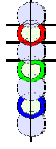


# Computational Geometry

## Well Separated Pair Decomposition and Its Applications



# References

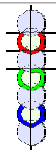
A decomposition of multidimensional point sets with applications to  $k$ -nearest-neighbors and  $n$ -body potential fields

Paul Callahan and Rao Kosaraju

Journal of the ACM, Vol 42, No 1, January 1995, pp 67–90.

Available on the net:

<http://www.w.acm.org/pubs/citations/journals/jacm/1995-42-1/p67-callahan/>



# Definitions

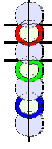
$A, B$  : two sets of points in  $d$  dimensions  
 $S$  :  $A$  parameter

$R(A)$  : Smallest axis parallel rectangle containing all the points of  $A$   
 $b(A, r)$ : Ball centered at the center of  $R(A)$  having radius  $r$

$l_i(A) = l_i(R(A))$  : The length of the  $i$ -th dimension of  $R(A)$

$l_{\min}(A) = l_{\min}(R(A)) = \min_{i=1}^d l_i(R(A))$  : The size of the smallest dimension

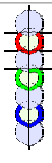
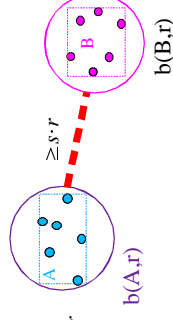
$l_{\max}(A) = l_{\max}(R(A)) = \max_{i=1}^d l_i(R(A))$  : The size of the largest dimension



# Well Separated

$A$  and  $B$  are **well separated** if there exists an  $r$ , so that:

- $A \subseteq b(A, r)$
- $B \subseteq b(B, r)$
- $dist(b(A, r), b(B, r)) \geq s \cdot r$

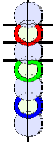


## Interaction Product of A and B

$$A \otimes B = \{(p, p') \mid p \in A, p' \in B, \text{ and } p \neq p'\}$$

$\{(A_i, B_j), \dots, (A_k, B_m)\}$  realization of the interaction product  $A \otimes B$  If:

- i.  $A_i \subseteq A$  and  $B_j \subseteq B$  for all  $i=1, \dots, k$ .
- ii.  $A_i \cap B_j = \emptyset$  for all  $i=1, \dots, k$ .
- iii.  $(A_i \otimes B_j) \cap (A_m \otimes B_m) = \emptyset$  for all  $i, j$  such that  $1 \leq i < m \leq k$ .
- iv.  $A \otimes B = \cup_{i=1}^k A_i \otimes B_j$ .
- v. Realization is **well Separated** if  $A_i$  and  $B_j$  *well separated* for  $i=1, \dots, k$ .



## Well Separated Pair Decomposition

**Input:** A set of points P, a parameter  $s$ .

**Output:** A WSPD of the points.

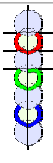
**Idea:** Represent P using a binary tree. A node represent a set made out of all the elements stored in the leafs of the subtree.

**Decomposition:** A set of pairs of nodes of the tree.

**Observation:** We can use quadtree, but...

**We have no performance guarantee**

**Idea:** Modify the quadtree so it have linear size.



## Fair Split Tree

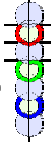
P: set of points

If  $|P| = 1$  then a single node is created: P.

$\widehat{R}(P)$  : Outer rectangle – for the root of the FS tree: A cube centered at the center of  $R(A)$  of size  $l_{max}(P)$  (the longest edge of the bounding rectangle  $R(A)$ )

We recursively partition P by a axis parallel hyperplane into two non-empty subsets.

$\widehat{R}(A)$  : Outer rectangle of a (non) root node A – the rectangle containing A out of the two rectangles formed by the split of p(A) (the parent of A) of  $\widehat{R}(p(A))$

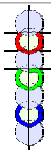


## Fair Split

A – node in the tree (we refer to a node in the tree by the set of points it represents)

**Fair Split:** cut of  $\widehat{R}(A)$  by a hyperplane in distance at least  $l_{max}(P)/3$  from its two parallel faces.

