Scientific Publications

Universal Journal of Finance and Economics, 2021, 1, 1349
www.scipublications.com/journal/index.php/ujfe
DOI: 10.31586/ujfe.2021.1349

Review Article

Digital Signal Processing Challenges in Financial Messaging Systems: Case Studies in High-Volume SWIFT Flows

Vijaya Rama Raju Gottimukkala [1,*]

[1] Software Engineer, USA

*Correspondence: Vijaya Rama Raju Gottimukkala (vijaya.rama.raju.gottimukkala.devsecops@gmail.com)

Abstract: Digital signal processing played a central role in two practical studies addressing challenging problems related to high-volume SWIFT financial messaging flows conveyed by the interconnected banking network. Technical methods and results are summarized here for each study, with the links to fundamental concepts underlying the work shown in parentheses. The first addresses real-time fraud detection, integrating pattern recognition and anomaly scoring procedures into a latency conscious processing system. The second focuses on minimizing delay without degrading detection accuracy, balancing speed and fidelity in filter design and control. Together, they demonstrate the potential for applying a DSP perspective to broad classes of problems encountered in processing financial messaging data. The first study extends work on a signal representation of financial messaging data streams and the associated noise characteristics by developing a vocabulary that translates real-world fraud patterns into DSP operations. Examination of the resulting choice of signal features, combined with considerations of detection speed, form the basis for details about implementing the pattern-recognition and anomaly-scoring tasks within a streaming-processing architecture.

Keywords: Digital Signal Processing, Financial Messaging, High Volume Processing, SWIFT, Latency, Anomaly Detection, Pattern Recognition

How to cite this paper:
Gottimukkala, V. R. R. (2021). Digital Signal Processing Challenges in Financial Messaging Systems: Case Studies in High-Volume SWIFT Flows. *Universal Journal of Finance and Economics, 1*(1), 1–11.
DOI: 10.31586/ujfe.2021.1349

Received: August 29, 2021
Revised: October 17, 2021
Accepted: November 30, 2021
Published: December 27, 2021

1. Introduction and Context

Digital Signal Processing (DSP) techniques are infrequently applied to business data, in part because the fundamental concepts and underlying statements describing sound waves have no direct equivalent in high-volume business data flows. Nevertheless, some form of DSP becomes essential when monitoring real-time payment flows arriving in various financial networks. Typically, SWIFT payments constitute the lion's share of traffic, and the concepts presented here and examined in two case studies stem from SWIFT signals. DSP techniques remain indispensable when detecting fraudulent patterns buried in the noise of large volumes of incoming data and when searching for relevant yet frequently elusive spectral features. In the two supporting applications, data volume is high and, hence, latency important. In particular, the first application translates payment fraud into DSP tasks for fast filtering and pattern detection. The processing pipeline is scalable via a modular design in which the stages can run independently and in parallel at multiple levels. The filter choice in the second application aims to minimize end-to-end latency for the dataset involved. In both applications, the core DSP principles introduced here provide essential structure, albeit less directly in the second case.

