

# Divide-and-Conquer Algorithms

Note Title

9/3/2007

1. Divide - divide problem into subproblems (smaller input for same problem)
2. Conquer - recursively solve subproblems (or directly solve when input reaches a termination condition)
3. Combine - use subproblem sols to solve given problem

Mergesort  $a[0], \dots, a[n-1]$

1. Divide - split array in half

input (problem) sort  $a[p] \dots a[r]$

Subproblem  $q = \lfloor (p+r)/2 \rfloor$

sort  $a[p] \dots a[q]$  +  $a[q+1] \dots a[r]$

stop  
 $n=1$  ↓

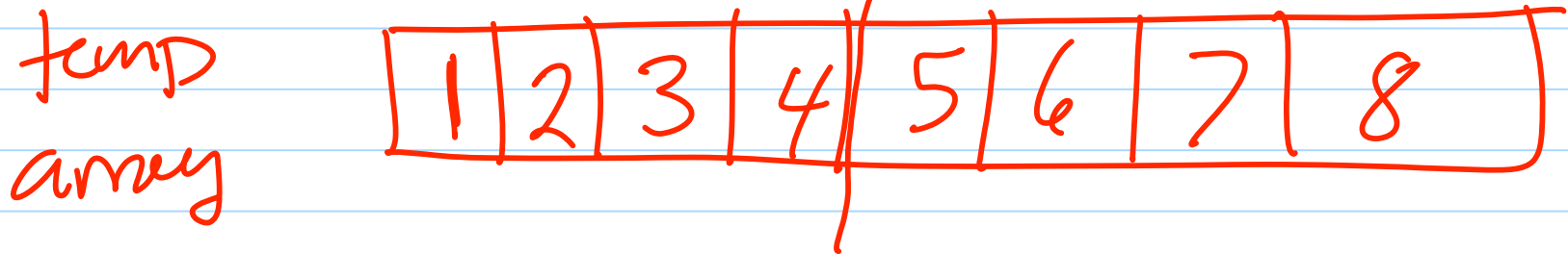
2. Recursively sort two subarrays

3. Merge two sorted subarrays

## Example Execution

6 4 3 8 | 1 7 2 5

3 4 6 8 | 1 2 5 7  
~~3~~ ~~4~~ ↑ ~~1~~ ~~2~~ ↑



This takes linear time (in size of resulting subarray)

# Closest pair of points

Divide split into left half and a right half of  $\sim n/2$  points each

Recursively find closest pair on each of these two subproblems

Combine - Use subproblem sols to find closest pair in given point ~~set~~ set

yStrip



dist of  
closest pair  
found so  
far

$P_1$

$P_2$

$d$

consider each point in yStrip  
looking upward

```
For (i=0; i < numInStrip - 1; i++) {  
  j = i + 1;
```

```
  while (j < numInStrip &&  
         yStrip(j).y - yStrip(i).y < d) {
```

```
    d = min(d, distance point  
            i & point j)
```

```
    j++
```

```
  }
```

