Data Streaming Architecture for Visualizing Cryptocurrency Temporal Data

Ajay Bandi

Northwest Missouri State University, Maryville MO 64468, USA
ajay@nwmissouri.edu

Abstract. The utilization of data streaming is becoming essential in mobile computing applications to reduce latency and increase bandwidth. Vast amounts of data are generated continuously from the websites of stock markets and financial institutions. The data's meta-analysis is critical for investors and needs to analyze in a short time. Traditionally, it requires several heterogeneous resources with high storage capacity to process and compute the data. Data streaming helps to capture, pipeline, and compute the data without storing it. This research aims to visualize the continuous updates to the cryptocurrency temporal data using aggregations and simple response functions. The cryptocurrency data is collected from multiple data sources. A macro-enabled Excel external live data from web feature, C3.js, and Tableau tools are used to capture and pipeline the streamed data in real-time to make better decisions. The results show that the visualizations are dynamically updating in the events of trades in cryptocurrencies over time. Data streaming researchers and practitioners benefit from extending the streaming architecture methodology and dataflow to other domains.

Keywords: Mobile Edge Computing · Data Streaming · Cryptocurrency · Visualization · Architecture · Big Data

1 Introduction

In the context of big data, "data streaming" means a continuous generation of data from various data sources. It is also referred to or interchangeably used as streaming data [13]. In various domains, it is essential to observe and react to events in real-time. Events such as analyzing the trends for investing in stock markets, identifying unauthorized transactions in bank accounts, detecting cyber-attacks, analyzing social media posts for privacy and intelligence purposes, and monitoring sensor data to detect wear-out parts in machinery are a few examples.

The streaming solutions are necessary for high speed and voluminous data. Data streaming is essential for mobile edge computing (MEC), an alternative to centralized cloud computing, to reduce latency [10]. In MEC, the storage and computation happen on the Internet's edge of mobile devices to overcome network communication challenges such as latency and bandwidth. The leading global market intelligence of communications and technology, International

Data Corporation (IDC), forecasts that there will be 42 billion IoT devices, and the estimated data generated from these devices is 80 zettabytes ($8x10^{22}$) by the year 2025 [14]. Storing and computing such voluminous real-time data is highly challenging. Data streaming benefits the capturing, process, and bringing insights into massive data without storing it to improve MEC latency. The data stream processing requires latency in milliseconds to seconds using simple response functions and aggregation techniques. In contrast to traditional data warehousing, data streaming uses non-relational data to capture, pipeline, and then compute the results using simple functions.

The continuous data generated from mobile applications [1] such as IoT apps, sensors, and vehicular systems makes it hard to capture, process, analyze, and get insights from real-time data without storing it. In some instances of data streaming, responding to the events can be done automatically. However, there is a need for human intervention to detect the changes, especially in the temporal data, when there is a continuous generation of vast amounts of data. To overcome this, there is a need to visualize the data in real-time to monitor trends. In this paper, the proposed a method is to visualize the real-time temporal data of changes in different cryptocurrencies' prices using macro-enabled Excel and visualization tools. The rest of this manuscript is organized as follows: Section 2 presents the related work. Section 3 describes the methodology and the implementation of the visualization of the streaming data. Section 4 presents the results, and section 5 concludes with future work.

2 Related Work

Data streaming has wide variety of applications in various domains. Ehrlinger et al. [6] discussed industrial streaming data. They conducted a case study in a production plant that generates huge amounts of data. Their study dealt the stability of the production process using machine learning algorithms to handle semantic shift of streaming data. Data streaming has wide variety of applications in various domains. The literature of the data streaming architectures concentrated on reducing latency, accuracy, efficiency, resource allocation. Henning and Hasselbring [7] focused on reliability and scalability. They proposed a data stream aggregation based architecture to protect the scalability and reliability of the streaming process. The authors defined these quality attributes' requirements that can be considered for the multi-layer and multi-hierarchical systems. The suggested architecture is applied to an industrial case study and observed a linear relationship with the sensor data and reliability.