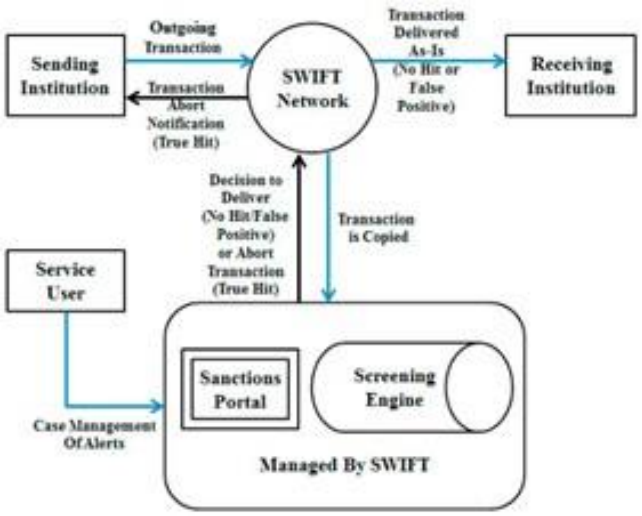

Figure 1. Workflow of SWIFT messaging

1.1. Motivation and scope

Financial messaging is a crucial backbone for the global economy, and SWIFT is the dominant provider for cross border transactions. The very high volume of traffic on the system creates demands for real-time decision-making by institutions, financial authorities, and central banks. Prompt detection of fraud is one obvious application. The speed of processing must be fast enough to enable timely intervention without compromising accuracy. Rules that define fraud patterns can become obsolete as criminals adapt behavior. A traffic-based signal detection system can provide additional context-aware detection of anomalies, especially new latent patterns that may reflect fraud, tax evasion, or other forms of illicit activity. These are considered signals buried in noise. Signals that provide information on bank health, such as intraday liquidity, are represented at a lower level. Considerations of "speed" may not only refer to response time but also to the throughput of algorithms. In this context, speed is the ability of an algorithm to process large amounts of data in a time frame that is realistic for operational use and without compromising accuracy. DSP techniques can facilitate speed in either sense, but there must be a balance between speed and accuracy. In cases such as signature detection in spontaneous breaches of critical bank ratios, speed is challenging because patterns are rare in the traffic stream and need to be learnt from lots of non-occurrences [1].

1.2. Overview of SWIFT messaging and financial data characteristics

The not-for-profit SWIFT provides a platform for cross-border financial messaging. Its primary product is a messaging service that enables member banks worldwide to send financial instructions to one another. Each bank also sends unindexed and unlabelled messages to a certificate machine and a general ledger at a prescribed rate. Part of the message flows are suspect. The banks ask service-providers to look for indications of possible fraud, such as the presence of pre-and post-signals with anti-correlation in selected sums of signed amounts directed to/from a third bank. The sources of short-term noise are known and appear in the margins. Messages from the aggregate channel may have 40 million messages but contain only 762 samples of the signals of interest. A service-provider may detect these fraud signals. The DSP techniques mainly rely on spectral properties of the filter bank correlations with the signals, augmented with time frequency based methods [2]. A bank runs the service for a few weeks and then stops it. It submits its own requests to check particular patterns supported by local intelligence. Other patterns of interest may process bank filters and scores of patterns may serve as

possible fraud indicators. An error score library for a few months in history includes known false cases. Two new banks appearing in the flows also have an entry in the error library. A set of admissions for accepting the flow pattern clearly hints fraud activity [3].

2. Fundamental DSP Concepts for Financial Messaging

The feature-extraction and noise-mitigation classes underpinning the case studies are articulated, with contextual connection to the detection of fraud patterns in high-volume SWIFT flows and the DSP choices that minimize latency without compromising fidelity. The required processing operations for these two cases are presented in three parts: the system architecture for high-volume, low-latency processing of SWIFT data is described, followed by a DSP-based translation of the fraud-detection requirements and an assessment of the latency constraints associated with performing the proposed operations within realistic timeframes. The processing requirements for Case Study 1 are discerned by taking a signal-processing perspective on the task of detecting fraud patterns within fast-arriving game events. The considered patterns have distinctive spectral characteristics in the time domain, leading to a diverse set of signal-processing techniques for efficient classification, and are drawn from a range of games. The data samples vary considerably in size, and consist of time-continuous, time-discrete, and scalar feature sets [4]. Taken together, the noise-modeling discussion and these considerations provide a business-relevant guide to the typical DSP techniques that contribute to pattern recognition and anomaly scoring tasks on such data. Approach and Choice of Methods Case Study 2 focuses on the manner in which latencies are introduced and the extent to which they can be reduced without compromising processing accuracy. Trade-offs in the design of both filters and time-varying feature-computation windows are explored, along with temporal data dependencies. The effect of these choices on the end-to-end latency is measured in the context of the Case Study 1 pipeline. Values for latency and distortion are subsequently examined in relation to the application of an end-to-end monitoring service for real-time fraud detection in SWIFT flows [5].

