TPCTC 2011

3rd International Conference on Performance Evaluation and Benchmarking

Measuring Performance of Complex Event Processing Systems

Torsten Grabs, Ming Lu
Microsoft StreamInsight
Microsoft Corp., Redmond, WA
{torsteng, milu}@microsoft.com

Agenda

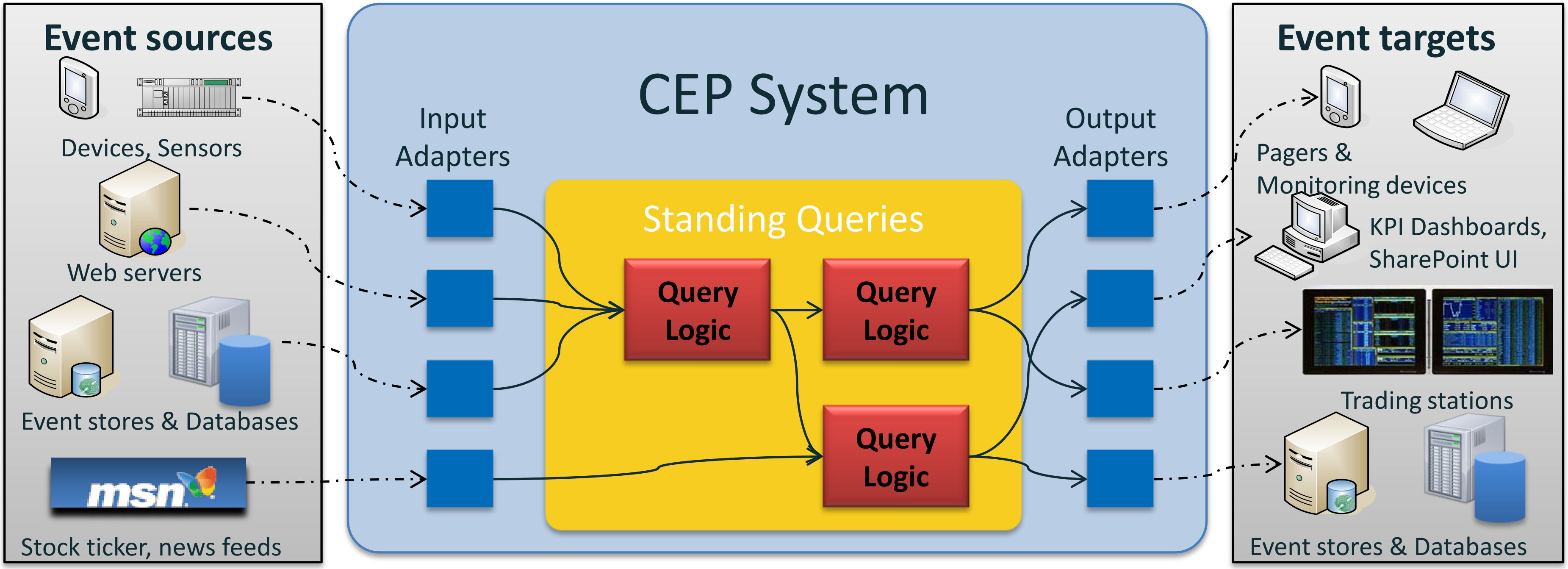
- Motivation
- CEP systems and performance
- System vs. information latency
- Measuring information latency
  - Event history
  - Prefixes over event histories
  - Event horizon
  - Result prefixes over the event horizon
  - Information latency metric
- CEP benchmarking with information latency
- Conclusions and future work

Motivation

- Complex Event Processing (CEP) used for data stream processing in many industries
  - Utilities (power, smart grids)
  - Finance (algorithmic trading, risk management)
  - Oil & gas (monitoring assets such as oil rigs)
  - IT monitoring (data center health)
- Main CEP benefits
  - Continuous and incremental processing
  - Low latency: near real-time results
  - High data rates: process fire hose of constantly arriving data
- Good performance is crucial across industry scenarios listed above
- Goal of this paper
  - Suggest metrics to assess and compare performance of CEP systems
  - Focus on latency, particularly information latency