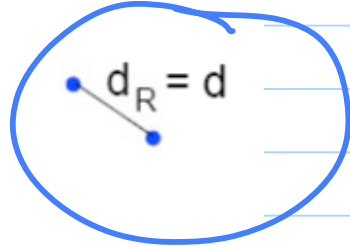
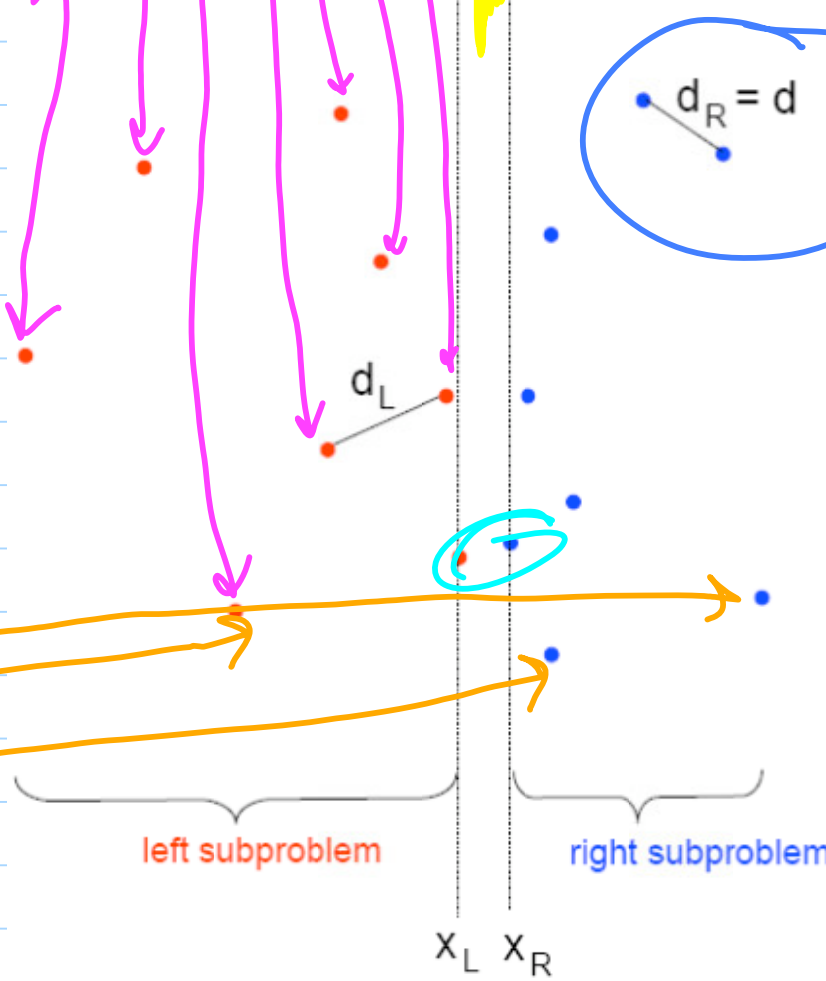
```
}
```



