Module 8: Miscellaneous Topics

Lecture 38: Incremental Nearest Neighbor

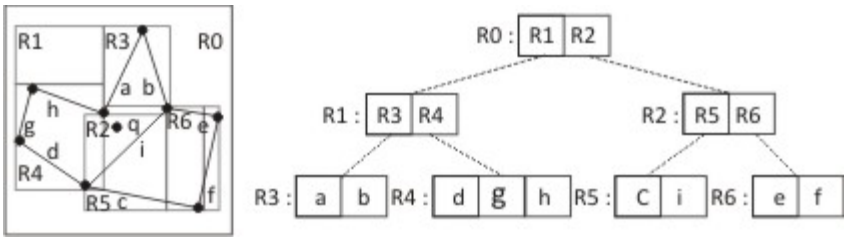
Prev topic | Next topic | Prev page | Next page

The Lecture Contains:

Incremental nearest neighbor

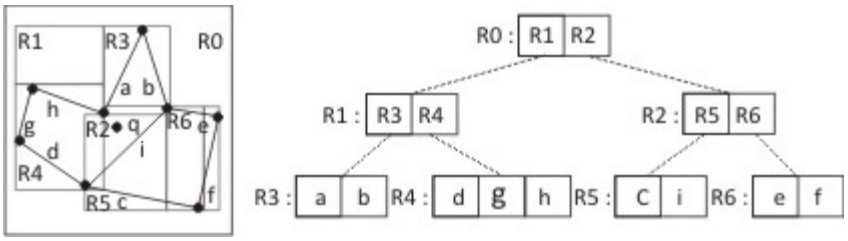
- Producing sorted lists required for FA and TA can be done efficiently by incrementally returning nearest neighbors
- Incremental NN algorithm follows best-first strategy
- Assume a tree-based index structure
- Maintain a min heap of nodes and leaves ordered according to their *minimum* distances to query
- Initialize it with the root node
- Whenever a call for the *next* nearest neighbor is made
    - Look at top element of heap
    - If it is a leaf, return the object and stop
    - If it is an internal node, delete it, and insert its children

Example



Extract	Insert	Heap	Return
		$(R_0, 0)$	-
R_0	R_1, R_2	$(R_1, 0), (R_2, 0)$	-
R_1	R_3, R_4	$(R_2, 0), (R_4, 11), (R_3, 13)$	-
R_2	R_5, R_6	$(R_5, 0), (R_4, 11), (R_3, 13), (R_6, 44)$	-
R_5	$[c], [i]$	$([i], 0), (R_4, 11), (R_3, 13), (R_6, 44), ([c], 53)$	-
$[i]$	i	$(R_4, 11), (R_3, 13), (i, 21), (R_6, 44), ([c], 53)$	-

Example



Extract	Insert	Heap	Return
$[i]$	i	$(R_4, 11), (R_3, 13), (i, 21), (R_6, 44), ([c], 53)$	-
R_4	$[d], [g], [h]$	$(R_3, 13), ([h], 17), (i, 21), ([d], 30), (R_6, 44), ([c], 53), ([g], 74)$	-
R_3	$[a], [b]$	$([a], 13), ([h], 17), (i, 21), ([b], 27), ([d], 30), (R_6, 44), ([c], 53), ([g], 74)$	-
$[a]$	-	$([h], 17), (i, 21), ([b], 27), ([d], 30), (R_6, 44), ([c], 53), ([g], 74)$	a (1NN)
$[h]$	-	$(i, 21), ([b], 27), ([d], 30), (R_6, 44), ([c], 53), ([g], 74)$	h (2NN)
i	-	$([b], 27), ([d], 30), (R_6, 44), ([c], 53), ([g], 74)$	i (3NN)