# Heads Or Tails: A Framework To Model Supply Chain Heterogeneous Messages

Sonya Leech, David Malone
Maynooth University
Kildare, Ireland
sonya.leech.2021@mumail.ie,david.malone@mu.ie

Jonathan Dunne
IBM
Dublin, Ireland
Jonathan_Dunne@ie.ibm.com

***Abstract*—The electronic exchange of business to business information (e.g. purchase orders, inventory data and shipment notices between departments or organizations) can eliminate the need for human intervention and paper copy trails. Incorporating Electronic Data Interchange (EDI) standards into an organization can drastically improve the efficiency of processing times. Modelling the behaviour of EDI messages within a Supply Chain network's queuing system has many purposes, from understanding the efficiency of queue behaviour to process re-engineering. In this paper we demonstrate that these messages are heterogeneous, suffer from correlation, are not stationary and are challenging to model. We investigate whether a parametric or non-parametric approach is appropriate to model message service and inter-arrival times. Our results show that parametric distribution models are suitable for modelling the distribution's tail, whilst non-parametric Kernel Density Estimation models are better suited for modelling the head.**

## I. Introduction

Queuing systems help businesses within the Supply Chain domain improve productivity and turnover. It enhances client satisfaction by allowing for the processing of large volumes of messages with persistent storage [1]. When demand exceeds supply, queuing systems provide a more streamlined experience by preventing job loss and supporting queue and job prioritization. This is integral to avoiding Supply Chain distribution problems by keeping the system steady-state [2].

A recent report on future trends for the Supply Chain suggests that resiliency is an essential requirement for customer needs [3]. Queuing systems are ubiquitous across industry domains: telephone communications [4], road traffic flows [5], hospital waiting lists [6], and banking transactions [7]. However, like all computing systems, queues are not immune to performance and reliability problems, including latency, bottlenecks, scalability, and the challenge of random arrival of incoming messages [8].

Our interest originated from problems arising within a Supply Chain network, where message processing times are not clearly understood. In practice, these heterogeneous EDI messages suffer from numerous failed message re-tries and throttling, causing bottlenecks within the enterprise messaging system. The benefits of modelling these messages and their queue behaviour could lead to a simplified understanding of the system. Given the sheer volume and velocity of incoming messages processed through the system, some abstraction will be necessary. In our example system, one might see over two million messages processed on a typical day. Depending on size, these messages may be split into smaller sizes and lead to potentially over thirty-two billion jobs processed through the Enterprise queuing system. Another important area where these EDI Supply Chain messages are important is within the realm of smart contracts and distributed ledgers. Applying smart contracts using Blockchain can reduce fraudulent transactions and help trace misplaced items within a transactions life cycle [9].

This paper studies the challenges we faced when modelling EDI transactions, including correlation, message bundling and heavy-tailed data. We consider different techniques for filtering, splitting, and grouping the data to facilitate parametric and non-parametric modelling. We also attempt to classify the events at the head of the distribution.

We aim to address the following questions. First, can EDI messages be modelled using parametric, or, failing that, non-parametric techniques? Second, can the service time (ST) and inter-arrival times (IAT) of an EDI message be modelled by a parametric method? If not, can non-parametric techniques be used? Third, are we able to classify these messages?

This paper is structured as follows. Section II describes the background and related research. Section III describes the dataset and methods we took and the limitations of our dataset. Section IV and Section V provide results and a discussion. Section VI concludes and notes future work.

## II. Background And Related Research

### A. Electronic Data Interchange

EDI is defined as *the inter-organizational, computer-to-computer exchange of business documentation in a standard, machine-processable format* [10]. These messages are part of a Supply Chain messaging architecture component. It enables electronic exchange and processing of these heterogeneous business documents through the network. Our research is centered around these heterogeneous EDI messages (e.g. purchase orders, invoices, shipment notices). Different EDI message standards exist, including X12, EDIFACT, TRADACOMS [11]. In our analysis, most of our messages were of the type X12.

We will discuss systems that support queueing of EDI transactions next. There is surprisingly little literature on the modelling of EDI transactions. Most of the literature

ISSN 2305-7254

concentrates on the benefits of switching to an EDI system [12], [13].

### B. Queuing Applications

Enterprise Queuing applications are used within the Supply Chain domain. We briefly consider four: ActiveMQ [14], Kafka [1], RabbitMQ [15], and IBM Message Queue(MQ) [16].

RabbitMQ is an open-source distributed message broker. It supports multiple messaging protocols and offers flexible queue routing with multiple exchange types [17]. Rabbit MQ has a broad customer base with over thirty-thousand companies running the software [18].

Kafka is an open-source distributed event streaming platform. It takes raw input messages categorized by Kafka and transforms them for further consumption or follow-up processing. It is scalable, fault-tolerant and can store and process streams of data with a guarantee of zero message loss [19]. Kafka is often used by high-end social networking companies. Twitter use it as part of their stream-processing infrastructure [20]. LinkedIn uses it for streaming data in news feeds and offline analytical systems [21]. Netflix uses it for data collection [22]. AWS use it as part of their cloud platform [23]. Kafka has a broadly similar market share to Rabbit MQ with over thirty-thousand companies using Kafka today [24].

Active MQ is an open-source Java-based standalone message broker. It supports high availability, load-balancing, and asynchronous messaging [25]. Active MQ has roughly half the market share of customers of Kafka and Rabbit MQ, with just over fourteen thousand customers [18].

IBM Message Queue (MQ) belongs to the WebSphere family. It is part of the middleware stack. It supports point-to-point, publish/subscribe and file transfer methods for its messaging and queuing operations. IBM MQ can transport any type of data as a message [16].

### C. Queuing Theory

Queue theory studies an orderly list of one or more jobs [26]. For stream-based job types, these jobs are processed through single or multi-server queuing systems. The arrival of such jobs may be random or based on pre-defined schedules. The method of service, such as 'first in first out', 'last in first out' and 'shortest job first' can also be important [27].

Modelling incoming jobs via service times and inter-arrival times gives insight into delays, queue sizes and how a heavy influx may impact a system. When modelling, queue length, idleness and wait times are fundamental metrics [28].

### D. Distribution Fitting

Distribution fitting is the study of fitting a probability distribution to a dataset given measurements of a random quantity that one can use to make inferences about the sample population [29]. With the fitted distribution, one can predict the probability of specific events.

Two common methods for estimating distribution parameters are the Method of Moments Estimation (MME) and Maximum Likelihood Estimation (MLE). MME was introduced by Pearson and matches the moments of a distribution to the empirical moments [30]. MLE, proposed by Fisher, estimates the parameters of a probability distribution by maximising the likelihood of the observed data [31]. Research undertaken by Fisher shows that MLE is generally more efficient than MME because MLE's variance is smaller [31].

### E. Goodness of Fit Testing

When a dataset is presented, it can be helpful to understand how well the underlying data fits a known probability distribution. There are methods to assess a distribution's goodness of fit to a data set.

The Cramer-Von Mises (CVM) test criterion is a non-parametric test that examines the goodness of fit of a cumulative distribution function (CDF) compared to an empirical density function (EMF) [32]. This significance test will determine whether data is drawn from a known CDF.

The Kolmogorov–Smirnov (KS) test measures the distance between the EMF of the sample and the CDF of a reference distribution or between the EMF of two samples [33]. KS tests are useful when testing whether a set of observations are from a specified continuous distribution [34].

The Anderson–Darling (AD) test is also a statistical test of whether a particular PDF fits the data [35]. This test is a modification of the KS test, giving more weight to the tails of the empirical data.

### F. Heavy Tailed Estimation

A heavy-tailed distribution is when the tails of the data are not exponentially bounded [36]. Heavy tailed distribution was first introduced in financial data from Mandlebrot in 1963 [37].

