2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR)

A Panel Data Set of Cryptocurrency Development Activity on GitHub

Rijnard van Tonder*, Asher Trockman†, Claire Le Goues*
*Carnegie Mellon University, †University of Evansville
rvt@cs.cmu.edu, asher.trockman@gmail.com, clegoues@cs.cmu.edu

***Abstract*—Cryptocurrencies are a significant development in recent years, featuring in global news, the financial sector, and academic research. They also hold a significant presence in open source development, comprising some of the most popular repositories on GitHub. Their openly developed software artifacts thus present a unique and exclusive avenue to quantitatively observe human activity, effort, and software growth for cryptocurrencies. Our data set marks the first concentrated effort toward high-fidelity panel data of cryptocurrency development for a wide range of metrics. The data set is foremost a quantitative measure of developer activity for budding open source cryptocurrency development. We collect metrics like daily commits, contributors, lines of code changes, stars, forks, and subscribers. We also include financial data for each cryptocurrency: the daily price and market capitalization. The data set includes data for 236 cryptocurrencies for 380 days (roughly January 2018 to January 2019). We discuss particularly interesting research opportunities for this combination of data, and release new tooling to enable continuing data collection for future research opportunities as development and application of cryptocurrencies mature.**

***Index Terms*—cryptocurrency, open source software, github, software metrics, software quality**

I. Introduction

Cryptocurrencies are a significant development in recent years, featuring in global news, the financial sector, and academic research [1]–[3]. In the last three years, hundreds of new cryptocurrencies have appeared and many adopt an open source approach to software development. Popular cryptocurrencies hold a significant presence in open source development. For example, software repositories for Bitcoin and Cardano rank in the top 10 most popular projects (by GitHub stars) for the C++ and Haskell languages, respectively. Software is central to how cryptocurrencies function, and thus their openly developed software artifacts present a unique and exclusive avenue to quantitatively observe human activity, effort, and software growth. Despite this opportunity, little effort has gone into collecting related data and metrics.

Our data set marks the first concentrated effort toward a panel (i.e., longitudinal) data set of cryptocurrency development for a wide range of metrics. Our data set is foremost a quantitative measure of developer activity for budding open source cryptocurrency development. We collect (among other features) the number of daily commits, contributors, lines of code changes, stars, forks, and subscribers for thousands of repositories over hundreds of cryptocurrency projects on GitHub. We also include financial data for each cryptocurrency: the daily price and market capitalization.

Our contribution includes new tooling for collecting, aggregating, and exploring the data metrics of interest. Our method actively polls GitHub and financial market data every 24 hours.[1] This enables (a) collecting exclusive data compared to existing archival techniques (e.g., GHTorrent [4] misses repository subscriber information, which is unfortunately not exposed in GitHub's event stream API), (b) online data collection from financial markets over time, and (c) configurable data aggregation and exploration over multiple repositories.

We describe our data collection method in Section II and data set content in Section III. We discuss on research applications in Section IV; Section V discusses potential future improvements, and concludes.

II. Data Collection

Data source. We collected all of our software metric data from GitHub, where the vast majority of open source cryptocurrency development takes place.[2] We collected cryptocurrency market capitalization, price, and volume from CoinMarketCap,[3] currently one of the leading sites indexing cryptocurrencies.

Metrics collected. Table I summarizes the metrics we collected for each repository every 24 hours. The metrics broadly communicate size and growth of projects (in terms of, e.g., developer activity or repository stars) and recency of active development over time. The number of repository forks, subscribers, and last update are all short-lived values over time: GitHub provides only a single field that is overwritten as the values are updated. Our data set is unique in storing the full spectrum of these values over time for 236 popular cryptocurrencies. The data set further includes the number of commits and lines of code added and deleted over the previous 24 hours. We also calculate the number of unique active developers by commit in the last 7 days on each day.

We also collect statistics calculated by GitHub (`/:repo/stats`), which are generally of a coarser granularity. GitHub statistics do not track popularity metrics, fine-grained developer activity, or live 24-hour updates. The GitHub statistic data is, however, convenient and complementary for tracking a subset of our finer-grained collection over a longer period of time. The `contributors` endpoint provides a list

[1]The time interval can be configured by the user.

[2]Other code hosts, like BitBucket and GitLab, are not currently supported. Some metrics that we track (e.g., stars) are not supported in BitBucket.