# Example Execution

pts By X

pts By Y

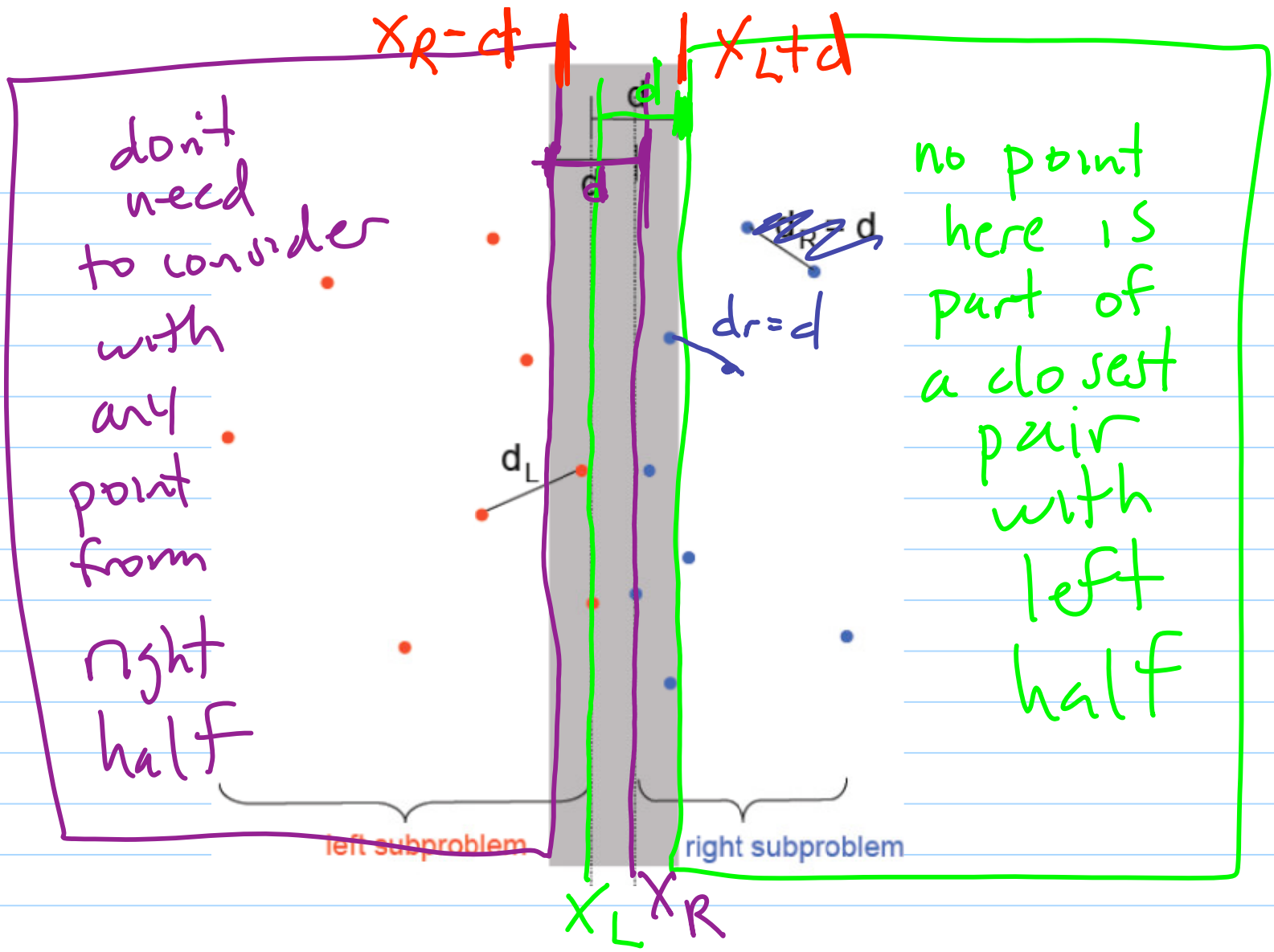


$$d = \min(d_L, d_R)$$

left subproblem

right subproblem

$x_L$   $x_R$



don't need to consider with any point from right half

no point here is part of a closest pair with left half

$x_R - d$  |  $x_L + d$

left subproblem | right subproblem

$x_L$   $x_R$

## Correctness

Terminate if  $(n=1)$  return  $\infty$   
if  $(n=2)$  return dist  
of 2 pts

1. If closest pair of points are both on left we find in recursive call for left. Same for right.

2. a) We've argued that any pair of points not in  $\gamma$  strip need only be considered by 1.

b) We've argued that any pair in  $\gamma$  strip that could have  $\text{dist} < d$  is considered.

Correctness (cont.)

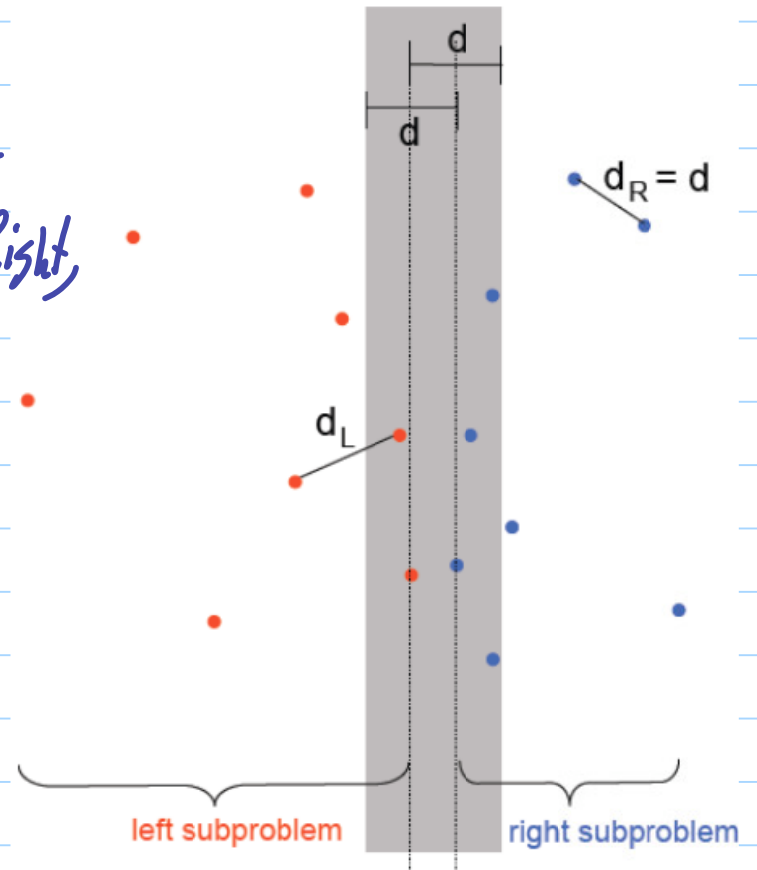
$T(n)$  = time complexity for  $n$  points