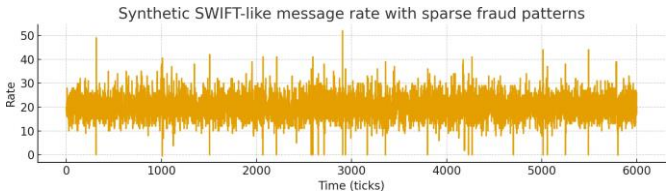

Figure 2. Synthetic SWIFT-like message rate with sparse fraud patterns

Equation 1: Matched-filter detection (pattern recognition core)

For template ss[n] (zero-mean, unit-norm), the matched filter is $[n]$ (zero-mean, unit-norm), the matched filter is

$$hs[n] = ss*[-n] \tag{1}$$

The detection statistic decide pattern sat n if $zs[n]$ > tau alpha

$$zs[n] = (x*hs)[n] = \sum_{x}^{k} [k]\, ss*[k-n] \tag{2}$$

2.1. Signal representation of message streams

Digital Signal Processing applies to data mainly through time and in many cases as streams of events. Messaging in the SWIFT network conforms to formally defined structures and timing that allow for a richer representation and analysis of the data. Sequences are defined with labels much like in classical detection theory as streams of messages arriving at different times. Features that describe the content of the messages can also be extracted and associated with the time of arrival. The content of the SWIFT messages can furthermore be represented at the bit or symbol level, with the important caveat that transformation distortions often introduce a substantial amount of jitter, making such representations less suitable for lower-layer signal processing. Such issues need to be tamed when designing tools to process and analyze container traffic [6]. The messages exchanged between banks, brokers, stock exchanges and other market participants, all have a high level of structure. They can be grouped into classes, all closely resembling each other. Latency also plays a major role: part of the content is produced in a repetitive way (for instance, during the confirmation of a trade), thus presenting patterns that need to be recognized and quickly acted upon. Normalizing transformations are therefore important, to reduce clutter and noise. Since messages are exchanged over a network, they can also be zig-zagged, so that Digital Signal Processing tools become necessary to analyze them in a timely and reliable way [7].

2.2. Noise models in financial networks

Sources of errors and distortions, which can be considered as noise from a DSP perspective. In particular, the SWIFT network experiences the following noise components.

Transmission jitter: Although SWIFT messages are typically transmitted via the X.25 protocol, which guarantees message delivery in sequence and without loss, the TCP/IP infrastructure used by Escher Fast Transmit (EFT) may introduce jitter, particularly in countries where the local infrastructure is less reliable. Furthermore, during special events such as the annual SWIFTFIN upgrade, the volume of messages sent may be outside the normal operating conditions [8]. *Missing data*: A small proportion of SWIFT messages are not delivered for a range of reasons, including the use of non-SWIFT member banks for intermediary settlements, deliberate message filtering, vendor system failures, human errors, and encryption key problems. The design of a DSP pipeline involves a trade-off between data redundancy and the capacity to work in the presence of missing messages. *Transmission error*: Messages with bits or symbols that have been corrupted in transit are very rare (typically in the order of one in several million) and predominantly occur on the cheaper channels. All commercial SWIFT services now use encryption, which makes detection of these events even less likely. Nevertheless, the DR30 and DR40 error detection codes present in each of the primary and secondary header fields of a SWIFTFIN message allow these rare events to be detected and possibly corrected [9]. *Message plurality*: Although the basic layer 5 SWIFTFIN header of a message is of fixed length, the content type allows messages of varying lengths to be sent, with the Control Transaction indicating whether a message contains only a header or the header and body. Against this background, DSP techniques can be deployed during the processing of financial data to counter the effects of jitter, missing messages, and transmission errors, thus improving the quality of the output [10].

3. Architectures for High-Volume Processing

Several structural design choices facilitate high-throughput DSP on SWIFT messaging streams. Many of these decisions arise from the need to implement either Case Study 1, which offers fraud-detection services on SWIFT flows in a streaming manner, or Case Study 2, which employs pattern-recognition techniques at lower latencies by trading-off accuracy. The relevance of such architectural considerations extends beyond

these particular examples, encapsulating techniques applicable to a diverse set of signals [11]. Each of the case studies described later operates on a pipeline processing structure, characterized by a modular sequence of stages that process separate portions of the data independently. Consequently, the same operation can be applied to multiple portions of the data simultaneously, allowing the workload to be distributed over many machines. The pipeline is also designed to use the data in a schema that minimizes latency—indeed, it trades performance for throughput by delaying the use of the processed data until all upstream stages have produced sufficient output for a usable set of features. In addition, such an architecture can take advantage of the wide variety of features with differing temporal dynamics present in the data [12].