In our case, if enough messages have long processing times, this may lead to poor performance. When modelling data, the distribution's tails may form a different parametric fit than the rest of the data. If we need to split our data into shorter and longer intervals, we take a simple approach of splitting the data into a head and a tail using simple criteria of $> n$ seconds. More generally, Hill proposed a method allowing inferences about the tails of data [38].

### G. Hurdle Distribution

When modelling count data, one should often give special consideration to zero value. Depending on the dataset, it may have different interpretations. Some count models suffer from excess zeros. A Hurdle model can support modelling excess zeros in the data by combining a left-truncated count component with a right-censored hurdle component [39]. A Hurdle distribution can be preferable to Poisson distribution due to the additional flexibility [40].

### H. Kernel Density Estimation

Kernel Density Estimation (KDE) is a non-parametric method to estimate the density of a random variable. When the sample population does not fit a known probability distribution, KDE can be used. KDE can indicate the data being either multi-modal or having a degree of skewness [41].

To implement KDE, one needs to choose the kernel and bandwidth. Different kernels apply different weights based on how far the data is from each point. Some common kernels are Epanechnokiov [42], Gaussian, Uniform, Box, and Triangle.

The bandwidth determines how spread out the kernels are. Large bandwidths can cause a high degree of bias in the distribution [41]. There are different algorithms to define the bandwidth, including Silverman's Rule of Thumb [41], Sheather & Jones [43], and Park & Marron [44].

For the accuracy of the KDE models, we used visualization techniques and the area under the curve as a method to assess a best-fit. The MISE was not appropriate for statistically confirming a Goodness Of Fit as we do not know the true density of the data.

### I. Related Work

We review some of the literature around performance modelling of queueing applications for EDI messages.

LinkedIn wanted to move from a batch-oriented system to a real-time publish-subscribe system. They needed a system that could process 10 billion messages per day with a peak demand of 172,000 messages per second. They initially explored ActiveMQ, but instead, they developed Kafka [45].

Wu, Shang & Wolter analyzed the performance of Kafka using specific tuning parameters [46]. Their analysis concluded that the performance varied significantly depending on message type and system infrastructure. They also noted a strong correlation between packet sizes and sending intervals.

Henjes, Menth & Zepfel conducted a study of the capacity of the IBM MQ JMS Server [47]. They considered different filters, message sizes and the number of pubs/subs. They found that message size had a significant impact on the message and data throughput of the server. The number of topics had little influence on server capacity. They did, however, find that the message replication grade and the number of filters have a significant impact on server capacity.

## III. Data Set And Methods

Modelling the behaviour of a queue can assist in improving the stability and reliability of queuing systems. Identifying the distribution of inter-arrival and service times allows modelling the behaviour of a queue and the prediction of congestion issues. We demonstrate data wrangling techniques that will enable us to fit our data into parametric and non-parametric models.

### A. System Description and Data Collection

The study presented in this paper uses an enterprise dataset from a cloud-based Supply Chain Network. Within our Supply Chain Network, a message can traverse multiple paths until it reaches its destination. As the message passes each point on its path, multiple queues may ingest the message. First, we analyze the messages from source to destination and from one endpoint to another.

Fig. 1 provides a greater intuition of the flow of a message on a Supply Chain Network. We can see from the figure that multiple queues interlock the communication channels between the various entities. Our analysis was focused on the entities in shaded grey. The combined Translation Service, Rules Engine and Message Dispatcher were provided on seven servers. In addition, there was the SAP Server.

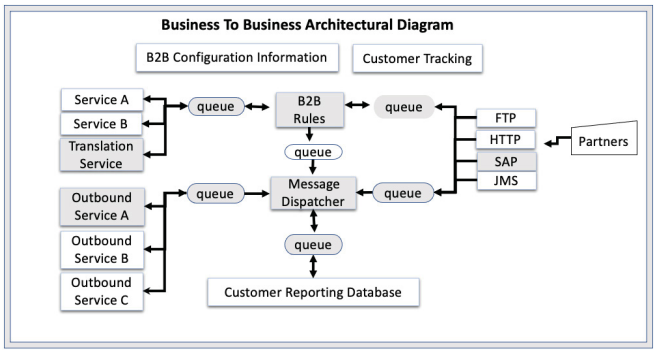


Fig. 1. Supply Chain Network Architectural Message Flow

We collated four different datasets from a Cloud Supply Chain Network organization. Fig. 1, a logging application called Graylog aggregates both the SAP and Outbound Message Service information. The Message Dispatcher, B2B Rules Engine, and the Translation Service log files are aggregates using a separate process. The log files contained a mixture of structured, unstructured and XML data. We stripped a lot of the data from within the log files to reduce noise. We then filtered to help classify the start and end of a message and the different steps the message took. We noted associated beginning and ending times, cumulative counts (to determine message splits, see below). Where one message generated other messages, they were tagged as a child/parent message. This data-wrangling technique allows for the traversal of each messages lifetime within the network, as represented by Fig. 2.

We wanted to focus our efforts on the complete end-to-end flow of a message from its source to its final destination. Initially, we took the SAP Service as our starting point. However, for the data available to us, we noted only 527 incoming messages that we could trace end to end. Instead, we chose to look at the Translation Service, of which 90% of all incoming messages make use.

Fig. 2 shows an example of the processing complexities of a single message flow end to end. The first entity to the left shows the start of the message from the source. The message then travels to the B2B Rules Engine and the Message Dispatcher displayed by the oval entities. From here, the message gets sent to three different queues.

These parent and child messages then get sent to the Translation Service represented by black rectangles. The Translation Service processes the message through the Translation queues, then back to the B2B Rules Engine / Message Dispatcher, where the Rules Engine initiates a Command Request and a Data Request shown with a white rectangle with grey text. After the Command Request and Data Request complete, the Rules Engine and Message Dispatcher send the message to the

client's trading partner, which is the last entity to the right of the image. As the MQ system did not log when a message left the queue, we used the Rules Engine and Message Dispatcher Command Request and Data Request to determine when the message left the queue.

From Fig. 2 we see that one message entered nineteen queues as part of its lifecycle. This unobfuscates the flow of an EDI message in a way that does not seem to be presented in previous literature.

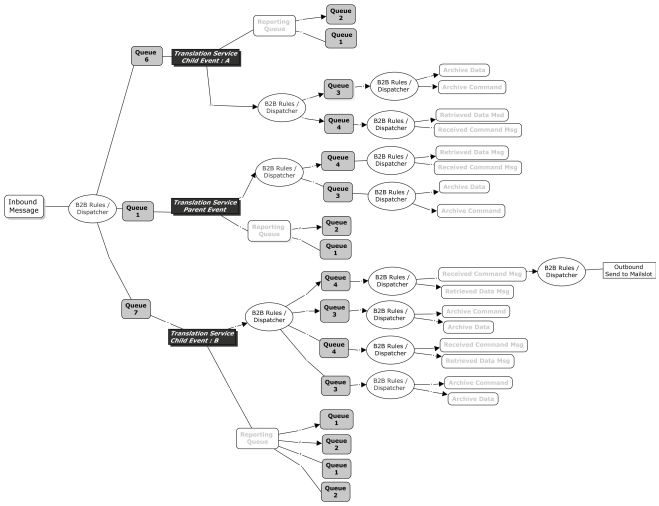


Fig. 2. B2B Low Level Message Flow, Note the number of queues with the grey rounded rectangle, show how often one message enters a queue through its lifecyle

The Translation Service illustrated in Fig. 1 offers different functionality. It can translate messages from one format to another (e.g. X12 to CSV) and can concatenate or extract documents from a message. It can split one input file into multiple output files. This reduces the size of the message processed through the queues, thus reducing job sizes and facilitating parallelism.

Fig. 3 shows the steps a message takes within the Translation Service. We have selected a simple message, as a more complex message result in a larger number of steps. Step 4 is where the actual translation occurs. Document extraction begins in Step 6. In Steps 16 to 18, messages are sent to the Translation queue, as noted in Fig. 2 with the grey rounded rectangles with the titles "Queue 3" and "Queue 4". Again, the order of the steps does not appear to have been noted in the literature review.

