increasing the Type-II error rate. The α significance level controls the rate that perfect distribution fits will be rejected when the K-S test is functioning properly, while the α_2 significance level will control that rate that

the K-S test malfunctions by not running the enough iterations of the Monte Carlo process. In this sense, α_2 is the maximum (worst-case) probability that the result of the K-S test would have changed if the distribution of the K-S statistic could have been directly computed instead of being inferred from the Monte Carlo process.

Because of the complex interplay between these two significance levels, it is no longer obvious that the Type-I error rate of the K-S test will still equal α. (Note that the Type-I error rate would differ from α slightly in any event, because the accept-reject threshold $\alpha \cdot n$ must be rounded to the nearest integer, causing the effective value of α to be different. However, here we are concerned about the effects of choosing α_2.)

To determine if this procedure results in a well-behaved statistical test, we performed simulations of our K-S test where the null hypothesis is always true. Because q is uniformly distributed across empirical datasets in this case (see Section 2.4.4), this simulation can be performed without actually drawing any synthetic datasets or fitting any data. For this reason, rather than doing Monte Carlo trials, we were able to track all possible binomial trial outcomes and compute their probabilities, allowing us to simulate our K-S test and compute an exact Type-I error rate for a perfectly fitting dataset with a given q. We then numerically integrated over all q using R's numerical integration function [26] in order to compute a Type-I error rate for the entire process with a given assignment of α and α_2. (The numerical integration estimated that the computed Type-I error rates were accurate to within 0.0040)

The results of this simulation can be seen in Figure 2.3 and Table 2.1. Note that the the Type-I error rate rises as expected when α is increased, and only small differences exist between the two values. We tried several reasonable choices α_2, but they did not have a large effect on this difference. Therefore, we observe that the significance level chosen when running our Kolmogorov-Smirnov test behaves in the

	α_2			
α	.01	.05	.1	.2
.05	.069	.067	.064	.064
.1	.100	.100	.099	.099
.15	.149	.148	.148	.145
.2	.198	.197	.196	.195
.25	.249	.247	.246	.245
.3	.300	.296	.295	.290

Table 2.1: Effects of α and α_2 on the Type-I error rate of the K-S test. Ideally, the Type-I error rate would be equal to α.

same way that significance levels are expected to behave in other statistical tests.

2.7.3 Proper functioning of optimization algorithms

The null hypothesis for the K-S test is that the data was drawn from a distribution identical to the fitted distribution. Note that this may not be the desired null hypothesis – it is often desirable to know if the data was drawn from from a given distribution *family*, and not just if the data was drawn from a given *distribution*.

The problem stems from the fact that optimization is not guaranteed to return the parameter assignment that results in the best fit. If the optimization algorithm returns a poor fit for a particular dataset, then the K-S test may reject the distribution fit, even if the data was drawn from the desired distribution family. Although the test's level of significance is guaranteed to be maintained (per the argument in Section 2.4.3), a poorly performing optimization function could lead to undesirable results (such as a dataset that was drawn from a specific probability distribution, but is incorrectly fitted to another distribution in the same family, and then rejected.)

Given some assumptions, though, we can informally argue that the K-S test will function expected given one of the following:

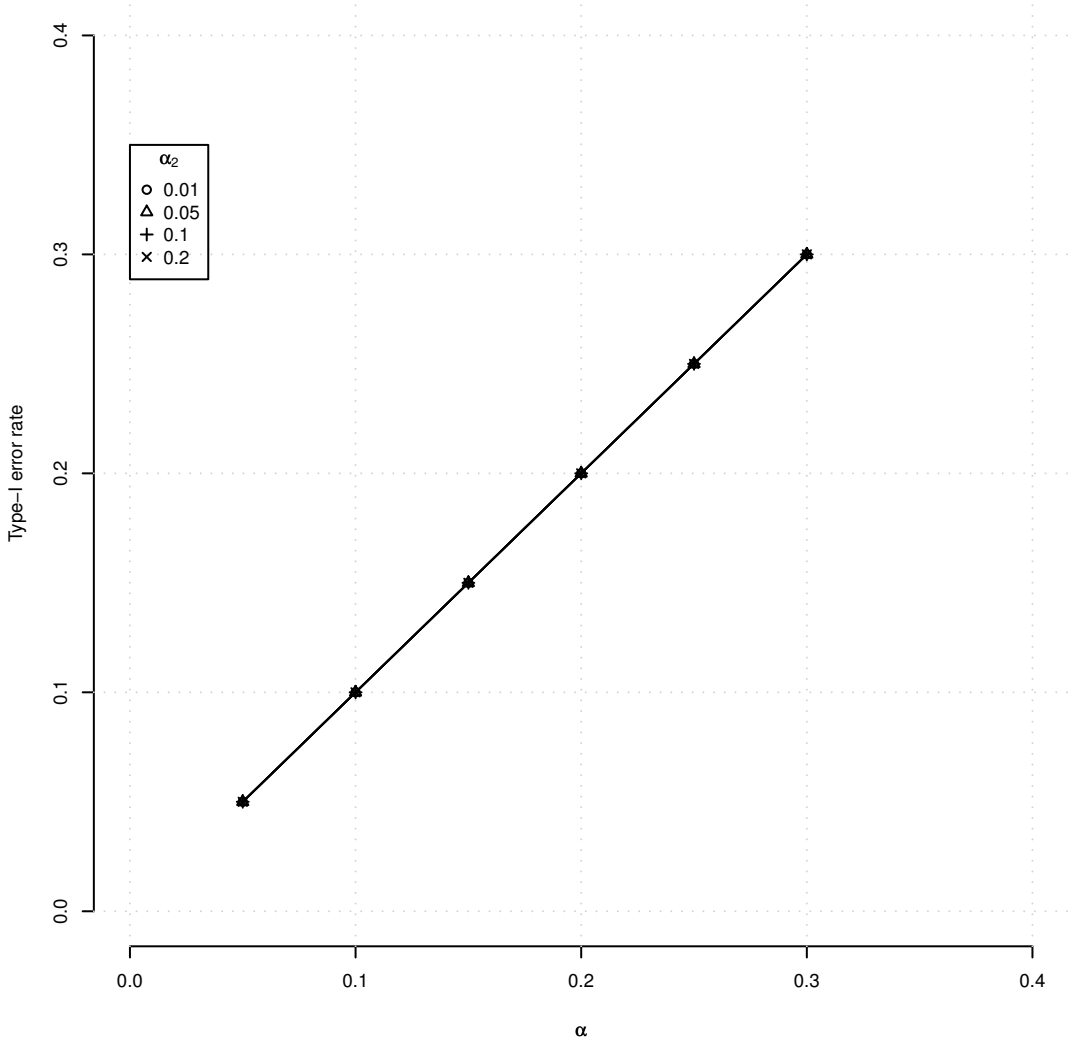

Figure 2.3: Effects of α and α_2 on the Type-I error rate of the K-S test. Ideally, the Type-I error rate would be equal to α.

- The optimization function returns an optimal solution
- The solutions are non-optimal by roughly the same amount throughout the parameter space

Note that the same optimization algorithm is used to optimize both the original and the synthetic datasets when performing the K-S test. Assume that the optimization algorithm performs poorly and results in a fit with a K-S statistic that is ϵ higher than desired. If the fits of the synthetic datasets are similarly poor and also result in fits with K-S statistics that are ϵ higher than desired, then the probability distribution fit will correctly be accepted or rejected.

Even if the optimization function experiences inconsistent errors, the Type I error rate will still equal α, as long as we state the null hypothesis appropriately (that the empirical data was drawn from the fitted distribution.) This is because, in the null hypothesis case, these inconsistent errors will affect every synthetic dataset (as well as the empirical dataset) equally.

This serves as further caution against misinterpreting the resulting p-value of the K-S test. The p-value cannot be interpreted as the "probability" that the distribution fit is valid. The only statement that we can make is that if the data was drawn from a distribution identical to the one that the fitting algorithm returned, then the probability that the goodness-of-fit test will reject the fit is exactly α, regardless how badly the optimization algorithm works.

2.7.4 Experimentwise error rates

Some experiments performed with our framework may entail fitting multiple datasets to a probability distribution, or fitting a dataset to multiple probability distributions, in order to discover which datasets have a statistically significant fit to which distributions. The dangers of performing multiple statistical tests in one

experiment are well-known because incorrectly handling the possibility of a false null hypothesis rejection could affect the integrity of the experiment. Methods to control this experimentwise error rate often entail computing the overall probability of a spurious null hypothesis rejection (Type I error) [41]. The probability of a single spurious null hypothesis rejection can be controlled by setting a conservative significance level α in the statistical test, which will be based on the number of tests that are being performed.

This is complicated in our case, because the null hypothesis is the hypothesis that the data came from the fitted distribution, and the null hypothesis is rejected by discovering that the data is not consistent with its fit. Therefore, Type I errors occur when a good fit is incorrectly rejected, while Type II errors occur when a fit is accepted even if the data was not drawn from the fitted distribution. The researcher can directly control the frequency of Type-I errors by adjusting the α of the test, while the probability of Type-II errors cannot be generally defined or expressed (as discussed in Section 2.4.3.) Because the Type-II error rate therefore cannot be directly controlled, the experimentwise error rate cannot be controlled either.

When performing multiple tests (which could entail fitting multiple probability distributions to one dataset, or fitting one distribution to multiple datasets), the probability of getting a Type I error will increase beyond the α. Normally, methods to control the experimentwise error rate will bring the Type I error rate down to α; however, these methods should not be applied to experiments that use our framework. This is because a more "conservative" experiment is one where fewer spurious fits are confirmed – in other words, increasing the Type I error rate actually makes the experiment *more conservative* in these cases. The problem lies in the fact that our tests cannot confirm that the empirical data came from the fitted distribution – they can only exclude the possibility, by determining that the data

is not consistent with the distribution. When the fitted distribution is not rejected, we can only conclude that the fit is *plausible*. Therefore, in our experiments, we have not adjusted for the experimentwise error rate.

2.8 Probability distribution families

In this section, we describe the probability distribution families that we implemented and included with our framework. Most of these distributions are continuous probability distributions, meaning that we apply the continuity correction described in Section 2.3.4 to the continuous equations.

2.8.1 The Pareto distribution

The Pareto distribution family contains distributions with density functions of the form [32]:

$$(\alpha - 1)x_{min}^{\alpha-1}x^{-\alpha}$$

The Pareto distribution family is important in Wiki research because researchers have often used it to describe the tails of heavy-tailed distributions generated by real-world processes. [32]

It has been claimed that this distribution has been found in data extracted from Wikipedia, such as the number of times that an article has been edited [5] or the number of distinct authors of an article [39], as well as in non-wiki applications such as the distribution of income or file sizes [19]. Such distributions arise from lower-bounded multiplicative processes, in the same way that the normal distribution arises from additive processes [19].

It has been observed that the tail of an empirical dataset often fits the Pareto distribution, without the head fitting the distribution. For this reason, our Pareto distribution fitting code only attempts to fit a subset $x >= x_{min}$ of data points.

This is because when taking the product of a Pareto function and a slowly varying function, the Pareto component of the function will dominate for large x [32]. Therefore, a wide variety of functions will successfully fit a Pareto distribution if only the tail of the distribution is considered. The procedure used to determine this x_{min} is described in Section 2.6.1.

In our later experiments, we exclude partial fits that only describe a small number of data points (when x_{min} is too large) because such fits can be statistically significant but practically uninteresting.

