Searchlight: Context-Aware Predictive Continuous Querying of Moving Objects in Symbolic Space

Kenneth Fuglsang Christensen, Lasse Linnerup Christiansen, Torben Bach Pedersen, Jeppe Pihl
Center for Data-intensive Systems (Daisy), Aalborg University, Denmark

**Motivation**

- Moving objects in mixed indoor/outdoor semantic/symbolic spaces
- Requirements:
  - Uniform model
  - Context awareness
  - Continuous queries
  - Predictive/historical
- Applications:
  - Campus services
  - Flight baggage
  - Emergency services
  - ...

**Continuous Query Processing**

- Push model
- 2-phase incremental evaluation
  - Patch generation
  - Periodical patch application
- Interval Patches
  - Window-based queries
  - Triple \((a_{\alpha}, b_{\beta}, b_{\eta})\)
- Instance Patches
  - Time instance-based queries
  - Tuple \((a_{\alpha}, l_{\eta})\)

**Searchlight Graph**

- Graph-based symbolic model
  - Nodes: locations
  - Edges: uni/bidirectional movement
  - Edge cost: travel time
- Contextual object keywords
  - “Danish”, “female”
- Movement-affecting object keywords
  - “student”, “bike”
- Contextual location keywords
  - “canteen”, “auditorium”
- Context-aware and adaptive edge costs
  - Keyword and time-dependent

**Searchlight Queries**

**Range query**: “report every 15 minutes which security guards have visited the CS Dept. at AAU during the past 4 hours”
ADD SecurityServiceMonitor AS EVERY 15 min
SELECT FROM (CS) BETWEEN -4 hrs AND 0 hrs
WHERE OBJECT KEYWORDS IN (security-guard)

**Aggregate query**: “how many people are predicted to be in the Biology Department at AAU during the next 60 minutes”
ADD PeopleCount AS EVERY 5 min
SELECT COUNT FROM (Biology) BETWEEN 0 min AND 60 min

**Position query**: “monitor the current location of each disabled person inside the CS Dept. at AAU every 5 seconds”
ADD LocateDisabledPersons AS EVERY 5 sec
SELECT POSITION OF (person) IN 0 min
WHERE OBJECT KEYWORDS IN (disabled) AND LOCATION KEYWORDS IN (CS)

**Prediction**

- **Context-aware Predictor**
  - Based on historic movement
- **Route archive**: stores previous routes based on keywords+start location
- **Predict()**: candidate route lists based on movement-affecting keywords
- **BestRoute()**: best route based on weighted score given candidate route lists
- **Weighted score**
  - Keywords
  - Route length
  - Use count
  - Temporal distance

**Experiments**

- Throughput of 200-250K readings/sec over 4.7K locations with up to 100 concurrent queries and fast response time on std. PC
- Efficient incremental evaluation
- Good multi-core scale-up
- Effective prediction
- 1-2 orders of magnitude faster than closest competitor

**Acknowledgements**

Supported by the BagTrack project, funded by The Danish National Advanced Technology Foundation under grant no. 010-2011-1