### B. Data Overview and Limitations

As noted, we focus on the Translation Service, which processes around 2 million messages per day. Our dataset was 13.5 hours, from midnight on the 30th of November up until 13:30 of that same day. Table I shows the volume of messages we analyzed. From the Translation Service, we observed data going into two different queues. The CMD queue is the command queue, where commands are sent. The Data queue is where the corresponding data is sent. Every

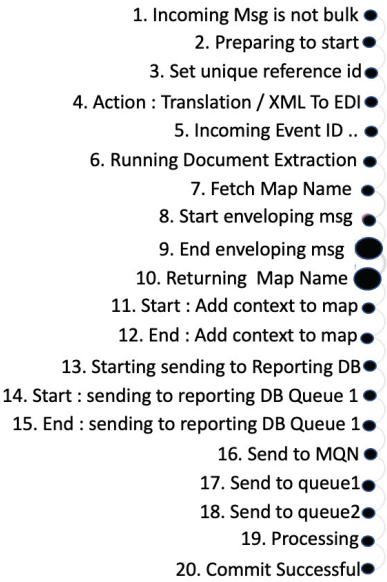


Fig. 3. Message Translation Steps

message is associated with a CMD and Data Queue entry. If we refer back to Fig. 2, this single message hits these queues eight times in total, represented by the rounded grey rectangles titled "Queue 3" and "Queue 4".

TABLE I. TRANSLATION SERVICE ALL DATA MESSAGE VOLUME

| Date | CMD Queue | Data Queue |
|---|---|---|
| 30th Nov | 1,036,938 | 1,036,956 |

We note a number of practical limitations of the data set. First, it was challenging to trace a message from end to end. The was no unique identifier between the different log files and Graylog that allow us to trace all relevant information. While every effort was made to identify unique characteristics that trace individual messages through the system, our method is somewhat ad hoc.

Second, because of the logging level in the system, we could only directly identify when the organization sent a message to the queue. We had to use log files from other applications to infer when the message left the queue.

Finally, collecting data for longer periods from this system is challenging. Due to the sheer volume of messages hitting the Rules Engine and the Message Dispatcher, there was a limitation in the amount of data we could gather. Consequently, the Rules engine and the message dispatcher only keep data for a few hours before the log files are recycled.

### C. EDI Modelling

When fitting data to parametric models, the easiest case is when time-series data shows no dependence on previous values. The independence of arrivals is also a common assumption in queueing models. If a correlation exists, the

time-series data may be deemed non-stationary, and different techniques may be used to handle correlation. Thus we check for correlation whilst checking the distribution fit of the data. Table II shows the Spearman's rank correlation co-efficient test statistics for non-normal data.

TABLE II. SPEARMAN RANK CORRELATION

| Co-efficient Start Range | Coefficient End Range | Result |
|---|---|---|
| 0.05 | .29 | Weak |
| 0.30 | .49 | Medium |
| 0.50 | 1 | Strong |

We shall now describe an approach to our analysis in the following set of subsections.

*1) Normal And Busy Periods:* Based on information from the team operating this system, we knew there was a period where the system had been considered to have a performance problem. When analyzing the data, we found a period where the number of messages queued was always bigger than zero and often growing. Accordingly, we broke the data into two periods, busy and normal. Table III shows our busy period was just under forty minutes, whilst our normal period was just over twelve hours. There is a similar volume of messages per second going into each queue ($\approx$ 22 messages per second). This suggests the busy period is caused by a change in STs rather than IATs. Arrivals to the CMD and Data queue are generally similar in both periods, though there is a slight discrepancy of eighteen messages between the two queues in the busy period. The difference may be due to some messages containing large attachments. If the message is big, it is paginated, causing more data messages.

TABLE III. TRANSLATION DATA DIFFERENT TIME PERIODS

| Period | Start Time | End Time | CMD Queue | Data Queue |
|---|---|---|---|---|
| Normal | 12AM | 12PM | 984,183 | 984,183 |
| Busy | 12:50PM | 1:29PM | 52,755 | 52,773 |

*2) Message Split Count:* We noted a significant number of messages that were part of a bundle, i.e., part of a group with zero seconds between them. We consider splitting the messages into groups according to how many messages were in the bundle.

First, Split = "1" is when a single message comes into the system, and one message gets sent to the queue. These are not part of a bundle and may be small in size. Split = "2" is when one message comes into the system, and two messages get sent to the queue. Finally, Split = "Other" is when one message comes into the system, and three or more messages get sent to the queue. The maximum bundle size we saw was 1,617. These are typically large bundled messages. If we refer back to Section I, thirty-two billion jobs (2 million * 1617) can come from two million messages.

Table IV shows how many messages belong to each split group over the normal period. We note that most messages are in the Split = 1 group.

TABLE IV. TRANSLATION DATA PER SPLIT

| Splits | Normal Period | Percentage |
|---|---|---|
| 1 | 558,549 | 57% |
| 2 | 233,421 | 24% |
| other | 192,213 | 19% |

*3) Hurdle Modelling:* A considerable volume of messages was processed in zero seconds. When attempting to fit the data to a distribution, using a hurdle model, we considered removing these zeros from the data to improve fit and to facilitate overdispersion. Table V shows the percentage of messages removed using a hurdle model.

TABLE V. NORMAL PERIOD: HURDLE MODEL

| Splits | Normal Period | Data Removed |
|---|---|---|
| 1 | 511,670 | 9% |
| 2 | 112,295 | 48% |
| other | 119,39 | 6% |

*4) Message Bundle:* As we noted above, there are a number of messages that result in a bundle. When calculating the arrival time and service time of these messages, it may make sense to treat them as a single message. In this case, we use the first message in the bundle to calculate the IAT and ST, as the other messages appear to have zero duration.

*5) Scheduled Versus Un-Scheduled Messages:* We checked our data for the normal period for other signs of burstiness. In Fig. 4 we show the frequency of jobs arriving based on the minute within the hour. We note that period 00 has the highest frequency of messages. Additional minutes 30-33 and 43-46 also show signs of extra scheduled jobs.

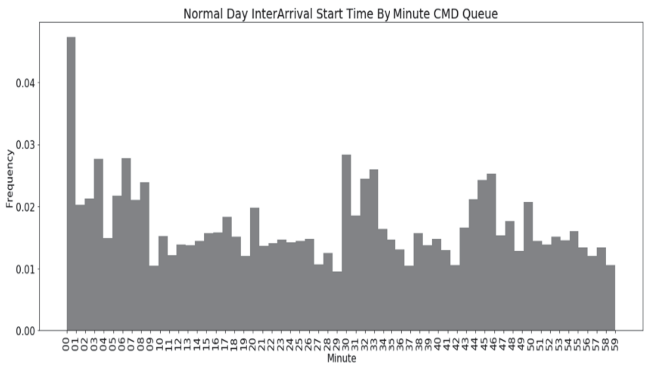


Fig. 4. Normal Period : Burstiness in Data

These scheduled jobs are likely to require separate modelling, so we removed these scheduled times from our model before attempting to model random arrivals.

*6) Map Count:* If we refer back to Fig. 3, we note that in step 7, the Translation Service fetches a map for the message. A map is an XML information document relating to the job (e.g. cost, shipment, details etc.). Some maps may also contain

programmatic loops. As different maps may influence messages duration, we consider fitting different models according to these maps.

In practice, not every message is associated with a map, and many maps can be associated with one message. Table VI shows the total count of maps for each split. We note that the maximum amount of maps for one message was 30.

TABLE VI. NORMAL PERIOD: MAP COUNT

| Split | Total Maps | Max Maps per job |
|---|---|---|
| 1 | 286,622 | 2 |
| 2 | 135,095 | 4 |
| Other | 14,556 | 30 |

### D. Parametric Modelling

We will consider modelling service times and inter-arrival times parametrically. We consider a range of possible distributions (Table VII) and data transformations (Table VIII) when fitting. We determine the parameters for each distribution, and then an Anderson-Darling Goodness Of Fit test can be used to determine if the data fits that distribution.