## Analysis of Asymptotic Time Complexity

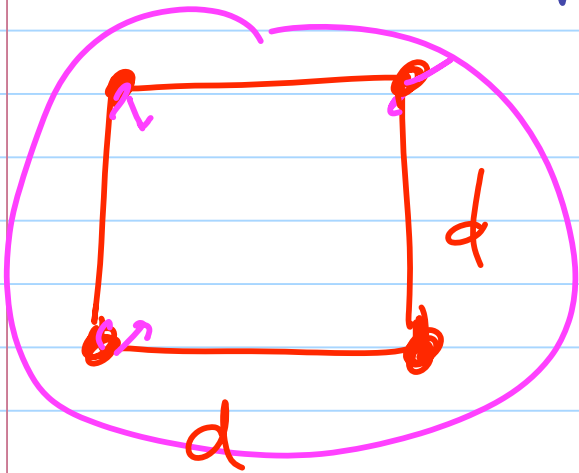
Divide can be done  
in linear time. You must  
create pts By X Left, pts By X Right,  
pts By Y Left, pts By Y Right  
*careful*

Recursive calls  
 $2T(n/2)$

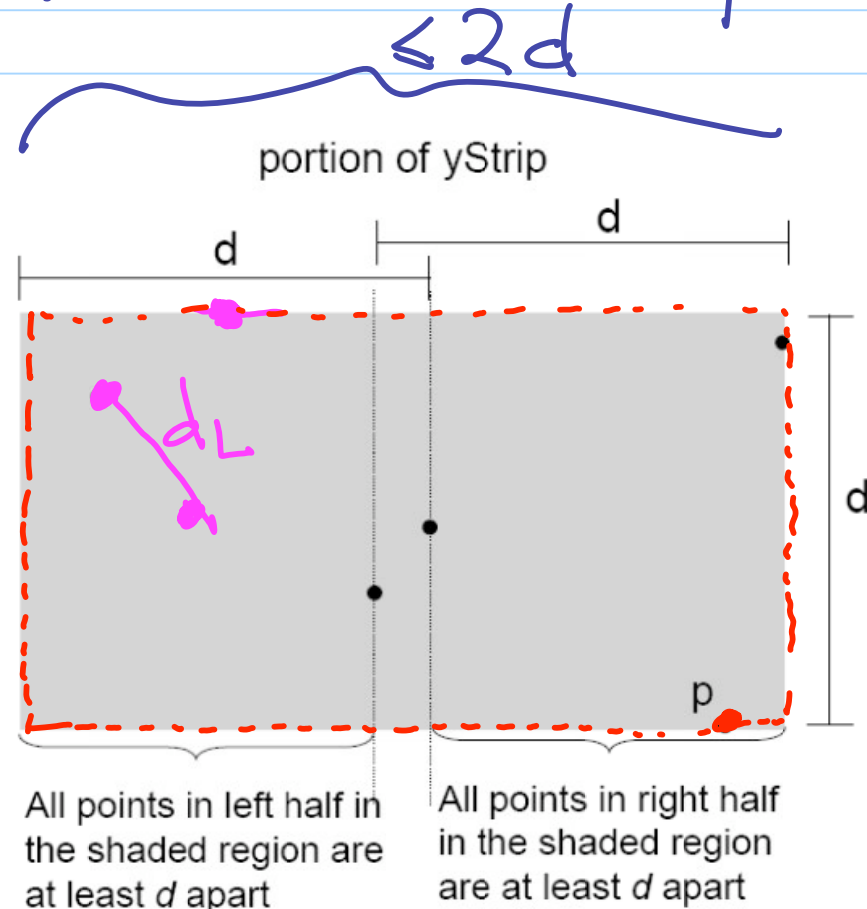
Combine - argue this  
is linear



# Time Complexity of Combine Step $\leq 2d$



Consider at most  
5 points in  
"while" loop for  
"point i"



# Expressing Time Complexity using Recurrence equation

$$T(n) = \underbrace{C_1 n}_{\text{divide}} + 2T\left(\frac{n}{2}\right) + \underbrace{C_2 n}_{\text{combine}}$$

constant

constant

$$= 2T\left(\frac{n}{2}\right) + Cn$$

$C_1 + C_2$