Tiburtino et al. [12] presented a new method called Xtreaming. This method aims to update the visualizations without visiting multidimensional data exactly once continuously. Yang et al. [15], and Ben-Elizer et al. [5] addressed the robust streaming algorithms in the insertion-only model. Ragan, Stamps, and Goodall [13] proposed a focus and context awareness to visualize the streaming data with visual aggregation technique. However, the data resolution was diminished to represent the context for a longer time. Their results shows a negative impact for

the contextual awareness in visualization of streaming data. With the emergence of cloud [11], and edge computing [4] applications, there is a need for data processing and visualizing without storing it. The data streaming aggregation, append-only techniques are used to visualize the cryptocurrency temporal data.

3 Methodology

The data processing lambda architecture is used to visualize the real-time change in cryptocurrency values [8]. The three main components of a lambda architecture [9] are the batch layer, speed layer, and service layer, as shown in Figure 1. The new data is fed to both the batch and speeding layers simultaneously from multiple data sources. These sources include but are not limited to IoT applications, mobile apps, vehicle communication with moving parts of traffic, satellites, sensors, and other devices. This speed component includes the inbuilt external streaming data function of the macro-enabled Excel. The batch layer has the complete data that is immutable, append-only, and serves as the historical data.

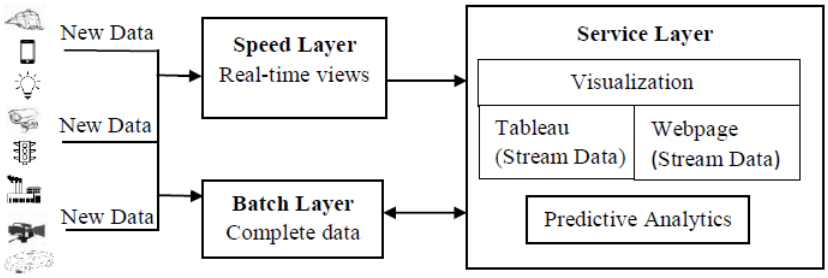

Fig. 1. Lambda architecture for data streaming

The speed layer contains the most newly added data not yet thoroughly recorded and classified by the serving layer. This layer holds the data that the service layer is currently recording and new data that arrived after the current indexing. It is common to notice the delay in the latest data added to the system and the same data queried by the service layer — technologies such as Apache Spark, Flink, Amazon Kinesis are used to reduce the latency between the speed and service layers. The service layer contains the regular additions of indexed data and is queriable for the end-users to visualize and perform predictive analytics. The standard queries' results are also updated to the complete data in the batch layer and direct usage to end-users.

The research goal is to visualize the streaming data. This research proposed the dataflow diagram, as shown in Figure 2, to visualize the continuous real-time data using lambda architecture. The horizontal cylinders represent the streamed

data. The next subsections describe the steps involved in the proposed dataflow diagram.

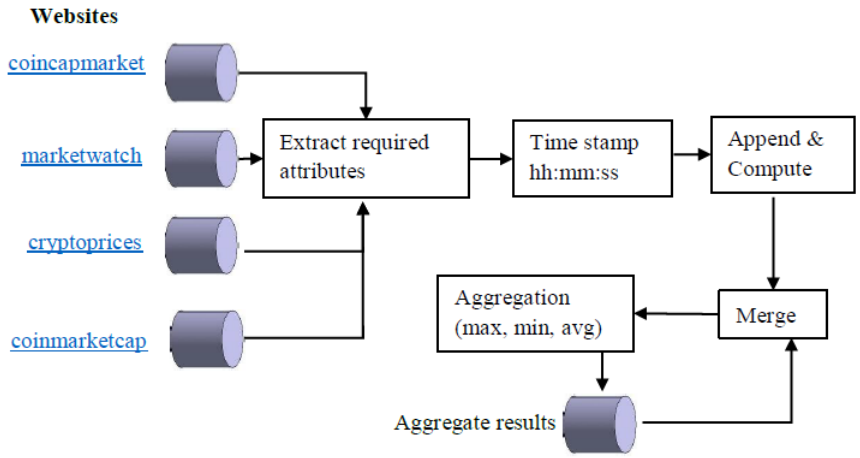

Fig. 2. Dataflow diagram of data streaming and computation

3.1 Tools Used

To implement a visualizing streaming data project, the following tools are used.