TABLE VII. PARAMETRIC TESTS

| | | | |
|---|---|---|---|
| Normal | Log | Log Logistics | Logistic |
| Cauchy | Gamma | Burr | Inverse Burr |
| Exponential | Beta | Weibull | Pareto |

TABLE VIII. DATA TRANSFORMATIONS

| | | |
|---|---|---|
| Log() | Sqrt() | Exp() |
| Log(Log) | Sqrt(Exp) | Sqrt(Log) |

Our service times are extracted from the data as the time a message starts to the time the following message starts. Inter-arrival times are the time between when one message gets sent to the queue and when the following message gets sent to the queue. While fitting, we noted that our data was heavy-tailed and right-skewed. As previously mentioned, we considered a head and tail approach. For both STs and IATs, we considered the head of the data to be values of one second or under.

### E. Non-Parametric Modelling

When a parametric approach does not provide a useful technique to model a dataset, a non-parametric approach can sometimes be useful. We apply Kernel Density Estimation (KDE). We used Silvermans Rule of Thumb, Sheather & Jones, Biased Cross-Validation, Un-Biased Cross-Validation, and Direct Plug-in methods for the bandwidths. Finally, our study will focus on modelling the service times with limited modelling on the inter-arrival times.

### F. Message Classification

While reviewing the dataset for over-dispersion, we observed a number of interesting features in the STs that were visible when plotted on a log scale (e.g. Fig. 5 and 6) where the message Split =' 1'. We note that there appeared to be stratification of times into different overlapping distributions.

It may be interesting if these different distributions represent different 'classes' of messages. A business needs to understand baseline metrics, and getting insight into these messages from the head of the distribution motivates our research question classifying them.

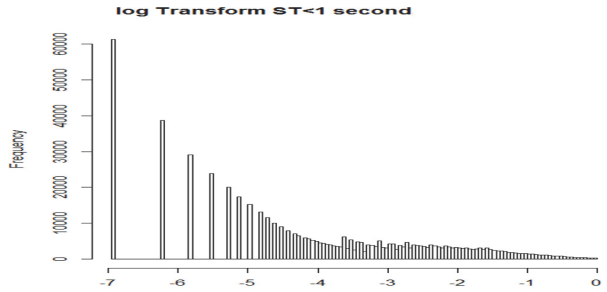


Fig. 5. Service Times - Log Transform

It appears that the messages from -7 to -4 form one group, so to better understand the different classes of messages, we zoom in on the range from -4 to -1.5 and note potentially five distinct types of messages as represented in Fig. 6. We can crudely split these into different groups as shown in Table IX.

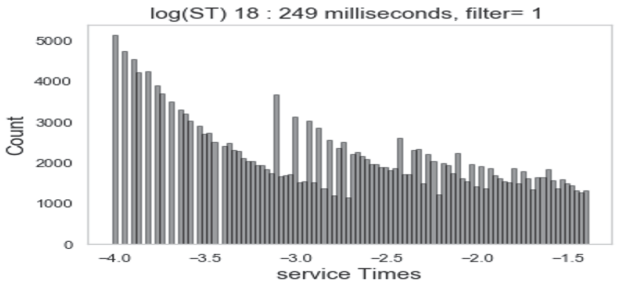


Fig. 6. Service Times - Log Transform, Filter =1

TABLE IX. MESSAGE CLASSIFICATION GROUPING

| Group | Start | End | Range (ms) | Mean (ms) | Customers |
|---|---|---|---|---|---|
| 1 | -7 | -4 | 0.001 – 18 | 0.005 | 651 |
| 2 | -4 | -2.7 | 18 – 67 | 0.036 | 635 |
| 3 | -3.0 | -2.2 | 49 – 110 | 0.307 | 577 |
| 4 | -2.0 | -1.7 | 137 – 182 | 0.158 | 429 |
| 5 | -1.7 | -1.5 | 182 – 223 | 0.205 | 417 |
| 6 | -1.5 | 0 | 223 – 999 | 0.404 | 524 |

## IV. RESULTS

We will now describe the results of our analysis.

### A. EDI Modelling

*1) Normal and Busy Periods:* Table X shows summary statistics for the service and inter-arrival times for each queue during both normal and busy periods. We note that the maximum duration of service and inter-arrival times is 458s (7.6 min) which is significant compared to all average values.

TABLE X. NORMAL/BUSY PERIOD - ST & IAT IN SECONDS.MILLISECONDS

| | Normal Period | | Busy Period | |
|---|---|---|---|---|
| | CMD Queue | Data Queue | CMD Queue | Data Queue |
| **Service Time** (s) | | | | |
| Min | 0.00 | 0.00 | 0.00 | 0.00 |
| Mean | 0.04 | 0.04 | 0.04 | 0.04 |
| Max | 22.80 | 22.80 | 458.44 | 458.44 |
| | | | | |
| **Inter-Arrival Time** (s) | | | | |
| Min | 0.00 | 0.00 | 0.00 | 0.00 |
| Mean | 0.04 | 0.04 | 0.04 | 0.04 |
| Max | 22.43 | 22.43 | 446.84 | 446.81 |

We note from Fig. 7 and Fig. 8 that the histograms have one bin where the majority of the data lies. On this scale, adding more bins does not change the shape of the histogram, as most messages took under one second to process. Fig. 7 and Fig. 8 clearly do not show enough details to allow us to discern useful patterns, other than the data having a long tail.

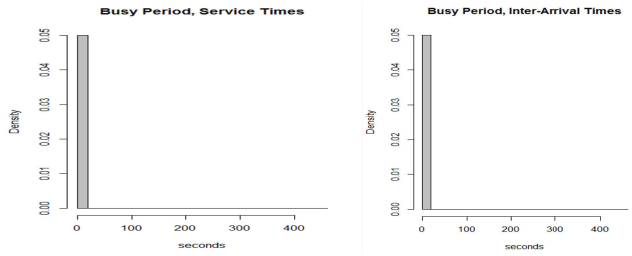


Fig. 7. Histogram: Busy Periods, IAT and ST

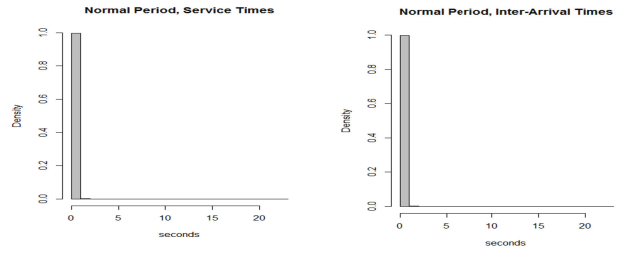


Fig. 8. Histogram: Normal Periods, IAT and ST

*2) Message Split Count:* Based on our discussion in Section III-C, we split the service and inter-arrival times into its head and tail. Fig. 9 shows improved definition in terms of both service and inter-arrival times. Both the head and the tail of the data show more bins containing an observable amount of data, and, consequently, we are able to discern more.

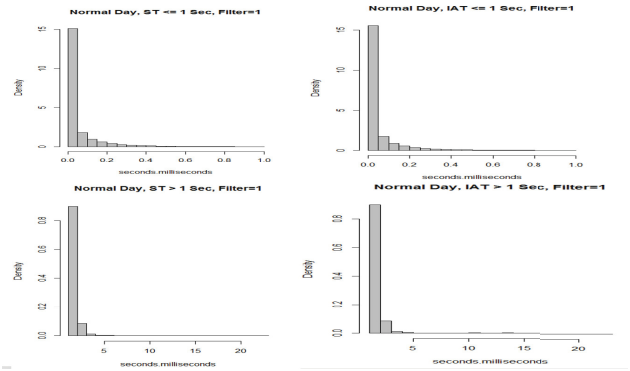


Fig. 9. Messages - Heads And Tails

We plotted the autocorrelation of each time series to look for correlation with a cutoff of between thirty and sixty lags. We have also done crosscorrelation tests between the service times and the interarrival times and noted that the tests confirmed crosscorrelation existed past sixty lags. We checked the data for correlation, grouping the messages by their split count. Table XI shows that there is no correlation in the tails of the model except for IAT where Split = '2'. We note that the head of the data does present correlation. We approach modelling this part of the data with caution unless correlation can be removed or understood.