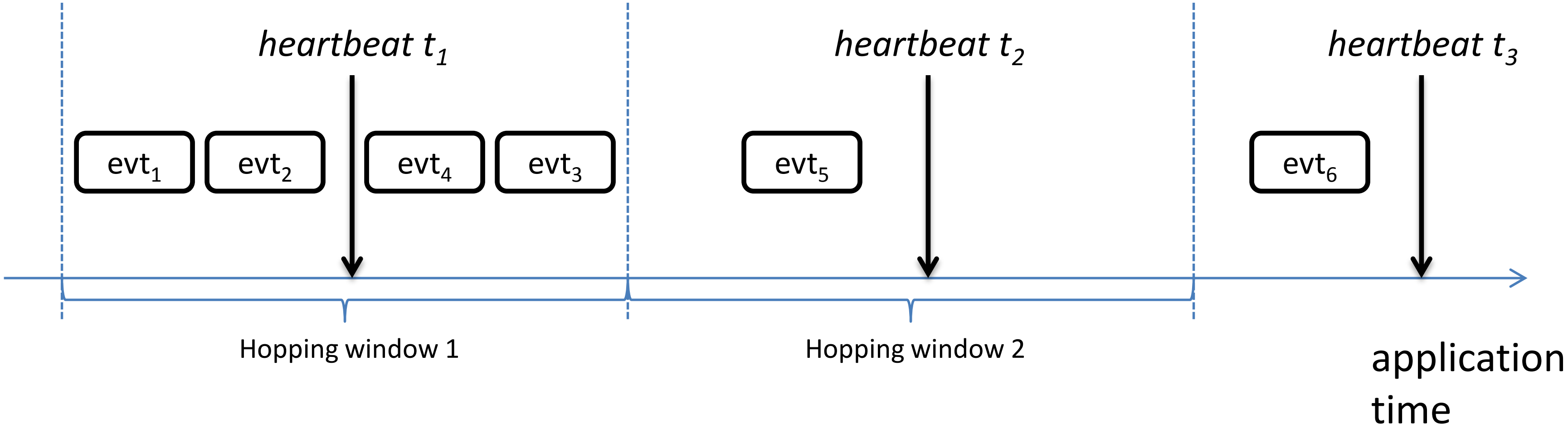
CEP Systems and Performance

- *Data rate*: How many input events per second can the system process (while keeping steady state)? [BEAST 1998, SPECjms2007]
- *Latency*: How long does it take until the effects of an input event appear in the output? [STAC]
- *Resource consumption*: How many resources (CPU, memory, IO etc.) does the CEP system consume while doing its processing?