2.8.2 The Zeta distribution

The Zeta distribution family contains distributions with PDFs of the form [32]:

$$\frac{1}{\zeta(\alpha, x_{min})} x^{-\alpha}$$

where ζ is the Hurwitz zeta function.

The Zeta distribution family, is similar to the Pareto distribution family in the sense that the probability mass function will appear as a straight line on a log-log scale. However, unlike the Pareto distribution, it is a true discrete probability distribution, with a probability mass function only defined on the natural numbers. The Pareto distribution can be used to approximate the Zeta distribution for discrete datasets, in the same way that the normal distribution is used to approximate the binomial distribution. However, using the true Zeta distribution yields better results [32].

Use of the Zeta distribution is better for datasets with many small values, because it avoids the distorting effects of a continuity correction. However, its computation is slow because of the need to compute the generalized Zeta function.

2.8.3 The Pareto Lognormal distribution families

The Pareto Lognormal distribution family contains distributions with density functions of the form [27]:

$$f_1(x) = \alpha x^{-\alpha-1} A(\alpha, \nu, \tau) \Phi\left(\frac{\log x - \nu - \alpha\tau^2}{\tau}\right)$$

where Φ is the CDF of the normal distribution with mean 0 and standard deviation 1, and:

$$A(\theta, \nu, \tau) = \exp(\theta\nu + \frac{\theta^2\tau^2}{2})$$

The Double Pareto Lognormal distribution family contain distributions of the form [27]:

$$\frac{\beta}{\alpha+\beta} f_1(x) + \frac{\alpha}{\alpha+\beta} f_2(x)$$

where f_1 is as above, and f_2 is:

$$f_2(x) = \beta x^{\beta-1} A(-\beta, \nu, \tau) \Phi^c\left(\frac{\log x - \nu + \beta\tau^2}{\tau}\right)$$

where Φ^c is the complementary CDF of the normal distribution mentioned previously.

While fitting power-law (Pareto) tails to distributions has been effective in many situations, it can be considered unsatisfactory in the sense that the fit describes the tail of the data well, but doesn't describe the head. This indicates that, although the true distribution of the data has a Pareto component, the distribution has other components not described by a Pareto distribution. For these cases, we examine other distributions with Pareto-style tails, sometimes called *generalized Pareto distributions* [15]. These distributions exhibit Pareto-tail behavior when describing data points with large values, while using alternative shapes for smaller data points (in the head). We speculate that such distributions could fit entire datasets

when the Pareto distribution only fits the tail of the dataset.

The double Pareto lognormal distribution was suggested in [19] to be an alternative for data that has a power-law tail but a lognormal head. When used to analyze income distribution, this distribution describes a population, growing at a fixed rate, with a log-normally distributed income upon entering the workforce and income growth proportional to current income. [27] In computer science, the double Pareto distribution was used to propose a generative model for the distribution of file sizes [19], which were previously modeled with lognormal or Pareto distributions.

The Pareto lognormal distribution was developed by Colombi [27] as a predecessor to the double Pareto lognormal distribution to model the distribution of incomes. This distribution family describes distributions of the product of a Pareto-distributed random variable and a log-normal distributed random variable [15]. Out of general interest in generalized Pareto distributions, we also implemented the Pareto lognormal distribution in our framework.

2.8.4 The Levy distribution

The Levy distribution family contains distributions with density functions of the form [21]:

$$\sqrt{\frac{\gamma}{2\pi}}\frac{1}{(x-\delta)^{3/2}}\exp\left(-\frac{\gamma}{2(x-\delta)}\right)$$

Like the Pareto distribution, the Levy distribution has a shift parameter δ which data points below this parameter have zero probability. Unlike the Pareto distribution, the Levy distribution exhibits a power law tail while having a parabola-like shape in the head, raising the possibility that a dataset described by a fitted Pareto tail could better be described by this family of distributions.

2.8.5 Burr distributions

The 3-Parameter Burr distribution family contains distributions with density functions of the form [2]:

$$\frac{\alpha\beta}{x}\left(\frac{x}{\sigma}\right)^{\beta}\left(1+\left(\frac{x}{\sigma}\right)^{\beta}\right)^{-(\alpha+1)}$$

The 2-Parameter Burr distribution family contains distributions with density functions of the form [35]:

$$\frac{\alpha\beta}{x}x^{\beta}\left(1+x^{\beta}\right)^{-(\alpha+1)}$$

The families of Burr distributions are Pareto-family distributions that have been used to model stochastic processes such as inequality in household income. The 2-parameter Burr distribution is identical to the classical Burr Type XII distribution [35], while the 3-parameter distribution adds a scaling parameter as suggested by [2]. We implemented these distributions for the analysis of wiki data because the families of Burr distributions are generalized Pareto distributions [2].

2.8.6 The Log-Normal distribution

The log-normal distribution family contains distributions with density functions of the form [42]:

$$\frac{1}{S\sqrt{2\pi}x}\exp\left(-\frac{\left(\ln x-M\right)^{2}}{2S^{2}}\right)$$

We have chosen to implement log-normal distributions in our framework because past research has found that some quantities in Wikipedia are distributed in this way. It was found that the number of edits per article in Wikipedia is log-normal by a mechanism where the number of new edits is a varying proportion of the number of total edits [47], and that the distribution of Wikipedia article lengths is log-normal [39]. Stochastic processes which can be modeled as the product of

many independently distributed random variables can be modeled with log-normal distributions. [42]

2.8.7 The Log-Series distribution

The log-series distribution family contains distributions with PDFs of the form [43]:

$$-\frac{\theta^n}{n \ln(1-\theta)}$$

Although we have not found any research where log-series were used to fit quantities in wikis, this family was included in our framework for completeness because it is one of the few well-known discrete probability distributions. Because it is a truly discrete probability distribution, continuity corrections and similar adjustments do not need to be applied.

2.8.8 The Normal and Cauchy distributions

The Normal distribution family contains distributions with density functions of the form [21]:

$$\frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

The Cauchy distribution family contains distributions with density functions of the form:

$$\frac{1}{\pi}\frac{\gamma}{\gamma^2 + (x-\delta)^2}$$

The Normal and Cauchy distributions are not expected to be of interest to wiki researchers, but they were included in our framework for completeness. The properties of these distributions are well-known and will not be discussed here. Because these distributions have a non-zero probability of generating negative data, and our framework only works with non-negative, discrete data, the distributions

are truncated below 0 and the probabilities are multiplied by a constant such that the revised density function is a valid probability distribution despite the absence of these values. Although the resulting distribution is technically not a normal or Cauchy distribution, there is precedent for using the normal distribution to model non-negative phenomena in the real world (such as when the normal distribution is used to approximate the binomial distribution [41].)

Chapter 3

Collecting wiki data

The previous section of this paper described how the collected wiki data can be analyzed to uncover the distributions and mathematical models that describe the behavior of wiki users. In this section, we describe how data can be collected from wikis for analysis.

Previous wiki research has focused almost exclusively on Wikipedia, mostly because it is easy to collect and analyze its data. This is because Wikipedia makes database dumps available to researchers for analysis[1], while obtaining dumps of other wikis would require the cooperation of their individual webmasters. Also, because Wikipedia content is licensed under a free content license [2], research results can be published and shared without sanitization. However, the result of this Wikipedia focus is a lack of information on how wikis other than Wikipedia are being used, information that could increase the knowledge available to wiki practitioners by making the increasingly sophisticated models and analysis of Wikipedia applicable to wikis in general.

Much of this Wikipedia research has focused on descriptive statistical characteristics on Wikipedia, often describing patterns of behavior in Wikipedia by fitting

[1] `http://en.wikipedia.org/wiki/Wikipedia_database`
[2] `http://creativecommons.org/licenses/by-sa/3.0/`

probability distributions to Wikipedia data. Discovering such distribution fits is valuable in that knowing how a particular phenomenon is distributed gives some insight into the underlying user behavior that resulted in the pattern seen in the wiki. For example, Wilkinson and Huberman [46] proposed a stochastic model that produces a log-normal distribution for the lengths of articles with a given age.

Although conducting research on Wikipedia provides insight into one of the most popular websites on the internet, analyzing Wikipedia at the expense of other wikis results in a lack of findings that can be generalized. Because the wiki format itself may encourage common patterns of use and collaboration styles, quantitative analysis of large numbers of wikis can uncover behaviors and patterns common to many wikis, in the same way that quantitative analysis of Wikipedia gives insight into the underlying processes that resulted in the encyclopedia that exists today.

3.1 The WikiCrawler

Because wikis present themselves to the user as ordinary websites , machine-readable database dumps are not generally available. Clearly, some other means of data collection is required to perform automated quantitative analysis on large numbers of wikis. This led us to develop webcrawler-like software that facilitates the mass analysis of wiki pages. We have identified several desirable characteristics that a webcrawler that downloads wiki pages should have:

- The webcrawler can map wiki pages to the URLs that represent them, causing each page to be downloaded exactly once
- Instead of indiscriminately following links, the webcrawler should parse the appropriate indices to obtain lists of items to process
- The webcrawler should be sensitive to the data analysis that will be performed, and only download the pages required to make the required measurements.

Normal web crawler	**WikiCrawler**
All pages are parsed to find HREF link tags	Index pages are parsed to find lists of wiki pages
Text content is extracted from pages to enable searching	Statistics are extracted from pages
Crawler works on valid HTML pages	Crawler works on pages from a particular wiki

Table 3.1: Comparison between ordinary web crawlers and the WikiCrawler

- The webcrawler should be able to parse the contents of downloaded wiki pages to extract the data that will be analyzed
- The webcrawler should be scalable enough to complete studies that require data from hundreds of wikis and hundreds of thousands of pages

Note that the software we desire is quite different from general-purpose webcrawlers, which simply follow links to download as many HTML pages as possible [7]. In this sense, we have generalized the concept of web crawling to encompass the collection of domain-specific wiki data.

Because the WikiCrawler extracts wiki-specific features from wiki pages, we had to target a specific wiki platform and write our parsers to analyze wiki pages from that platform. We decided to support Mediawiki, a popular open-source platform used to host wikis. Mediawiki has gained popularity because it is the platform underlying Wikipedia, and the data collected from Mediawiki installations will be similar to data extracted from Wikipedia.

Despite the fact that Mediawiki permits extensive user interface customization, most Mediawiki installations use consistent HTML element IDs on their UI elements. For example, the two wikis depicted in Figure 3.1 look different, but similarities in the underlying HTML allow information to be extracted from both. There are a few variations in the HTML generated by different versions of Mediawiki, but most of

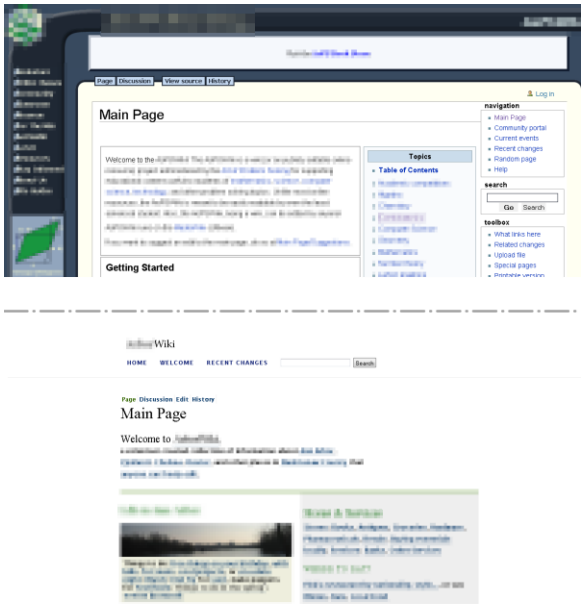

Figure 3.1: Two examples of wikis hosted by Mediawiki

the processing rules are identical across versions. For this reason, most Mediawiki wikis can be analyzed by the WikiCrawler automatically.

