Lecture 5

Merge Sort

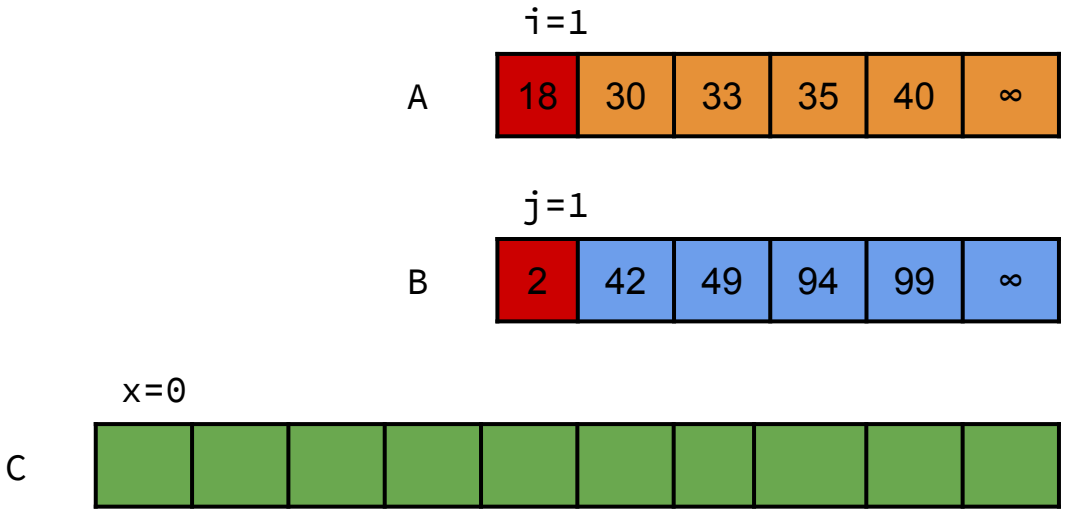
Merge Pseudocode

```
Merge(A,p,q,r):
    n1=q-p+1
    n2=r-q
    Let L[1...n1+1] and R[1...n2+1] be new arrays
    for i=1 to n1:
        L[i]=A[p+i-1]
    for i=1 to n2:
        R[i]=A[q+i]
    L[n1+1]=∞
    R[n2+1]=∞
```

Merge Pseudocode Continued.

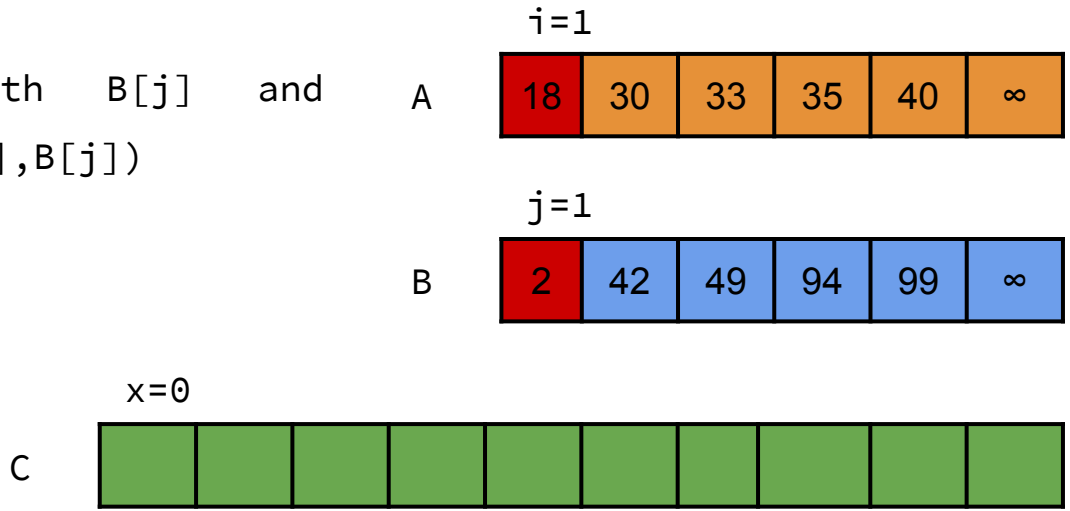
```
    i=1
    j=1
    for k=p to r:
        if L[i]≤R[j]:
            A[k]=L[i]
            i=i+1
        else:
            A[k]=R[j]
            j=j+1
```

Simulation of Merge



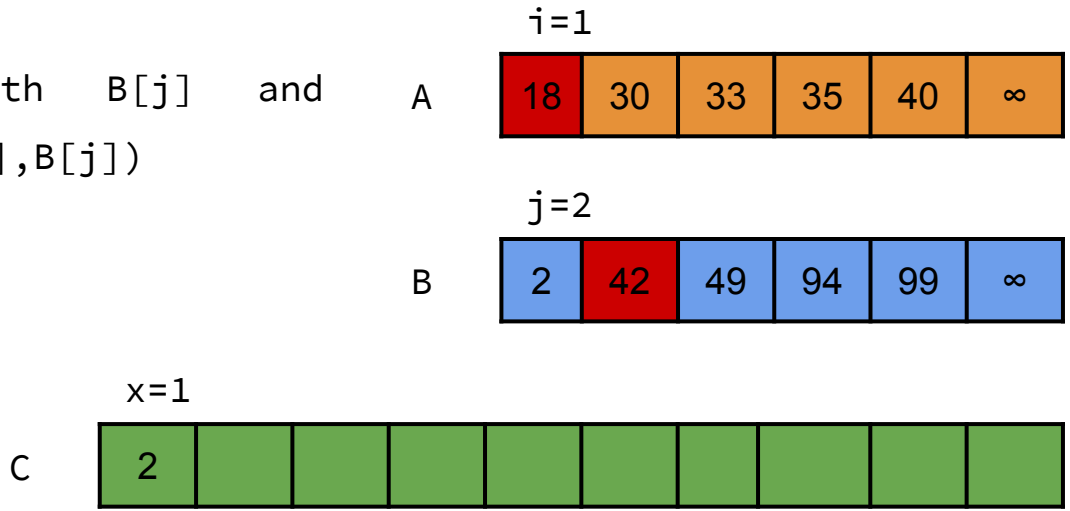
Simulation of Merge

Comparing A[i] with B[j] and setting C[x]=min(A[i],B[j])



