Querying Massive Trajectories by Path on the Cloud

Ruiyuan Li[1,2], Sijie Ruan[1,2], Jie Bao[1], Yanhua Li[3], Yingcai Wu[4], Yu Zheng[1,2,5]

[1]*Urban Computing Group, Microsoft Research, China*;
[2]*Xidian University, China*; [3]*Worcester Polytechnic Institute, USA*;
[4]*Zhejiang University, China*; [5]*Chinese Academy of Sciences, China*

Background

- **Massive trajectories are generated continuously**

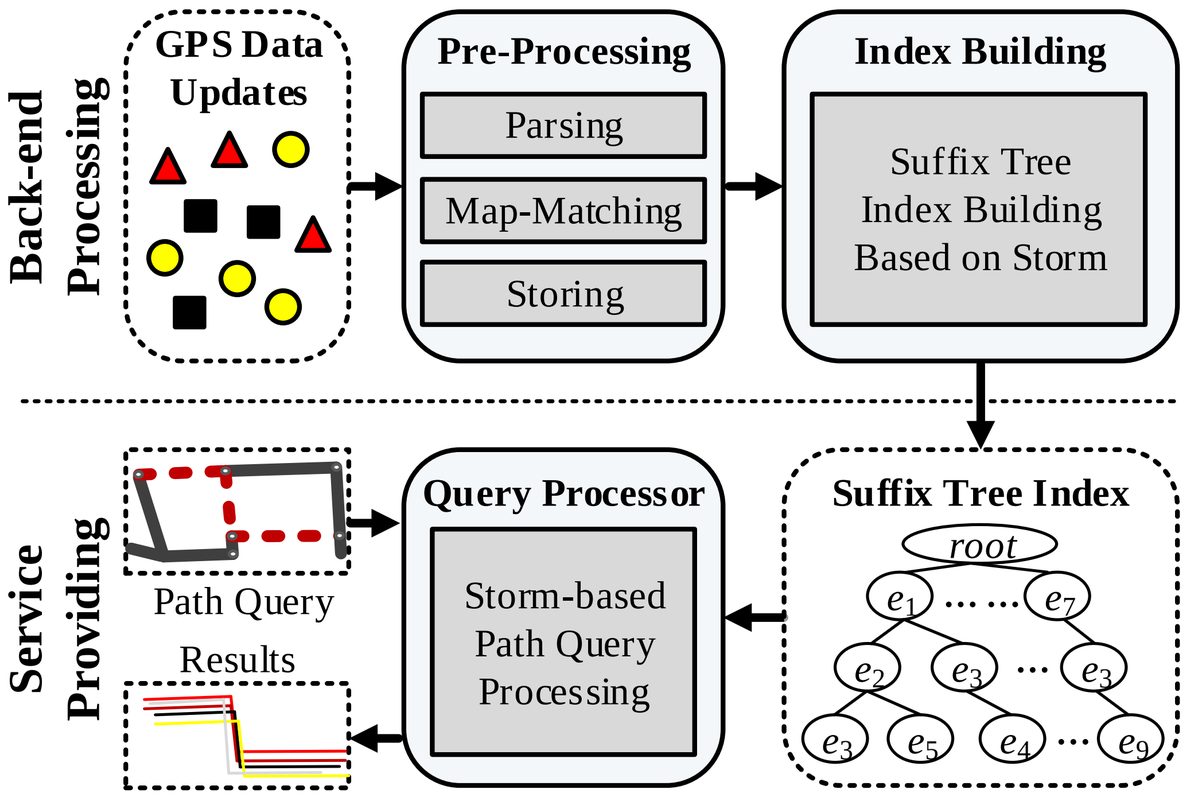
- **Path Query Definition**
  - **Given**
    - Path $\boldsymbol{P}$ = {e_i, e_j,…,e_k}
    - Time interval [t_s, t_e]
    - Trajectories $\boldsymbol{T}$ = {tr_1, tr_2, ⋯, tr_n}
  - **Find**
    - All of sub-trajectories of tr_i in $\boldsymbol{T}$ that passed path $\boldsymbol{P}$ within time interval [t_s, t_e]
- **Application Scenarios**