The WikiCrawler was developed as a part of our previously published research [34].

3.2 Architecture of the WikiCrawler

The WikiCrawler is a Java application which ingests a list of seed URLs, identifies wikis to study, downloads and parses all relevant data from the wikis, and exports the data for further analysis. All operations of the crawler are managed through an integrated workflow which parallelizes all download and processing activities. The overall architecture of the WikiCrawler can be seen in Figure 3.2. The WikiCrawler is built with an event-driven architecture, where components register event handlers with a data abstraction layer, which then triggers events when data objects are added or changed. In this way, there is a minimum of dependencies between WikiCrawler components, and components maintain a minimal amount of

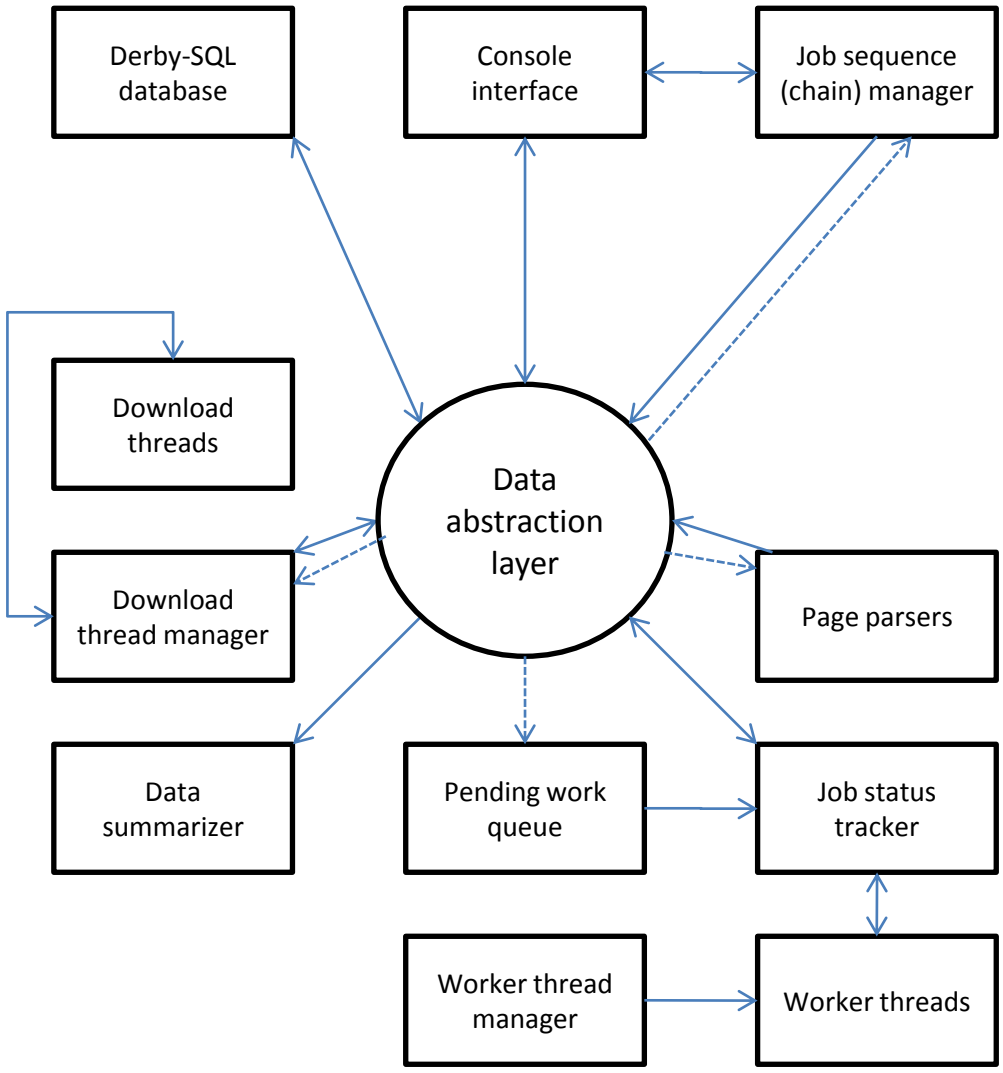

Figure 3.2: Block diagram showing the architecture of the WikiCrawler. All database access is mediated by a data abstraction layer that allows for concurrent access to the database by multiple threads while preventing sequences of calls that will lead to deadlocks. Solid arrows represent the flow of data between the components, while dotted lines represent events triggered by the data abstraction layer, which notifies components of the WikiCrawler that a particular piece of data has changed.

internal state, allowing the crawler to be parallelized if necessary.

WikiCrawler operations are performed on objects called *pages* and *sites*, representing individual wiki pages and entire wikis. The page objects keep track of any pending downloading or processing work that the WikiCrawler must perform, through the use of status variables indicating if pages are awaiting a download or awaiting processing. Pages and sites are externally represented by rows in an Apache Derby database and internally represented as data objects.

The operator of the crawler can initiate *jobs* that cause pages within a site to be downloaded and processed in a particular way. For example, one job counts the number of words in each article of the wiki, while another job counts the number of times each article has been revised. Both jobs will begin by downloading the contents of every page in the wiki, but the latter job requires the revision histories of the pages to be downloaded as well, so the revision histories will be downloaded after the page downloading is complete. In this way, the currently running job defines the actions to be taken after a page has been downloaded and parsed. Parser and job code is implemented in a hierarchical object model, allowing parsers and jobs written for one task to be easily reusable for other tasks.

Every site with activity has a status variable representing the currently running job. Different sites can have different jobs running at the same time, and the progress of a particular site's job is completely defined by the status variables of the pages within that site. In this way, our crawler architecture becomes scalable – machines with high network bandwidth can download 20 or more wiki pages simultaneously, while multiprocessing machines can parse and analyze multiple pages and wikis at once. Built-in limits prevent more than two pages from being downloaded from the same host simultaneously, to avoid overloading the servers that we crawl. Because the current crawler state is represented in the database, the crawler is resistant to crashes and reboots, although pages being processed during the crash are flagged

and must be cleaned up manually.

The crawler is controlled by a console interface, where we implemented a simple command language allowing objects within the crawler (pages and sites) to be examined and manipulated. *Summary* commands generate customizable status reports, while *selection* commands retrieve sets of pages and sites matching certain criteria. Hence, *manipulation* commands, which can modify objects and start jobs, are vectorized so they can operate on entire sets of pages and sites simultaneously. We found this to be useful as we debugged our crawler and troubleshooted issues with the wikis that we crawled, as described in Section 3.4.2.

3.3 Extracting data from HTML wiki pages

Wiki pages that the WikiCrawler downloads will be in the form of HTML documents. To operate in a manner that would extract machine-readable data from the pages, the crawler must perform two tasks: obtain URLs of pages that will contain the desired data, and extract the relevant data from the downloaded HTML pages.

Because the WikiCrawler only operates on wikis running the Mediawiki platform, the functions performed by the wiki software are common across all wikis. We must be able to retrieve: an article page (which contain the text of one wiki page), the list of all articles in the wiki, the list of all users in the wiki, the edit history of an article, and the edit history of a user. Because the WikiCrawler is not a general-purpose webcrawler and it only downloads pages containing the aforementioned data, the crawler must craft specific URLs that will contain the information of interest.

We use two strategies for constructing the URLs of pages to download. For the WikiCrawler to analyze a wiki, it must first be given a *seed* (or bootstrapping) URL that corresponds to some article in the wiki. When the page for the seed

URL is retrieved, two pieces of information are extracted from the page: the canonical title of the article (as presented to the user by Mediawiki) and the canonical URL of the retrieved page (because all Mediawiki pages contain a self-link that loads the current page again.) From the canonical title and the canonical URL of one page, the crawler constructs a template that allows arbitrary wiki pages to be retrieved by title. Although the same Mediawiki software was running on all sites that we analyzed, user customizations and web server configurations result in URLs being constructed differently. For example, the following URLs could refer to pages with the same title on three different wikis: `http://en.wikipedia.org/wiki/Main_page` `http://wiki2.example.com/wiki/index.php/Main_page` `http://wiki3.example.com/w/index.php?title=Main_page`

After inferring the mapping between URLs and articles in a wiki, the WikiCrawler gains the ability to generate URLs that retrieve the lists of pages and users in the wiki, as well as pages showing a list of edits made by a particular user. This is because Mediawiki puts these lists in the same namespace as ordinary articles, with titles such as `Special:All pages`.

All other URLs (such as URLs to retrieve the revision history of a page) are obtained by locating a link to the desired page on a page that has already been downloaded and parsed, and then reading the `href` attribute of this link. In this way, the WikiCrawler is immune to further changes and customizations in the URL formatting used by Mediawiki.

Features are extracted from the downloaded HTML documents by parsing the HTML documents and locating the individual HTML elements that are known to contain the desired data on Mediawiki installations. Although wikis may vary in appearance, all analyzed wikis are running the Mediawiki software, and appearance changes are usually implemented by writing custom CSS files, rather than modifying the HTML code itself. This means that the downloaded HTML is consistent across

Figure 3.3: A list of recent revisions to the Main Page on Wikipedia. Because the WikiCrawler parses these lists when collecting revision histories, the "newer" and "older" links are automatically followed for paging purposes.

wikis, and HTML element IDs that remain consistent can be used to find the desired data. For example, the link to the revision history of a wiki page is always contained in a `UI` element with an ID of `ca-history`. Downloading lists of things, such as revisions to a particular wiki page, is more challenging because retrieving the entire list requires paging (Figure 3.3). Each page of the list must be downloaded and analyzed separately, and the positions of the next/previous page buttons are inferred from the index parameter found in the button's URL parameters. (The text of the button is not used for this purpose because the text can be customized for localization.)

3.4 Experiences running the WikiCrawler

By gathering URLs, downloading wiki pages, and extracting features from them, it is possible for the WikiCrawler to collect all necessary data from a wiki without human intervention. However, we encountered some situations where these automatic processes failed and manual intervention was required. Here we describe these experiences:

3.4.1 Recovering from intermittent failures

File downloads from wikis sometimes failed because of a temporary server error or a connection timeout. Such failures resulted in empty or incomplete pages which failed to parse, preventing their associated sites from finishing their jobs. The WikiCrawler's command console allowed us to locate such pages and manually reset their states. A rare but troublesome failure mode was when the connection would get into a state where Java's built-in HTTP client would permanently lock up, despite exceeding the specified timeout. Recovering from this rare situation required us to manually stop and terminate the WikiCrawler.

3.4.2 Handling persistent errors

Due to minor bugs in the Mediawiki software or non-standard extensions to Mediawiki, some wikis had pages which would fail with a server error whenever we attempted to download the page. Dealing with this condition requires discretion – if the crawler is unable to download a legitimate page, data will be missing from the statistics we were trying to collect. A common cause of this error was a bug in Mediawiki that allows users to create a wiki page with illegal characters in the name, preventing the page from ever being viewed or modified. In these case, we manually bypassed these pages, because they did not represent legitimate data. However, in cases where many legitimate pages could not be viewed due to database corruption or another bug, we struck the entire site from our analysis (as seen in Section 4.1).