[3]https://coinmarketcap.com

2574-3864/19/$31.00 ©2019 IEEE
DOI 10.1109/MSR.2019.00037

Metric	GitHub Endpoint
# of repository stars # of repository forks # of repository subscribers Time since last repository update†	`/:repo`
# of commits in the last 24 hours # of contributors in the last 7 days	`/:repo/commits`
# of lines of code added & deleted in the last 24 hours	`/:repo/commits/:sha`
# of contributors during repository lifetime	`/:repo/stats/contributors`
# of commits in the last year	`/:repo/stats/participation`
# of lines of code added & deleted in the last year	`/:repo/stats/code_frequency`

TABLE I: Metrics collected for each repository every 24 hours and the associated GitHub endpoint we queried. Short-lived data that is not historically available on GitHub include stars, forks, subscribers, and repository update times. We record the update time (†) as the most recent update time by either recent commit time, or the update time reported by `:repo` (since the latter also considers, e.g., user updates to a repository description, but can sometimes be stale with respect to recent commits).

of contributors over a repository's lifetime. GitHub limits the contributor list size to 100 contributors. The `participation` endpoint provides the weekly aggregate number of commits for the past year. This data provides a coarse historic log of commit activity, but is also transient (bounded by the last 52 weeks). The `code_frequency` endpoint provides the weekly aggregate of code changes over the repository lifetime.

In addition to the time-varying data in Table I, we also recorded two pieces of static metadata for each respository: (1) whether it is a forked repository and (2) the programming language (determined by GitHub's Linguist library[4]).

A single repository may have multiple branches. Branches facilitate various development workflows (e.g., Gitflow) and certain branches may become stale over time. We collect data for the *default* branch of the repository as recorded by the `:repo` endpoint on GitHub (this is typically the master branch, but it sometimes deviates to accomodate alternative workflows). GitHub statistics are also calculated on the repository's default branch.

Included projects and collection timeline. We started collecting data on January 21, 2018. Our procedure to include projects went as follows: (1) We ranked the top 339 cryptocurrencies by market capitalization (as per CoinMarketCap) on January 21, 2018; (2) We manually went through the 339 cryptocurrency entries and confirmed those with repository links to GitHub (as listed on CoinMarketCap). Of these, 236 cryptocurrencies hosted code on GitHub; the remainder either did not develop open source artifacts, or were hosted on unsupported code hosting sites. We registered these projects in our data collection tool. On GitHub, cryptocurrency-related software is either developed in a single repository (associated with a single user GitHub account) or across multiple repositories (associated with a GitHub organization account). Our tool thus registers either an organization account or a single repository link per cryptocurrency. For organizations, our tool collects metrics across all associated repositories. We recorded the metrics in Table I for each repository every 24 hours. The data set cuts off on February 4, 2019. We did not add additional cryptocurrencies once data collection started (i.e., software metrics for new cryptocurrencies launched mid-year 2018 are excluded in the current data set).

Collection interval and method. We implemented a cron job that records metrics for each repository by actively polling the endpoints in Table I every 24 hours. This approach ensures that we (a) collect metrics that are not available on the event stream (e.g., watcher subscriptions) and (b) obtain live, fine-grained commit and code changes at 24-hour intervals. Although commit data can be retrieved retroactively, our data collection is "online" and robust against destructive history rewrites (like force pushes).

One challenge to live collection is that GitHub places a rate limit of 5000 API requests per GitHub application (or user token) per hour. Recording data for all 236 projects would require 29 user tokens if we started all of the requests simultaneously. However, since we chose to collect data at a 24 hour granularity we did not need to start them all at once. Instead, we split our retrieval jobs by cryptocurrency (i.e., GitHub account) across a 14 hour period so that we could reuse a token roughly every hour. In this setup, we needed three user tokens. Thus, other users of our tool require only three user tokens to capture at the same fidelity and number of repositories represented in our current data set. We recorded the cryptocurrency rankings listed by CoinMarketCap every 24 hours, available through CoinMarketCap's API (a single URL). This data includes market capitalization, price, and volume for each cryptocurrency. We ran our cron job on a small Digital Ocean droplet.[5]

Tool operation. We performed all data collection by implementing our own commandline tool[6] consisting of three commands.

The *data persistence* command saves the metrics in Table I (at the current time) to disk. The command accepts one or more cryptocurrency identifiers, and a GitHub authentication token to issue additional requests when rate limiting takes effect. Cryptocurrency identifiers (and associated GitHub organizations) are manually entered in a configuration file and easily extended.