- Macro-enabled Excel – To extract live data from the websites. VBA and live data extraction. Visual Basic for Applications (VBA) Scprit is used to curate data, perform aggregation operations, conservation of Excel to .csv file, and connect it with web pages for visualizations.
- Papa Parse – It is a fast and powerful CSV parser for the browser that supports web developers and streaming large files.
- C3.js – D3 based reusable chart library for customized visualization purpose on the web pages
- Tableau – A business intelligence and visualization tool to visualize the temporal data using line and multiline charts.

3.2 Data

The different cryptocurrencies such as Bitcoin, Ethereum, Tether, XRP, Chain Link, Bitcoin Cash, Binance Coin, and Cardano, Litecoin, Polkadot, Wrapped Bitcoin, USD coin, DOW Jones industrial average's stock price are considered in this research. Also, the change in the percentage of the current price of these currencies is used for computations.

3.3 Data sources

The cryptocurrencies and the stock price are colletecd from multiple data sources. The data is scraped from multiple websites (coincapmarket.com, marketwatch.com, coinmarketcap.com, cryptoprices.com). These websites are chosen to collect similar attributes related to the goal of this research project. The extracted data is cryptocurrencies and the stock price that changes in real-time whenever there is a change in the market price.

3.4 Data extraction

A macro-enabled spreadsheet (Microsoft Excel) is used to extract the data. The price of Bitcoin, Ethereum, Tether, XRP, Chain Link, Bitcoin, Bitcoin Cash, Binance Coin, Cardano, Litecoin, Polkadot are extracted from the coincapmarket.com website. The Dow Jones industrial average's stock price from marketwatch.com website. The USD coin currency value from the coinmarketcap.com website. The Wrapped Bitcoin's value obtains from the cryptoprices.com website. The data from the four different websites are extracted into the four different Excel sheets in one Excel workbook. Also, the change in percentage of the current price is extracted and compared to the price before 24 hours. This attribute is used to perform the calculations on the streamed data.

To collect the real-time data, embed the website URL by clicking the Data tab under the "Get External Data" group, select "From Web." Provide the website URL in the "New Web Query" window to import the website's raw data. Similarly, import the raw data from other websites in separate sheets. Identify the selected cryptocurrency values and stock prices from the raw datasheets. The new sheet extracts the selected values from the four sheets using Excel formulae (Example: Sheet1!D155). The data in the new sheet is the cleaned data and is later used for the visualization. Then, set up the connection properties for live reloading for every one minute.

3.5 Insert timestamp

Since the cryptocurrencies change dynamically in real-time according to the market trend, the timestamp is include by using the Visual Basic for Applications (VBA) Script within the Excel using the short cut (Alt+F11). The time is represented in the hh:mm:ss format in a 24 hour clock. The timestamp is an essential attribute the temporal data to represent the changes. The significance of the timestamp in this research is to identify the unique record of the data and the corresponding cryptocurrency price.

3.6 Append and Compute

After executing the above steps, the initial values of cryptocurrencies and stock prices are recorded in the respective columns' first row. The initial values are the values extracted from the website for the first time after execution. Then,

append the values in the new row for every one minute. The new row may have two possible values. 1) The new price of the cryptocurrency or the stock is based on the market. 2) If the value is not changed, the previous value will be recorded at a new timestamp. In this way, the live stream data is recorded and appended in the Excel for further computations and analysis. In other words, The first column is filled with the initial values of all the cryptocurrencies. The second column is the timestamp. From the third column onwards, consider the transpose of the first column cryptocurrencies to stream the given timestamp's respective values. For example, if the first column has ten cryptocurrency values, there will be ten respective columns from column 3 to 13. The pseudo code for appending streaming data is given in algorithm 1.

