Data Logging And Data Visualization

Using Perl

Special Assignment

Submitted by: 21bec084 Krinal Parmar

21bec104 Jeet Ranpara

Course code and name: 2EC501 Scripting Language

Submitted to: Prof. Vaishali Dhare

Abstract:

In the realm of data-driven decision-making, data logging and data visualization constitute fundamental elements. These tools enable the systematic collection and structured presentation of data, playing a pivotal role in facilitating informed decisions. This report explores the principles, applications, and methodologies associated with data logging and data visualization. Furthermore, it demonstrates the practical implementation of these concepts using the versatile Perl programming language, thereby illuminating a seamless path from data acquisition to meaningful visualization.

Introduction:

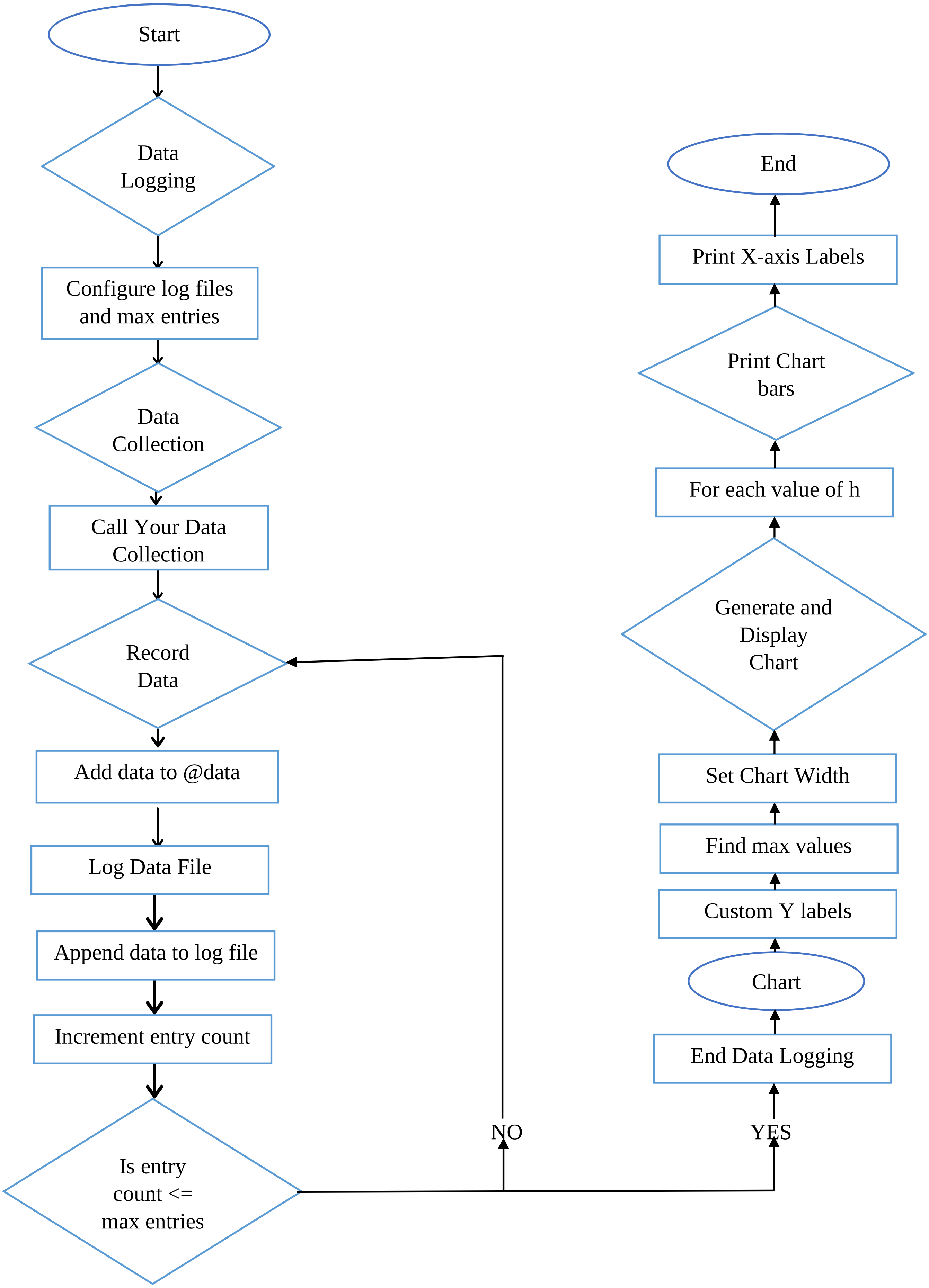
In the contemporary landscape of data-driven decision-making, the pivotal role played by data logging and data visualization cannot be overstated. These core components serve as the foundation for capturing, recording, and presenting data in a manner that facilitates comprehensive understanding and actionable insights.

