EFFICIENT SEARCH OF PATH-CONSTRAINED MOVING OBJECTS

Thuy Thi Thu Le and Bradford G. Nickerson

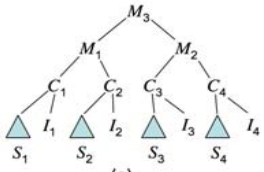
Faculty of Computer science, University of New Brunswick, Fredericton, New Brunswick, Canada

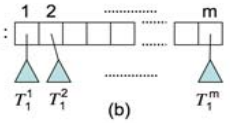
Motivation

- Queries on historical positions of moving objects on planar graphs are likely to be used in applications such as reconstruction, planning event, and training.
- How to efficiently store the position history of moving objects?
- How to improve the response time for query processing of moving objects?
- Define: Graph G with N moving objects on E edges.

Data Structure

- **Graph strip tree (GStree)**

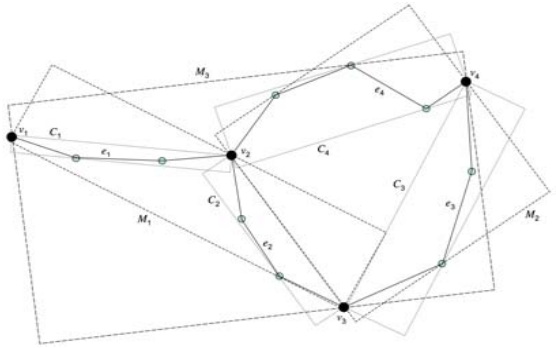




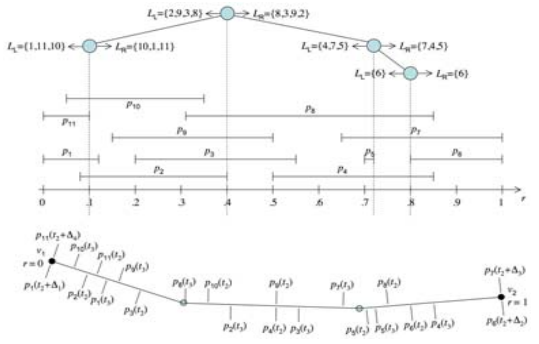
a) Each leaf C_i points to the strip tree S_i representing the graph edge e_i, and to the corresponding moving object interval sets I_i

b) Interval sets I_i are an array of size m. Each element of I_i points to an external memory interval tree T^i_j representing the moving objects during interval j on edge e_i

- Requires linear space in N and E.
- Example road network of New Brunswick consisting of a planar graph with 66,437 edges and 54,827 vertices (ratio of 1.21).

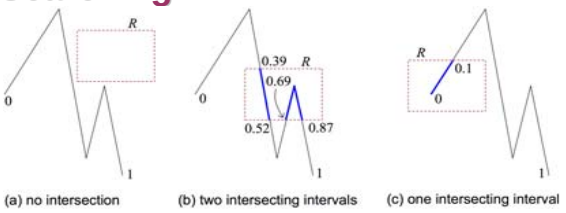


An example graph G with 4 edges e_1, ..., e_4 and 4 vertices v_1, ..., v_4. The edges are represented as strip trees, with C_1, ...,C_4 representing the root bounding boxes for each strip tree. The strip trees are merged bottom up in pairs to construct the GStree.



An example interval tree T^3_1 for e_1. There are 11 intervals representing 11 moving objects during the time interval $[t_2, t_3]$. Moving objects p_1, ..., p_6 are moving from vertex v_1 to vertex v_2. Moving objects p_7, ...p_{11} are moving in the opposite direction from v_2 to v_1.

Searching



- Query $Q_2 = (R, [t_1, t_2])$, where R is a rectangular query and $[t_1, t_2]$ is a time interval. Query Q_1 is Q_2 with $t_1 = t_2$.
- Start from the root node to the leaf nodes of the GStree. The rectangular query R is used first to find edges intersecting R.
- For each intersected edge, the time interval query $[t_1, t_2]$ is used in interval trees to find moving objects satisfying the query.
- Q_1 query requires $O(\log_B N/E + k)$ disk I/Os, where k is the number of disk blocks required to store the answer; B is the number of objects transmitted by one disk I/O.

Experimental Results (using main memory)

We ran queries on 3,288,689 entries of moving objects during 5 time steps. Tables show average search times (in seconds) and numbers of visited nodes on the GStree, the MON-tree, and the naïve search. h is the number of range searches (out of 400) that were averaged to obtain these results. D is the average number of visited nodes.

Q_1

Range	h	GStree		MON-tree		naïve
		time	D	time	D	time
1	179	0.00043	0.03	0.00009(0.22)	1.28(38.17)	0.123(284.96)
2	3	0.00233	7.00	0.00500(2.14)	100.67(14.38)	0.125(53.57)
3	32	0.00355	167.52	0.01187(3.35)	832.44(4.97)	0.123(34.79)
4	180	0.01941	1,659.21	0.25942(13.37)	10,595.50(6.39)	0.132(6.78)
5	6	0.07749	7,276.33	1.87738(24.23)	45,371.00(6.24)	0.160(2.07)

Q_2

Range	h	GStree		MON-tree		naïve
		time	D	time	D	time
1	179	0.00047	0.03	0.00012(0.26)	1.28(38.17)	0.122(257.61)
2	2	0.00300	9.50	0.00150(0.50)	81.00(8.53)	0.124(41.50)
3	23	0.00309	195.00	0.00804(2.61)	646.39(3.31)	0.123(40.00)
4	173	0.01722	2,287.62	0.15290(8.88)	9,249.58(4.04)	0.133(7.73)
5	23	0.05921	10,039.35	0.74206(12.53)	36,595.87(3.65)	0.165(2.79)

Open

- Search time complexity of Q_2 query?
- Experimental performance with I/O efficient interval tree?
- Use cache oblivious algorithms to improve search on massive GStree?

Acknowledgements

NSERC CRSNG

Harrison McCain Foundation