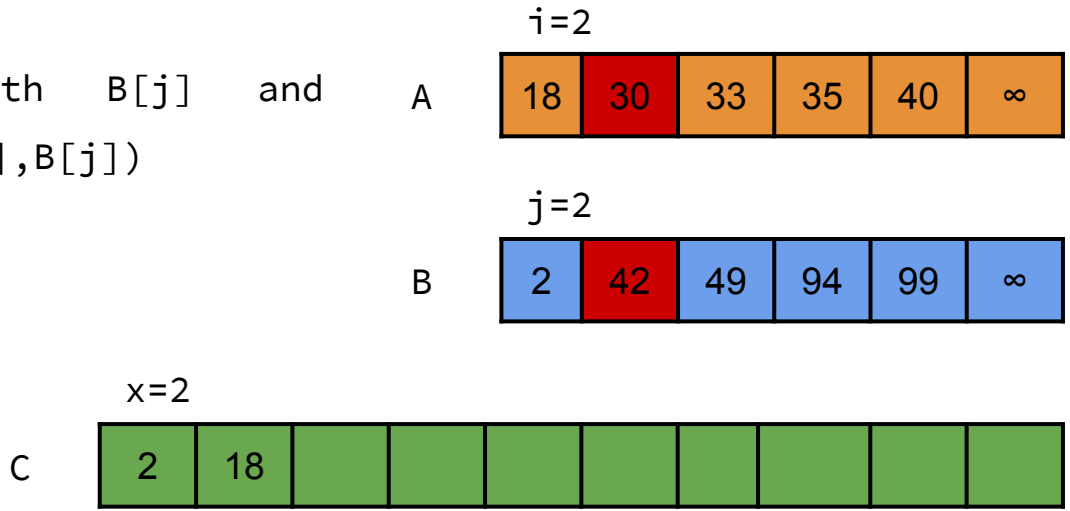
Simulation of Merge

Comparing A[i] with B[j] and setting C[x]=min(A[i],B[j])



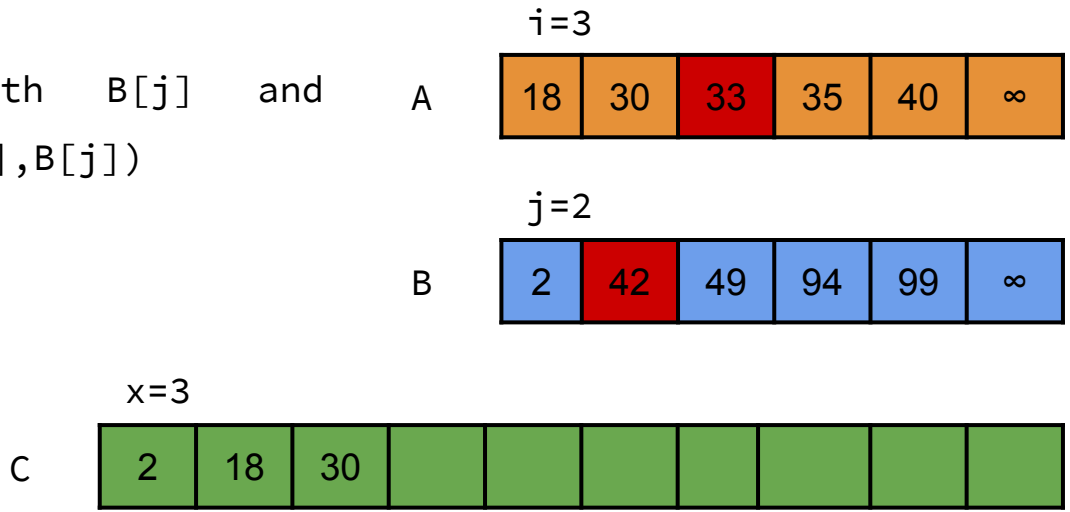
Simulation of Merge

Comparing A[i] with B[j] and setting C[x]=min(A[i],B[j])



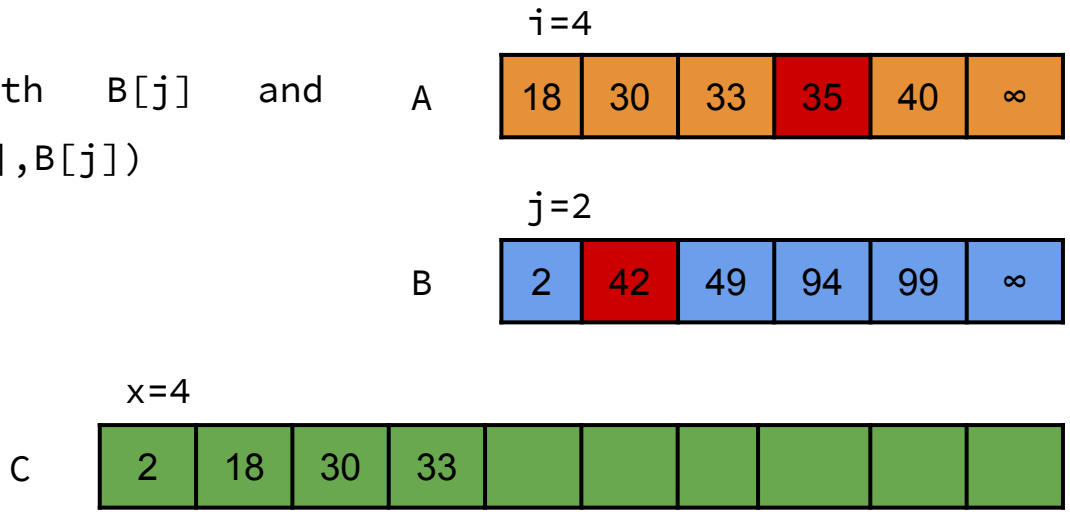
Simulation of Merge

Comparing A[i] with B[j] and setting C[x]=min(A[i],B[j])



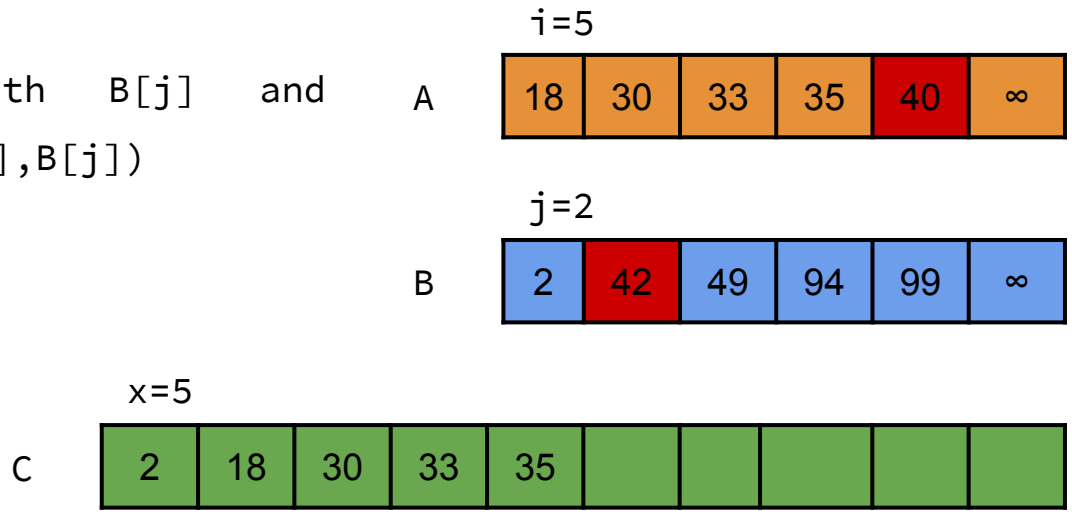
Simulation of Merge

Comparing A[i] with B[j] and setting C[x]=min(A[i],B[j])