The *export* command aggregates metrics over multiple repositories and exports data in simple CSV file format. We use a separate second command so that aggregation can be configured before emitting the CSV data. For example, we have found it useful to configure aggregation as including or excluding forked repositories.

[4]https://help.github.com/articles/about-repository-languages

[5]A single vCPU (running on Intel E5-2630 2.30GHz) and 1GB RAM.

[6]available at https://github.com/rvantonder/CryptOSS; https://doi.org/10.5281/zenodo.2595621

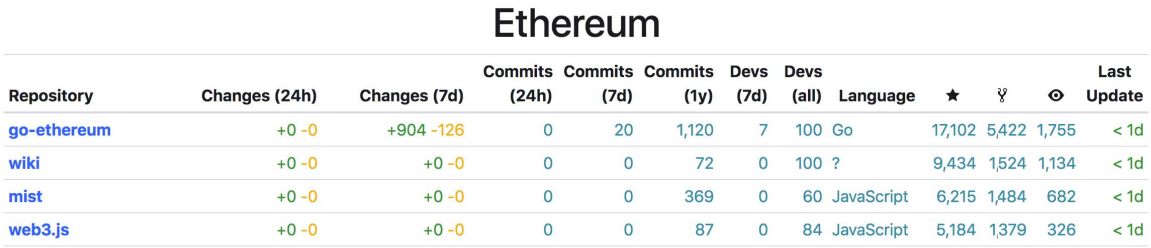

Fig. 1: A web view of just the top 4 Ethereum repositories, sorted by stars by clicking on the respective column. Our tool renders data in an interactive website for navigating across difference cryptocurrencies and sorting by various metrics.

The third command generates an HTML site from raw and aggregated data (see Fig 1). This is a convenience utility where users can interactively explore and debug collected data. The website renders tabular data for all cryptocurrencies on disk. The user can drill down and inspect individual metrics for all repositories belonging to a single cryptocurrency. All metrics, displayed by column, are individually sortable by clicking on the column header.

We implement a cron job and website generation script that wrap these three commands. The cron job polling interval can be configured by the user (for example, the metrics above can be collected every hour if desired, GitHub authentication tokens allowing).

We note that the data persistence command is broadly useful for actively polling the GitHub API at a chosen granularity to obtain fine-grained time series data, and is not restricted particularly to cryptocurrency repositories (the user can simply list alternative GitHub accounts in the configuration file). Similarly, we note potential for aggregating and exporting these metrics across repositories and organizations broadly.

III. Data Set Description

Our data set includes data collected for 236 projects, comprising more than 7,000 repositories over the course of 380 days. Various factors, not unfamiliar to mining software repositories,[7] affected data collection (e.g., projects moved on GitHub, network disruption, or our server temporarily running out of space). We thus normalized raw data for analysis convenience, and performed rudimentary recovery for lost data. The raw data set, processed data set, and recovered data set are made available at https://doi.org/10.5281/zenodo.2595588. We elaborate on challenges, data normalization, recovery, and content below.

Challenges and data recovery. We missed data for various reasons. Some cryptocurrencies changed the associated GitHub account (for example, Raiblocks rebranded to Nano), and we did not update our configuration to reflect such changes. The 24-hour commits over 7 days may not add up to the expected aggregate counts due to desynchronized queries or connections. We intermittently ran out of space on our server and unfortunately periodically overwrote data during compression (interspersed roughly once a month). In other cases, running large queries (e.g., for Ethereum) would unpredictably run out of memory. In total, we lost data for 44 days out of 380 days during active collection.[8] This loss is greatly mitigated by the fact that the data is generally recoverable. For example, historical commit data can be recovered from GitHub (as long as the repository has not been deleted). Other data, such as stars, may be profitably queried from GHTorrent. Existing sources present compelling ways to recover missing data. In practice, this entails synchrony with our current data (to avoid inconsistencies) or expensive operations (e.g., cloning thousands of repositories) to perform small queries. Instead of spending considerable effort on interfacing with external sources, we performed data recovery with the existing data as follows. Data such as stars, watchers, and subscribers tend to increase monotonically. Using this assumption, we checked whether these values stayed the same for days before and after a missing day. If they were the same, we imputed the same value for the missing day. For commits and lines of changes, we used the 7 day aggregate values to deduce some missing days. We recovered values for 19 of the 44 days (25 days have historically available, but unrecovered values).