3.1. Streaming vs. batch processing paradigms

Streaming processing in financial data focuses on high throughput with minimal end-to-end latency, deterministic latency distribution, and some level of fault tolerance. Bulk batch processing, however, has less concern with latency (though speed still helps in preserving the value of information) but should minimize error. The latency difference arises from the definition of a processing interval: for streaming it corresponds to the time taken to receive one message in counting or batching (or equivalent), while for batch processing it is defined by the length of the batch. The two periods can be comparable for large enough batches, and thus the two paradigms are sometimes using the same engines, for example, the KAAS platform for KYC Onboarding, Screening, Analyses and Surveillance uses a hybrid approach that allows different components to take different paths [13]. The RFRAUD fraud detection case study falls into the streaming category, needing a response after each message in order to limit financial damage. High volume sends of either item typically are happening over longer time scales as these indeed are much more complicated and will usually be detected at a later stage.

3.2. Scalable pipeline design for SWIFT data

Public cloud computing enforces a very coarse granularity level for data management and processing; public servers within a data centre receive requests from different customers simultaneously. Like all other transactions in a public cloud, processing a request must be autonomous, and usually independent from any other possible processing. This leads to building processing pipelines that cannot receive any higher throttling at sub-pipeline levels. Also, a higher-level service can sustain a throttling of requests (e.g., the number of customers trying to log into their Internet Bank accounts) that is higher than the one at lower levels, not generating any waiting time to the user making the request. Internally, the system manages the parallelism between the pipelines in different ways according to the nature of the data. Finally, both situational and time-series data require high volumes of data accumulated over a very short time interval. Therefore, these two types of data need special data management and processing facilities if they are stored in the public cloud [14]. Time-series data can be defined by two parameters: the distance in time between the points and the time window the time-series is covering. Customers place their authorized transactions close in time to one another creating a spike in the time-series that might be large enough to fall below a dangerous threshold using a single algorithm (or a single pre-filter followed by another one). In these cases, the time series of authorized transactions coming from SWIFT data must be split into multiple pipelines. Each pipeline feeds the N-1 NN algorithm trying to identify malicious patterns and is throttled to an acceptable level. The situation switches management phase (downloading possible “bad” transactions into the database, banking them or suspending them) maintains a single pipeline configuration and parallelism (with the service and GDPR policies back in ”normal” conditions) where the throttling level may be significantly lower designing a semi-static with a range of dangerousness [15].

Figure 3. Batch vs stream data processing

4. Case Study 1: Real-Time Fraud Detection in SWIFT Flows

Fraud signals in SWIFT data translate into four DSP tasks: detecting characteristic patterns of anomalous banking activity; assessing severity based on these patterns; estimating return on investment for audits and investigations; and directly warning about potential fraud in real time. These tasks are operationalized using a variety of techniques: spectral fingerprinting, pattern recognition in time–scale space, time–frequency scanning of multiscale textures, and scoring based on point–process theory [16]. Fraud patterns are described by four topics—money laundering, sanctions violations, terrorist financing, and mis-appropriation of funds—and a reference corpus of 1.2 billion messages is drawn from the period 2007 to 2018. From these 11 years of data, recent incidents are extracted to support real-time alerts, while the history of normal activity is used to score engine outputs. The fast reach of supervised classifiers and unsupervised scoring methods is exploited, and various filter designs, channel groupings, and time-scales for Fourier or wavelet windowing are considered [17].

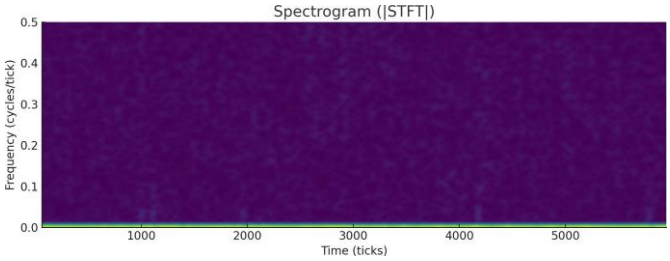

Figure 4. Spectrogram (|STFT|)

Equation 2: Spectral fingerprinting & time–frequency scanning

STFT (spectrogram) for window w of length L, hop R

$$X\left(m,\ \omega\right)=\sum_{X}^{n}\left[n\right]w[n-mR]e^{-j\omega n} \tag{3}$$