Data logging, as an initial step, entails the systematic recording of data over specified time intervals. It serves as the backbone of comprehensive data analysis, enabling precise storage and accessibility of information. Its significance is exemplified in an array of domains, ranging from scientific research to industrial process control and environmental monitoring.

However, the sheer accumulation of data, in its raw form, can be inscrutable and lack meaningful insight. It is at this juncture that data visualization assumes its indispensable role. By translating data into visual representations, data visualization transforms complex datasets into comprehensible and actionable information. Through the use of visual elements such as line charts, heatmaps, and bar graphs, data visualization makes data accessible and interpretable.

The purpose of this report is to delve into the principles of data logging and data visualization, highlighting their significance in facilitating data-driven decision-making. Moreover, this report offers a practical demonstration of these principles, utilizing the versatile Perl programming language to exemplify the entire data acquisition and visualization process. As we proceed, it is crucial to recognize that data, beyond being a collection of numbers and symbols, represents the voice of empirical reality, containing valuable narratives waiting to be uncovered. This report aims to shed light on these narratives and their critical role in informed decision-making.

Flowchart:



Start
Data Logging
Configure log files and max entries
Data Collection
Call Your Data Collection
Record Data
Add data to @data
Log Data File
Append data to log file
Increment entry count
Is entry count <= max entries
NO
YES
End Data Logging
Chart
Custom Y labels
Find max values
Set Chart Width
Generate and Display Chart
For each value of h
Print Chart bars
Print X-axis Labels
End

Code:

(without using library)

```perl
#!/usr/bin/perl
use strict;
use warnings;

# Data Logging Configuration
my $log_file = "data_log.txt";
my $max_entries = 10; # Maximum number of entries before stopping

sub your_data_collection_function {
    my $data = int(rand(25));  # Replace with your actual data collection logic
    return $data;
}

my @data;  # Store the collected data

# Erase or truncate the log file before starting
open(my $fh, '>', $log_file) or die "Could not open file '$log_file' $!";
close($fh);

my $entry_count = 0;

while ($entry_count <= $max_entries) {
    my $timestamp = time();
    my $data_point = your_data_collection_function(); # Replace with your data source or collection code
    push @data, $data_point;  # Store the collected data

    # Log to the file
    open($fh, '>>', $log_file) or die "Could not open file '$log_file' $!";
```

```
    print $fh "$timestamp $data_point\n";
    close($fh);

    $entry_count++; # Increm ent the entry count

    sleep(0.01); # Adjust the sleep duration as needed
}

# Horizontal Bar Chart Configuration
my @y_labels = (1 .. @data);  # Custom Y-axis labels

my $max_value = 0;

foreach my $value (@data) {
    $max_value = $value if $value > $max_value;
}

my $width = $max_value;  # Adjust the width as needed

# Create and display a horizontal bar chart using the collected data
for my $h (reverse(1 .. $width)) {
    printf("%2d |", $h);
    for my $i (0 .. $#data) {
        my $bar_height = int($width*$data[$i] / $max_value);
        if ($bar_height >= $h) {
            print " #";
        } else {
            print "  ";
        }
    }
```

```
    print "\n";
}

# Print X-axis labels
print "    +";
for my $i (0 .. $#data) {
    print "--";
}
print "\n";
for my $i (0 .. $#data) {
    printf("%4d", $i + 1);
}
print "\n";
```