Traffic Modeling

Anomaly Detection

Path Recommendation

System Overview



Pre-processing

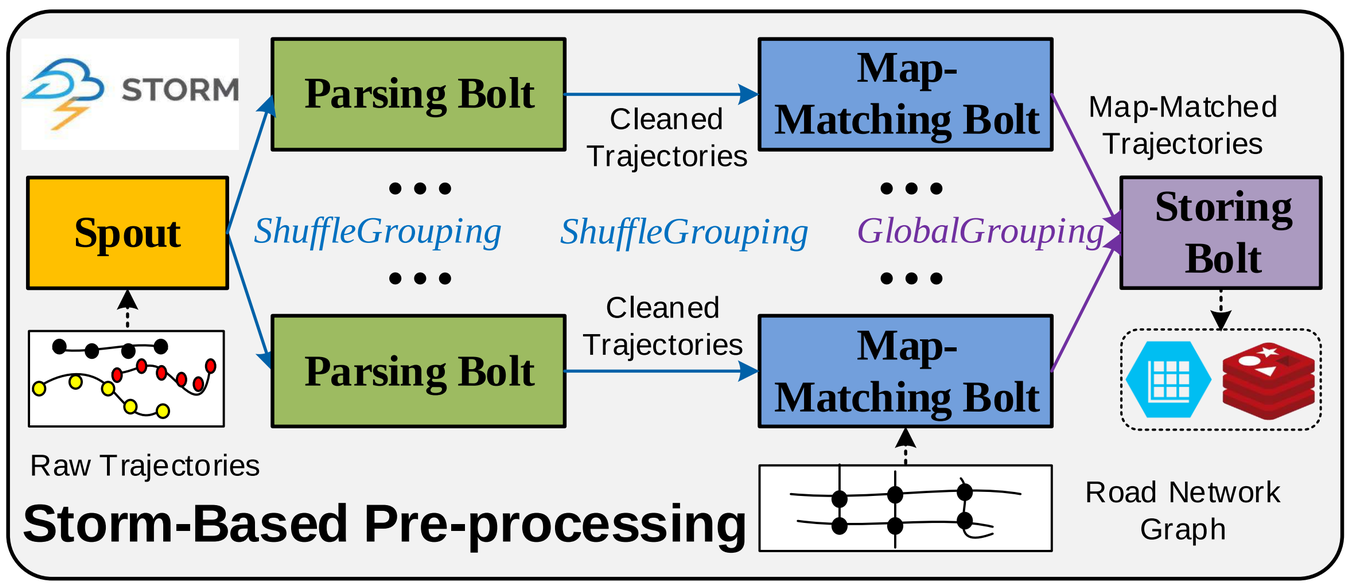
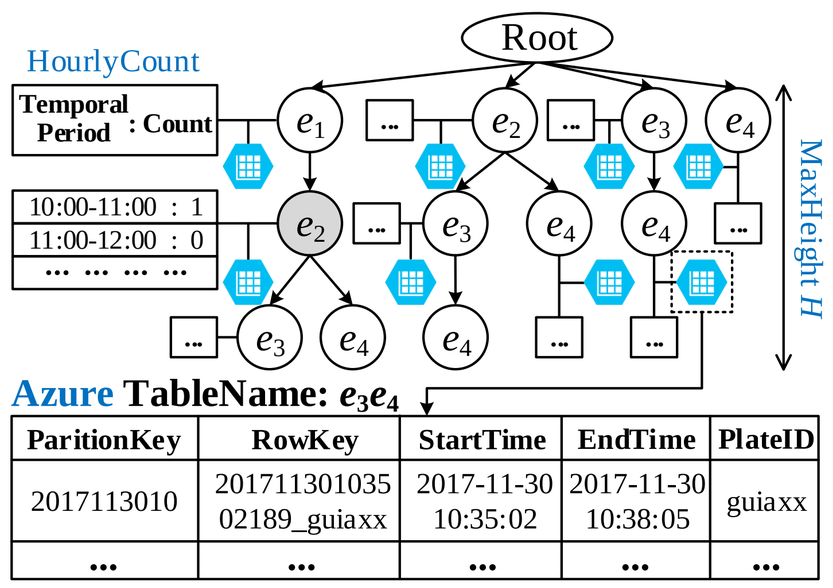
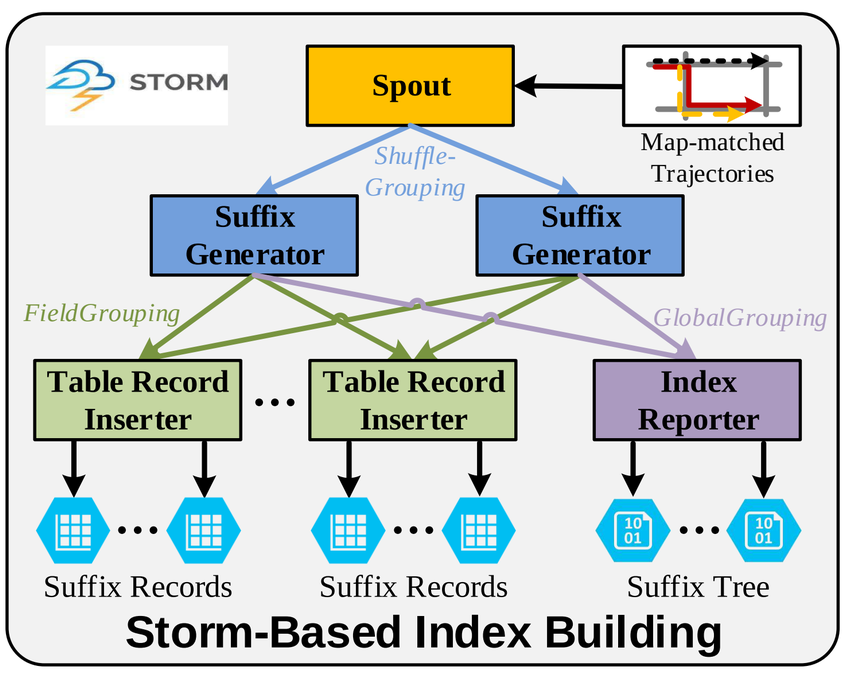