3.4.3 Handling modified wikis

Some wiki site administrators install Mediawiki plugins to add new features to the site. These plugins sometimes affected our analysis because they slightly changed the HTML structure of wiki pages, preventing us from parsing them. For

some common plugins, we added exceptions to our parser to detect the modifications and work around them. Other plugins (such as ones that restricted access to a segment of the wiki to a specific user group) prevented us from analyzing the wiki entirely, because they made it impossible to complete the data collection.

3.4.4 Detecting errors

To help ensure the integrity of the data, the WikiCrawler rejects data that is apparently erroneous, such as empty pages or users that have never shown any activity. Such situations sometimes arise because the wiki software experienced an error while retrieving the requested data; however, we discovered that these conditions sometimes exist in legitimate wikis. We worked around this by detecting the locale-specific messages that appear when these conditions legitimately exist. In cases where these messages are nonexistent or localized, we manually permitted the WikiCrawler to accept this data.

3.4.5 Mediawiki version differences

During our large-scale crawling of wiki pages, we discovered that the structure of HTML pages and URLs changed slightly between certain versions of Mediawiki. For example, when viewing the changes made by a particular user, older Mediawiki versions included a `go=prev` parameter in the URL of the button that moves to the previous page, while newer versions call the parameter `dir=prev`. These issues were discovered and corrected by manually inspecting pages that had failed parsing, modifying the parser to handle these cases, and then issuing commands through the WikiCrawler console to attempt parsing these pages again.

3.4.6 Wikis modified during crawling

One reoccurring problem that we experienced was dealing with wikis that were modified while they were being crawled. While we intend for our analysis to snapshot wikis at a point in time, it is likely that extremely large and active wikis will change during crawling because it can take 12-24 hours to download every page in a large wiki. This causes errors in cases where wiki pages were renamed or deleted after downloading their corresponding index entries, which results in failure of the crawler. In these cases, we instructed the crawler to re-download the index of pages, which usually triggered the downloading of several new pages that didn't exist in the previous index.

Chapter 4

Collecting a wiki corpus

We developed the WikiCrawler in order to build a corpus of wikis by downloading and parsing the contents of several hundred randomly chosen wikis across the internet. Aside from facilitating the analysis in the current research, this corpus is also available to the broader research community in order to explore other wiki-related research questions.

The data for this wiki corpus was collected as a part of our previously published research [34].

4.1 Finding wikis

In order to build a corpus of wikis, we first obtained a list of wikis to analyze and gathered seed URLs for these wikis for the crawler.

The S23[1] website, which has listings for tens of thousands of public wikis, has

[1] `http://s23.org/wikistats/`

been used for past research that compared multiple wikis, such as [29] and [30]. However, we did not use this site, or similar sites such as WikiIndex[2] because it was unclear if the wikis in the directories were collected by humans, which could result in a selection bias in our corpus. Similarly, we did not choose to obtain data from a wiki hosting service such as Wikia [3] [13] for similar reasons.

Instead, we used search engines to prepare a list of wikis that would be free from the bias of direct human selection. We obtained a list of candidate wikis to analyze by using the Yahoo and Microsoft Live search web services to retrieve the first 1000 results for the string `Main_Page`. Because Mediawiki creates a page called `Main_Page` by default, this allows us to easily find Mediawiki instances. Using search engines to form our sample ensures that the only bias is that the wikis were popular enough to appear in a search result.

Filtering non-wikis and duplicates between the two search engines out of our 2000 seed URLs, we found 1445 wikis which could potentially be analyzed. Wikimedia projects were excluded because they can be downloaded and analyzed more easily by using the database dumps mentioned earlier.

The Robot Exclusion Standard[4] allows websites to indicate that robots and crawlers should not visit. (This standard is not enforced by technical means; therefore, compliance is voluntary.) We rejected 77 of the 1445 wikis because our crawling would have violated this protocol.[5]

Because Mediawiki is an open-source product, users often customize it to improve the appearance of the wiki or add new features. These customizations sometimes confound the HTML parsing component of our crawler, which checks for inconsistencies in the downloaded pages to compensate for this. 182 sites were ex-

[2] `http://www.wikiindex.org/`

[3] `http://www.wikia.com/Wikia`

[4] `http://www.robotstxt.org/orig.html`

[5] This is an apparent contradiction, because our list of wikis originally came from search engines. This would mean that major search engines are violating this protocol, or that the wikis can be accessed from an alternate URL that falls outside of the restrictions.

cluded due to such inconsistencies, or because a password was required to access one or more wiki pages, preventing accurate statistics from being compiled. In addition, we manually removed 3 sites because they contained illegal or pornographic content, or because they were duplicates (which happens when multiple virtual hosts are backed by the same wiki.) In the end, 1183 wikis were available for analysis.

4.2 The study population

We estimated the sizes of the 1183 available wikis by examining the lists of main namespace articles. (The actual number of articles in the wiki is usually lower than this estimate, due to redirection pages that contain no content themselves, among other anomalies.)

The size estimates are depicted in Figure 4.1. Note that the sizes of wikis with more than 300 pages are distributed with a discrete power law distribution ($p = .284$, $\alpha = 1.77$, $x_{min} = 300$). This is consistent with the finding by Roth [29] that the sizes of wikis tracked on S23 have a power law distribution.

The number of wikis with less than 300 pages is smaller than what the power law distribution would predict, possibly because the search engines we used were more likely to return larger wikis than smaller ones.

4.3 Sampling a subset of wikis

Although we could have downloaded all 1183 wikis in our study population, this would have been excessive for this research because of the large amount of data to be collected (the 1183 wikis collectively contained over 6.2 million articles). For this reason, we selected a random sample of the wikis in our study population to build the corpus.[6] We first excluded wikis not falling within a desirable size range,

[6]Despite the small number of wikis analyzed, we still downloaded more than 820,000 HTML pages and the corpus consumes 33 GB on disk.

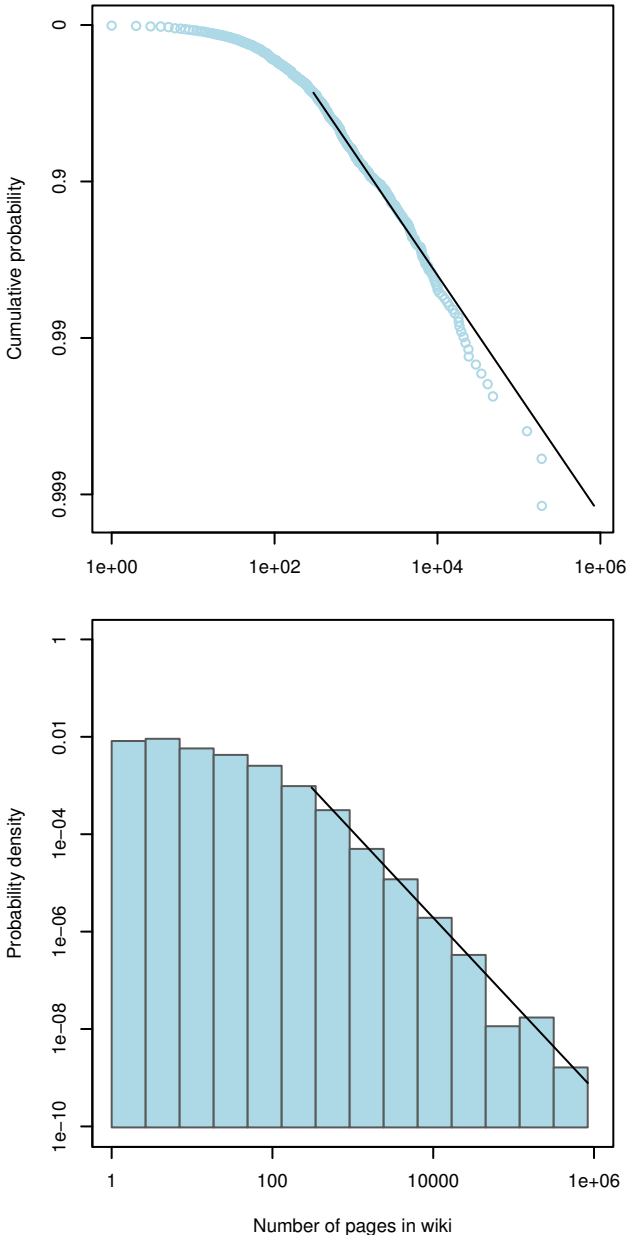

Figure 4.1: Sizes of wikis in our study population, depicted as a CDF (Cumulative Distribution Function) and a histogram. The goodness of the fit can easily be estimated by comparing the empirical CDF (plotted as dots) with the CDF of the fitted power law function, depicted as a straight line. (The Kolmogorov-Smirnov statistic can be visualized as the maximum vertical distance between the dots and line.) A histogram is also provided, as it is somewhat more intuitive for visualizing the distribution of the population.

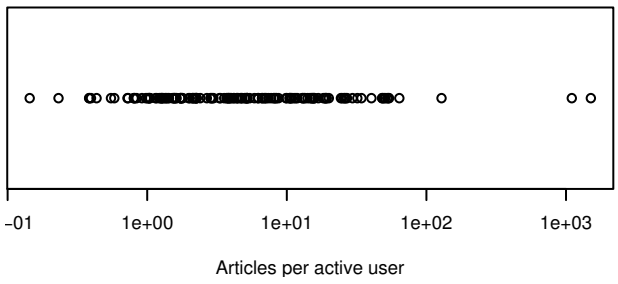

Figure 4.2: Number of articles per active user. Each dot represents one wiki.

only choosing among wikis with 50 or more pages (because very small datasets cannot be meaningfully analyzed) and fewer than 32000 pages. (In the case of wikis with more than 32000 pages, manually inspecting them revealed that they were large because they were generated by robots which converted large amounts of database data to a wiki format, instead of being organically created by a user base.)

We randomly selected 181 of the remaining wikis for our analysis. The processing of 30 wikis failed because persistent errors were experienced while crawling (see Section 3.4.2), leaving 151 wikis for analysis.

4.4 Choosing wikis to study

When studying a large population of wikis, it is important to determine if the wikis are being used in the way that wikis are intended to be used (for collaborative content development) or if many of them are being used as simple content-management systems that discourage editing by ordinary users. This situation was previously seen when excluding wikis with more than 32000 pages from the study sample, and we propose a metric to determine if other wikis are being used in the same way. For each wiki, we divided the number of articles by the number of active users and plotted the results in Figure 4.2. (Active users are users that edited the wiki at least once. We exclude inactive user accounts because they could have been automatically generated.)

This measure was distributed along a continuum of 0.4-54.0 articles per active user, with four outliers at 64, 128.3, 1100.0, and 1507.0 articles per active user. Inspecting the outliers shows that the third and fourth outlier wikis were automatically generated from a database while the first and second outlier wikis are knowledge bases that users cannot directly edit. Spot checks of other wikis (including the data points closest to the outliers) do not reveal any similar cases (One wiki with 52.3 articles per user was partially generated but had later attracted an active user base.) Therefore, we conclude that measuring the number of articles per active user is an effective way of detecting artificially generated wikis, and that the wikis in our sample are largely being authored collaboratively.

Chapter 5

An analysis of the collected data

After developing the WikiCrawler and generating a wiki dataset, we performed a preliminary analysis of the dataset, in order to explore some of its properties and demonstrate how more targeted experiments could be performed on the data. To facilitate this analysis, we wrote a toolkit in the R language to import the WikiCrawler data and compute some statistics (such as the Gini coefficient) on each of the wikis in the dataset. In conjunction with the probability distribution fitting framework described in Section 2, a complete wiki data analysis can be performed without leaving the R environment.

