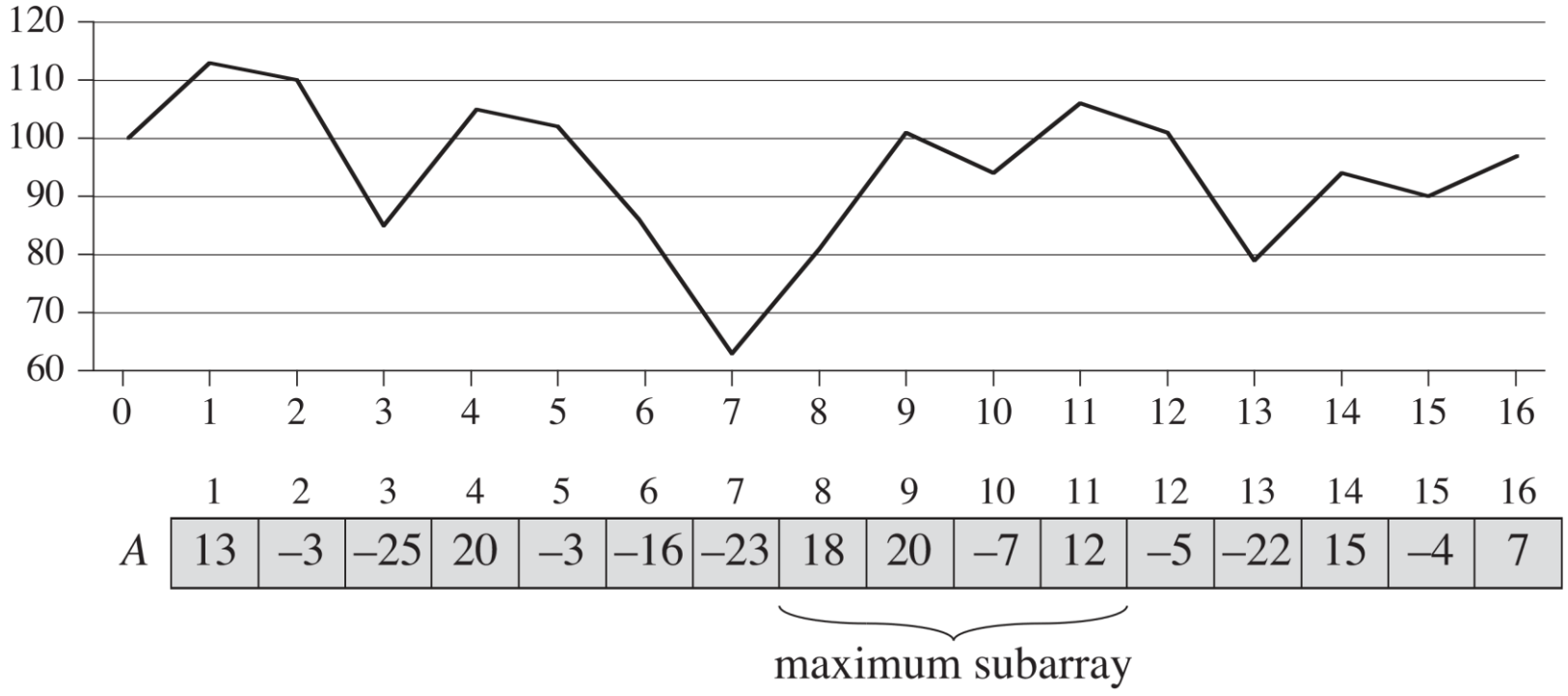
# *Dynamic Programming-1*

Department of Computer Science, Tsinghua University

# Maximum Subarray

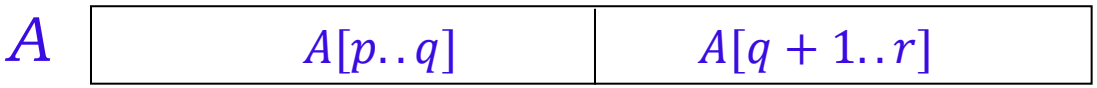
- Given an array $A$ of $n$ numbers, find the nonempty, contiguous subarray of $A$ whose values have the largest sum.