Table-Based Suffix Tree

Thousands of tables need to be accessed. I/O bottleneck!

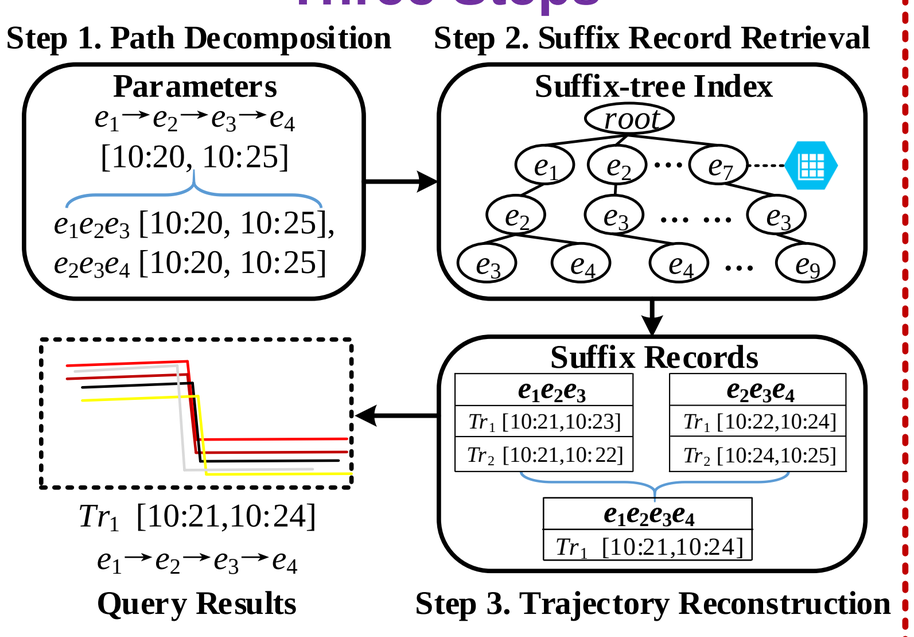


Index Building



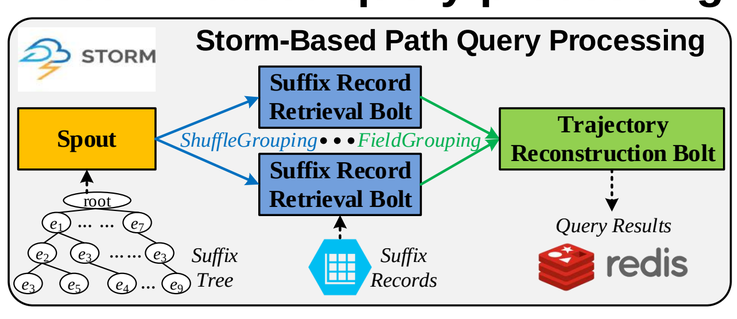
Query Processing

Three Steps

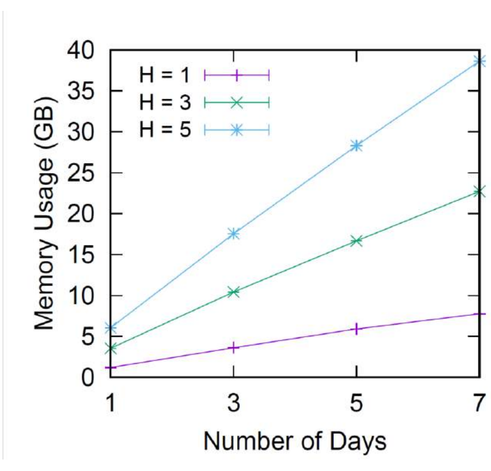


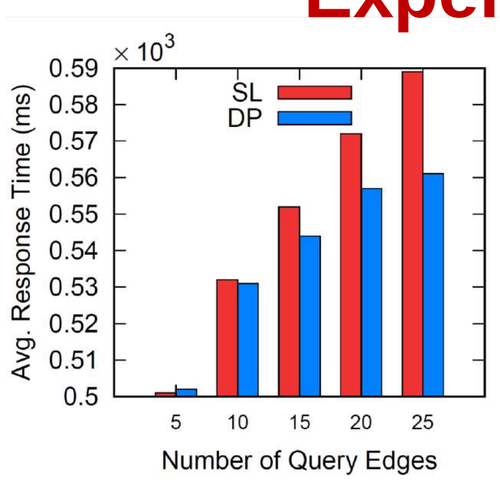
Two Techniques

- **Path decomposition methods**
  - Sliding Window method
    - Window size of *H*, step size of *H* - 1