Data set content and normalization. The final CSV file is 362MB[9] and contains the data from January 21, 2018 to February 4, 2019 for 236 projects. The file contains just over 3 million entries, one line per entry, with each entry corresponding to a single repository on a particular day. Each entry contains 24 comma-separated fields; the fields are listed in Listing II. To make the data set convenient for analysis, we normalized entries with null values across all days for all unique cryptocurrency and repository name pairs. Each date thus has 7,931 rows.

Aggregate metrics over multiple repositories per project can be insightful. Unfortunately, there is no easy way to attribute activity on forks to a project. In some cases, software development on forks deserve to be attributed to a particular cryptocurrency (e.g., the Litecoin fork of Bitcoin should be attributed to the Litecoin project). In other cases, forked dependencies (like `nixpkgs`) should generally not be attributed to a project. Our tool *can* generate aggregate values (choosing

[7]see "Data processing" at http://ghtorrent.org/faq.html.

[8]Missed dates can be found at https://doi.org/10.5281/zenodo.2595588

[9]The raw compressed data is 5.1GB.

date	cryptocurrency_name
symbol	market_cap_rank
price_usd	market_cap_usd
repo_name	language
is_forked	
stars	forks
watchers	last_updated
commits_24h	commits_7d
changes_24h_loc_added	changes_7d_loc_added
changes_24_loc_removed	changes_7d_loc_removed
contributors_7d	contributors_all_time
changes_1y	
commits_1y_loc_added	changes_1y_loc_removed

TABLE II: The 24 fields in the CSV data set. The **language** and **last_updated** are strings; **is_forked** is a boolean; all other fields are integers. The value of **contributors_all_time** is capped at 100.

whether to include forks or not), but because of these challenges we expose only whether a repository is forked in the CSV dataset.

IV. Research Applications

Although metrics such as lines of code and commit history do not necessarily speak to software quality (indeed, evaluating software quality is a longstanding research problem [5]–[8]), quantitative metrics can be useful as covariates for software quality predictors [9] and maintenance [10]. Moreover, developers use metrics such as star and commit counts as signals of popularity and quality on GitHub [11], and research suggests that popularity is positively associated with cryptocurrency prices [12].

Hence, as future work with our data set, we propose studies on the temporal relationship between open source development activity, popularity, and cryptocurrency financial data (e.g., as in [13]). This could involve regression modeling to find the most important correlates of cryptocurrency price, or more complex models such as recurrent neural networks to predict future price. Our data set could further facilitate the realistic testing of algorithmic trading strategies, similar to Garcia et al. [14].

Previous studies have investigated the intervention effect of adding continuous integration tools [15] and badges, a signal of quality [16], to open source projects. These studies use the interrupted time series design [17] to analyze the change in level and slope of a studied measure after such an intervention. Similarly, future studies could investigate the change in cryptocurrency prices associated with the addition of development tools or visible signals on repository pages.

Future studies could analyze the conditional variance of the collected measures in addition to the conditional mean; cryptocurrency prices may, for example, be more volatile in periods of intense development activity. Further, our data set could be paired with additional cryptocurrency data, such as the trading volume or number of active users [12]. One challenge is to leverage the small but potentially important differences in highly correlated metrics such as stars, forks, and watchers (a unique contribution of our data set).

Broadly, cryptocurrency prices provide one operationalization of value for corresponding software projects, which can be viewed in relation to different development practices [15], [16] programming language preferences, and governance structures [18]. We may ask: Do centralized teams tend to develop more valued cryptocurrencies than decentralized teams? To what extent does language preference or team size play a role? Our data set can help answer such questions.

V. Discussion and Conclusion

Our data set marks a first concentrated effort to quantify effort and interest behind the software development of cryptocurrencies. We include software and financial metrics for more than 7,000 repositories over a period of 380 days. We used a 24-hour active polling method to access transient data (e.g., subscribers); otherwise unavailable in existing data sets. Our method suffered some gaps during collection, though much of the missing data is historically recoverable. Going forward, additional infrastructure can ensure the quality and scale of data collection. An open challenge remains monitoring changes in cryptocurrency organizations (such as rebranding or renaming) and reflecting such changes in the data set. Unifying data collection and metrics over different code hosts (e.g., GitLab, BitBucket) presents a further challenge and opportunity for improving our data set. Finally, some cryptocurrencies do not have a single, definitive location for software development (e.g., Bitcoin Cash), making it a challenge to monitor activity and growth. Our hope is nevertheless that the initial data set in this paper spurs greater interest for the continuing collection and analysis of software artifacts behind the cryptocurrency phenomenon.