Algorithm 1: Pseudo code for appending streaming data

```
Declare and initialize int LastFilledCell address of last filled cell in a
 column
      ▷ N is total number of different crypto currencies or the number of
 values in the column one
for int i=0; i<=N.size; i++ do
    if LastFilledCell.value != N[i] then
        Append to LastFilledCell+1.value = N[i].value;
End
```

The change in the value of the cryptocurrency price is calculated based on the change in percentage value. The result is the value of the cryptocurrency before 24 hours. The change in percentage value is streamed on the website and extracted during the data extraction phase. The calculated change in the value of the price will help the investor for better decision-making. For example, The value of the Bitcoin at 14:20:00 is $20,000, and the percentage change is +2% compared to the previous day at the same time. The change in value is $400.

3.7 Merge and Aggregation

While capturing the websites' streaming data, aggregation functions are used and calculated the average, maximum and minimum values of the Bitcoin and the timestamp. In the VBA Script, Average, Max, min functions of the `WorksheetFunction` class are used to calculate aggregation values. The computed values are merged in the original data and compared with newly added data to calculate the final results for streamed data.

3.8 Aggregation results

The results derived from the aggregation step is compared with the newly streamed data to calculate the new maximum and minimum values. Consider the overall past data to recalculate the new average of the streamed data entry. The VBA pseudo code for aggregation results is given algorithm 2.

Algorithm 2: Pseudo code for calculating aggregations on streaming data

```
Create a column named max in Excel
Declare and initialize int MaxColLastFilledCell address of last cell in
 max column
                                  ▷ N is total number of different crypto currencies
                                        ▷ column is size of a respective column
for (int i=0; i<=N.size; i++) do
    for (int j=0; j<=column.size; j++) do
        if MaxColLastFilledCell.value < N[i] then
            Append MaxColLastFilledCell.value = Max(column)
            Get timestamp and append

                                                                  ▷ to find min
int MinValue  9999.
                                ▷ this will be declared in excel in min column cell 1
Declare and initialize int MinLastFilledCell address of last cell in min
 column
                                  ▷ N is total number of different crypto currencies
                                        ▷ column is size of a respective column
]for (int i=0; i<=N.size; i++) do
    for (int j=0; j<=column.size; j++) do
        if MinLastFilledCell.value > N[i] then
        Append MinLastFilledCell.value = min(column);
        Get timestamp and append

                                                              ▷ To find Average
Declare and initialize int AvgLastFilledCell address of last cell in
 average column
Append AvgLastFilledCell.value = Average(column)
                                        ▷ column is size of a respective column
```

4 Results and Discussion

The cleaned dataset is used in the speed layer for real-time views and then pipelined to the service layer. Visualizations and prediction analytics are performed in the service layer. For the real-time visualization of the cryptocurrencies' temporal data, the first step is converting the macro-enabled Excel sheet into .csv files. The pseudo code for the conversion in VBA Script is given in algorithm 3. Then, the cleaned data in the .csv file is used to visualize on the web page. Papa Parse is used to parse the .csv files and C3.js to visualize. Papa Parse library is a fast and powerful CSV parser for the browser that supports web workers and streaming large files. It is easy to use and parse CSV files directly to local or over the network. C3 is a JavaScript library that builds on top of D3. C3 makes it easy to generate D3-based charts by wrapping the code

required to construct the entire chart. The pseudo code for visualizing streaming data using Papa Parse and C3.js is given in the algorithm 4.

Algorithm 3: This is a pseudo code for converting .xlsm to .csv file

```
Start:
Declare Excel.Worksheet ws;
Declare String SaveToDirectory;
Initialize SaveToDirectory as "Streaming-Visualization \Web
 Visualization \data"
foreach ws in ThisWorkbook.Worksheets do
 | ws.SaveAs SaveToDirectory & ws.Name, xlCSV
end
End
```

Algorithm 4: This is a pseudocode for visualizing streaming data.

```
Start:
function FUNCTION(parseData(createGraph))
Read the CleanedDataset.csv using Papa Parse library
Call createGraph (CleanedDataset.data)
End Function
   function FUNCTION(createGraph(data))
                          ▷ data is excel data from CleanedDataset.csv
Initialize time of an array type.
Initialize array variables for all the cryptocurrencies in
CleanedDataset.csv
for var i = 1; i < data.length-1; i++ do
 | Add the time stamp from the CleanedDataset.csv to time array
 | Add cryptocurrencies data into its respective arrays
end
Generate multiline chart using c3 library taking time on X-axis and
cryptocurrencies on Y-axis
       function FUNCTION(timedRefresh(timeoutPeriod))
 ▷ timeoutPeriod is the period of time in milliseconds for refreshing the
webpage
                  ▷ Web page will reload for every 60 seconds i.e. 6000ms
End Function
End
```