**Splitting rule:** Cut hyperplane in the middle of the longest edges of  $R(A)$  (**regular** bounding rectangle).

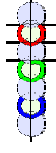
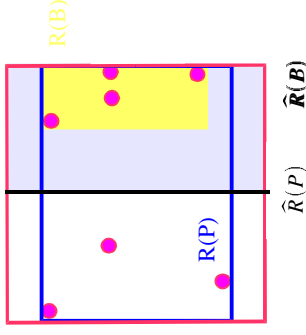


## Just the place for a figure...

... and a bit of text

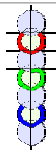
- Split always separate at least one points from therest of the points – tree have  $n-1$  internal nodes.
- Height of tree can be linear.
- Fast splitting using  $d$  binary trees.
- Each stage takes

$$O(n_{small} \log n_{small})$$



## Fair Split Trees

- Can be computed in  $O(n \log^2 n)$  time.
- Running time can be improved to  $O(n \log(n))$
- Can be used to compute WSPD of the points, by observing that:
  - Each pair of points, must have a node where they are in the same set, but in different children of this node.
  - Thus, compute for all the nodes in the tree the WSPD of the left (child) set, with the right (child) set



## WSPD Algorithm

Procedure ComputeWSPD( $P, P'$  : nodes in the tree)

Begin

If  $P$  and  $P'$  are well separated then  
return  $\{P, P'\}$

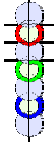
Swap  $P, P'$  if necessary so that  $l_{max}(P) \geq l_{max}(P')$

Since  $|P| > 1$  let  $P = P_1 \cup P_2$

ComputeWSPD( $P_1, P'$ )

ComputeWSPD( $P_2, P'$ )

end.

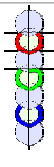


## Lemma 4.1

- **C – d–cube**
- $S = \{A_1, \dots, A_l\}$  so that  $A_i \cap A_j = \emptyset$  and  $l_{max}(p(A_i)) \geq l(C)/c$
- $R(A_i) \cap C \neq \emptyset$
- $K(c, d) = \max_{S, P} |S|$

Then  $K(c, d) \leq (3c+2)^d$

Intuitively, not too many disjoint "big" nodes can intersect a cube.



## Lemma 4.1 – Proof

For any node  $A$  in  $T$ :  $I_{\min}(\widehat{R}(A)) \geq \frac{I_{\max}(P(A))}{3}$   
 proof by induction on the point in the tree where  $I_{\min}(\widehat{R}(A))$  is created

Namely, since  $\widehat{R}(A_i) \cap \widehat{R}(A_j) = \emptyset$ , then each

$\widehat{R}(A_i)$  contains a cube of size  $\frac{I_{\max}(P(A))}{3} \geq \frac{I(C)}{3c}$  that intersects  $C$ .

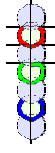
All those cubes are disjoint, then by volume consideration, we have

$$K(c,d) \leq (3c+2)^d$$



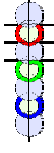
## Observations

- ◆  $\{P, P'\}$  a pair considered by alg'
- ◆  $I_{\max}(P) \geq I_{\max}(P')$
- ◆ Well separated if distance between bounding balls  $\geq s(\sqrt{d}/2)I_{\max}(P)$
- ◆  $d(P, P')$  – min distance between  $R(P)$  and  $R(P')$
- ◆ Stronger condition:  $d(P, P') \geq (s(\sqrt{d}/2) + \sqrt{d})I_{\max}(P)$
- ◆ By induction:  $I_{\max}(P(P')) \geq I_{\max}(P)$
- ◆  $S_P$  – set of all  $P'$  considered with  $P$  ( $P$  is about to split)



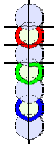
## Observations II

- ◆  $C$  – a cube of size  $(s\sqrt{d}+2\sqrt{d+1})I_{\max}(P)$  centered around  $R(P)$
- ◆ Since  $P' \in S_P$  – pair  $\{P, P'\}$  not well separated  $\rightarrow R(P')$  intersects  $C$
- ◆ All the elements of  $S_P$  are disjoint
- ◆ Reminder:  $I_{\max}(P(P')) \geq I_{\max}(P)$  and by lemma 4.1...
- ◆  $|S_P| \leq K(s\sqrt{d}+2\sqrt{d+1}, d) \leq (3(s\sqrt{d}+2\sqrt{d+1})+2)^d = O(1)$
- ◆ Overall number of pairs considered (and thus generated) is linear.



## Result

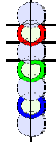
- ◆ One can compute a fair split tree in  $O(n \log(n))$  time.
- ◆ One can compute in  $O(n)$  time a linear number of pairs that form a well separated pairs decomposition of the given point set. (note – the representation of the sets of points in the pairs of decomposition is implicit).