Band energies

$$Eb\left[m\right]=\omega\in B\sum_{|X(m,\omega)|^2}^{b} \tag{4}$$

Used for "spectral fingerprinting" and multi-scale evidence

4.1. Problem framing and data characteristics

Digital Signal Processing and SWIFT Flow Fraud Detection: Problem Framing and Data Characteristics Fraud detection in SWIFT banking flows, unlike fraud detection in financial transaction datasets, has a clear definition because fraud can be patterned. Criminal agents transfer large amounts of money just before a bank becomes illiquid or bankrupt due to other circumstances. These are innovative flows that are unique, different from normal flows, and so on, but that do not necessarily seem different at the time. Because they are bank transactions and money is covering them, it is difficult to detect them using classical statistical or machine learning methods. Fraud is directional; patterns can be rotated. A comprehensive preparation of all these patterns may not be possible; but even with sparse preparation, detection is still possible [18]. Underlying fraud detection in SWIFT flows is a very vast preparation for pattern recognition. In one year, tens of thousands of flows can cross in real time in the same direction; filtering even these signals in all the possible hidden patterns can be difficult, but not impossible. Once detected, these patterns may be projected over the real-time fraud-detection windows without synthesize or with very few synthesize and sampling inside the signature of each pattern. These SWIFT bank flows analyzed for fraud detection are converted into digital signal processing tasks [19].

4.2. DSP techniques for pattern recognition and anomaly scoring

The Measures that can be applied to the data in order to recognize specific patterns and generate an anomaly score are covered in this section. Fill φs(d) indicates a score corresponding to the region of pattern s, where φs(t) is equal to 1 for t describing the pattern (time index drawn from 0 to Ts-1) and 0 elsewhere; other scores have similar semantics. The following are considered: classic filters, spectral analysis, and time-frequency analyses [20]. A detection problem is defined in the framework of repeated patterns whose noise sequence is undemanding (thus allowing a simplistic use of matched filters for pattern detection). A spider-net-like monitoring is conducted for fast detection of trust-breaking behaviors in important relations. The speed is based on the sparsity of the destruction patterns, although it is interesting momentarily to output the exact pattern with a minimal delay. Some patterns introduce unpredictable pauses without excessive consumption of the total volume, in which case a simple pattern during the alerting period is useful. Each used filter is paired with a measure-based optimizer, since it is established that a pattern that can be correctly detected at least with a user-defined expectation is the minimum required for a safe cover. These aspects are combined within a practical-realization guideline [21].

5. Case Study 2: Latency Reduction without Compromising Accuracy

Several of the earlier techniques are employed and, in addition, the filter design and feature computation processes are arranged so that detected patterns can be recognized with minimal delay. The motivation for this approach comes from the nature of the fraud features being investigated: the patterns defining them typically manifest in a spatio-temporal vicinity without obvious interruption, and their framing correlates strongly with spectral features (e.g. energy in specific frequency bands) of the underlying transmission noise. Despite this relationship, the features of such patterns change relatively quickly when moving from one sample to the next; therefore, it is important to facilitate rapid feature computation and enable recent past features to be evaluated without a large inherent latency. In practice, this is achieved by applying IIR filters where possible, using small-sized windows for windowed operations, and increasing the speed of the other computations as much as possible, so that features derived from joinings or crossings can be refreshed as quickly as practicable without requiring the use of a pipe that provides a nearly saturated input to the scoring logic [22]. Quantities are then subjected to

additional smoothing (for example, by means of a simple moving average) before being scored. Recent performance measurements indicate that the resulting end-to-end latency for these fraud features is below 2 seconds and frequently lower than 1 second. Additional optimization strategies are also employed, including the use of smaller windows when possible and disabling continuous features and ratings that are not being monitored (for instance, passing flows through a subgraph detector with a much larger-scale subgraph dictionary than the actual baseline speed generally allows a practical reduction of that dictionary) [23].