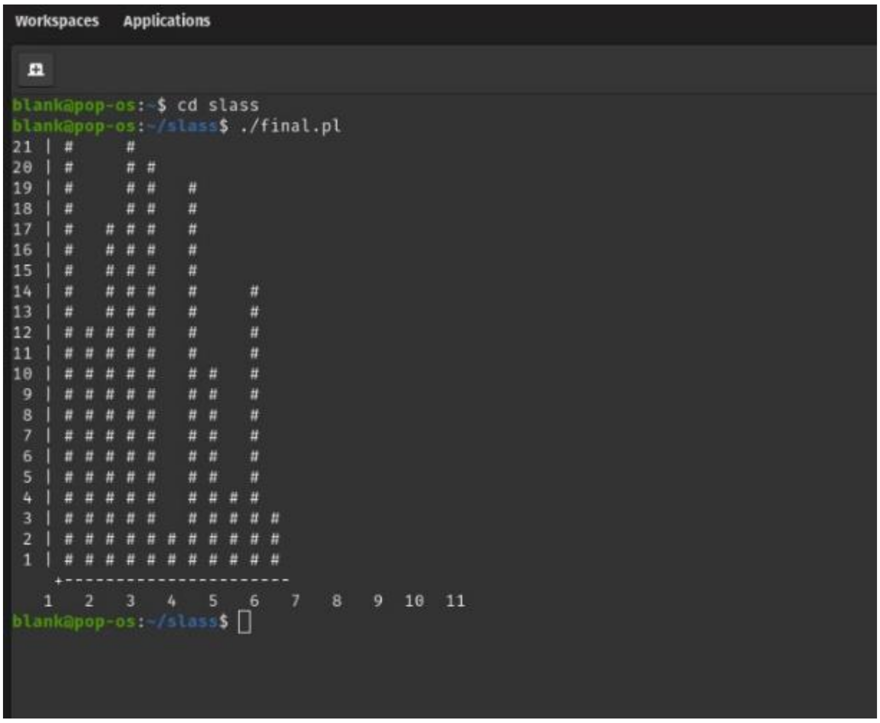
(with using library)

```
#!/usr/bin/perl
use strict;
use warnings;

my $log_file = "data_log.txt";
my $max_entries = 25; # Maximum number of entries before stopping

sub your_data_collection_function {
    my $data = int(rand(100));  # Replace this with your actual data collection logic
    return $data;
}

# Erase or truncate the log file before starting
open(my $fh, '>', $log_file) or die "Could not open file '$log_file' $!";
close($fh);
```

```perl
my $entry_count = 0;

while ($entry_count < $max_entries) {
    my $timestamp = time();
    my $data = your_data_collection_function(); # Replace with your data source or collection code

    # Log to the file
    open($fh, '>>', $log_file) or die "Could not open file '$log_file' $!";
    print $fh "$timestamp $data\n";
    close($fh);

    $entry_count++; # Increment the entry count

    sleep(1); # Adjust the sleep duration as needed
}

# Use gnuplot to create a text-based graph
open($fh, '<', $log_file) or die "Could not open file '$log_file' $!";
open(GNUPLOT, "| gnuplot");

print GNUPLOT "set terminal dumb\n";
print GNUPLOT "plot '$log_file' using 1:2 with lines\n";
close(GNUPLOT);

close($fh);
```

Output:

(without using library)



(with using library)

Implementation and Observation:

This system aims to capture and represent data over time, ultimately creating a horizontal bar chart without the use of a library and line chart while using the library. It serves as a foundational tool for data logging and visualization in real-world applications.

Data Collection Implementation:

Data collection starts with a function, `your_data_collection_function`, generating simulated data. This function acts as a placeholder for integrating real data sources, such as sensors or databases.