Acknowledgments

This work is partially supported under NSF grant number CCF-750116. All statements are those of the authors, and do not necessarily reflect the views of the funding agency.

References

[1] N. Abbatemarco, L. M. D. Rossi, and G. Salviotti, "An econometric model to estimate the value of a cryptocurrency network. the bitcoin case," in *European Conference on Information Systems: Beyond Digitization - Facets of Socio-Technical Change*, ser. ECIS '18, 2018, p. 164.

[2] J. Bonneau, A. Miller, J. Clark, A. Narayanan, J. A. Kroll, and E. W. Felten, "Sok: Research perspectives and challenges for bitcoin and cryptocurrencies," in *IEEE Symposium on Security and Privacy*, 2015, pp. 104–121.

[3] M. Apostolaki, A. Zohar, and L. Vanbever, "Hijacking bitcoin: Routing attacks on cryptocurrencies," in *IEEE Symposium on Security and Privacy*, 2017, pp. 375–392.

[4] G. Gousios, "The GHTorrent dataset and tool suite," in *Mining Software Repositories*, ser. MSR '13, 2013, pp. 233–236.

[5] E. Kalliamvakou, D. E. Damian, K. Blincoe, L. Singer, and D. M. Germán, "Open source-style collaborative development practices in commercial projects using github," in *International Conference on Software Engineering*, ser. ICSE '15, 2015, pp. 574–585.

[6] S. Wagner, K. Lochmann, L. Heinemann, M. Kläs, A. Trendowicz, R. Plösch, A. Seidl, A. Goeb, and J. Streit, "The quamoco product quality modelling and assessment approach," in *International Conference on Software Engineering*, ser. ICSE '12, 2012, pp. 1133–1142.

[7] A. J. Albrecht and J. E. G. Jr., "Software function, source lines of code, and development effort prediction: A software science validation," *IEEE Trans. Software Eng.*, vol. 9, no. 6, pp. 639–648, 1983.

[8] S. Slaughter, D. E. Harter, and M. S. Krishnan, "Evaluating the cost of software quality," *Commun. ACM*, vol. 41, no. 8, pp. 67–73, 1998.

[9] J. Rosenberg, "Some misconceptions about lines of code," in *International Software Metrics Symposium*, ser. METRICS '97, 1997, p. 137.

[10] D. M. Coleman, D. Ash, B. Lowther, and P. W. Oman, "Using metrics to evaluate software system maintainability," *IEEE Computer*, vol. 27, no. 8, pp. 44–49, 1994.

[11] L. Dabbish, C. Stuart, J. Tsay, and J. Herbsleb, "Social coding in GitHub: transparency and collaboration in an open software repository," in *Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work (CSCW)*. ACM, 2012, pp. 1277–1286.

[12] L. Kristoufek, "What are the main drivers of the bitcoin price? evidence from wavelet coherence analysis," *PloS one*, vol. 10, no. 4, p. e0123923, 2015.

[13] A. Trockman, R. van Tonder, and B. Vasilescu, "Striking Gold in Software Repositories? An Econometric Study of Cryptocurrencies on GitHub," in *International Conference on Mining Software Repositories*, ser. MSR '19, 2019.

[14] D. Garcia, C. J. Tessone, P. Mavrodiev, and N. Perony, "The digital traces of bubbles: feedback cycles between socio-economic signals in the bitcoin economy," *Journal of the Royal Society Interface*, vol. 11, no. 99, p. 20140623, 2014.

[15] Y. Zhao, Y. Zhou, A. Serebrenik, V. Filkov, and B. Vasilescu, "The impact of continuous integration on other software development practices: A large-scale empirical study," in *International Conference on Automated Software Engineering*, ser. ASE. IEEE, 2017, pp. 60–71.

[16] A. Trockman, S. Zhou, C. Kästner, and B. Vasilescu, "Adding sparkle to social coding: an empirical study of repository badges in the npm ecosystem," in *International Conference on Software Engineering (ICSE)*. ACM, 2018, pp. 511–522.

[17] W. R. Shadish, T. D. Cook, D. T. Campbell *et al.*, *Experimental and quasi-experimental designs for generalized causal inference/William R. Shedish, Thomas D. Cook, Donald T. Campbell.* Boston: Houghton Mifflin,, 2002.

[18] M. L. Markus, "The governance of free/open source software projects: monolithic, multidimensional, or configurational?" *Journal of Management & Governance*, vol. 11, no. 2, pp. 151–163, 2007.