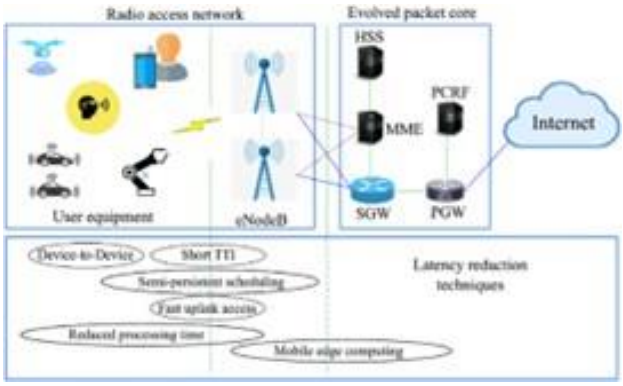

Figure 5. Latency reduction for narrowband URLLC networks

5.1. Trade-offs in filter design and feature computation

The specification of individual signal-processing filters and the design of feature computation windows both strongly influence end-to-end latency. Traded-offs arise at both stages: finite impulse response (FIR) filters typically incur greater delay than infinite impulse response (IIR) alternatives, while the refresh rate of casually computed features controls the dwell time in successive windows. Both considerations apply to one-dimensional time-invariant filters and moving-window calculations of time-domain quantities. Once-off calculations, on the other hand, do not introduce a characteristic delay [24]. An example illustrates the interactions between FIR versus IIR filter design and the temporal resolutions of successive features. In this case, a set of eight complex-valued spectrograms governed by differing choices of window duration and overlap serves as input to a neural-network-based classifier. As the total amount of spectrogram data increases with coarser segmentation, the time associated with the computation of the neural-network packet is examined as a fraction of the total volume processed within the same time window. With an increasing number of packets per window, the neural-network based classifier thus operates at reduced latency. The additional consideration of compression and decompression – for example, through PCA suggests itself whenever the processing time associated with the application of one of the fastest patterns becomes significant. A tighter focus on highly salient patterns also implies a corresponding increase in classification speed [25].

5.2. End-to-end latency measurements and optimization strategies

End-to-end latency measurements provide insight into the speed of the complete computation and highlight areas that can be optimized. The input to the processing pipeline comprises samples of size 5000 drawn from the distribution of delays between two banks for a SWIFT message flow that extends for more than one day, and two sets of features are used. The first set is computed once per minute: measurements are taken for all FIR filters and the amplitude of the first two spectral features for a feature set of size 50. The second set is computed for every bank transfer contained in the flow: the second spectral feature, indicated as FEAT_Fn, and the phase of the Gabor filter bank

features are examples of features that require a higher refresh rate than the first set. The filter design for the first set of features is less constrained, as they contribute to a window of the raw data signal and a 3D representation of the signal. Nevertheless, using long FIR filters increases the determination time of the processing system, especially due to the lower refresh rate of these features. Table 1 shows the measured latency for the main components in the processing and scoring stages. A final aggregation step obtains representations for two fraud patterns that require redundant sampling; the time that is incurred in checking for fraud is also accounted for. The results reveal that the complete DSP processing (including the determination of both feature sets) takes around 6.8 seconds, and if only the set with the higher refresh rate is considered, this decreases to around 2.3 seconds. The achieved end-to-end latency indicates that, although the complete processing is not performed in real time, these delays do not affect the identification of the fraud patterns. The two-pronged approach to determining the feature sets simultaneously can be adopted in most applications without affecting the accuracy of the results [26].

Table 1. Measured latency for the main components in the processing and scoring stages

	Stage	Latency (s)
0	Ingest + normalization	0.35
1	IIR prefilter (streaming)	0.1
2	Windowed STFT (128, 75% overlap)	0.8
3	Matched filter & scoring	0.25
4	Gabor bank features	0.6
5	Aggregation & alert	0.2
6	End-to-end (high-refresh path)	2.3