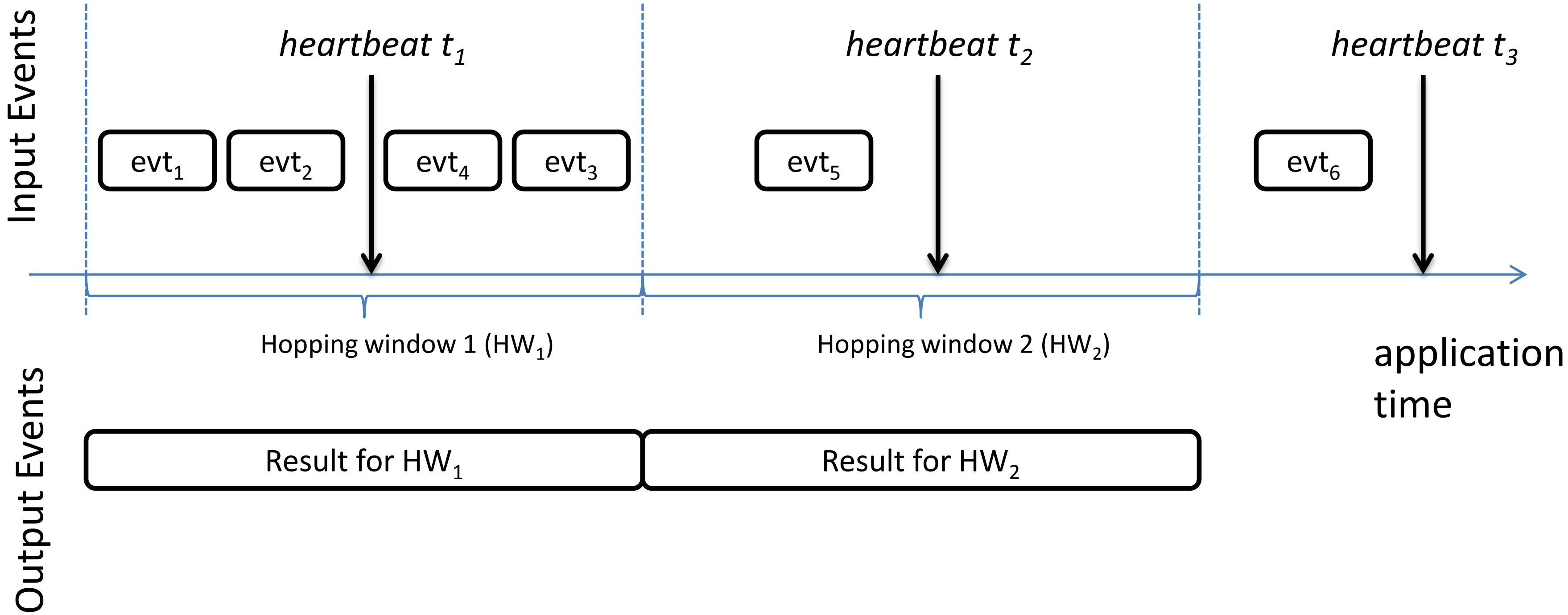
System vs. Information Latency

- System latency:
  - Delays of results because of the time spent processing (on the CPU)
  - Easy to measure through event lineage [Chandramouli et al. 2011]
- Information latency:
  - Delays introduced by waiting
  - No agreed upon definition up to now
- Example:
  - Query: aggregate over hopping window
  - Data: events with timestamps and payloads as below



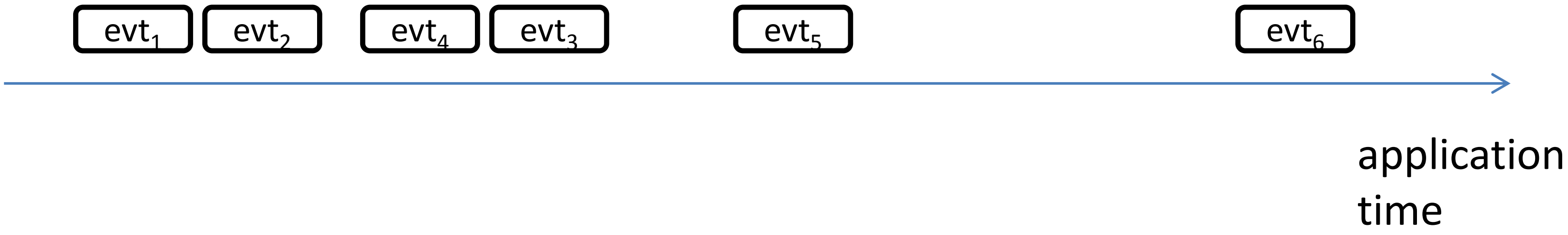
Example: Information Latency

- Events arrive over time (system time) and carry timestamps (application time)
- Heartbeats communicate that all events up to application time *t* have been received
- The big question: are results being produced as early as possible or could the system do better?