# Brute-force

```
Maximum-Subarray-Brute-Force(A)
1 n = A.length
2 maxsofar = −∞
3 for i = 1 to n
4       sum = 0
5       for j = i to n
6             sum = sum+ A[j]
7             maxsofar = max(maxsofar, sum)
8 return maxsofar
```

# D&C Solution




***1. Divide:*** partition $A$ into two halves

***2. Conquer:***

- find the maximum sum in the left half (say $m_{left}$)
- find the maximum sum in the right half (say $m_{right}$)

***3. Combine:***

- find the maximum sum cross the boundary (say $m_{cross}$)
- $\mathbf{min}(m_{left}, m_{right}, m_{cross})$.

# D&C Solution

```
Maximum-Subarray(A,p,r)
1 if p == r
2       return A[p]
3 q = ⌊(p+r)/2⌋
4 lmax = sum = 0
5 for i = q downto p
6       sum = sum+ A[i]
7       lmax = max(lmax, sum)
4 rmax = sum = 0
5 for i = q+1 to r
6       sum = sum+ A[i]
7       rmax = max(rmax, sum)
8 return max(lmax+rmax, Maximum-Subarray(A,p,q), Maximum-
Subarray(A,q+1,r) )
```


In-class exercise: the running time of Maximum-Subarray.

# DP Solution

① Make a choice to split the problem into one or more subproblems;

The maximum subarray contains $A[n]$ or not?

② Just assume you are given the choice that leads to an optimal solution $S$;

Yes/No

③ Given this choice, try to best characterize the remaining subproblems;

$Maxending[n]$ / $Max\_sub[n-1]$

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| $-2$ | 5 | $-2$ | 7 | $-2$ | $-10$ | 9 | $-1$ |

$$Max\_sub[i] = \max(Maxending[i],\ Max\_sub[i-1])$$

# DP Solution

| −2 | 5 | −2 | 7 | −2 | −10 | 9 | −1 |
|---|---|---|---|---|---|---|---|

$Maxending[p] = A[p]$

**for** $i = p+1$ **to** $r$

$Maxending[i] = \max(Maxending[i-1] + A[i],\ A[i])$

$Maxending$

| −2 | | | | | | | |
|---|---|---|---|---|---|---|---|

$Max\_sub[p] = A[p]$

**for** $i = p+1$ **to** $r$

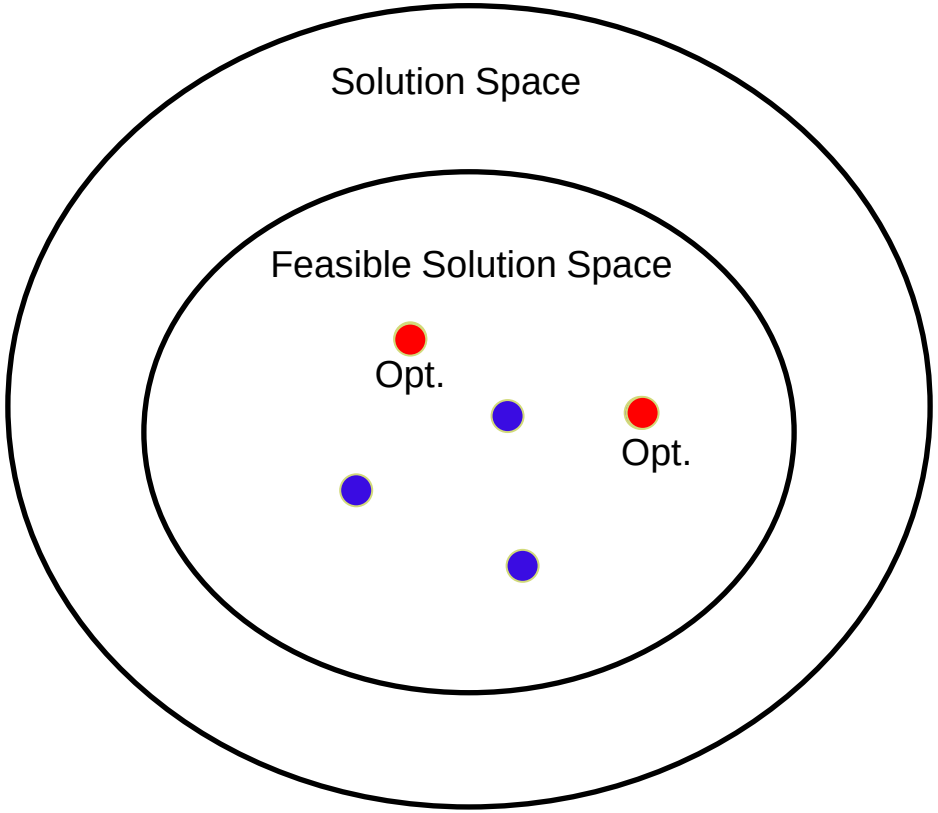
$Max\_sub[i] = \max(Maxending[i],\ Max\_sub[i-1])$