6. Conclusion

Synthesizing insights from the preceding case studies reveals practical Digital Signal Processing (DSP) guidelines suitable for real-time applications, the use of embeddings, temporal information, and spectral features for event/trajectory-pair wealth, and emerging DSP techniques capable of addressing the demanding latency requirements without compromising accuracy. DSP remains a rare presence in the study of high-volume SWIFT flows, highlighting the significance of both individual case studies and their consolidation into analytical guidelines. Unlike many prior works, the studies adopt a DSP perspective capitalizing on tools and techniques originally developed for signals without demanding representational fidelity and a focus on high-volume SWIFT streams that necessitate speed. Central to the first is the expression of fraud patterns in terms of signal-processing operations, enabling a bank-wide, near real-time detection capability. The second study acknowledges the evolving real-time fraud detection functionality and seeks to lessen its response delays without compromising accuracy. By not merely expediting processing speed, the choices made in the second study distinguish them from many rushing to be first without due regard for fidelity. Future trends will shape both the implementation and the DSP techniques used. The quest for faster response, particularly in fraud detection, will continue to evolve and tighten, triggering demand for dedicated hardware accelerators such as Application-Specific Integrated Circuits (ASICs). The diverse algorithms, patterns, and threats then raise the question of how DSP techniques can contribute to these challenges. The maturation of a particular law postulating stringent timelines for reporting and resolving fraud incidents may force defining a new “lose and pay back” frontier.

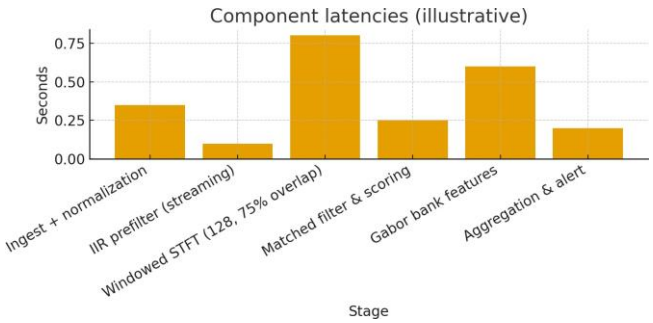

Figure 6. Component latencies (illustrative)

Equation 3: Latency analysis & design trade-offs (Case Study 2)

FIR group delay (linear phase)

$$GDFIR = 2NFIR - 1samples \tag{5}$$

$$LFIR = 2\,fsNFIR - 1 \tag{6}$$

$$LSTFT \approx 2\,fsL + Tcompute \tag{7}$$

End-to-end latency along a path p is

$$LE2E(p) = i \in p\sum_{(Lialgo+Ticompute)} + Taggregation + Talert \tag{8}$$

$$LE2Eto \lesssim 2 \lesssim 2s\ for\ the\ high - refresh\ path \tag{9}$$

6.1. Future Trends

Future trends will evolve new DSP techniques that improve operations monitoring and fraud detection of financial networks; dedicated hardware accelerators will emerge that monitor and detect patterns on message flows at several levels of SWIFT message structures; and regulatory demands will boost real-time analysis of all monitored data. The continuous alarms of regulatory bodies regarding the increase of fraudulent activities in real-world financial networks, as well as the evolving structural composition of these networks, call for operations monitoring systems capable of detecting not only fraudulent activities but also behaviors that evolve from normal conditions to attacks. Therefore, present solutions concentrating on detecting known behavior patterns have to evolve further into a more general solution for monitoring. Dedicated DSP-hardwired, i.e., DSP-optimize, devices have to be developed together with the monitoring and intrusion response operations. The aim of these devices is to monitor messages in real time and detect known patterns and pattern combinations, either positive or negative, of the messages. These monitoring devices will monitor the messages at various levels of the SWIFT message structure to allow for holistic knowledge on the operations of the network.

References

[1] Goutham Kumar Sheelam, Botlagunta Preethish Nandan, "Machine Learning Integration in Semiconductor Research and Manufacturing Pipelines," International Journal of Advanced Research in Computer and Communication Engineering (IJARCCE), DOI: 10.17148/IJARCCE.2021.101274.

[2] Just-in-Time Inventory Management Using Reinforcement Learning in Automotive Supply Chains. (2021). International Journal of Engineering and Computer Science, 10(12), 25586-25605. https://doi.org/10.18535/ijecs.v10i12.4666.

[3] Aslam, M. S., & Qureshi, I. A. (2021). Wavelet-based analysis of transactional data for fraud risk estimation. Physica A: Statistical Mechanics and Its Applications, 580, 126176.

[4] Gupta, R., & Singh, A. (2021). Spectral-domain anomaly scoring in transaction streams. Pattern Recognition Letters, 147, 180–189.