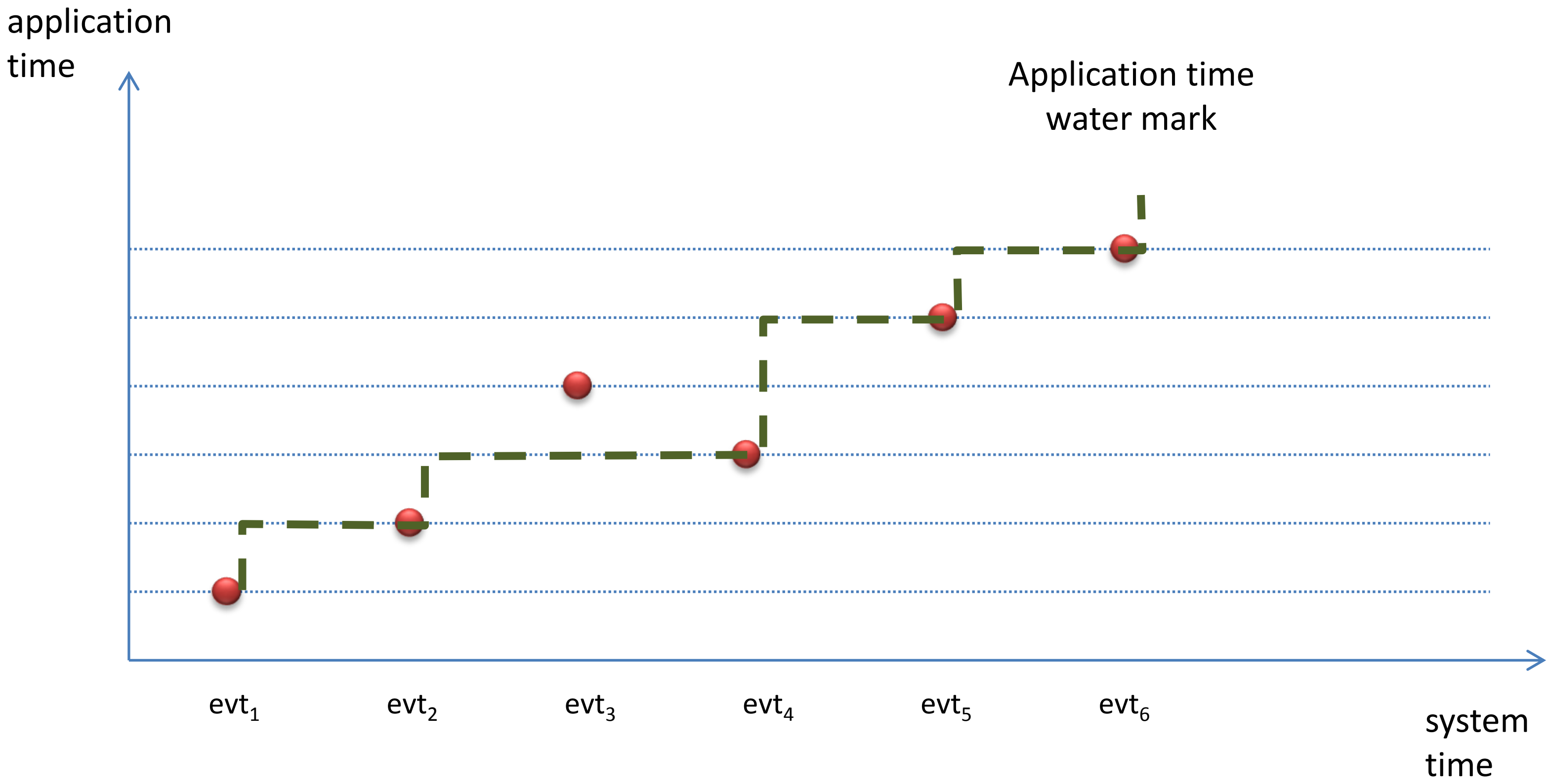
Event History

- How can we define and measure information latency?
- *Event history*: capture how events arrive over time
  - System time and application time
  - Input and output events: input vs. output histories
- *Prefix of event history*: subset of the events, beginning "from the left", i.e., with increasing time, up to a certain point
- Example of an event history:



Event Horizon

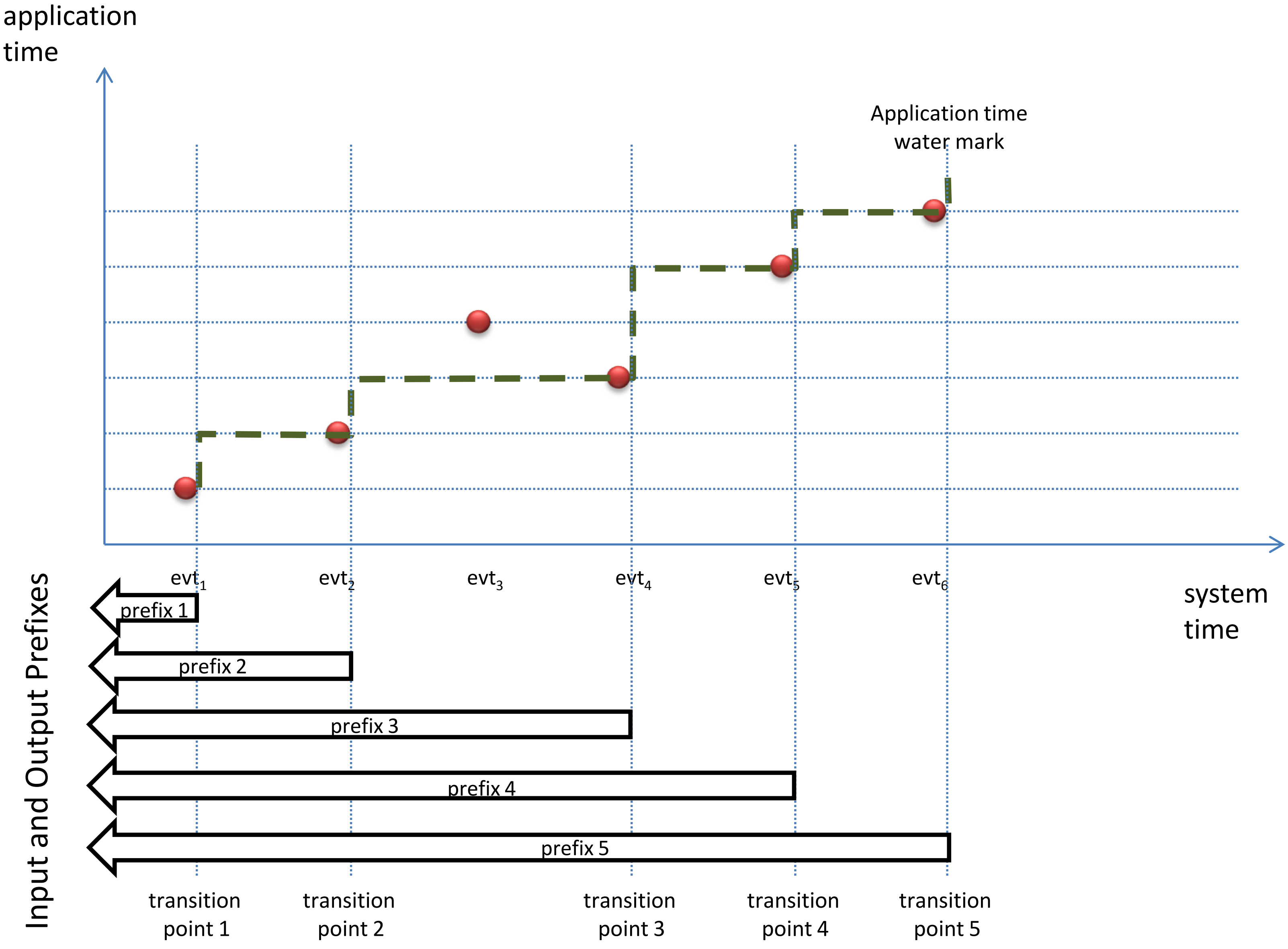
- *Event horizon*: Correlate system time and application for event histories
- *Water mark line*: progress of application time over system time as per the timestamps of the events in the history
- *Transition point*: application time where water mark moves up at