The analysis in Sections 5.1 to 5.5 was previously published by us. [34]

5.1 Summary statistics of the wiki corpus

It is useful to visualize the number of authors, users, and words in our sample of wikis, in order to observe the relationships between these basic measures. To do

this, we used the WikiCrawler to count the articles, registered users, and words in each wiki chosen for analysis. Figure 5.1 depicts the number of articles and users in each analyzed wiki, along with the average article sizes. (This final article count is not subject to the inaccuracies in the estimate mentioned previously.) Performing a Spearman's rank correlation test on the two depicted relationships shows a moderate correlation between user and article counts. ($p = 6.93 \cdot 10^{-10}$, $\rho = .474$), while the same test showed the number of users and the average article length to be slightly correlated ($p = .028$, $\rho = .178$). These results indicate that having a large user base is associated with a large number of articles appearing in a wiki, with a less clear association between the number of users and the lengths of existing articles. Studying if new users tend to create new articles over adding content to existing ones is a topic for further research.

5.2 Concentration of work across wiki users

Past wiki researchers have studied the concentration of work in Wikipedia (the degree that a small number of users is responsible for a large proportion of the content.) In one such study, Kittur *et al.* [12] examined the claim that Wikipedia reflects "the wisdom of crowds", and concluded that the most active, "elite" users are continually declining in influence. Other quantitative questions about wikis, such as the hypothesis that interest in Wikipedia is plateauing and the long-term viability of the project is threatened [22], could also benefit from measuring the concentration of work (because measuring *how* interest is distributed could add more insight than simply measuring the *quantity* of interest observed.) For this reason, we revisit these measurements of concentration of work, and apply them to a larger sample of wikis than Wikipedia alone.

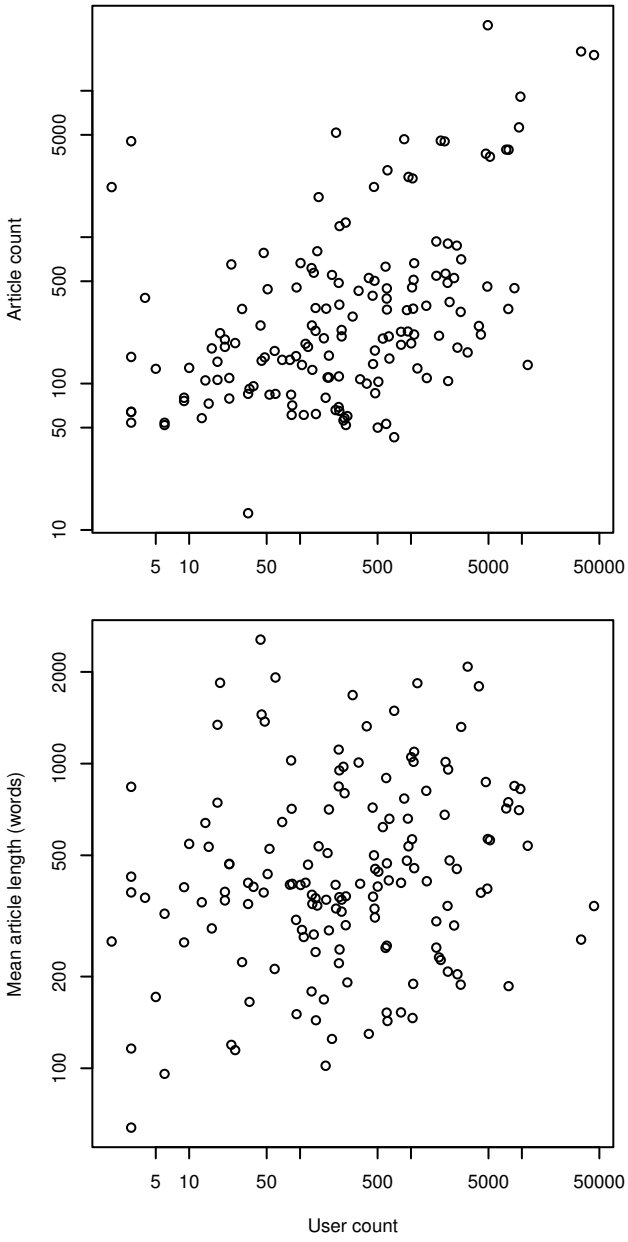

Figure 5.1: Numbers of articles and average article length of wikis compared with the number of users. Each dot represents one wiki.

5.3 Gini coefficients

Originally developed to measure economic inequality, wiki researchers have used the Gini coefficient to determine the degree that few users make a large proportion of the contributions to Wikipedia. The Gini coefficient is a unitless quantity that measures how fairly a variable (in our case, wiki edits) is distributed across a population (in our case, users or articles), with 0 signifying perfect equality and 1 signifying the largest possible inequality (which would mean that all edits are concentrated in a single user or article) [9].

To compute the Gini coefficient on a discrete dataset, we use the formula presented in [9], which is a revision of the traditional formula that reduces the bias for small datasets:

$$\frac{n \sum_{i,j} \|y_i - y_j\|}{2\,(n-1)\,n^2 \bar{y}}$$

where y is the sequence of values and n is its length. (Wikipedia research such as [25] did not include the $\frac{n}{n-1}$ correction factor when calculating the Gini coefficient, but its effect is infinitesimal in large datasets such as Wikipedia.)

Ortega *et al.* [23], noted that 90% of the revisions were made by 10% of the users of the English Wikipedia, resulting in a high Gini coefficient of .9360. Ortega indicates that this high degree of inequality persists on a monthly basis [25], casting doubt on the theory that wiki content represents "the wisdom of crowds"

To exclude wikis that were too small for a meaningful inequality measurement, we measured inequality in the subset of sampled wikis that had more than 50 active users and 300 pages (totaling 50 wikis). In Table 5.1, we present the inequality in the contribution levels of all users as measured by the Gini coefficients. A large amount of inequality was found, with most wikis having a Gini coefficient greater than .96, which is slightly higher than the Gini coefficients found for Wikipedia editions in [25] (although comparing Gini coefficients of very large and very small

$(.57, 0.88]$	3
$(0.88, 0.92]$	6
$(0.92, 0.96]$	11
$(0.96, 0.98]$	15
$(0.98, 1]$	15

Table 5.1: Gini coefficients of sampled wikis, measured across all users (0.0 would indicate that all users contributed equally while 1.0 would represent that a single user contributed everything.)

$(.36, 0.80]$	7
$(0.80, 0.84]$	6
$(0.84, 0.88]$	13
$(0.88, 0.92]$	16
$(0.92, 0.98]$	8

Table 5.2: Gini coefficients of sampled wikis, measured across active users

wikis can be misleading, as noted in [9]).

In Table 5.2, we calculate the Gini coefficients of wikis if inactive users (who never edited the wiki) are excluded. (The presence of these users may skew the results because wikis can share a user database with other information systems, resulting in the appearance of many inactive users.) When excluding users who never edited, the inequality is smaller but still notable, with a median above .87.

5.4 Power law behavior and user editing activity

The Gini coefficients of the wikis we studied showed that wiki users contribute content to wikis at highly unequal levels. While the Gini coefficient is good for measuring this inequality, it cannot further characterize this inequality or provide insight into why it exists. By expressing user contribution levels with a curve and fitting a probability distribution to this curve, more insight is provided into the data,

and it becomes possible to model the behavior of wiki users by considering models that generate data distributed according to the fitted probability distribution.

We first fit power law tails (as described in Section 2.8.1) to the distributions of user activity levels. We chose the Pareto distribution because it is often associated with high inequality, and because previous research in wikis has found that this distribution fits several quantities in Wikipedia, such as the number of edits to individual articles and the number of unique contributers to articles. Such distributions have also been found in a number of phenomena on the world wide web [17].

We performed power-law fits for the user contribution distributions of the same 50 wikis analyzed in Section 5.3. To prevent the discovery of trivial fits where a power-law tail encompasses only the last few data points, we restricted (see Section 2.6.1) the possible values of x_{min} so the power law distribution describes at least half of the unique values (or "counts") that were observed in the empirical distribution. All fits were performed at $\alpha = .10$ (done by [32] in a similar analysis) and $\alpha_2 = .01$, as described in Section 2.4.5.

We found that the user contribution distributions in 39 out of the 50 wikis had tails which were consistent with power law distributions. Figure 5.2 presents several examples of wikis that do and do not have clear power law distributions for user contributions. Note that in the lower-right wiki, the number of users making more than 24 edits is well-predicted by the power law fit, while the number of users making fewer edits would be overestimated by the fitted distribution. These partial fits have also been seen in [39], [6], and other Wikipedia studies that observed power law distributions in various phenomena.

To explore these cases where the empirical distribution deviates from power law behavior as the number of contributions shrinks, we compared the number of users represented by the unfitted portion of the empirical distributions with the number

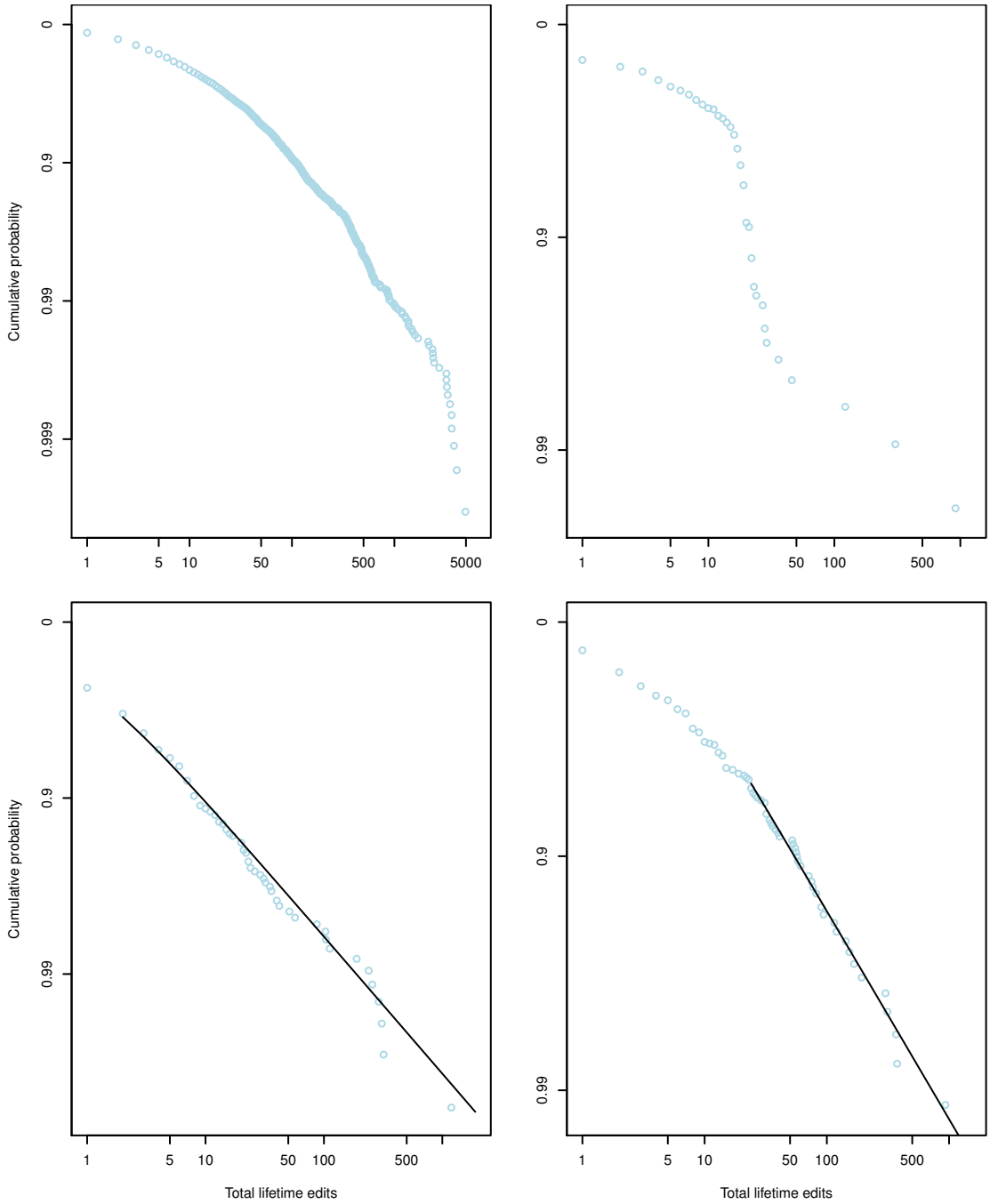

Figure 5.2: Sample cumulative distributions of user contribution counts. The top two graphs are examples of wikis where user contributions are not consistent with power law distributions, while the bottom two graphs are examples of wikis where they are. Note that the wiki in the lower-right quadrant had a partial fit for the power law tail. The graphs are to be interpreted as in Figure 4.1

of users that would have been expected in that portion if the Pareto distribution had held. 34 wikis had less probability mass than expected in this portion of the empirical distribution, while 6 wikis had more.