TABLE XI. CORRELATION CHECKS BY SPLIT

| | Correlation indicated by ACF | | |
|---|---|---|---|
| Test | Split = 1 | Split = 2 | Split = Other |
| ST <= 1 second | True | True | True |
| IAT <= 1 second | True | True | True |
| ST > 1 second | False | True | False |
| IAT > 1 second | False | False | False |

*3) Hurdle Modelling:* We removed the transaction messages of zero seconds duration from the model. This was done for all messages in the head of the data. We performed this task in consideration of Hurdle modelling and due to the fact that several standard transformations can not be performed without either removing the zeros or shifting the data. Even after removing transactions of zero duration, the histogram does not change substantially (see Fig. 10 left) as the majority of messages are in the millisecond range. Interestingly, removing these zeros from the data does also not remove correlation from the models (Fig. 10 right).

To investigate this correlation in service times further, we will now explore the data by splitting it in other ways.

*4) Message Bundle:* When modelling by message bundle, our analysis identified the first, up to and including the second last message was processed in zero seconds of duration. We noted that only the last part of the bundled message had a duration greater than zero seconds. We looked at modelling

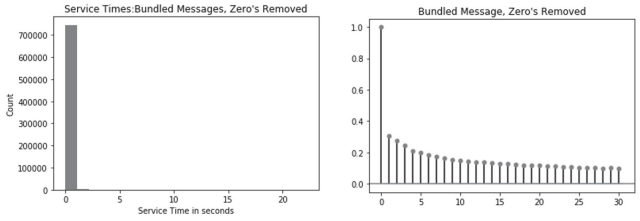


Fig. 10. Distribution Head : Message Bundle With Hurdle Implementation

by message bundle to see if a correlation still existed in the data, as per previous Fig. 10 and correlation was still present.

Thus, bundles of messages alone are not a full explanation of correlation in the data.

*5) Scheduled Versus Un-Scheduled Messages:* Referring back to Fig. 4, we noted different periods where the frequency of messages is higher than those of other periods. In particular, we note the 00 minutes, minutes 30–33 and minutes 43–46 appear to represent busier periods, which could contribute to correlation in our data. We investigated the impact of the removal of these scheduled messages from our data. As per Table XII, we were not able to remove correlation from our data based on the results of the table.

TABLE XII. SERVICES TIMES - CORRELATION CHECKS BY SCHEDULE

| Test | Correlation indicated by ACF |
|---|---|
| Hurdle Implementation | True |
| Removed minutes 00 | True |
| Removed minutes 30-33 | True |
| Removed minutes 43-46 | True |

Table XIII is a summary of the different filtering techniques we have applied and the results of the correlation tests.

TABLE XIII. RE-CAP OF CORRELATION TEST RESULTS

| Seq | Correlation Exist | Volume | % | Hurdle Dist. | Split = 1 | Split = 2 | Split = other | 00 Schedule Removed | 30-33 Schedule Removed | 43-46 Schedule Removed | <=1 Second | > 1 Second |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | True | 558,312 | 57 | False | True | False | False | False | False | False | True | False |
| 2 | True | 233,373 | 24 | False | False | True | False | False | False | False | True | False |
| 3 | False | 192,208 | 20 | False | False | False | True | False | False | False | True | False |
| 4 | False | 511,669 | 52 | True | True | False | False | False | False | False | False | False |
| 5 | True | 112,294 | 11 | True | False | True | False | False | False | False | False | False |
| 6 | True | 11,938 | 1 | True | False | False | True | False | False | False | False | False |
| 7 | False | 487,212 | 50 | True | True | False | False | True | False | False | False | False |
| 8 | True | 109,971 | 11 | True | False | True | False | True | False | False | False | False |
| 9 | True | 11,526 | 1 | True | False | False | True | True | False | False | False | False |
| 10 | False | 440,629 | 45 | True | True | False | False | True | True | False | False | False |
| 11 | True | 100,447 | 10 | True | False | True | False | True | True | False | False | False |
| 12 | True | 10,560 | 1 | True | False | False | True | True | True | False | False | False |
| 13 | False | 394,702 | 40 | True | True | False | False | True | True | True | False | False |
| 14 | True | 928,75 | 9 | True | False | True | False | True | True | True | False | False |
| 15 | True | 9,623 | 1 | True | False | False | True | True | True | True | False | False |
| 16 | True | 558,312 | 57 | False | True | False | False | True | False | False | True | False |
| 17 | True | 233,373 | 24 | False | False | True | False | True | False | False | True | False |
| 18 | True | 233,373 | 24 | False | False | False | True | True | False | False | True | False |
| 19 | False | 235 | 0 | False | True | False | False | False | False | False | False | True |
| 20 | False | 47 | 0 | False | False | True | False | False | False | False | False | True |
| 21 | False | 6 | 0 | False | False | True | False | False | False | False | False | True |
| 22 | False | 216 | 0 | True | True | False | False | True | True | True | False | True |
| 23 | False | 44 | 0 | True | False | True | False | True | True | True | False | True |
| 24 | False | 6 | 0 | True | False | False | True | True | True | True | False | True |
| 25 | True | 394,485 | 40 | True | True | True | False | True | True | True | True | False |
| 26 | True | 92,831 | 9 | True | False | False | False | True | True | True | True | False |
| 27 | True | 9,617 | 1 | True | False | False | True | True | True | True | True | False |

*6) Map Count:* We now look at the implication of the map on the correlation by splitting the data according to how many maps were applied. When analyzing the map information, we noted 40% of transactions have no map name, and 96% of transactions have either 0 or 1 maps associated with them, and 3.5% of transactions have more than one map. From Table XIV we note that all tests passed correlation except where the map count for a message was greater than three, which represents 0.14% of the messages.

TABLE XIV. SERVICE TIME : CORRELATION RESULT

| Test | Messages | % | Map Count | Correlation |
|---|---|---|---|---|
| 1 | 687,467 | 100.00 | all | Weak |
| 2 | 383,024 | 55.72 | 1 | Weak |
| 3 | 22,832 | 3.32 | 2 | Weak |
| 4 | 1,311 | 0.19 | $> 2$ | Weak |
| 5 | 24,143 | 3.51 | $> 1$ | Weak |
| 6 | 280,297 | 40.77 | $< 1$ | Weak |
| 7 | 663,321 | 96.49 | $\leq 1$ | Weak |
| 8 | 686,153 | 99.81 | $\leq 2$ | Weak |
| 9 | 686,495 | 99.86 | $\leq 3$ | Weak |
| 10 | 969 | 0.14 | >3 | True |

It appears that the number of maps is a contributing factor to the correlation we are seeing. With this insight, we consider parametric and non-parametric modelling. Before doing that, we will briefly look at the data on a larger scale.

### B. Parametric Modelling

*1) Modelling Service And Inter-Arrival Times:* Initially, we tried to fit parametric distributions and tested the fit on all data with no filtering techniques applied. However, some parts of the data are amenable to the fitting of a parametric distribution. For example, we found a reasonable fit to a parametric distribution using the message split count on the tail, representing 0.3% of the data. In this case, a filter was applied ( split=1), a hurdle model was implemented We can confirm that the tail of this filtered data fits a parametric Burr distribution using the Anderson-Darling test (see Table XV).