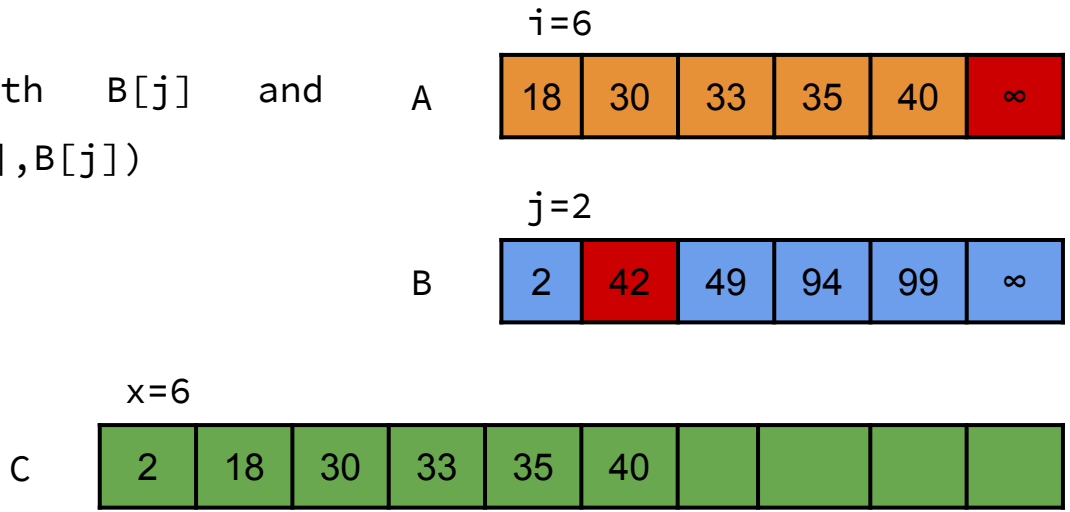
Simulation of Merge

Comparing A[i] with B[j] and setting C[x]=min(A[i],B[j])



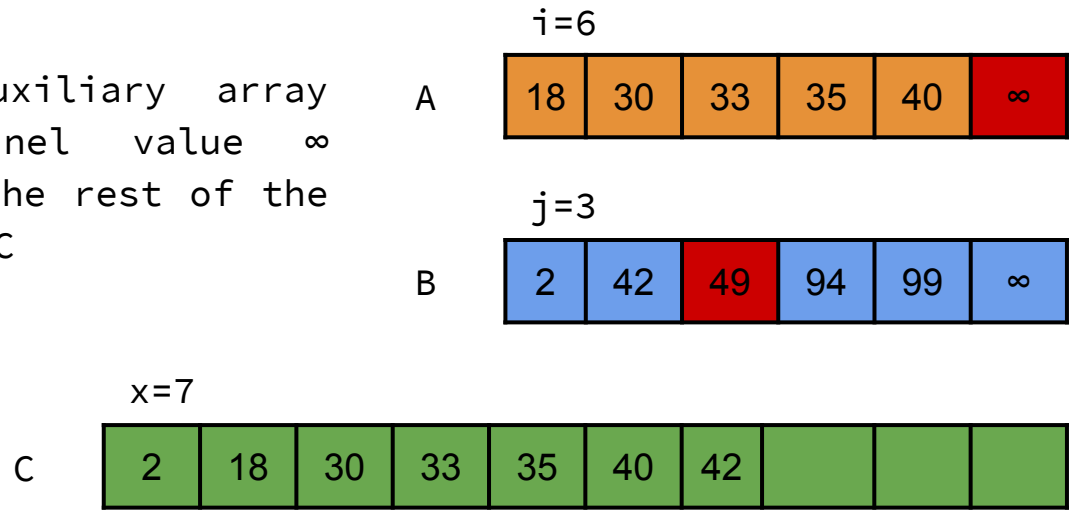
Simulation of Merge

Comparing A[i] with B[j] and setting C[x]=min(A[i],B[j])



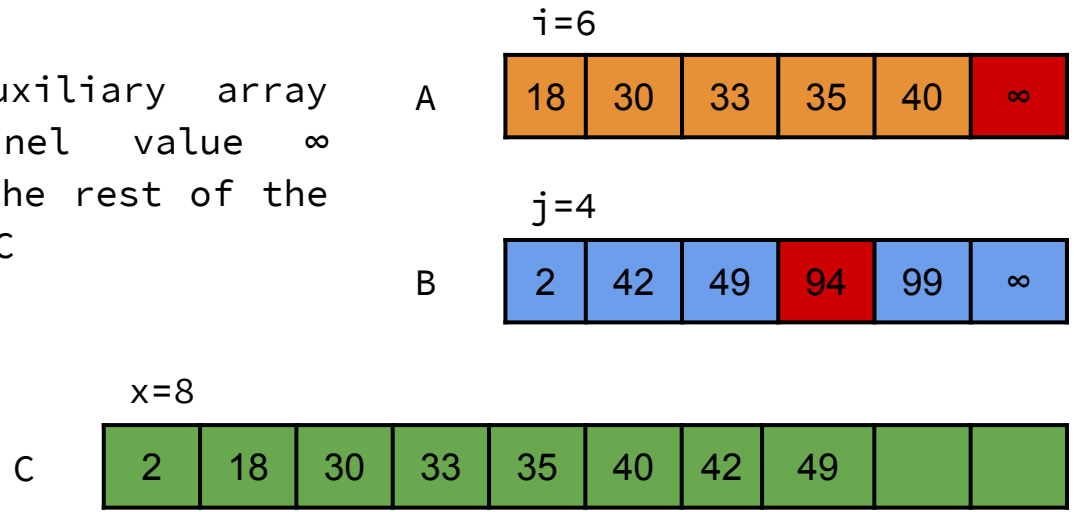
Simulation of Merge

As one of the auxiliary array reached the sentinel value ∞ therefore, copying the rest of the values to the array C



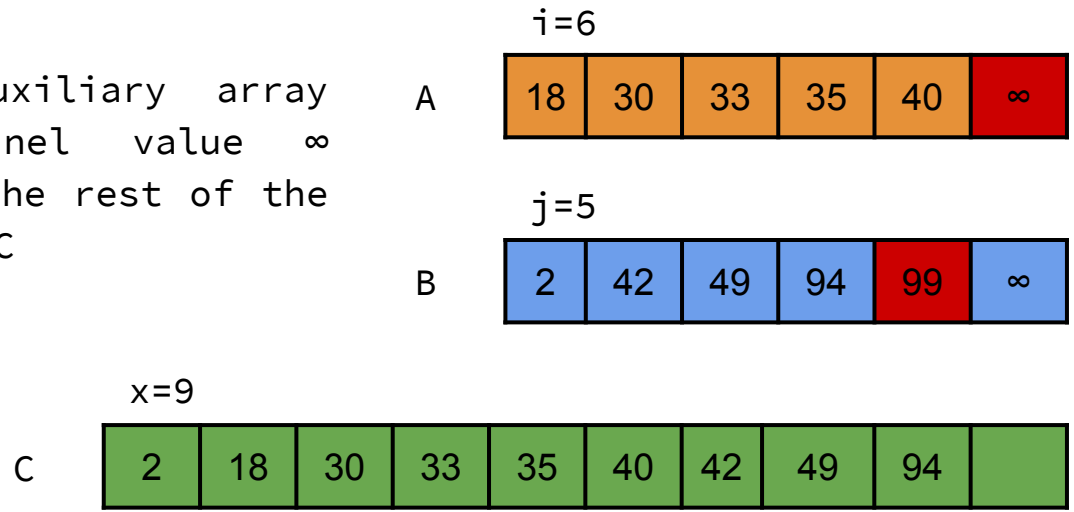
Simulation of Merge

As one of the auxiliary array reached the sentinel value ∞ therefore, copying the rest of the values to the array C