This indicates that if the studied wikis had more "occasional editors", then the fitted power-law tail would better describe the observed distribution. This could be a simple consequence of the Pareto being an inappropriate fit for the distribution, with the underlying distribution of edits conforming to another power-law family distribution with more flexibility in the curve's head. However, we also consider the possibility that the empirical data is deviating from the power law due to factors which discourage occasional contributions, which depress the count of users who made few edits. One such factor is that many wikis allow users to edit wiki pages without a username, which provides a convenience for the occasional contributer, but may be less attractive to frequent contributers who want their work to be recognized and attributed. Because it is difficult to associate anonymous contributions with an individual, the number of occasional editors may be undercounted, because many such users may have edited the wiki anonymously and therefore cannot be tracked. Other wikis that disallow anonymous contributions may introduce a barrier to entry that has the effect of deterring occasional contributions altogether, depressing the count of occasional contributers in a similar manner.

5.5 Relationship between Gini coefficients and power-law tails

We have seen that users make highly unequal amounts of wiki contributions as measured by the Gini coefficient, and that most of the user contribution distributions can be fitted with power-law tails. Next, we explore the relationship between the parameters of these power-law tails and the measured Gini coefficients, in order to determine the extent that these are two ways of measuring the same phenomenon.

For each wiki, we calculated the Gini coefficient of a synthetic power law tail

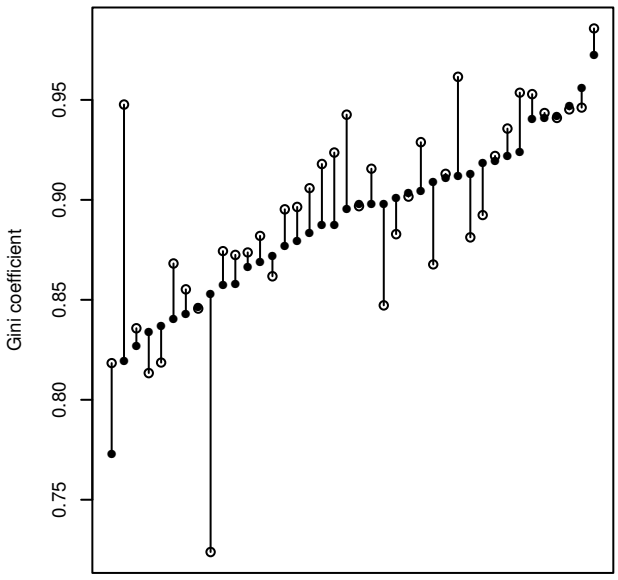

Figure 5.3: Changes in Gini coefficient. Closed circles represent the actual Gini coefficient of a wiki's user contribution levels, while open circles represent the Gini coefficient that would have been predicted from the parameters of the fitted power law distribution.

with the same distribution parameters as the fitted distribution for that wiki and the same number of users as in that wiki. We found that the Gini coefficients of the synthetic tails were very close to those of the actual data, with a median absolute difference of only .017. As seen in Figure 5.3, a few of the Gini coefficients rose or dropped sharply when switching to the synthetic dataset, but most changed only slightly, and all stayed within a general neighborhood (.13) of the original value.

Ortega [25] noted that in Wikipedia, the Gini coefficient of work performed within successive time intervals slowly rises and eventually stabilizes. This was cited as evidence that the contribution levels of Wikipedia users are not equalizing over time, as claimed in [12]. We have demonstrated that the Gini coefficient is mostly determined by the number of users being measured and the parameters of the power law distribution that describes their concentration of work. This suggests that monitoring the changing parameters of the user contribution distribution may offer provide additional insight into the evolution of inequality in Wikipedia, insight

that cannot be gained by only monitoring the Gini coefficients.

5.6 Other distributions in wiki data

In the Section 5.4, we demonstrated that 39 out of the 50 wikis that we studied had user contribution distributions with statistically significant power law tails. Next, we sought to extend this finding by attempting to fit a variety of probability distribution families to these same user contribution distributions. We also extended the scope of this investigation by fitting the distributions of edits per article (having previously fit distributions of edits per user).

Fitting a greater variety of probability distribution families to this data is useful because finding a pattern of such fits can motivate the development of a *generative model* of user activity that produces the observed distributions. To the extent that many of the studied wikis fit the same distribution family, such a model could be generalizable, providing researchers with another tool to explain how users behave when using wikis. With this aim, previous research has attempted to model the process in which Wikipedia articles accumulate edits. Wilkinson and Huberman [46] discovered that the number of edits per article within a time slice is log-normally distributed, leading to a proposed stochastic process that produces the empirical distribution of concentration of work across Wikipedia articles. Outside of wiki research, power-law fits led to the proposal of models that explained the appearance of numerous phenomena on the World Wide Web [17]. For example, one common model that explains how power law distributions develop in graphs is the *preferential attachment* model, which specifies that new nodes tend to attach to nodes with many edges [19].

As mentioned in Section 2.6.1, our power-law tail fits only required that a subset of the data $x \geq x_{min}$ conforms to the power-law distribution. Because of this, we attempted to fit other distributions to our user contributions data, despite

the fact that 39 out of 50 distributions were already found to have power law tails. Finding a distribution that describes all of the observed data (and not just a subset of it) makes the development of a model more practical. For example, Mitzenmacher [19] argued that the distribution of file sizes fits a double-Pareto distribution, rather than a lognormal or Pareto distribution as previously claimed, and this finding led to a new generative model for the growth of computer files. [20]

We used our distribution fitting framework to fit the user and article edit distributions in all 50 wikis studied in Section 5.3 to every distribution described in Section 2.8. The framework's goodness-of-fit test then rejected some of the fits which were statistically insignificant at $\alpha = .10$ and $\alpha_2 = .01$, only leaving us with fits that were worth of further study. So the fitting algorithm would return after a reasonable amount of time, we set a maximum of 360 iterations, even if the binomial test's p-value was still above α_2 at that point. 60 out of the 1100 goodness-of-fit tests terminated early for this reason. As in Section 5.4, we also restricted partial distribution fits so the fitted distribution describes at least half of the unique values (or "counts") that were observed in the empirical distribution.

The results of this fitting process are depicted in Table 5.3 and in Figures 5.4 and 5.5, which present the number of wikis where the indicated probability distribution could plausibly be fitted. In the event that all of the wikis conformed to a hypothesized distribution perfectly, we would expect that 45 out of 50 of the wiki distributions would be plausible, because the the K-S test will reject perfect distributions at a rate α. Because it is unlikely that every wiki in our sample has identically and perfectly distributed data, the number of plausible distribution fits for a distribution should be less than 45.

Our results show that simple power law tails are the most broadly plausible distributions for both edits per user and edits per article. The discrete power law performed better for the edits per user (coming close to the theoretical maximum

Distribution	Edits per user	Edits per article
Levy	27	15
Double Pareto Lognormal	12	17
Pareto Lognormal	6	8
Log Series	1	1
3-parameter Burr	17	11
2-parameter Burr	11	1
Pareto	5	32
Zeta	39	34
Lognormal	3	12
Normal	0	0
Cauchy	1	6

Table 5.3: Number of wikis where the indicated distribution was plausible given the indicated dataset. A total of 50 wikis were processed.

of 45 plausible distribution fits), while the both the discrete and continuous power laws performed better for the edits per article. (The edits per article distribution contained larger values than the edits per user distribution, making the use of the continuous approximation to the discrete power law distribution more appropriate.)

We also found that, when considering the distribution of edits per user, the Levy distribution is a plausible distribution for the power-law tails that we discovered, as opposed to the simpler Pareto or Zeta distributions. In this distribution, 27 of the tails were consistent with the Levy distribution. In many cases (see Table 5.4), the tail of the distribution was consistent with both the Levy and the Pareto/Zeta distribution. This is not a contradiction because the K-S test only confirms that a fit is plausible for the data – it cannot confirm that the data was actually drawn from the given distribution. Because Levy distributions have power-law tails, it is possible that data could be described by both Zeta and Levy tails, with the Levy tail encompassing more data points. Our data showed a slight tendency toward