TABLE XV. AD TEST NORMAL PERIOD, ST, TAIL OF DATA

| AD Score | P-Value | Test |
|---|---|---|
| 1.2 | 0.3 | Pass |

We ran similar tests on the tail of the inter-arrival times where the inter-arrival times is $> 1$ second. We consider, for example, the results of attempting to fit distributions to the IAT with Split = 1, and with no hurdle model. The results of the AD tests conclude, as per Table XVI that this filtered data does not fit a parametric distribution. However, we observe that a no-transform and a square root transform (highlighted in bold) are relatively close to a Burr distribution but do not pass the AD test.

For the head of the data, i.e. ST $<= 1$ second, we could not parametrically fit the data to a parametric distribution, irrespective of transformation or implementing any splitting

TABLE XVI.
IAT $>$ 1, FILTER = 1 : AD TEST

| Ad test | data | Log (data+1) | Sqrt (data) | Exp (data) | Sqrt (log (data+1)) | Log (log (data+1) +1) | Sqrt( exp( data)) |
|---|---|---|---|---|---|---|---|
| | **AD Score** | | | | | | |
| Log Normal | Inf | 55 | Inf | Inf | 55 | 48 | Inf |
| Log Logis-tic | 16000 | 45000 | 59000 | Inf | 120000 | 80000 | Inf |
| Gamma | 34000 | 87 | Inf | 6600 | 66 | 61 | 7100 |
| Weibull | Inf | 6600 | 4800 | Inf | 24000 | 18000 | . |
| Expo-nential | 480 | 610 | 670 | Inf | 740 | 680 | Inf |
| Cauchy | 110 | 92 | 95 | 170 | 82 | 83 | 140 |
| Logistic | Inf | 56 | Inf | Inf | 46 | 46 | Inf |
| Pareto | 8200 | 610 | 670 | Inf | 740 | 680 | Inf |
| Burr | **3.9** | 6.6 | **4** | Inf | 6.6 | 8.5 | Inf |
| Inv Burr | 20 | 15 | 20 | Inf | 15 | 14 | Inf |

In the interests of space only AD scores have been shown. The majority p-values round 0.00

methods. Table XVII shows that the AD score on all of the tests is high (an AD score of 3.5 or below would be sufficient to pass the AD test).

TABLE XVII.
ST $<$ 1, FILTER = 1 : AD TEST

| Ad test | data | Log (data+1) | Sqrt (data) | Exp (data) | Sqrt (log (data+1)) | Log (log (data+1) +1) | Sqrt( exp( data)) |
|---|---|---|---|---|---|---|---|
| | **AD Score** | | | | | | |
| Log Normal | 4146 | 4279 | 4146 | 78917 | 4279 | 4427 | 78917 |
| Log Logis-tic | 1099486 | 153957 | 1703852 | 35541593 | 1778558 | 1198064 | 89237003 |
| Gamma | 681391 | 20247 | 7959 | 85258 | 6757 | 14646 | 81636 |
| Weibull | 410883 | 453321 | 615255 | 2912272 | 679932 | 492797 | 9870194 |
| Expo-nential | 131869 | 112365 | 9986 | 200079 | 10607 | 97906 | 216494 |
| Cauchy | 101117 | 101117 | 98590 | 42829 | 103782 | 96204 | 102433 |
| Logistic | 58318 | 54796 | 24413 | 63141 | 23026 | 51976 | 60585 |
| Pareto | 4300 | 4624 | 9986 | 200079 | 10607 | 4936 | 216494 |
| Burr | 4488 | 4682 | 4487 | 54435 | 4682 | 4903 | 54470 |
| Inv Burr | 5210 | 5651 | 5216 | 57655 | 5690 | 6089 | 57675 |

In the interests of space only AD scores have been shown. The majority p-values round 0.00

We also attempted parametric fits to subsets of the data based on the map count. Previously, we saw that splitting the service times by map count reduced correlation. Table XVIII shows the results of fitting followed by an AD test. With a transformation of square root and exponential for the parametric tests, the AD score is exceptionally high. None of the fits passed the AD test.

### C. Non-Parametric Modelling

As no suitable parametric model has been found, we set out to model the head of the data using KDE. We fitted KDE models to the head of the data where split=1. Examining the histograms, we found too much white space for convincing regardless of the bandwidth, kernel, bins, or custom breaks used. We then reduced the data size to a sample size of

TABLE XVIII.
MAP COUNTS = ALL - AD TEST

| | **data+1** | | **sqrt(data+1)** | | **exp(data+1)** | |
|---|---|---|---|---|---|---|
| Transformation | AD Test | P-Value | AD Test | P-Value | AD Test | P-Value |
| Lognormal | Inf | 0.00 | Inf | 0.00 | Inf | 0.00 |
| Log logistics | 53231368 | 0.00 | 132692775 | 0.00 | Inf | 0.00 |
| Gamma | Inf | 0.00 | Inf | 0.00 | Inf | 0.00 |
| Weibull | Inf | 0.00 | 40223101 | 0.00 | Inf | 0.00 |
| Exponential | 273165 | 0.00 | 293850 | 0.00 | Inf | 0.00 |
| Cauchy | 136839 | 0.00 | 135069 | 0.00 | 140489 | 0.00 |
| Logistic | Inf | 0.00 | Inf | 0.00 | Inf | 0.00 |
| Pareto | 278707 | 0.00 | 293850 | 0.00 | Inf | 0.00 |
| Burr | Inf | 0.00 | Inf | 0.00 | Inf | 0.00 |
| Inverse burr | inf | 0.00 | Inf | 0.00 | Inf | 0.00 |

fifty thousand messages, implemented k-fold techniques and re-applied the KDE modelling techniques. From Fig. 11 we observe that KDE does fit the tail of the data reasonably well but see a less convincing fit at the head of the first bin. The model appears to be under-predicting the data within the first bin.

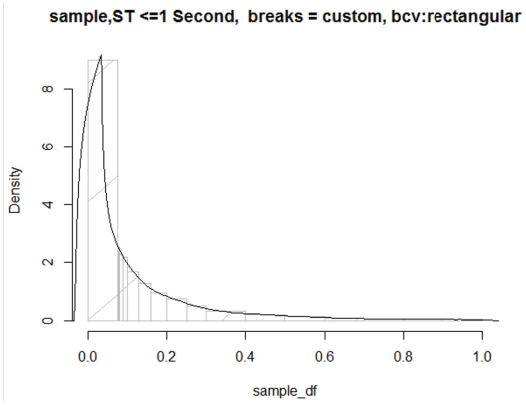


Fig. 11. KDE: ST<=1 Second

Encouraged by these results, we looked at the data grouped by the hour and were able to fit the data using KDE. Fig. 12 shows the resulting histograms, the histogram to the left overlayed with a KDE model uses a bandwidth selector of the Sheather-Jones "plug-in" estimator with a method of "dpi" and an Epanechnikov kernel. We used a break size of 50 on the histogram. The histogram to the right has a bandwidth selector of unbiased cross-validation using a rectangular kernel.

Fig. 13 also shows the resulting histograms, the histogram to the left uses a bandwidth selector of unbiased cross-validation and a triangle kernel. The histogram to the right has a bandwidth selector of Sheather-Jones "plug-in" estimator with a method of "dpi" and a triangular kernel.

Most data split by the hour permitted convincing KDE fits, except for 9 am, and 11 am, where we did not find a good match between the fitted distribution and the histogram.

