GC
10/23/08

Approx Nearest Neighbor

Arya-Mount Journal Paper

Last Time Cell-Decomp

Prob: Not clear how to get efficient point location

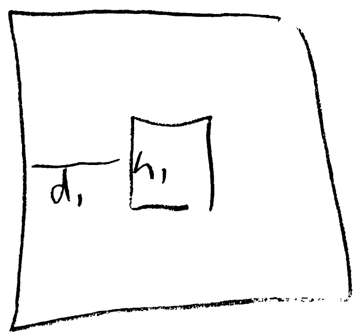
Recall: Box ≡ Rect 1) $\frac{ls}{ss} \leq 3$ 2) aligned

Cell ≡ 1) Box

2) Diff of 2 boxes (sticky)

Sticky: b_I sticky in b_O if

$d = 0$ or $d_i \geq h_i$



Split (b, S)

1) even

2) $b_1 \cap S \neq \emptyset$ & $b_2 \cap S \neq \emptyset$

else "not possible"

Shrink (b, S)

$$\underset{b' \subseteq b}{\text{Minimal}} \{ S \subseteq b' \wedge b' \text{ is a sticky box} \}$$

Note Shrink in $O(1)$ time

Prob for split: We need to split S into S_1 & S_2.

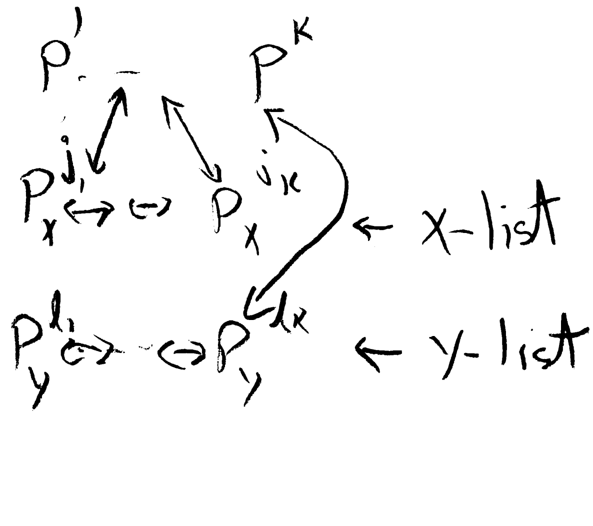
Idea! $S = \{p^1, \ldots, p^k\}$ $\quad p^i = (p^i_x, p^i_y)$

Maintain link lists L_x, L_y

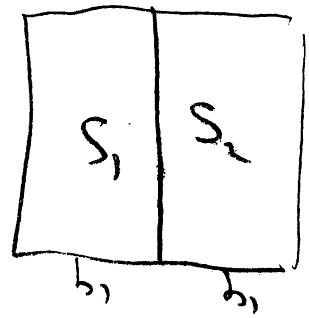
$L_x \equiv p^1_x \ldots p^k_x$

cross linked

$L_y \equiv p^1_y \ldots p^k_y$



Suppose we split vertically:



lets work on the x list first.

We need only break the list i.e.

find t st $P_x^{j_t} \in b_1$ & $P_x^{j_{t+1}} \in b_2$

Solution: work from both ends of list.

$\text{Cost} = \min\{|S_1|, |S_2|\}$

What about y-list?

Assume $|S_1| \leq |S_2|$

We splice out S_1 from the y-list

$\text{Cost} = O|S_1|$

In last week's alg we now need to sort y-list of S_1

If y-list also included rank, numbers $\in \{1, \ldots, k\}$

Then sorting would be $O(k)$ (bucket sort)

Prob: $|S_1| << k$.

New alg Idea: work on S_2 until $|S_2| \leq \frac{2}{3}k$.

Alg <u>centroid-shrink</u> (b, S, k) $|S|=k$ (case $b \equiv$ box)

While $|S| \geq \frac{2}{3}k$ do

$b = \text{Shrink}(b, S)$

$(b_1, S_1), (b_2, S_2) = \text{split}(b, S)$ $|S_1| \leq |S_2|$

centroid-shrink (b_2, S_2)

Case: b is a cell b_0 & b_I

Shrink $(b_0, b_I, S) \equiv$ shrink but include b_I.

Split $(b_0, b_I, S) \equiv$ if (b_1, S_1) & (b_2, S_2)

if $b_I \subseteq b_2$ & $|S_1| \leq |S_2|$ continue

else $((b_1, b_I), S_1)$ & centroid-shrink (b_2, S, k)

Cost $= T(K)$

Claim: $T(K) = O(K)$

Cost of shrink-splits $= \sum c_0 K_i$ where $\sum K_i \leq K$

At the end we do 2 spliceouts at a cost of $O(K)$

Balanced-Box Decomp (BBD)

$T(n) = O(n \lg n)$ $\quad T(n) \leq 2T(n/2) + c \cdot n$

Each cell is assigned a bded size list of points

$C = (b_0, b_I)$ gets a point from b_I

$T \equiv BBT$ query point q

$r \equiv root(T)$

$Q \equiv$ priority queue $(dist(box, q))$

Search(q, T)

enqueue(r)