## Finding k-nearest neighbor

- ◆ **Problem:** Given a set  $P$  of  $n$  points compute for each point its  $k$  nearest neighbor.
- ◆ Relation is not symmetric.
- ◆ **Lemma 8.1:**  $\{A, B\}$  in WSPD of  $P \otimes P$  so that  $(a, b)$  is a pair so that  $a \in A, b \in B$  and  $b$  is a  $k$ -nearest neighbor to  $a$ . Then  $|A| \leq k$ .

Surprise: We can find closest pair in  $O(n \log(n))$  time using WSPD!



## Lemma 8.2

$B$  – point set,  $O_B$  the center of the smallest  $d$ -ball that contains  $R(B)$ , assume radius = 1

$a, a'$  – two points so that the singeltons  $\{a\}, \{a'\}$  are WS from  $B$  and  $d(a, O_B) \geq d(a', O_B)$

$\alpha$  – angle between  $aO_B$  and  $a'O_B$

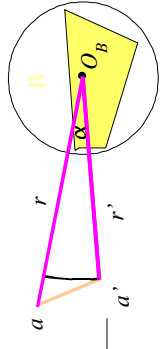
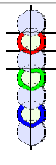
If  $\alpha < s/(s+1)$  then

$d(a, a') < d(a, b)$  for all  $b$  in  $B$ .

$d(a, a') < r - r' + r' * \alpha = r - (1 - \alpha)r'$

Substitute  $r' = s + 1, \alpha = s/(s+1)$  then  $d(a, a') < r - 1$

Finally,  $d(a, b) \geq r - 1$  for all  $b \in B$



## Lemma 8.3

$C$  – cone with diametrical angle  $\vartheta$  with apex in  $O_B$

$A \subseteq C$  – a set of points each one WS from  $B$

$P$  – a set containing  $A \cup B$

If  $\vartheta < s/(s+1)$  and  $X = \{a' \in A \setminus \{a\} \mid d(O_B, a') \leq d(O_B, a)\}, |X| \geq k$

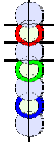
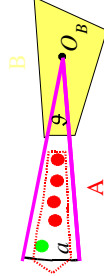
then no  $b \in B$  can be a nearest neighbor of  $a$  in  $P$ .

Intuitively: there are enough points in  $A$  close to  $B$ , to block  $B$  from participating in the nearest neighbors of  $a$ .

For any  $a' \in X$ , we have by Lemma 8.2

$d(a, a') \leq d(a, b)$  for any  $b \in B$

$X$  has  $k$  nearer points to  $a$  than any point of  $B$ .



## K Nearest Neighbor

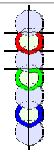
We have WSPD of  $P$  with  $s > 2$

$f(B)$  – a set of all  $a$  so that there exists  $A, B'$  in the WSPD so that  $a \in A, |A| \leq k, (A, B')$  in the WSPD, and  $B'$  is an ancestor of  $B$ .

$N(B) \subseteq f(B)$  – includes all  $a$ , so that there is a  $k$  nearest neighbor of  $a$  which is in  $B$ .

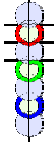
We compute  $N(B)$  recursively  $N(B)$  so that its size is  $O(k)$

Then, we can compute for each point its  $k$  nearest neighbor (using median)

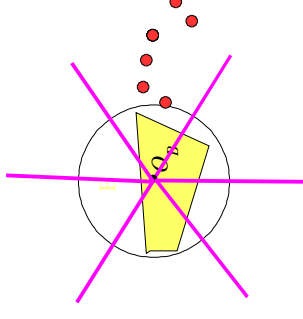


## Algorithm

- Initialize  $N$  to be a union of  $N(p(B))$  with all associated  $A$ , so that  $(A, B)$  is in the WSPD and  $A$  contains at most  $k$  elements.
- Partition the points of  $N$  into cones, and compute the  $k$  nearest points in each cone to the center of  $B$ .
- By Lemma 8.3 –  $N$  contains all feasible candidates, and its size is  $O(k)$  (number of cones is constant.)
- Set  $N(B)$  to  $N$ . Overall size of those sets is  $O(nk)$ .
- If  $(a, b)$  is a  $k$ -nearest neighbor then a msut be in  $N(\{b\})$ .
- $NN(x)$  – all  $b$  so that  $x$  is in  $N(\{b\})$ .
- Compute for  $x$  its  $k$  nearest neighbor out of  $NN(x)$



## Figure



## Result

One can compute in  $O(nk)$  the  $k$  nearest neighbor of each point, out of a set of  $n$  points in  $O(nk + n \log(n))$  time in any dimension, by a simple algorithm.

