УДК 004.052:004.75

SEMANTIC OBSERVABILITY IN STREAM PROCESSING: RUNTIME CORRECTNESS VERIFICATION FOR ONLINE MARKET MICROSTRUCTURE FEATURE ENGINEERING

Egor Kolychev[1] (master student)
Scientific supervisor – **PhD, Associate Professor Ivan Perl**[1]
[1]ITMO University
507367@niuitmo.ru

Abstract. Stream processing systems compute financial microstructure metrics in real time, yet existing observability monitors only infrastructure indicators and cannot detect semantic correctness violations. This work proposes a three-level observability model and demonstrates that formal definitions of metrics such as Order Flow Imbalance inherently define runtime-checkable invariants detecting data corruption invisible to infrastructure monitoring. A taxonomy of streaming operators for financial metrics is introduced, classifying them by state requirements and order sensitivity. Fault injection experiments show that standard monitoring misses realistic data quality failures while semantic invariants reliably detect them with O(1) per-event overhead.
Keywords: stream processing, observability, semantic correctness, runtime verification, market microstructure, Order Flow Imbalance

Stream processing systems are increasingly used to compute market microstructure metrics in real time: Order Flow Imbalance (OFI), Volume-Synchronized Probability of Informed Trading (VPIN), Trade Flow Imbalance, and others [1]. These metrics require maintaining state between events, strict processing order, and window semantics, making them sensitive to failures typical of distributed systems: event duplication, reordering, and data loss. Stanford, Kallas, and Alur demonstrated that advances in stream processing performance have not been matched by adequate attention to computation correctness: an error in a streaming application immediately produces wrong results for downstream consumers [2]. Existing observability tools – OpenTelemetry, Prometheus, consumer lag monitoring – track infrastructure metrics (latency, throughput, error rates) but cannot detect a situation where all system indicators are normal while computed values are incorrect. Batch validation tools (Great Expectations, dbt tests) operate post-hoc on materialized data. Academic Stream Runtime Verification systems (RTLola, MonPoly) provide formal guarantees over event streams but have not been applied to domain-specific financial streaming computations [3]. Thus, a gap exists between three fields – stream processing, financial mathematics, and runtime verification – each solving an adjacent but non-overlapping problem.

This work proposes a three-level observability model for streaming computations. Level 1 (infrastructure) covers standard monitoring: latency, throughput, consumer lag, implemented via OpenTelemetry and Prometheus. Level 2 (computation) covers state store size, checkpoint duration, and watermark progress – partially available in Apache Flink and Kafka Streams as built-in metrics. Level 3 (semantic) introduces domain-specific correctness invariants derived from the mathematical definition of the computed metric – absent from existing systems and constituting the main contribution of this work. The key insight is that a formal metric specification defines not only the computation algorithm but also a set of runtime-checkable invariants. For OFI [1], defined as the sum of contributions from consecutive changes in best bid/ask prices in the order book, the definition yields invariants: (a) in the absence of quote update events, the OFI increment equals zero; (b) the spread (ask minus bid) is non-negative at all times; (c) the total buy and sell volume matches the total matched trade volume. Violation of any invariant indicates data corruption: event duplication, reordering, or message

loss. The PGVal study confirms that even exactly-once semantics does not guarantee semantic correctness: Flink silently drops late events in join operations, and Kafka Streams discards all late events in multi-stream topologies [4].

To systematize semantic monitoring requirements, a taxonomy of microstructure metrics by streaming operator type is proposed: (1) windowed commutative aggregates, order-insensitive within a window (Trade Flow Imbalance, Realized Volatility); (2) order-dependent state machines where each event's contribution depends on the previous state (OFI); (3) join operators with temporal shift requiring delayed reference lookups (Spread Decomposition); (4) online estimators based on rolling regression (Kyle's Lambda); (5) path-dependent operators with volume-time windowing where bucket boundaries depend on cumulative volume (VPIN). Each class exhibits a specific set of semantic invariants and different sensitivity to ordering violations and data loss. The taxonomy enables determining the minimum set of runtime checks for each computation type.

Experimental verification is conducted on an OFI streaming pipeline. Under controlled fault injection scenarios – duplicate quotes from a feed provider, event reordering due to network disruption, reprocessing after Kafka partition recovery, and frozen data feeds – infrastructure metrics remain within normal bounds while semantic invariants detect correctness violations. Invariant checking executes in O(1) per event, ensuring minimal overhead for high-throughput systems.

The proposed semantic observability model detects correctness violations invisible to standard infrastructure monitoring. The approach is technology-agnostic: invariants are derived from the metric's mathematical definition and applicable to architectures of varying scale – from lightweight pipelines to distributed frameworks. The results can be applied in designing online financial market analytics systems requiring real-time correctness guarantees. Future work includes developing adaptive invariants calibrated to market regime from historical data, and formalizing automatic invariant derivation from arbitrary streaming metric specifications.

Literature

1. Cont R., Kukanov A., Stoikov S. The Price Impact of Order Book Events // Journal of Financial Econometrics. – 2014. – Vol. 12, No. 1. – P. 47–88.
2. Stanford K., Kallas K., Alur R. Correctness in Stream Processing: Challenges and Opportunities // Proceedings of CIDR. – 2022.
3. Bartocci E., Falcone Y. Lectures on Runtime Verification: Introductory and Advanced Topics // Lecture Notes in Computer Science. – Springer, 2018. – Vol. 10457.
4. Tahir J., Mayer R., Doblander C., Jacobsen H.-A. How Reliable Are Streams? End-to-End Processing-Guarantee Validation // Proceedings of the VLDB Endowment. – 2025. – Vol. 18, No. 3. – P. 585–598.