    - Minimize sub-path number
  - Dynamic Programming method
    - Minimize the max count of each sub-path
    - Bucket effect: the slowest one is a decisive factor
- **Storm-based query processing**

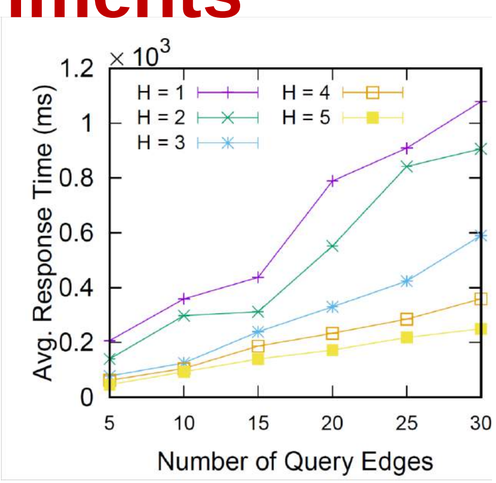


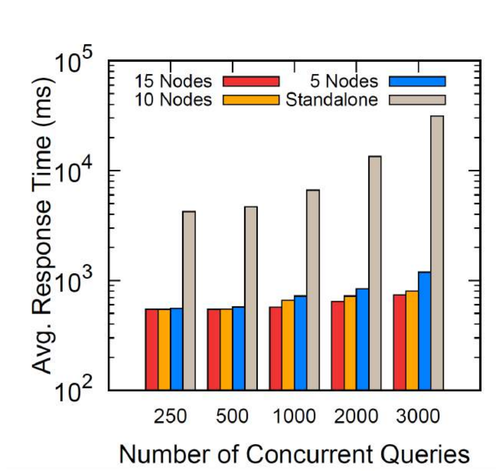
Experiments





H is a tradeoff between indexing and query.





Query performance improves over 60%.

Demo

Urban Traffic