### D. Message Classification

If we refer back to our hypothesis question on message classification Fig. 5 shows the head of the data transformed

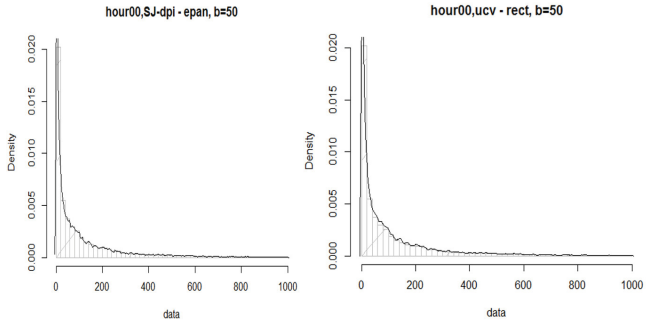


Fig. 12. KDE: Fitting By Hour

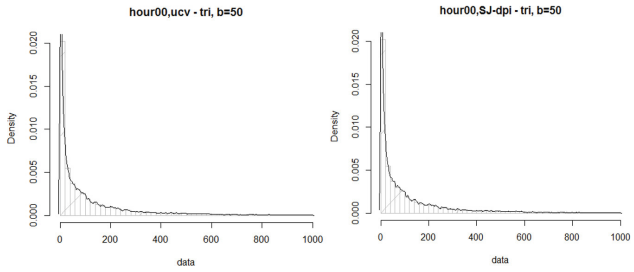


Fig. 13. KDE: Fitting By Hour

for all messages $>$ 0.000 milliseconds and $<$ 1 second. In Table XIX we have attempted to classify the messages between Group 1, which is from - 7 to - 4 (0.001 to 0.018 milliseconds), and Group 2, which is -4 to -2.7 (0.019 to 0.036 milliseconds).

TABLE XIX.
HEAD MESSAGE CLASSIFICATION: GROUP COMPARISON

| Group | Milli sec-onds | Doc Type Count | Max Map Count | File Size Bytes | Splits | Transl-ation Action | EDI Types |
|---|---|---|---|---|---|---|---|
| 1 | 0.001 – 0.018 | 2 | 2 | 60 bytes - 50 mil | 0 – 1521 | Defer, Doc Ex-tract, TX | X12, Edi-fact, Other, Idoc, Ean-com |
| 2 | 0.019 - 0.036 | 2 | 2 | 60 bytes – 11 mil | 0 – 889 | Defer, Doc Ex-tract, TX | X12, Edi-fact, Other, Idoc, Ean-com |

When comparing the two groups in Table XIX, we note that Group 1 has messages with bigger bytes sizes than Group 2. The number of message splits in Group 1 is different to that of Group 2. We note that the more times the message is split, the faster the duration. So, we can say that Group 1's messages are short in duration, and this may be due to the number of times the messages are split. Group 2's smaller split count may explain the increased processing time for the messages.

## V. DISCUSSION

We will reference each section back to the research question asked in Section I. At a high level, we noted that most transactions took under one second to process and contained many discrete values, and so it was useful to split the data into a head and tail part.

*1) Normal and Busy Periods:* We split the data into these two periods for parametric modelling as the busy period was outside of normal conditions. Nevertheless, the normal period with no transformations or filtering did not fit parametric or non-parametric models. However, when we split the data using the different EDI Modelling techniques, we fit much of the data using parametric and non-parametric models. We note that these EDI messages suffer from a high correlation, and caution should be taken when undertaking modelling. We considered splitting the data to understand the source of correlation. We have partly answered our first research question: Can EDI messages be modelled using parametric, or, failing that, non-parametric techniques? We see that we can predict the probability of specific events for random sample messages.

*2) Message Split Count:* We used the message split count to help split up the head of the data. We further noted that correlation was not fully explained by splitting on this feature. For the Split='1' group, the tail of the data fitted a parametric distribution, whilst the head of the data required further effort. This provides a further partial answer to the first research question: Can EDI messages be modelled using parametric, or, failing that, non-parametric techniques? Developers could use this splitting technique in EDI B2B messaging systems to determine if modelling the data by the number of splits shows an increase or decrease in message processing times. Data Scientists could use it for selecting data when fitting parametric models.

*3) Hurdle Modelling:* We implemented a Hurdle model to remove over-dispersion. We removed the zero duration messages as we were mainly interested in messages of a positive duration. This approach improved the fit of the models, but not enough to pass an AD test and did not explain the observed correlation. When speaking to developers about which messages they were concerned with modelling, they were only interested in messages of a duration greater than five minutes. This provides a further partial answer to our first research question: When modelling such messages, it is worth reflecting on which messages may be interesting according to the needs of the group that will use the model, and this may point to modelling a subset of the data.

### A. Message Bundle

We noted that some messages arrived in the system as part of a bundle, where only the last message had a duration greater than zero seconds. Keeping these bundled messages in the system and modelling each one individually may cause correlation in the data due to the time dependence of the

succession of messages and may also cause over-dispersion. We want to identify the causes for correlation and over-dispersion, so we consider removing them from the dataset for modelling. This approach might weaken correlation and reduce over-dispersion. However, implementing the message bundle technique did not fully remove correlation from the model, but it did weaken it. Again, this has partly answered our first research question: bundling messages may be useful to developers when understanding the overall service times for each bundle and so help with understanding the service times and inter-arrival times of the messages (especially when the message has been significantly split). SREs might use bundles to determine the performance load on the system based on modelling of the scheduled time periods, as these bundled jobs may benefit from separate modelling.

### B. Scheduled Versus Un-Scheduled Messages

Scheduling of messages is important for modelling queue behaviour and burstiness. Understanding the scheduled load versus random load helps support queue capacity planning. In isolation removing the scheduled messages from the data did not remove correlation, nor did it support parametric modelling. Again, we get further information on our first research question: EDI messages can have a scheduled component that may require separate modelling. SREs can use this insight to separately consider scheduled frequencies of messages over normal load. Data scientists and researchers can use it to better model incoming jobs via service times and inter-arrival times to better understand queue capacity.

### C. Map Count

A map is a useful feature of an EDI message. These maps have many different authors and editors. Depending on the skill level of the map editor and the content within these maps, they may take different times to process. We expected that as map count might be a proxy indicator for complexity, it might help explain correlation. We found that a combination of splitting the data by Map Count and using message bundling produced uncorrelated data. Thus, it seems like a useful technique for data scientists and researchers who want to break up correlations in EDI heterogeneous messages. So, we have a further partial answer to our first research question.

### D. Parametric Modelling

We noted that modelling all service times or the head of the distribution did not fit a known parametric distribution regardless of transformation or our EDI modelling techniques. However, we could fit the tail of the data to a known Burr distribution. The inter-arrival times of the tail of the data were close to a Burr-type distribution but failed the AD test.

We wanted to understand if service times and inter-arrival times of EDI messages could be modelled effectively, and a parametric approach was only suitable for part of the dataset. It appears further work is required to break up this dataset which appears to contain heavy-tailed and discrete components. Thus, we can conclude that the dataset should be modelled using a simple parametric method. Data scientists and researchers can use this information to guide the steps one can take to model these EDI messages effectively. We have partly answered our second research question: can the ST and IAT of an EDI message be modelled by a parametric method or non-parametric method?

### E. Non-Parametric Modelling

Modelling the head of our data for service times using KDE, we observe a lot of white space indicating that the model does not fit the data. However, breaking the data up by the hour, we observe that a KDE model provides better fits to the data. From that, we have partly answered our second research question: can the ST and IAT of an EDI message be modelled by a parametric method or non-parametric method? As suggested in the previous section, Data scientists and researchers can use this information to guide the steps one can take to model EDI messages using a non-parametric approach.

### F. Message Classification

Our third research question set out to answer whether we can classify EDI messages. On looking at the head of the data up to 0.036 milliseconds, we were able to classify these messages into two groups where group 1 had a shorter processing time than group 2. We also noted that the significant difference between the two groups was the number of times the message got split and the number of bytes in a message. Further work is required to classify these messages better. Message classification is desirable so that SREs can define baseline metrics for the different types of EDI messages coming into the system. The issue is the number of different ways one can classify the messages.

## VI. Conclusion

This study aimed to model B2B EDI messages and determine if modelling was appropriate by a parametric or non-parametric approach. Modelling this data set is quite challenging. We noted evidence of correlation throughout the dataset, and the understanding of correlation was difficult. We note that a “one size fits all approach” is not appropriate; instead, we would require a combination of approaches to fit such data adequately. We found that the head of our data could not be modelled using parametric techniques and that the tail of the data fitted a parametric distribution. We also found that a non-parametric approach was suitable to model a portion of our data based on how we sliced the data. This work provides a more refined study, specifically modelling heterogeneous messages within the Cloud Supply Chain domain. By using a parametric approach, we modelled the tail of the data using a Burr Distribution. Modelling the data by hour fitted a non-parametric model.