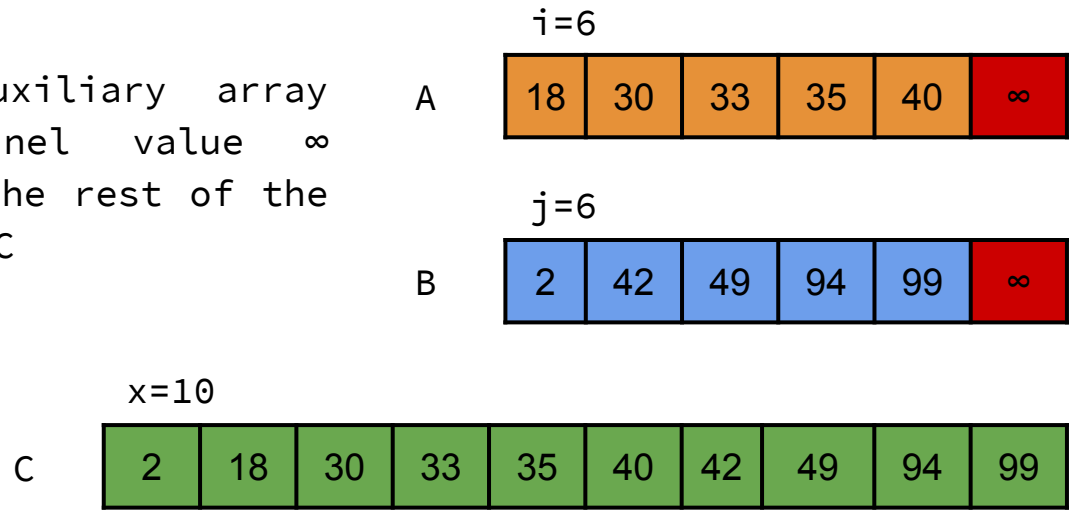
Simulation of Merge

As one of the auxiliary array reached the sentinel value ∞ therefore, copying the rest of the values to the array C



Simulation of Merge

As one of the auxiliary array reached the sentinel value ∞ therefore, copying the rest of the values to the array C

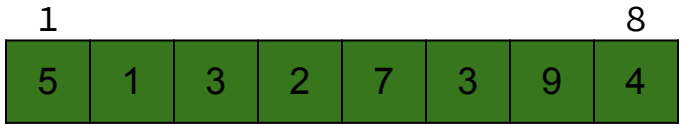


Merge Sort Pseudocode

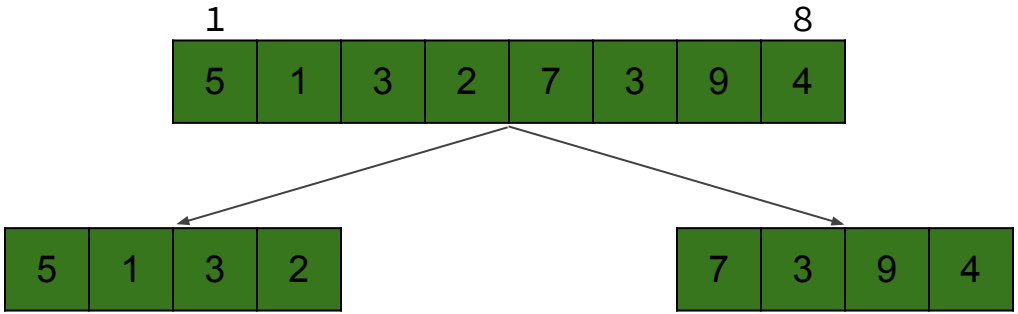
```
Merge_Sort(A,p,r):
    if p<r:
        q=⌊(p+r)/2⌋
        Merge_Sort(A,p,q)
        Merge_Sort(A,q+1,r)
        Merge(A,p,q,r)
```