Using Prefixes with Event Horizons

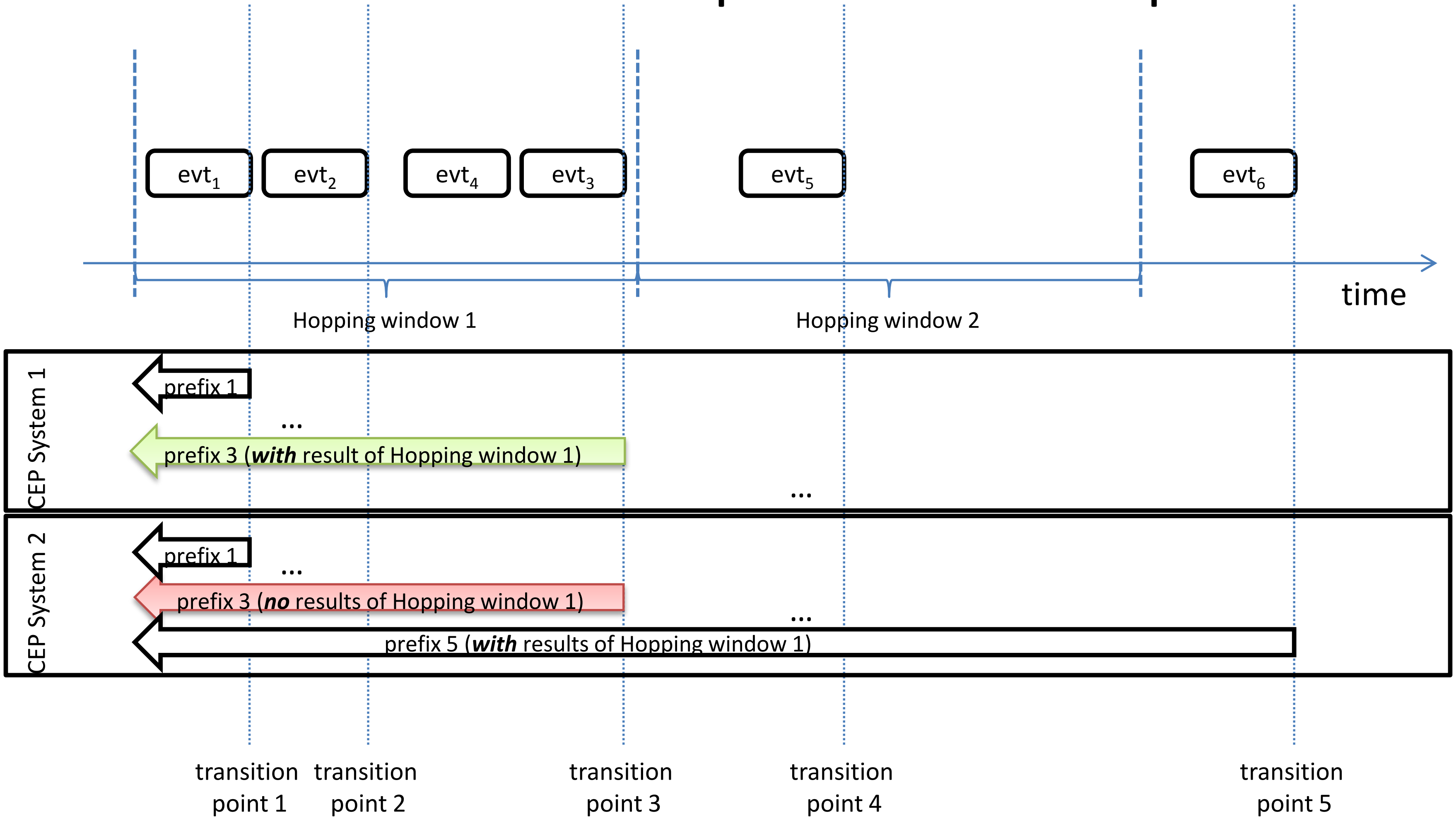
- Goal: Capture how much progress the results (i.e. the output) have made against the input
- Focus on the transition points in the event horizon
- Move iteratively through transition points from “left to right” (increasing system time) while aggressively issuing heartbeats as a transition point is reached
  - This defines a set of input history prefixes
  - This also defines a set of corresponding output history prefixes
- Observation:
  - A larger output prefix for a given input prefix is desirable.
  - It provides more information sooner than the system with the smaller output for the same prefix.

Example: Prefixes with Event Horizons



Comparing Result Prefixes

- Exploit the observation that larger result prefixes are better than smaller prefixes for comparisons



Measuring Information Latency

- Introducing “the oracle”
  - Has complete knowledge of all current and future input events
  - Can therefore produce the full result set at system start time
  - Serves as idealized reference point to compare with
- Relate actual CEP systems against the oracle
  - What portion of the result does the CEP system produce as compared to the oracle? Capture this as a ratio.
  - Doing this for all transition points in the history describes how well the actual system makes progress towards the full result
- This can easily implemented by iteratively replaying input prefixes up to transition point *i* for all transition points 1 through *j* where
  - k_i is the size of the output prefix at transition point *i*
  - *n* is the size of the full result set

$$\text{Information Latency} \coloneqq 1 - \prod_{i=1}^{j} \frac{k_i + 1}{n + 1} = 1 - \frac{1}{(n+1)^j} \cdot \prod_{i=1}^{j} (k_i + 1)$$

Benchmarking with Information Latency

- Plug information latency as a factor into performance metrics for CEP benchmarks
- Complements other parameters and metrics
  - Data rate
  - Event size
  - System latency
- Example: Worst case performance

$$Perf := \frac{DataRate_{max} \cdot EventSize_{max}}{SystemLatency_{max} \cdot InfoLatency}$$

- Balance performance metrics with pricing metric

Conclusions & Future Work

- CEP performance measurement so far focused on data rate and system latency
- Information latency needs to be included into CEP performance measurement
  - Unnecessary latency causes customer frustration
  - Information latency issue can be hard to detect
- Discussed a framework for information latency measurement and comparison in this paper
- We look forward to work with data streams researchers and other CEP system vendors to develop comprehensive CEP benchmarks

? Questions ?