Data Logging Configuration and Process:

Data is logged to a file, "data_log.txt," and stored in an array. Configuration involves setting the log file name and maximum entries (set to 10). A while loop controls data logging.

Horizontal Bar Chart Generation:

A horizontal bar chart is created, with customization options for Y-axis labels and chart width. The chart visually represents data points based on the maximum value within the dataset.

Advantages of Data Logging and Visualization:

- Better Decision-Making
- Spotting Patterns
- Real-time Awareness
- Simplified Communication
- Efficiency
- Historical Insights
- Customization
- Combining Data Sources
- Predictive Analysis

Disadvantages of Data Logging and Visualization:

- Information Overload
- Misleading Visuals
- Resource Intensive
- Learning Curve
- Privacy Concerns
- Maintenance
- Bias
- Cost
- Data Accuracy
- Interpretation Challenges

Application of Data Logging and Visualization:

1. Business Analytics:

- Sales and Marketing: Analyzing customer data, sales trends, and marketing campaigns to make informed decisions and boost revenue.

- Financial Analysis: Monitoring financial transactions and visualizing market data to make investment decisions.

- Supply Chain Management: Tracking inventory, shipments, and logistics data to optimize the supply chain.

2. Environmental Monitoring:

- Weather Forecasting: Collecting and visualizing weather data to predict and understand climate patterns.

- Air Quality Monitoring: Recording air pollution levels and visualizing the data to assess air quality in different locations.

3. Healthcare and Life Sciences:

- Patient Monitoring: Visualizing patient data like vital signs, test results, and treatment progress for improved healthcare decisions.

- Genomics: Analyzing genetic data and visualizing DNA sequences for research and medical applications.

4. Industrial Process Control:

- Manufacturing: Monitoring and visualizing data from production lines to improve efficiency and product quality.

- Energy Management: Tracking energy consumption and production to optimize usage and reduce costs.

5. Internet of Things (IoT):

- Smart Homes: Visualizing data from smart devices (thermostats, security systems, etc.) for home automation and control.

- Smart Cities: Monitoring data from IoT sensors for urban planning and resource optimization.

6. Scientific Research:

- Experimental Data Analysis: Visualizing data from scientific experiments and observations to draw conclusions.

- Astronomy: Analyzing data from telescopes and satellites to explore the cosmos.

7. Education and Training:

- Data Science and Statistics: Teaching and learning data analysis, statistics, and visualization techniques.

- Interactive Learning: Creating interactive educational tools for students to grasp complex concepts.

8. Finance and Investment:

- Stock Market Analysis: Visualizing financial market data to inform investment decisions.

- Portfolio Management: Tracking investment performance and optimizing portfolios.

9. Quality Control:

- Product Quality Assurance: Monitoring product quality through data analysis and visualization in manufacturing.

- Software Testing: Visualizing test data to identify software issues and improve quality.

10. Social Sciences:

- Economic Analysis: Visualizing economic data to understand trends, inequalities, and inform policy decisions.

- Demographics: Analyzing population data to study social trends and plan social programs.

11. Energy and Environment:

- Renewable Energy Monitoring: Tracking and visualizing data from renewable energy sources like solar and wind for efficient energy management.

- Water Resource Management: Monitoring and analyzing water usage, quality, and availability.

12. Smart Agriculture:

- Crop Monitoring: Visualizing data from sensors, drones, and satellite imagery for precision agriculture.

- Livestock Management: Monitoring animal health and performance using IoT devices.

Conclusion:

In conclusion, data logging and visualization presents a versatile and adaptable solution for the ease of work to store the data. Its capacity to capture and represent data over time, whether using simulated or real data, positions it as a valuable tool for data-driven decision-making across various domains. The code's customizability and potential for integration make it a practical asset, while optional provisions for error handling, testing, and documentation ensure its reliability. This system stands as a fundamental resource for harnessing the power of data in practical applications.