Simulation of Merge Sort

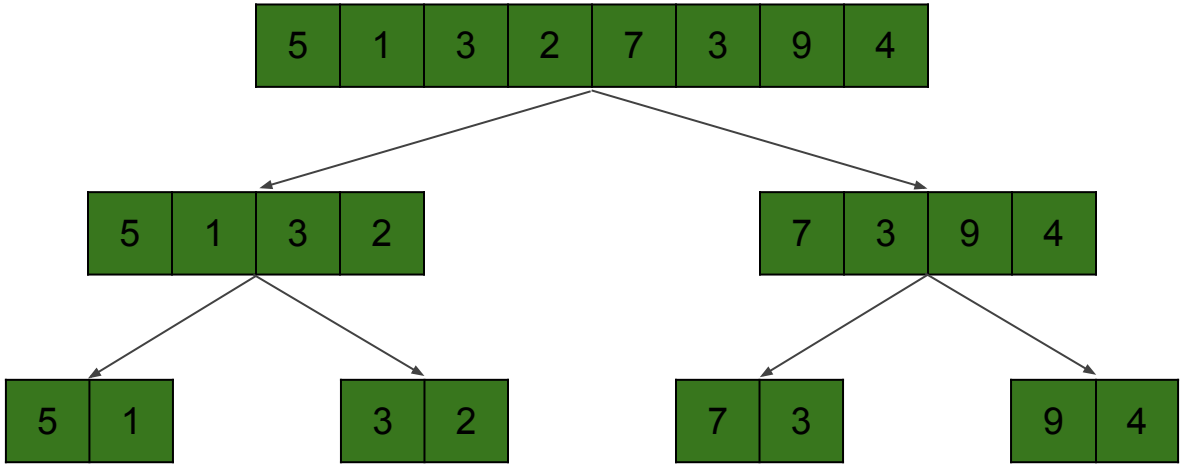
– – –



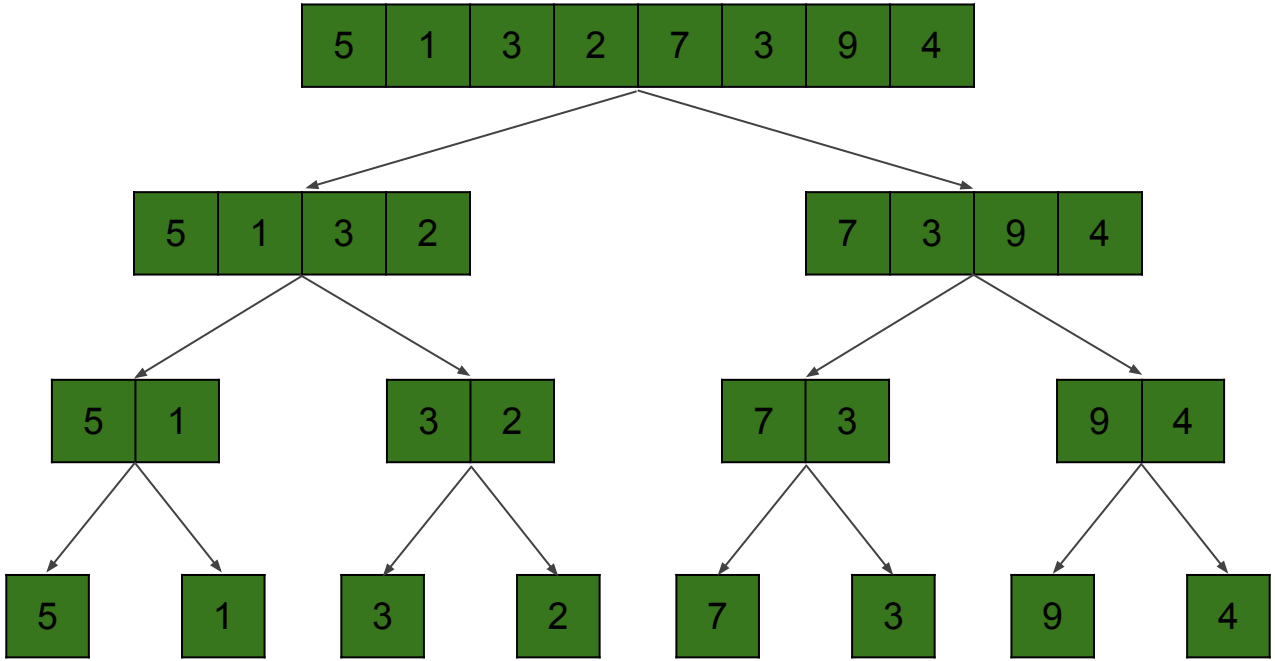
Simulation of Merge Sort



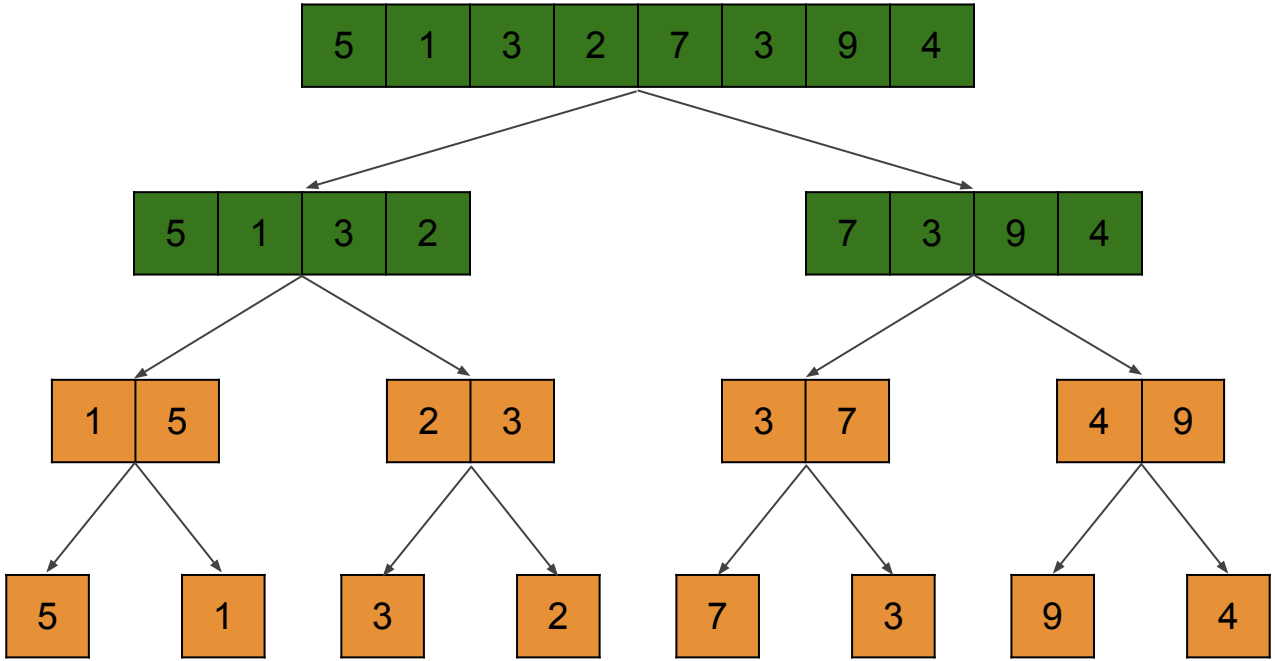
Simulation of Merge Sort



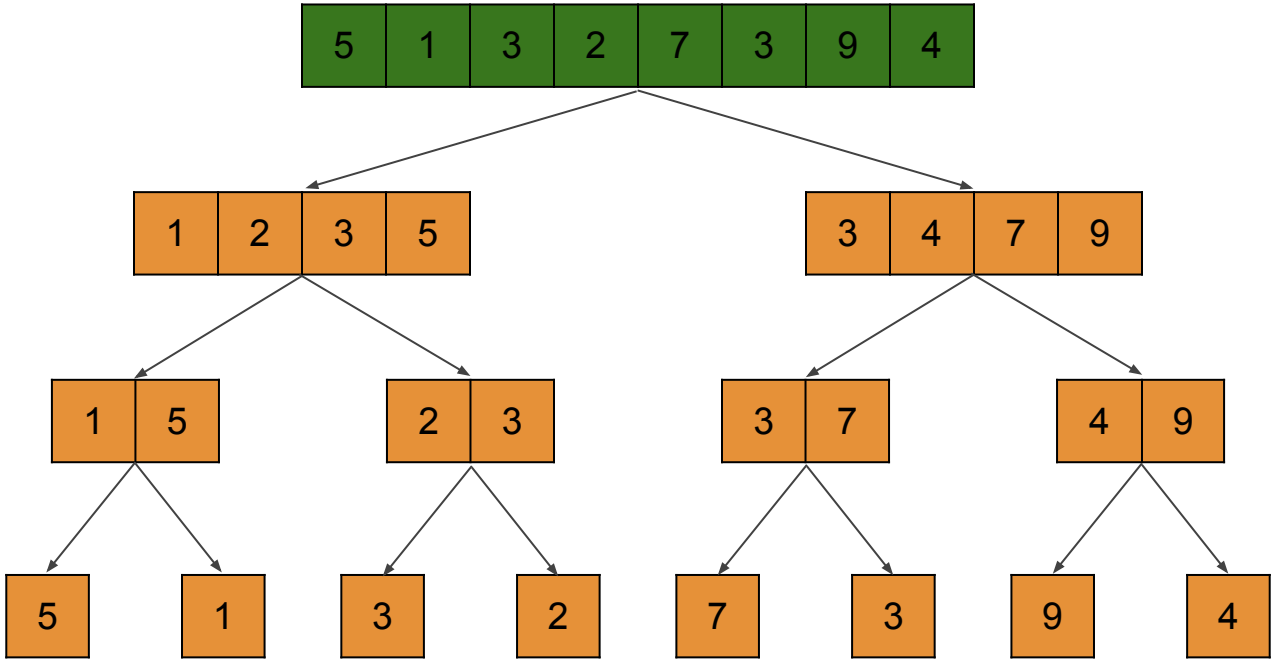
Simulation of Merge Sort



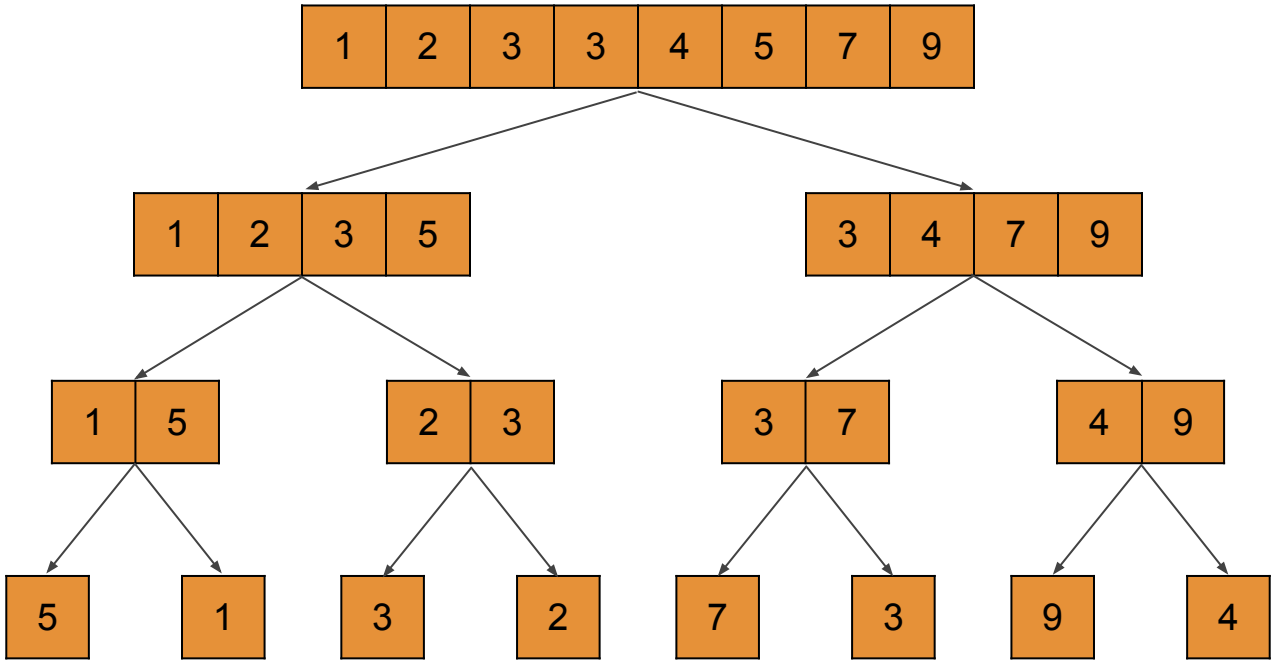
Simulation of Merge Sort



Simulation of Merge Sort



Simulation of Merge Sort

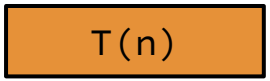


Analysis of Merge Sort

The height h of the merge-sort tree is $O(\log_2 n)$. At each recursive call we divide in half the sequence, The overall amount or work done at the nodes of depth i is O(n) .We partition and merge 2^i sequences of size $n/2^i$ and we make 2^i+1 recursive call Thus, the total running time of merge-sort is $O(n\log_2 n)$