In future work, we can investigate modelling the busy period service times and identify if that period fits a parametric distribution. Additionally, further work is required to identify the underlying correlation within queue service time data and

if we can remove the correlation whether a parametric or non-parametric modelling approach is appropriate.

## REFERENCES


[1] Ellis, S.,Bond, S., Marden, M., And Singh, H., Driving strategic value with IBM Sterling Supply Chain business network. https://www.ibm.com/downloads/cas/PZ7LROWL

[2] Hassellof, A., Why queue management systems are essential for modern businesses, https://ombori.com/blog/modern-queue-management-systems

[3] Top 10 future trends in supply chain and logistics, https://www.aacb.com/trends-in-supply-chain-and-logistics/

[4] Heyde, C. C. 'Agner Krarup Erlang,Statisticians of the Centuries',Springer New York,2001,pp. 328–330.

[5] Cremer, M., And Ludwig, J. (1986). A fast simulation model for traffic flow on the basis of boolean operations. Mathematics and computers in simulation, 28(4), 297-303.

[6] Iversen, T. (1993). A theory of hospital waiting lists. Journal of Health Economics, 12(1), 55-71.

[7] Hermanto, R. P. S., And Nugroho, A. (2018). Waiting-time estimation in bank customer queues using RPROP neural networks. Procedia Computer Science, 135, 35-42.

[8] Ross, S. M. (2014). Introduction to probability models. Academic press.

[9] Abeyratne, S. A., & Monfared, R. P. (2016). Blockchain ready manufacturing supply chain using distributed ledger. International Journal of Research in Engineering and Technology, 5(9), 1-10.

[10] Barber, D. (1995). Electronic data interchange. Library Technology Reports, 31(5), 499+. https://link.gale.com/apps/doc/A17937141/AONE?u=nuim&sid=bookmark-AONE&xid=7a08fede

[11] IBM, What is electronic data interchange (EDI)?, https://www.ibm.com/topics/edi-electronic-data-interchange

[12] Iacovou, C. L., Benbasat, I., & Dexter, A. S. (1995). Electronic data interchange and small organizations: Adoption and impact of technology. MIS quarterly, 465-485.

[13] Lim, D., & Palvia, P. C. (2001). EDI in strategic supply chain: impact on customer service. International Journal of Information Management, 21(3), 193-211

[14] Garg, N. (2013). Apache kafka. Packt Publishing Ltd.

[15] Toshev, M. (2015). Learning RabbitMQ. Packt Publishing Ltd.

[16] IBM, Introduction to IBM MQ,https://www.ibm.com/docs/en/ibm-mq/8.0?topic=overview-introduction-mq

[17] Rabbit MQ, https://www.rabbitmq.com/#features

[18] HG Insights, RabbitMQ, https://discovery.hgdata.com/product/rabbitmq

[19] Sax M.J. (2018) Apache Kafka. In: Sakr S., Zomaya A. (eds) Encyclopedia of Big Data Technologies. Springer, Cham. https://doi.org/10.1007/978-3-319-63962-8_196-1

[20] YANG, P., Building a machine learning logging pipeline with Kafka Streams at Twitter, https://www.confluent.io/blog/how-twitter-built-a-machine-learning-pipeline-with-kafka/

[21] Lee, J., How LinkedIn customizes Apache Kafka for 7 trillion messages per day, https://engineering.linkedin.com/blog/2019/apache-kafka-trillion-messages

[22] Sharma, N.,Featuring Apache Kafka in the Netflix studio and finance world, https://www.confluent.io/blog/how-kafka-is-used-by-netflix/

[23] Deutscher, M., AWS expands its serverless capabilities and adds a managed Kafka service, https://siliconangle.com/2018/11/29/aws-expands-serverless-capabilities-adds-managed-kafka-service/

[24] HG Insights, Apache Kafka,https://discovery.hgdata.com/product/apache-kafka

[30] Pearson, Karl. "Contributions to the mathematical theory of evolution." Philosophical Transactions of the Royal Society of London. A 185 (1894): 71-110.

[25] ACTIVE MQ,Flexible & Powerful open source multi-protocol messaging, https://activemq.apache.org/

[26] Allen, A. O. (2014). Probability, statistics, and queueing theory. Academic press.

[27] Laguna, M., And Marklund, J. (2019). Business process modeling, simulation and design. Chapman and Hall/CRC.

[28] Shortle, J. F., Thompson, J. M., Gross, D., Harris, C. M. (2018). 'Fundamentals of queuing theory' (Vol. 399). John Wiley And Sons.

[29] Casella, G., & Berger, R. L. (2021). Statistical inference. Cengage Learning.

[31] Fisher, R. A. (1925, July). Theory of statistical estimation. In Mathematical proceedings of the Cambridge philosophical society (Vol. 22, No. 5, pp. 700-725). Cambridge University Press.

[32] Razali, N. M., And Wah, Y. B. (2011). Power comparisons of shapiro-wilk, kolmogorov-smirnov, lilliefors and anderson-darling tests. Journal of statistical modeling and analytics, 2(1), 21-33.

[33] Massey Jr, F. J. (1951). The Kolmogorov Smirnov test for goodness of fit. Journal of the American statistical Association, 46(253), 68-78.

[34] Lilliefors, H. W. (1967). On the Kolmogorov-Smirnov test for normality with mean and variance unknown. Journal of the American statistical Association, 62(318), 399-402.

[35] T. W. Anderson and D. A. Darling, "A test of goodness of fit," Journal of the American statistical association, vol. 49, no. 268, pp. 765–769, 1954.

[36] Asmussen, S. R. (2003). "Steady-State properties of GI/G/1". Applied Probability and Queues. Stochastic Modelling and Applied Probability. 51. pp. 266–301. ISBN 978-0-387-00211-8.

[37] Mandelbrot, B. B. (1997). The variation of certain speculative prices. In Fractals and scaling in finance (pp. 371-418). Springer, New York, NY.

[38] Hill, B. M. (1975). A simple general approach to inference about the tail of a distribution. The annals of statistics, 1163-1174.

[39] Zeileis, A., Kleiber, C., and Jackman, S. (2008). Regression models for count data in R. Journal of statistical software, 27(8), 1-25.

[40] Arnold, B. C., Balakrishnan, N., Alegria, J. M. S., and Minguez, R. (Eds.). (2009). Advances in mathematical and statistical modeling. Springer Science and Business Media.

[41] Silverman, B. W. (2018). Density estimation for statistics and data analysis. Routledge.

[42] Epanechnikov, V. A. (1969). Non-parametric estimation of a multivariate probability density. Theory of Probability & Its Applications, 14(1), 153-158.

[43] Sheather, S. J., & Jones, M. C. (1991). A reliable data-based bandwidth selection method for Kernel Density Estimation. Journal of the Royal Statistical Society: Series B (Methodological), 53(3), 683-690.

[44] Park, B. U., & Marron, J. S. (1990). Comparison of data-driven bandwidth selectors. Journal of the American Statistical Association, 85(409), 66-72.

[45] Goodhope, K., Koshy, J., Kreps, J., Narkhede, N., Park, R., Rao, J., & Ye, V. Y. (2012). Building LinkedIn's real-time activity data pipeline. IEEE Data Eng. Bull., 35(2), 33-45.

[46] Wu, H., Shang, Z., & Wolter, K. (2019, August). Performance prediction for the Apache Kafka messaging system. In 2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS) (pp. 154-161). IEEE.

[47] Henjes, R., Menth, M., & Zepfel, C. (2006, July). Throughput performance of java messaging services using websphereMQ. In 26th IEEE International Conference on Distributed Computing Systems Workshops (ICDCSW'06) (pp. 26-26). IEEE.