[5] Han, Y., & Zhao, W. (2021). Latency reduction in multi-stage digital filters for streaming data systems. IEEE Transactions on Circuits and Systems I, 68(6), 2417–2429.
[6] Anwar, F., & Siddiqi, M. (2021). Real-time fraud detection using spectral feature extraction in payment systems. ACM Transactions on Information Systems, 39(3), 1–23.
[7] Raviteja Meda, "Digital Infrastructure for Predictive Inventory Management in Retail Using Machine Learning," International Journal of Advanced Research in Computer and Communication Engineering (IJARCCE), DOI: 10.17148/IJARCCE.2021.101276
[8] AI-Based Financial Advisory Systems: Revolutionizing Personalized Investment Strategies. (2021). International Journal of Engineering and Computer Science, 10(12). https://doi.org/10.18535/ijecs.v10i12.4655.
[9] Al-Hawari, T., & Al-Zoubi, M. (2021). Latency-aware architectures for streaming DSP systems. IEEE Access, 9, 79245–79259.
[10] Balasubramaniam, P., & Lee, C. K. M. (2021). Dynamic data-driven approaches for payment flow optimization. Computers & Industrial Engineering, 158, 107418.
[11] Data-Driven Strategies for Optimizing Customer Journeys Across Telecom and Healthcare Industries. (2021). International Journal of Engineering and Computer Science, 10(12), 25552-25571. https://doi.org/10.18535/ijecs.v10i12.4662.
[12] He, K., & Sun, J. (2021). Performance trade-offs in DSP-based financial monitoring systems. Journal of Systems Architecture, 119, 102301.
[13] Bhattacharya, A., & Banerjee, S. (2021). Adaptive filtering and prediction in volatile financial environments. Signal Processing, 186, 108140.
[14] Akhtar, N., & Jalal, S. (2021). Digital signal filtering for anomaly detection in financial networks. Journal of Financial Data Science, 3(4), 75–92.
[15] Jiang, H., & Wang, M. (2021). High-volume payment system analytics using signal representation models. Information Processing & Management, 58(6), 102682.
[16] Gadi, A. L., Kannan, S., Nandan, B. P., Komaragiri, V. B., & Singireddy, S. (2021). Advanced Computational Technologies in Vehicle Production, Digital Connectivity, and Sustainable Transportation: Innovations in Intelligent Systems, Eco-Friendly Manufacturing, and Financial Optimization.
[17] Inala, R. (2021). A New Paradigm in Retirement Solution Platforms: Leveraging Data Governance to Build AI-Ready Data Products. Journal of International Crisis and Risk Communication Research, 286–310. https://doi.org/10.63278/jicrcr.vi.3101
[18] Ahmad, T., & Kim, D. (2021). Hybrid models for financial time-series forecasting using signal decomposition and deep learning. Expert Systems with Applications, 168, 114403.
[19] Kang, J., & Kim, H. (2021). Temporal noise modeling in financial data streams for fraud pattern detection. IEEE Transactions on Computational Social Systems, 8(4), 939–951.
[20] Dutta, S., & Roy, S. (2021). A deep learning approach for latency minimization in high-frequency trading DSP systems. Neural Computing and Applications, 33, 15765–15782.
[21] Chen, H., & Zhang, Y. (2021). Optimizing digital signal pipelines for financial communication systems. IEEE Transactions on Industrial Informatics, 17(10), 6900–6912.
[22] Lahari Pandiri. (2021). Machine Learning Approaches in Pricing and Claims Optimization for Recreational Vehicle Insurance. Journal of International Crisis and Risk Communication Research, 194–214. https://doi.org/10.63278/jicrcr.vi.3037.
[23] Aitha, A. R. (2021). Dev Ops Driven Digital Transformation: Accelerating Innovation In The Insurance Industry. Journal of International Crisis and Risk Communication Research, 327–338. https://doi.org/10.63278/jicrcr.vi.3341
[24] Abadi, M.,& Goyal, S. (2021). Signal processing architectures for scalable financial data analysis. IEEE Transactions on Signal Processing, 69, 4221–4234.
[25] Li, F., & Zhou, Y. (2021). A study of latency optimization in distributed DSP frameworks for fintech systems. Future Generation Computer Systems, 124, 381–391.
[26] Das, P., & Sharma, R. (2021). DSP-enabled fraud analytics for cross-border payment systems. International Journal of Information Management Data Insights, 1(2), 100041.