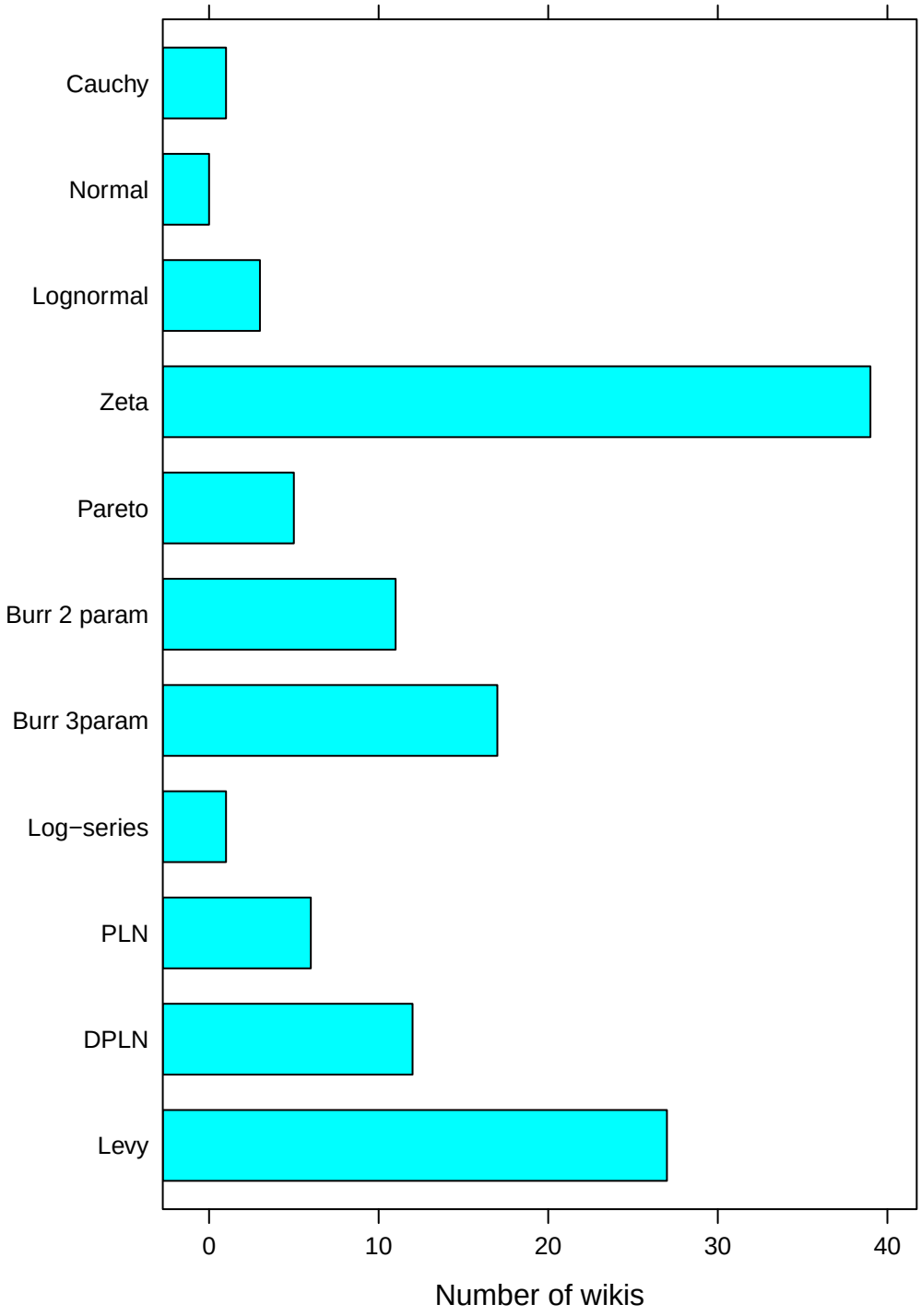

Figure 5.4: Number of wikis where the indicated distribution was plausible for the edits per user. A total of 50 wikis were processed.

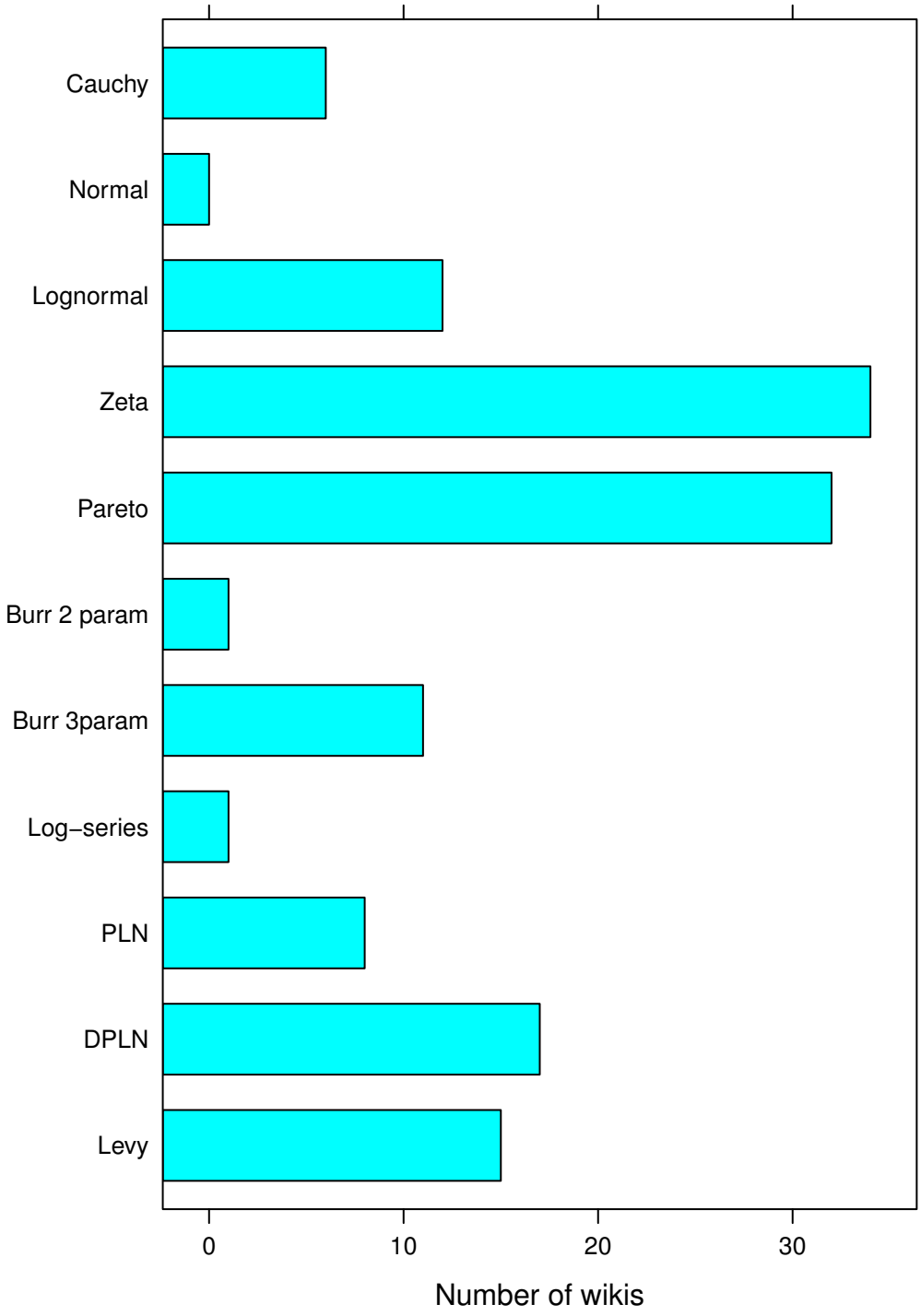

Figure 5.5: Number of wikis where the indicated distribution was plausible for the edits per page. A total of 50 wikis were processed.

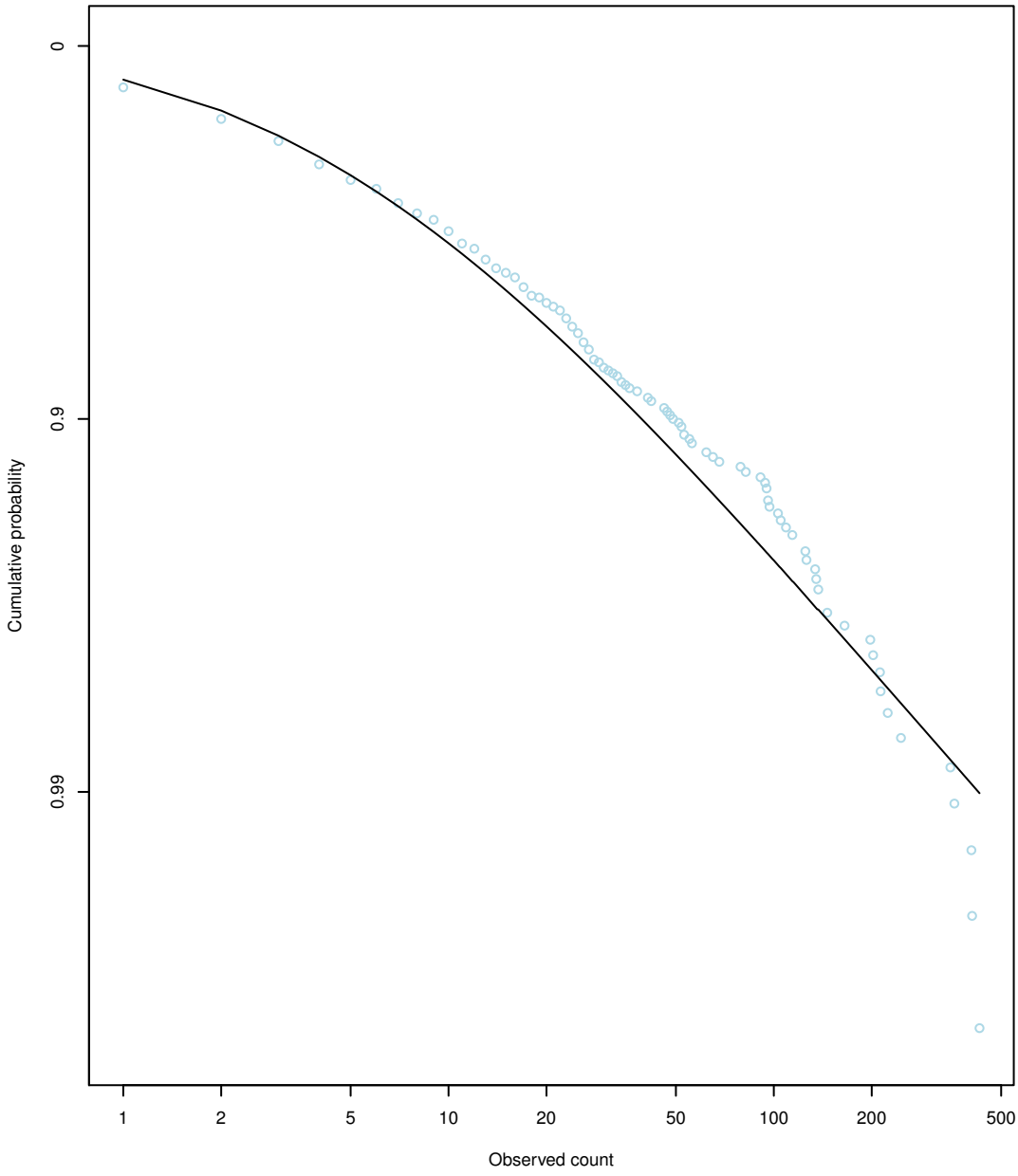

Figure 5.6: One wiki's distribution of edits per user, as fit to the double Pareto lognormal distribution. This data did not fit the Pareto or Zeta distribution, but the double Pareto lognormal distribution was a satisfactory fit.

Neither Zeta nor Levy	9
Zeta but not Levy	14
Levy but not Zeta	2
Zeta and Levy	25

Table 5.4: Number of wikis where plausible Zeta or Levy tails were found in the distribution of edits per user. A total of 50 wikis were processed.

this – among datasets that could be described with both Levy and Pareto tails, the Levy tail, on average, encompassed more data points than the Zeta tail did, with a median difference of 23 and a mean difference of 15.08. To better determine if the Levy or Zeta distribution is a better fit for our data, we would ultimately have to perform a modified likelihood ratio test, as described in [32], which is outside the scope of this work.

Finally, the three-parameter Burr distribution (in the case of edits per user) and the Double Pareto Lognormal distribution (in the case of edits per article) fit a sizable number of datasets, with 17 plausible fits each. Because both of these distributions exhibit power-law behavior in their tails, this plausibility may indicate that the Burr or Double Pareto Lognormal distributions are similar to a (still unknown) distribution that fits most of the data. However, we cannot rule out the possibility that the plausible fits were discovered due to the K-S test being insufficiently powerful for these distribution families.

Chapter 6

Conclusions and Future work

A major contribution of this research was to develop tools and methodology that wiki researchers can use to draw further conclusions on large numbers of wikis, as well as a corpus of wikis that facilitates easy hypothesis testing in cases where our publicly released data is sufficient. The WikiCrawler and the probability distri-

bution testing framework are general enough that they could be used for a variety of quantitative wiki research. The limited research questions that we considered in our analysis should not be construed as everything that can be done with these tools and data.

