**Sequence Mining**

- **Goal**: Discover subsequences as patterns in sequence data
- **Input**: Collection of sequences of *items*, e.g.,
  - Text collection (sequence of words)
  - Customer transactions (sequence of products)
- **Output**: Subsequences that
  - occur in $\sigma > 0$ input sequences (frequency threshold)
  - have length at most $\lambda > 0$ (length threshold)
  - have gap $\gamma \geq 0$ (contiguous or non-contiguous subsequences)

**Example:**
- $S_1$: Anna lives in Melbourne
- $S_2$: Bob lives in the city of Berlin
- $S_3$: Charlie lives London

**Hierarchies**

*Items* can be naturally arranged in a hierarchy:

- **Syntactic hierarchy**
  - DET
  - a
  - all
  - the
  - PERSON
  - Scientist
  - Politician
  - Melbourne
  - CITY
  - DSLR Camera
  - Canon5D
  - Nikon5100
  - Tripod
  - Photography
  - Product hierarchy

**Sequence Mining with Hierarchies**

- **Item hierarchies** are specifically taken into account
- **Items** in output sequences may belong to different levels in the hierarchy

**Example:**
- $S_1$: Anna lives in Melbourne
- $S_2$: Bob lives in the city of Berlin
- $S_3$: Charlie lives London

**LASH**

- **Distributed framework** for generalized sequence mining
- **Build** over MapReduce for large-scale data processing
- **MAP (partitioning)**
  - Data is divided into potentially overlapping partitions
- **REDUCE (mining)**
  - Partitions are mined independently

**Key features**

- Scales to very large datasets
- Novel hierarchy-aware form of item-based partitioning
- Optimized partition construction
- Customized local mining
- No global post processing

**Partitioning**

- **Key idea**: partition the output space
- **Items** are ordered by decreasing frequency e.g., PERSON < CITY < in < lives < ...
- Create a partition for each frequent item called pivot item
- Rewrite each input sequence for each partition
  - Fast rewrites (low overhead)
  - Makes partitions as small as possible
  - Reduces communication and skew

**Mining**

- **Traditional approach**
  - Mine using any GSM alg.
  - Filter non-pivot sequences
  - Inefficient
- **LASH's PSM approach**
  - Only mine pivot sequences
  - Requires no filtering

**Examples**

- **The New York Times Corpus**, syntactic hierarchy

**Experiments**

**The New York Times Corpus**, syntactic hierarchy

- **Multiple orders of magnitude faster**
- **PSM more than 3x faster** than traditional sequence miners
- **Good strong and weak scalability**