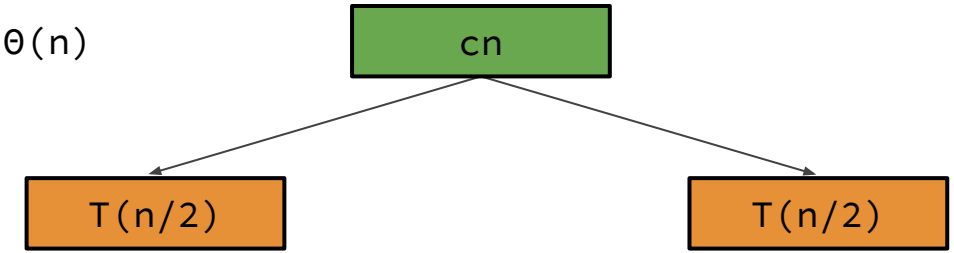
Analysis of Merge Sort Continued

$T(n) = 2T(n/2) + \Theta(n)$



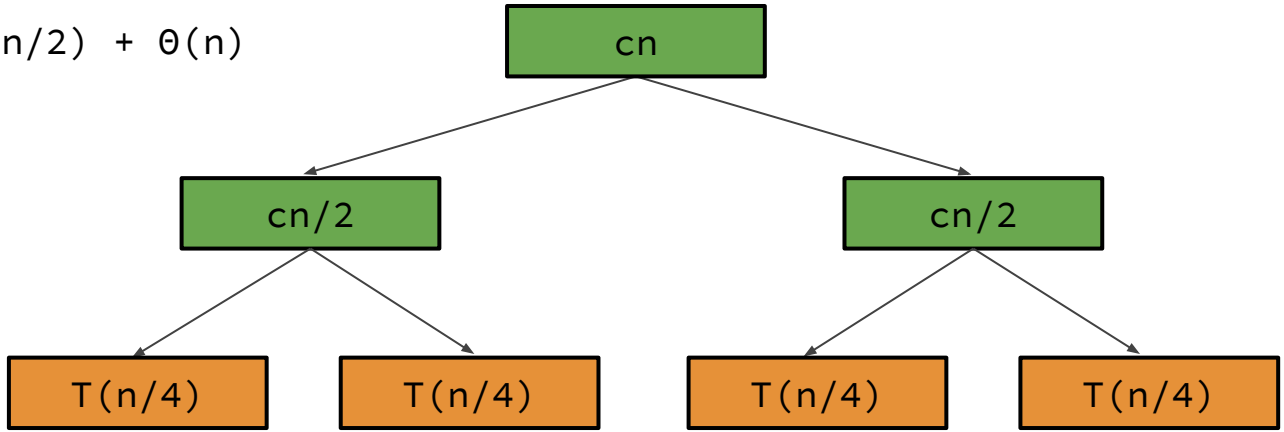
Analysis of Merge Sort Continued

$T(n) = 2T(n/2) + \Theta(n)$



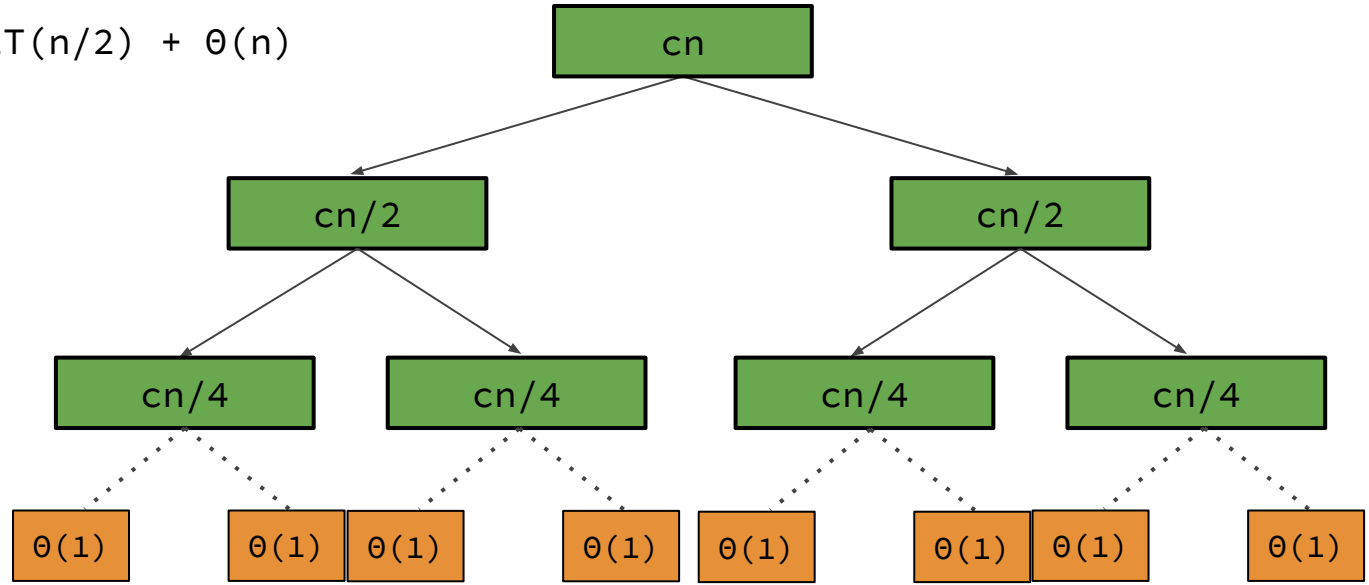
Analysis of Merge Sort Continued

$T(n) = 2T(n/2) + \Theta(n)$



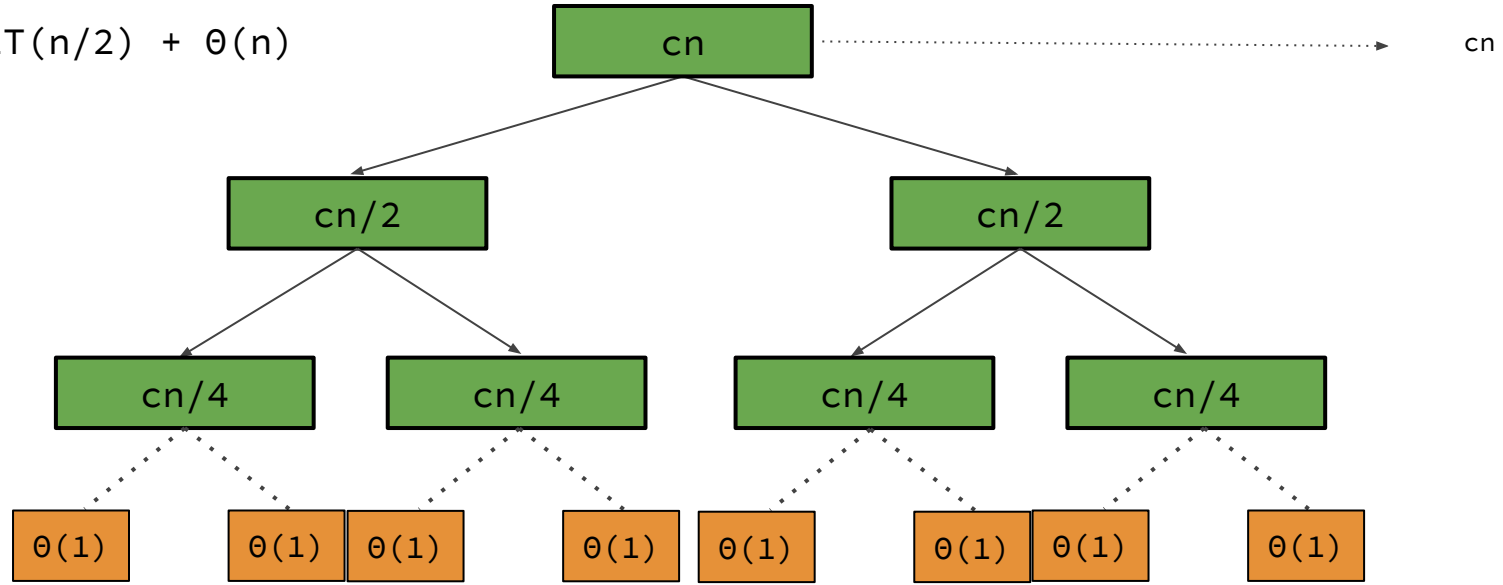
Analysis of Merge Sort Continued