The visualizations of the streaming data are shown in Figures 3 and 4. These charts are dynamic, and the visualizations update every one minute. The webpage is developed to automatically refresh every minute to update the data dynamically in the visualizations. The change in the cryptocurrency prices is temporal data as it changes according to the market trend. The resulted visualizations compare various cryptocurrencies in a multiline chart, and the Bitcoin's price change is presented in the line chart. The user can filter whatever the cryptocurrency they want to view. Figure 4 shows the line chart of Bitcoin's trend after filtering it from other cryptocurrencies. The real-time visualization

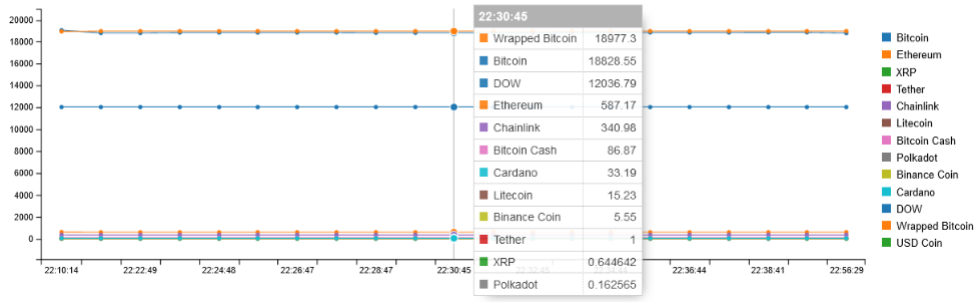

Fig. 3. Multiline chart for streamed data on a webpage

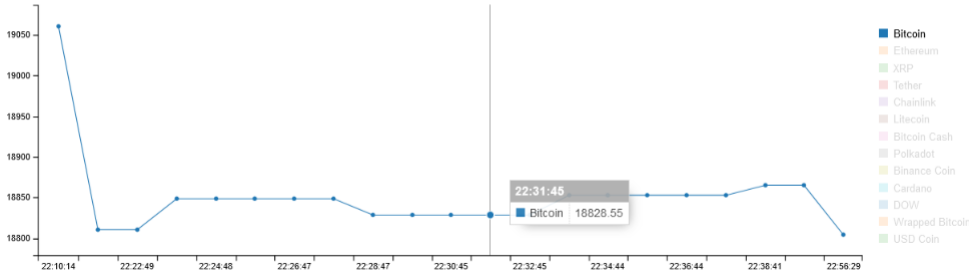

Fig. 4. Line chart for Bitcoin streamed data on a webpage

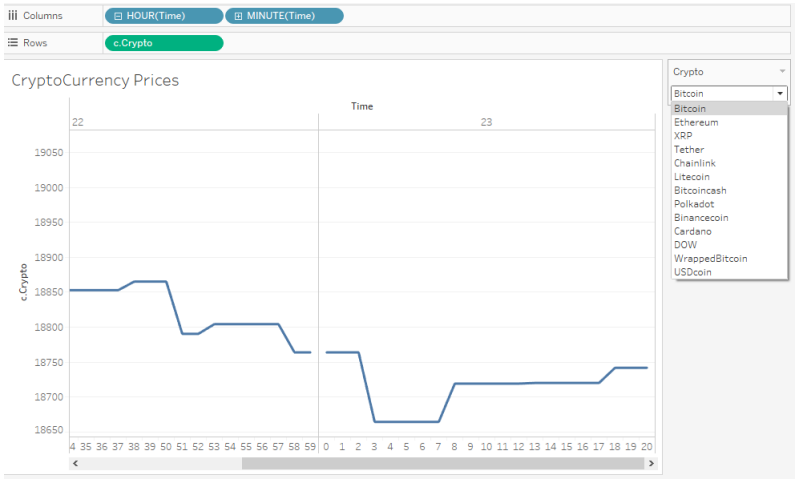

Fig. 5. Line chart for Bitcoin streamed data in Tableau

can be viewed in the video [2] and the project artifacts can found in the GitHub repository [3].

