Multiple Pattern Queries with Common Sub-Patterns

**Motivation**
- Observations
  - The cardinality of the sub-pattern matches varies over time
  - The crests and troughs often may not align well
- Intra-Query Event Correlation
  - Number of event instances of type $E_j$ follow an event of type $E_i$
  - Estimate the number of sub-pattern matches formed in a time interval
- Inter-Query Event Correlation
  - A ratio between
    - The number of sub-pattern matches computed with sharing
    - The number of sub-pattern matches computed independently
  - Estimate the degree of sharing possible across multiple patterns
- Redundancy Score
  - Estimate the degree of the redundant computation of sub-pattern
  - matches within a time interval using both Intra- and Inter-Query Event Correlations.

**SPASS Optimizer**
- Challenge 1. Non-Alignment of Cardinality of Pattern Matches

**SPASS Architecture**
- SPASS Optimizer builds an optimal sharing plan for entire pattern workload
- SPASS Runtime exploits iterative hierarchical processing to compute pattern matches

**SPASS Runtime**
- Key challenge - maintain result matches for sub-patterns
- Solution
  - Shared continuous sliding views store intermediate results of sub-patterns
  - Partial sub-pattern matches stored in sequence views
  - Subsequent reuse by accessing these materialized views associated with sub-patterns
- Concurrent reuse of shared continuous sliding views
  - View Validity Interval (VVI) – timestamp-based indicators associated with materialized views
  - View Lookup Interval (VLI) – a time interval to look up pattern matches

**Experimental Results**
- Window size and number of patterns increase, SPASS achieves more performance gains.
- On average, SPASS exhibits 17 times faster average execution time compared to the unshared approach.
- W1 is characterized by 4 sets of 5 patterns sharing common prefixes across the queries
- W2 consists of 4 sets of 5 patterns with common suffixes.
- W3 has queries with mixed common sub-patterns.

**Conclusion**
- SPASS Optimizer leverages event correlations to find an effective sharing plan.
- SPASS Runtime then execute this shared pattern plan by exploiting the shared continuous sliding view technology.
- SPASS achieves many folds performance improvement in CPU utilization compared to state-of-the-art techniques.