$T(n) = 2T(n/2) + \Theta(n)$



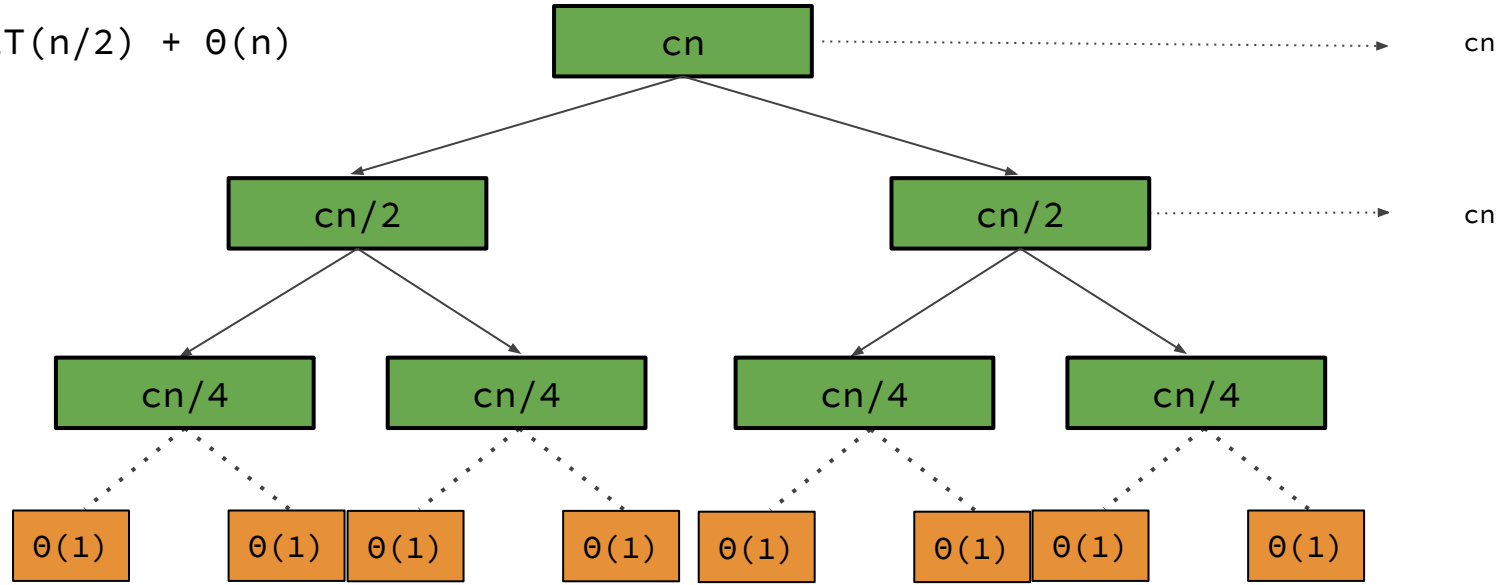
Analysis of Merge Sort Continued

$T(n) = 2T(n/2) + \Theta(n)$



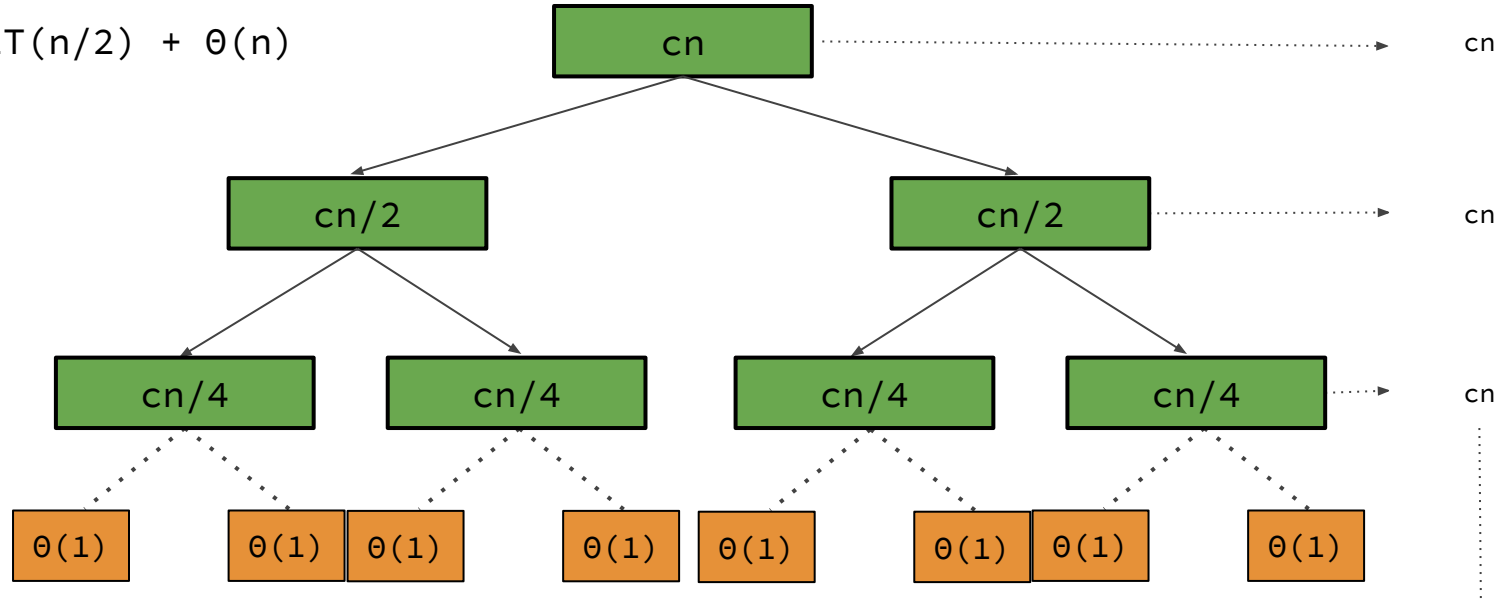
Analysis of Merge Sort Continued

$T(n) = 2T(n/2) + \Theta(n)$

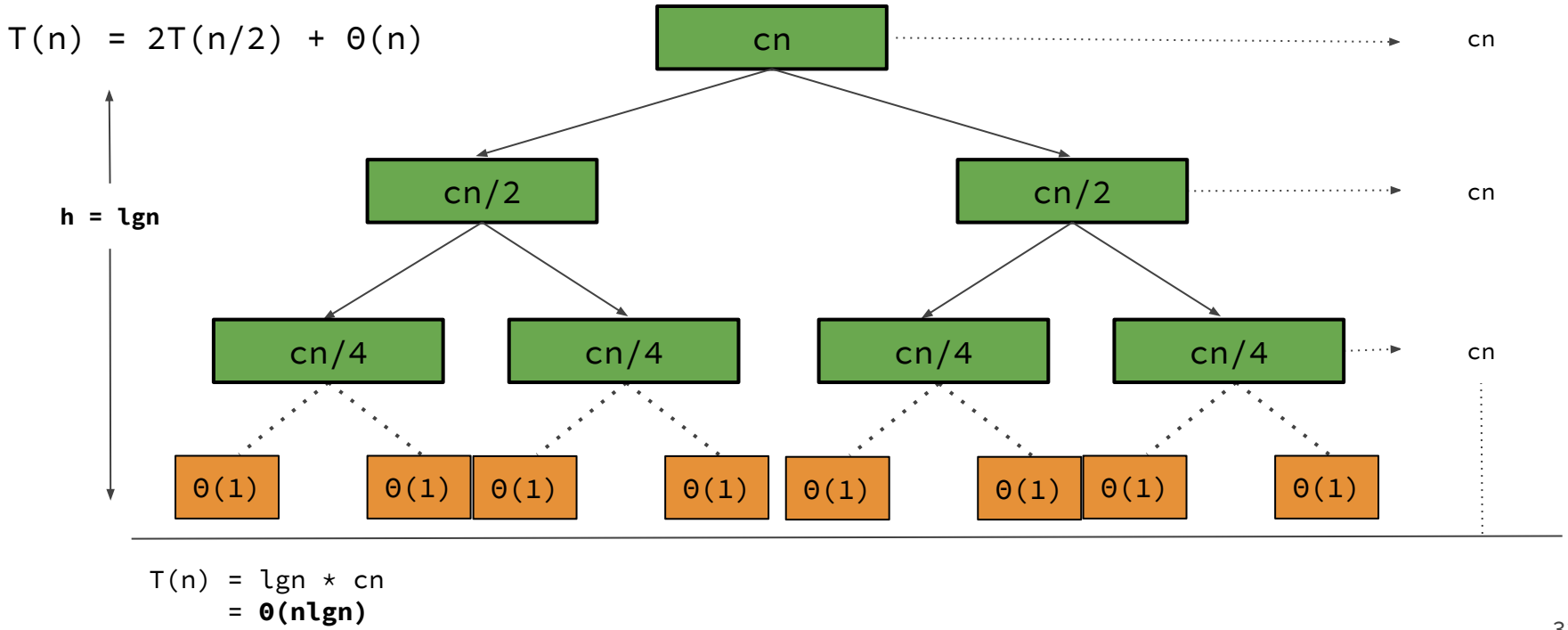


Analysis of Merge Sort Continued

$T(n) = 2T(n/2) + \Theta(n)$



Analysis of Merge Sort Continued



T(n) = lgn * cn
= **Θ(nlgn)**