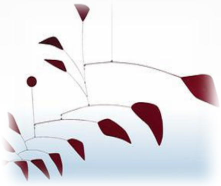
EECS 3101 - Design and Analysis of Algorithms



Shahin Kamali

Tutorial

York University

Picture is from the cover of the textbook CLRS.

Highway Billboard Placement

- Consider a highway of M kilometers. The task is to place billboards on the highway such that revenue is maximized. The possible sites for billboards are given by number $x_1 < x_2 < \ldots < x_{n-1} < x_n$ specifying positions in kilometers measured from one end of the road.
  - If we place a billboard at position x_i, we receive a revenue of $r_i > 0$.
  - The constraint is that no two billboards can be placed within d kilometers or less than it.
  - **Example:** $M = 15, n = 5, d = 5,$ $(x, r) = (6, 3), (8, 6), (12, 5), (14, 3), (15, 5),$
  - **Potential answers:**
    $\{(6, 3), (12, 5)\} \rightarrow$ revenue: 8
    $\{(8, 6), (15, 5)\}$: revenue: 11

Highway Billboard Placement

- **Step 1:** define subproblems; let *Profit*(i) define the maximum revenue for an input formed by the first i billboards ($1 \leq i \leq n$). We want to find *Profit*(n).

Highway Billboard Placement

- **Step 1:** define subproblems; let *Profit*(i) define the maximum revenue for an input formed by the first i billboards ($1 \leq i \leq n$). We want to find *Profit*(n).
- **Step 2:** devise a recursive formula for Profit i

Highway Billboard Placement

- **Step 1:** define subproblems; let *Profit*(i) define the maximum revenue for an input formed by the first i billboards $(1 \leq i \leq n)$. We want to find *Profit*(n).
- **Step 2:** devise a recursive formula for Profit i
  - Base case: we have $Profit(0) = 0$.

Highway Billboard Placement

- **Step 1:** define subproblems; let $Profit(i)$ define the maximum revenue for an input formed by the first i billboards ($1 \leq i \leq n$). We want to find $Profit(n)$.
- **Step 2:** devise a recursive formula for Profit i
  - Base case: we have $Profit(0) = 0$.
  - If we reject the i'th billboard, we get a candidate solution with value $Profit[i-1]$.

Highway Billboard Placement

- **Step 1:** define subproblems; let $Profit(i)$ define the maximum revenue for an input formed by the first i billboards ($1 \leq i \leq n$). We want to find $Profit(n)$.
- **Step 2:** devise a recursive formula for Profit i
  - Base case: we have $Profit(0) = 0$.
  - If we reject the i'th billboard, we get a candidate solution with value $Profit[i-1]$.
  - If we accept the i'th billboard, we get a revenue $r[i]$. In addition, let $pred$ be the larges value $j < i$ s.t. $x[i] - x[j] > d$ (and 0 if no such j exists). In addition to $r[i]$ we can get a profit of $Profit[pred[i]]$ from other billboards.

    Example: $pred[5] = 3$ in the above example (recall that $d = 5$).

$$Profit[i] = \begin{cases} 0 & i = 0 \\ \max\{Profit[i-1], Profit[pred[i]] + r[i]\} \end{cases}$$

Submatrix Sum

- Given an $M \times N$ matrix A and two coordinates (p, q) and (r, s) representing top-left and bottom-right coordinates of a submatrix of it, calculate the sum of all elements present in the submatrix. Here, $0 \leq p < r < M$ and $0 \leq q < s < N$.

  Example: for $(p, q) = (2, 2)$ and $(r, s) = (3, 3)$, sum is $8 + 1 + 1 + 3 = 13$

	1	2	3	4	5	6
1	3	7	6	2	2	4
2	9	8	1	6	1	2
3	2	1	3	5	4	3
4	4	9	3	7	3	5
5	1	8	2	5	6	3

Submatrix Sum

- Given an $M \times N$ matrix A and two coordinates (p, q) and (r, s) representing top-left and bottom-right coordinates of a submatrix of it, calculate the sum of all elements present in the submatrix. Here, $0 \leq p < r < M$ and $0 \leq q < s < N$.

  Example: for $(p, q) = (2, 2)$ and $(r, s) = (3, 3)$, sum is $8 + 1 + 1 + 3 = 13$

	1	2	3	4	5	6
1	3	7	6	2	2	4
2	9	8	1	6	1	2
3	2	1	3	5	4	3
4	4	9	3	7	3	5
5	1	8	2	5	6	3

- We want to report the sum in $O(1)$. This requires **pre-processing** the input!

Submatrix Sum

- **Pre-processing:** process the matrix **once** and use the results of your calculation in answering any query.

	1	2	3	4	5	6
1	3	7	6	2	2	4
2	9	8	1	6	1	2
3	2	1	3	5	4	3
4	4	9	3	7	3	5
5	1	8	2	5	6	3

Submatrix Sum

- **Pre-processing:** process the matrix **once** and use the results of your calculation in answering any query.
  - Take an auxiliary matrix $sum[][]$, where $sum[i][j]$ will store the sum of elements in the matrix from $(1, 1)$ to (i, j), e.g., $sum[3, 2] = 3 + 7 + 9 + 8 + 2 + 1$.

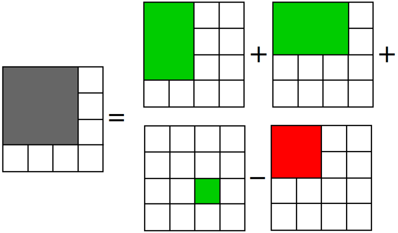
	1	2	3	4	5	6
1	3	7	6	2	2	4
2	9	8	1	6	1	2
3	2	1	3	5	4	3
4	4	9	3	7	3	5
5	1	8	2	5	6	3

Submatrix Sum

- **Pre-processing:** process the matrix **once** and use the results of your calculation in answering any query.
  - Take an auxiliary matrix *sum*[][], where *sum*[*i*][*j*] will store the sum of elements in the matrix from $(1, 1)$ to (i, j), e.g., $sum[3, 2] = 3 + 7 + 9 + 8 + 2 + 1$.
  - We calculate the value of *sum*[*i*][*j*] using the following relation:

$$sum[i][j] = \begin{cases} 0 & \text{if } i < 0 \text{ or } j < 0 \\ sum[i][j-1] + sum[i-1][j] + A[i][j] - sum[i-1][j-1] & \text{if } i, j > 0 \end{cases}$$

	1	2	3	4	5	6
1	3	7	6	2	2	4
2	9	8	1	6	1	2
3	2	1	3	5	4	3
4	4	9	3	7	3	5
5	1	8	2	5	6	3



Submatrix Sum

- **Real-time Queries:** use pre-computed matrix *sum* to answer queries in constant time.

Submatrix Sum

- **Real-time Queries:** use pre-computed matrix *sum* to answer queries in constant time.
- To calculate the sum of elements present in the submatrix formed by coordinates $(p, q), (p, s), (r, q), and (r, s)$ in constant time, we apply the relation below:
  - $total = sum[r][s] - sum[r][q-1] - sum[p-1][s] + sum[p-1][q-1]$