$Max\_sub$

| −2 | | | | | | | |
|---|---|---|---|---|---|---|---|

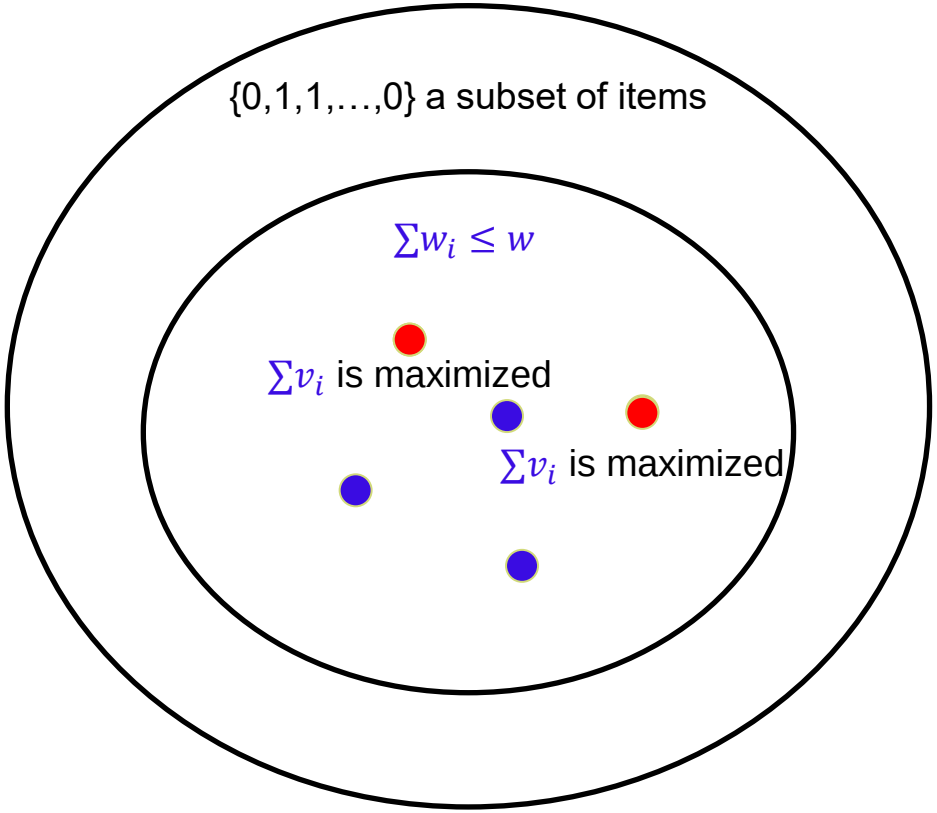
# Optimization Problems

- For any instance of a problem, there are multiple feasible solutions, each solution has a cost value, and the task is to find the solutions with the optimal cost value.




# Knapsack

- Given $n$ items: $a_1, a_2, \ldots, a_n$, and their weights: $w_1, w_2, \ldots, w_n$, and values: $v_1, v_2, \ldots, v_n$. A knapsack of capacity $w$. Find a subset of items of total weight $\leq w$ and of maximum value.




# Group Discussion Knapsack

- Given $n$ items: $a_1, a_2, \ldots, a_n$, and their weights: $w_1, w_2, \ldots, w_n$, and values: $v_1, v_2, \ldots, v_n$. A knapsack of capacity $w$. Find a subset of items of total weight $\leq w$ and of maximum value.

① Make a choice to split the problem into one or more subproblems;

The optimal subset contains $a_n$ or not?

② Just assume you are given the choice that leads to an optimal solution $S$;

Yes/No

③ Given this choice, try to best characterize the remaining subproblems?