On the C3 chats, the Y-axis represents the dynamic increment of the time stamp, and it changes as the new data arrived for every minute. The X-axis represents the cryptocurrency value that automatically adjusts based on the change in the prices. The tool tip in Figures 3 and 4 shows the time stamp and the corresponding cryptocurrency values at that instance. In Tableau visualizations, the X-axis represents the time for every minute (0 to 59) for every hour. The Y-axis represents the cryptocurrency value. In Figure 5 the line chart shows from 22:35 to 23:20. After 22:59, the new hour 23 started and continued to represent the data for every minute to visualize the streaming data's dynamic changes.

5 Conclusions and Future Work

Software systems that analyze the continuously generated data, also called streaming data, need to visualize in real-time to make better decisions. This research presented a method to visualize cryptocurrencies that lambda architecture for data streaming. The implementation method uses aggregations, append-only, and simple response functions of data streaming for temporal data. The results show that the visualizations are updating for every minute dynamically. The implementation of the architecture and the artifacts are provided in the GitHub repository. In the future, this research will be extended to measure and reduce the latency along with the security aspects of the data streaming.

References

1. Bandi, A., Fellah, A.: Design issues for converting websites to mobile sites and apps: A case study. In: 2017 International Conference on Computing Methodologies and Communication (ICCMC). pp. 652–656 (2017). https://doi.org/10.1109/ICCMC.2017.8282547
2. Bandi, A.: https://youtu.be/Ivc6X4KalRc
3. Bandi, A.: https://github.com/bandiajay/DataStreamingVisualization
4. Bandi, A., Hurtado, J.A.: Edge computing as an architectural solution: An umbrella review. In: 26th Annual International Conference on Advanced Computing and Communications. Springer (2020)
5. Ben-Eliezer, O., Jayaram, R., Woodruff, D.P., Yogev, E.: A framework for adversarially robust streaming algorithms. In: Proceedings of the 39th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems. pp. 63–80 (2020)
6. Ehrlinger, L., Lettner, C., Himmelbauer, J.: Tackling semantic shift in industrial streaming data over time pp. 36–39 (2020)
7. Henning, S., Hasselbring, W.: Scalable and reliable multi-dimensional sensor data aggregation in data streaming architectures. Data-Enabled Discovery and Applications **4**(1), 1–12 (2020)
8. Horvat, N., Ivković, V., Todorović, N., Ivančević, V., Gajić, D., Luković, I.: Big data architecture for cryptocurrency real-time data processing (2020)
9. Lin, J.: The lambda and the kappa. IEEE Internet Computing **21**(5), 60–66 (2017). https://doi.org/10.1109/MIC.2017.3481351

10. Mahadev, S.: The emergence of edge computing. Computer **50**(1), 30–39 (2017)
11. Maraparedy, R., Bandi, A., Tirumala, S.S.: Cloud computing architectures: A retrospective study. Journal of Innovation in Computer Science and Engineering **2**(1), 1–5 (2012)
12. Neves, T.T., Martins, R.M., Coimbra, D.B., Kucher, K., Kerren, A., Paulovich, F.V.: Xtreaming: an incremental multidimensional projection technique and its application to streaming data. arXiv preprint arXiv:2003.09017 (2020)
13. Ragan, E.D., Stamps, A.S., Goodall, J.R.: Empirical study of focus-plus-context and aggregation techniques for the visualization of streaming data. In: Proceedings of the International Conference on Advanced Visual Interfaces. pp. 1–5 (2020)
14. Shirer, M., MacGillivray, C.: The growth in connected IoT devices is expected to generate 79.4zb of data in 2025, according to a new idc forecast, https://www.idc.com/getdoc.jsp?containerId=prUS45213219
15. Yang, R., Xu, D., Cheng, Y., Wang, Y., Zhang, D.: Streaming algorithms for robust submodular maximization. Discrete Applied Mathematics (2020)