The WikiCrawler allows researchers to study the characteristics of many wikis over time, allowing for research that predicts the future growth of wikis based on current characteristics. In [29], Roth *et al* tracked a few basic statistics on many wikis over a time period, concluding that wikis with small, active user bases grow more quickly. Because the WikiCrawler allows researchers to collect detailed page-level and summary statistics, the same types of analysis could be performed with far more sophistication. For example, researchers could use statistics such as Gini coefficients to predict future wiki popularity, and the popularity of individual pages could be predicted in a similar manner. This could lead to wiki assessment tools, allowing corporate wiki practitioners to measure the health of a wiki, based on the ultimate outcomes of wikis with similar statistical profiles.

Our tools could also facilitate future discoveries that define additional metrics, fit additional probability distributions to them, and use the results to develop models of user behavior. In cases where it is only necessary to know the probability distributions of the dataset's tails, additional tests could be performed to determine if Pareto or Levy distributions are more plausible (as discussed in Section 5.6.) If a distribution that fits the entire dataset (and not just the tail) is required, we found that the three-parameter Burr distribution or the Double Pareto Lognormal distribution may be suitable starting points for finding a distribution; however, better distribution fits may be found by more carefully choosing the quantity being distributed. For example, instead of simply fitting a distribution to the number of edits per wiki article, Wilkinson and Huberman [46] fitted a distribution to the number of Wikipedia edits to an article within a time slice. This methodology was more

successful, resulting in the discovery that this quantity was log-normally distributed. Similarly, fitting probability distributions to model the behavior of users on many wikis may be more successful when performing distribution fits on time-slices of data in this manner.

Finally, the WikiCrawler and the corpus of wiki data provide an easy way to revisit past Wikipedia research and determine if its conclusions can be extended to wikis in general. This research explored the relative importance of occasional and frequent contributers in wikis, revisiting work such as [23] and [25] which was limited to Wikipedia. Other Wikipedia research [14] [38] examined the behavior of Wikipedia users when they disagreed and engaged in "edit wars" over a particular article – a concept which could be generalized to other wikis. More Wikipedia research that could easily be generalized examines how authors self-organize into networks or cliques based on shared interests [11] [3]. In short, some of the previous research only focused on Wikipedia because of the convenient availability of data and tools to analyze it, and our data and tools allow for the same research to be applied to other wikis. (Note that some Wikipedia research involves the Wikipedia category taxonomy [10], the Wikipedia governance structure, or assessment of the quality of articles in Wikipedia [46]. Such research could not be generalized without some modification.)

Ultimately, this research should only be seen as a first step toward a greater understanding of wikis. Through the basic methodology that we introduce in this paper – analyzing existing wikis to provide information that will assist wiki practitioners in the future – we hope that future research findings will allow for wiki best practices to be empirically supported in a way not possible until now.

The software and data described in this thesis are downloadable from our web site. At the time of writing this thesis, the URL was `http://seam.cs.umd.edu/stuckman`.

Bibliography

[1] Method of small p-values. `http://www.quantitativeskills.com/sisa/papers/paper5.htm`. From Quantitative Skills – Consultancy for Research and Statistics.

[2] Pareto distribution. In Samuel Kotz, Norman Lloyd Johnson, and Campbell B. Read, editors, *Encyclopedia of statistical sciences*, volume 6. Wiley, 1985.

[3] Robert P. Biuk-Aghai. Visualizing co-authorship networks in online wikipedia. In *Communications and Information Technologies, 2006. ISCIT '06.*, pages 737–742, 9 2006.

[4] Erik Brynjolfsson, Yu Jeffrey Hu, and Michael D. Smith. From niches to riches: The anatomy of the long tail. *MIT Sloan Management Review*, 47(4):67–71, 2006.

[5] Luciana S. Buriol, Carlos Castillo, Debora Donato, Stefano Leonardi, and Stefano Millozzi. Temporal analysis of the wikigraph. In *WI '06: Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*, pages 45–51, Washington, DC, USA, 2006. IEEE Computer Society.

[6] Luciana S. Buriol, Carlos Castillo, Debora Donato, Stefano Leonardi, and Stefano Millozzi. Temporal analysis of the wikigraph. In *WI '06: Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*, pages 45–51. IEEE Computer Society, 2006.

[7] Junghoo Cho and Hector Garcia-Molina. Parallel crawlers. In *WWW '02: Proceedings of the 11th international conference on World Wide Web*, pages 124–135, New York, NY, USA, 2002. ACM.

[8] George Deltas. The small-sample bias of the gini coefficient: Results and implications for empirical research. *Review of Economics and Statistics*, 85(1):226–234, 2003.

[9] George Deltas. The small-sample bias of the gini coefficient: Results and implications for empirical research. *Review of Economics and Statistics*, 85(1):226–234, 2003.

[10] Todd Holloway, Miran Bozicevic, and Katy Börner. Analyzing and visualizing the semantic coverage of wikipedia and its authors: Research articles. *Complex.*, 12(3):30–40, 2007.

[11] Rut Jesus, Martin Schwartz, and Sune Lehmann. Bipartite networks of wikipedia's articles and authors: a meso-level approach. In *WikiSym '09: Proceedings of the 5th International Symposium on Wikis and Open Collaboration*, pages 1–10, New York, NY, USA, 2009. ACM.

[12] A. Kittur, E. H. Chi, B. A. Pendleton, B. Suh, and T. Mytkowicz. Power of the few vs. wisdom of the crowd: Wikipedia and the rise of the bourgeoisie. In *CHI '07: Proceedings of the Computer/Human Interaction Conference*. ACM, 2007.

[13] Aniket Kittur and Robert E. Kraut. Beyond wikipedia: coordination and conflict in online production groups. In *CSCW '10: Proceedings of the 2010 ACM conference on Computer supported cooperative work*, pages 215–224, New York, NY, USA, 2010. ACM.

[14] Aniket Kittur, Bongwon Suh, Bryan A. Pendleton, and Ed H. Chi. He says, she says: conflict and coordination in wikipedia. In *CHI '07: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 453–462, New York, NY, USA, 2007. ACM.

[15] Christian Kleiber and Samuel Kotz. *Statistical Size Distributions in Economics and Actuarial Sciences*. Wiley, 2003.

[16] Frank J. Jr. Massey. The kolmogorov-smirnov test for goodness of fit. *Journal of the American Statistical Association*, 46(253):68–78, 1951.

[17] Alberto Medina, Ibrahim Matta, and John Byers. On the origin of power laws in internet topologies. *SIGCOMM Comput. Commun. Rev.*, 30(2):18–28, 2000.

[18] Ronald E. Miller. *Optimization: Foundations and applications*. Wiley, 1999.

[19] Michael Mitzenmacher. A brief history of generative models for power law and lognormal distributions. *Internet Mathematics*, 1:226–251, 2004.

[20] Michael Mitzenmacher. Dynamic models for file sizes and double pareto distributions. *Internet Mathematics*, 1(3):305–333, 2004.

[21] J. P. Nolan. *Stable Distributions - Models for Heavy Tailed Data*. Birkhäuser, Boston, 2010. In progress, Chapter 1 online at academic2.american.edu/∼jpnolan.

[22] Felipe Ortega. *Wikipedia: A Quantiative Analysis*. PhD thesis, Universidad Rey Juan Carlos, 2009.

[23] Felipe Ortega and Jesus M. Gonzalez-Barahona. Quantitative analysis of the wikipedia community of users. In *WikiSym '07: Proceedings of the 2007 international symposium on Wikis*, pages 75–86. ACM, 2007.

[24] Felipe Ortega, Jesus M. Gonzalez-Barahona, and Gregorio Robles. The top-ten wikipedias - a quantitative analysis using wikixray. In *ICSOFT 2007: Proceedings of the 2nd international conference on software and data technologies*, pages 46–53, 2007.

[25] Felipe Ortega, Jesus M. Gonzalez-Barahona, and Gregorio Robles. On the inequality of contributions to wikipedia. In *HICSS '08: Proceedings of the Proceedings of the 41st Annual Hawaii International Conference on System Sciences*, page 304. IEEE Computer Society, 2008.

[26] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2009. ISBN 3-900051-07-0.

[27] William J. Reed and Murray Jorgensen. The double pareto-lognormal distribution – a new parametric model for size distributions. *Communications in Statistics - Theory and Methods*, 33(8):1733–1753, 2004.

[28] Vito Ricci. Fitting distributions with R. `http://cran.r-project.org/doc/contrib/Ricci-distributions-en.pdf`, 2005. CRAN document.

[29] Camille Roth. Viable wikis: struggle for life in the wikisphere. In *WikiSym '07: Proceedings of the 2007 international symposium on Wikis*, pages 119–124. ACM, 2007.

[30] Camille Roth, Dario Taraborelli, and Nigel Gilbert. Measuring wiki viability. In *WikiSym '08: Proceedings of the 2008 international symposium on Wikis*. ACM, 2008.

[31] Thomas P. Ryan. *Modern Engineering Statistics*. Wiley, 2007.

[32] Cosma R. Shalizi and M.E.J. Newman. Power-law distributions in empirical data. *SIAM Review, to appear*, 2009. arXiv:0706.1062.

[33] Sander Spek, Eric O. Postma, and H. Jaap van den Herik. Wikipedia: organisation from a bottom-up approach. *CoRR*, abs/cs/0611068, 2006. `http://arxiv.org/abs/cs/0611068`.

[34] Jeff Stuckman and James Purtilo. Measuring the wikisphere. In *WikiSym '09: Proceedings of the 5th International Symposium on Wikis and Open Collaboration*, New York, NY, USA, 2009. ACM.

[35] Pandu R. Tadikamalla. A look at the burr and related distributions. *International Statistical Review*, 48(3):337–344, 1980.

[36] Simon Urbanek. *multicore: Parallel processing of R code on machines with multiple cores or CPUs*. R package version 0.1-3.

[37] Adriaan van den Bos. *Parameter estimation for scientists and engineers*. Wiley, 2007.

[38] Fernanda B. Viégas, Martin Wattenberg, and Kushal Dave. Studying cooperation and conflict between authors with history flow visualizations. In *CHI '04: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 575–582, New York, NY, USA, 2004. ACM.

[39] J. Voss. Measuring wikipedia. In *International Conference of the International Society for Scientometrics and Informetrics*, Katholieke Universiteit Leuven, Leuven, Belgium, 2005. International Society for Scientometrics and Informetrics.

[40] Glenn A. Walker. *Common Statistical Methods for Clinical Research with SAS Examples*. SAS Publishing, 2002.

[41] Ronald E. Walpole, Raymond H. Myers, Sharon L. Myers, and Keying Ye. *Probability and statistics for engineers and scientists*. Prentice Hall, 2002.

[42] Eric W. Weisstein. Log normal distribution. `http://mathworld.wolfram.com/LogNormalDistribution.html`. From MathWorld – A Wolfram Web Resource.

[43] Eric W. Weisstein. Log-series distribution. `http://mathworld.wolfram.com/Log-SeriesDistribution.html`. From MathWorld – A Wolfram Web Resource.

[44] Eric W. Weisstein. Maximum likelihood. `http://mathworld.wolfram.com/MaximumLikelihood.html`. From MathWorld – A Wolfram Web Resource.

[45] Eric W. Weisstein. Stochastic optimization. `http://mathworld.wolfram.com/StochasticOptimization.html`. From MathWorld – A Wolfram Web Resource.

[46] Dennis M. Wilkinson and Bernardo A. Huberman. Assessing the value of cooperation in wikipedia. *First Monday*, 12(4), 4 2007.

[47] Dennis M. Wilkinson and Bernardo A. Huberman. Cooperation and quality in wikipedia. In *WikiSym '07: Proceedings of the 2007 international symposium on Wikis*, pages 157–164, New York, NY, USA, 2007. ACM.