Pseudo-code

float cloest\_cross\_pair(point\* middle,float dmin,int size){

    float fd=dmin;

    for(int i=0;i<size;i++){

        int j=i+1;

        while((middle[j].y-middle[i].y)<=dmin and j<size){

            float d=get\_distance(middle[i],middle[j]);

            if(d<=fd){

                update fd and add points into pair\_arr

            }

            j++;

        }

    }

    return fd;

}

float closet\_pair(point \*p,int size){

    if(size<=3){

        float d1,d2,d3,dmin1;

        d1=get\_distance(p[0],p[1]);

        if(size==2){

            pair\_arr[pair\_size].a=p[0];

            pair\_arr[pair\_size].b=p[1];

            pair\_size++;

            return d1;

        }

        else{

            d1=get\_distance(p[0],p[1]);

            d2=get\_distance(p[1],p[2]);

            d3=get\_distance(p[0],p[2]);

            dmin1=min(d1,d2);

            dmin1=min(dmin1,d3);

            if(dmin1==d1){

                store the first and second point

            }

            if(dmin1==d2){

                store the second and third point

            }

            if(dmin1==d3){

                    store the first and third point

            }

            return dmin1;

        }

    }

    else

    {

        int medin=size/2;

        int n1=medin,n2=size-medin;

        point \*left=new point[n1];

        point \*right=new point[n2];

        use for loop to copy all number into new array

        float dL=closet\_pair(left,n1);

        float dR=closet\_pair(right,n2);

        float dmin2=min(dL,dR);

        float low=p[medin].x-dmin2;

        float high=p[medin].x+dmin2;

        int index=0,size\_middle=0;

        point \*middle=new point[200000];

        while(index<size){

            find all points in middle band

        }

        dmin2=cloest\_cross\_pair(middle,dmin2,size\_middle);

        return dmin2;

    }

}

int main(int argc,char \*\*argv){

read file

merge sort in x direction

merge sort in y direction

float dim=closet\_pair()

output information into file

}

Analysis

1. We merge sort in x direction and y direction in main function. It would take 2\*NlogN.
2. In closet\_pair function, we copy all the array, so it would take N. Then, we find points in middle band, It would take n, and n is smaller than N. Then we call the function cloest\_crorss\_pair, it would take C\*n times.
3. Overall, we have N+n+C\*n, according to dominate rules. We only take N.
4. So it would be T(N)=T(N/2)+O(N), as a result, I would take O(NlogN)