Design and Analysis of Algorithms

Lecture-31

Dharmendra Kumar (Associate Professor)
Department of Computer Science and Engineering
United College of Engineering and Research,
Prayagraj

Divide and Conquer

Divide and Conquer Approach

- The divide-and-conquer paradigm involves three steps at each level of the recursion:
- **Divide** the problem into a number of sub-problems that are smaller instances of the same problem.
- **Conquer** the sub-problems by solving them recursively. If the sub-problem sizes are small enough, however, just solve the sub problems in a straightforward manner.
- **Combine** the solutions to the sub-problems into the solution for the original problem.

Divide and Conquer Approach

We will solve the following problems using divide and conquer approach :-

- Sorting
  - Merge sort
  - Quick sort
- Searching
  - Binary search
- Matrix Multiplication
- Convex Hull

Binary search

- **Binary search** is the most popular Search algorithm. It is efficient and also one of the most commonly used techniques that is used to solve problems.

- Binary search works only on a sorted set of elements. To use binary search on a collection, the collection must first be sorted.

- When binary search is used to perform operations on a sorted set, the number of iterations can always be reduced on the basis of the value that is being searched.

Binary search process

Binary search algorithm

```
Binary-search(A, n, x)
l = 1
r = n
while l ≤ r
do
        m = ⌊(l + r) / 2⌋
        if A[m] < x then
                l = m + 1
        else if A[m] > x then
                r = m − 1
        else
                return m
return unsuccessful
```

Time complexity **$T(n) = O(\lg n)$**

Matrix Multiplication (Divide and Conquer Method)

To multiply two matrices A and B of order nxn using **Divide and Conquer approach**, we use to multiply two matrices of order 2x2.

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \qquad B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$$

Where, A_{ij} and B_{ij} are $\frac{n}{2} \times \frac{n}{2}$ matrices for i,j = 1,2.

Resultant matrix C will be

$$C = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}$$

Where,

$C_{11} = A_{11}B_{11} + A_{12}B_{21}$ $\qquad$ $C_{12} = A_{11}B_{21} + A_{12}B_{22}$

$C_{21} = A_{21}B_{11} + A_{22}B_{21}$ $\qquad$ $C_{22} = A_{21}B_{21} + A_{22}B_{22}$

Matrix Multiplication

Clearly, computation of c_{ij} consists of two multiplications of two $\frac{n}{2}$ x $\frac{n}{2}$ matrices and one addition of two $\frac{n}{2}$ x $\frac{n}{2}$ matrices. Therefore, the algorithms for this is the following:-

```
SQUARE-MATRIX-MULTIPLY-RECURSIVE(A, B)
n = A.rows
let C be a new n × n matrix
if n == 1
    c11 = a11 · b11
else partition A, B, and C as in equations (4.9)
    C11 = SQUARE-MATRIX-MULTIPLY-RECURSIVE(A11, B11)
          + SQUARE-MATRIX-MULTIPLY-RECURSIVE(A12, B21)
    C12 = SQUARE-MATRIX-MULTIPLY-RECURSIVE(A11, B12)
          + SQUARE-MATRIX-MULTIPLY-RECURSIVE(A12, B22)
    C21 = SQUARE-MATRIX-MULTIPLY-RECURSIVE(A21, B11)
          + SQUARE-MATRIX-MULTIPLY-RECURSIVE(A22, B21)
    C22 = SQUARE-MATRIX-MULTIPLY-RECURSIVE(A21, B12)
          + SQUARE-MATRIX-MULTIPLY-RECURSIVE(A22, B22)
return C
```

Matrix Multiplication

Time complexity of this algorithm is computed as following:-

$$T(n) = \theta(1) \qquad \text{if } n=1$$

$$= 8T(n/2) + \theta(n^2) \qquad \text{if } n > 1$$

After solving this recurrence relation, we get

$$T(n) = \theta(n^3)$$

Strassen's Matrix Multiplication Algorithm

Strassen's algorithm has three steps:

1) Divide the input matrices A and B into $\frac{n}{2} \mathrm{x} \frac{n}{2}$ sub-matrices.
2) Using the sub-matrices created from the step above, recursively compute seven matrix products P_1, P_2, … P_7. Each matrix P_i is of size $\frac{n}{2} \mathrm{x} \frac{n}{2}$.

   $P1 = A_{11}(B_{12}-B_{22})$ $\quad$ $P2 = (A_{11}+A_{12})B_{22}$

   $P3 = (A_{21}+A_{22})B_{11}$ $\quad$ $P4 = A_{22}(B_{21}-B_{11})$

   $P5 = (A_{11}+A_{22})(B_{11}+B_{22})$ $\quad$ $P6 = (A_{12}-A_{22})(B_{21}+B_{22})$

   $P7 = (A_{11}-A_{21})(B_{11}+B_{12})$

3) Get the desired sub-matrices C_{11}, C_{12}, C_{21}, and C_{22} of the result matrix C by adding and subtracting various combinations of the P_i sub-matrices.

$C_{11} = P_5+P_4-P_2+P_6$ $C_{12}= P_1+P_2$

$C_{21}= P_3+P_4$ $\quad$ $C_{22}= P_1+P_5-P_3-P_7$

Strassen's Matrix Multiplication Algorithm

Strassen_Matrix_Multiplication(A, B, n)

If n=1 then return AxB.

Else

1. Compute A_{11}, B_{11}, ………………., A_{22}, B_{22}.
2. $P_1 \leftarrow$ Strassen_Matrix_Multiplication(A_{11}, B_{12}-B_{22}, n/2)
3. $P_2 \leftarrow$ Strassen_Matrix_Multiplication(A_{11}+A_{12}, B_{22}, n/2)
4. $P_3 \leftarrow$ Strassen_Matrix_Multiplication(A_{21}+A_{22}, B_{11}, n/2)
5. $P_4 \leftarrow$ Strassen_Matrix_Multiplication(A_{22}, B_{21}-B_{11}, n/2)
6. $P_5 \leftarrow$ Strassen_Matrix_Multiplication(A_{11}+A_{22}, B_{11}+B_{22}, n/2)
7. $P_6 \leftarrow$ Strassen_Matrix_Multiplication(A_{12}-A_{22}, B_{21}+B_{22}, n/2)
8. $P_7 \leftarrow$ Strassen_Matrix_Multiplication(A_{11}-A_{21}, B_{11}+B_{12}, n/2)
9. $C_{11} = P_5+P_4-P_2+P_6$
10. $C_{12} = P_1+P_2$
11. $C_{21} = P_3+P_4$
12. $C_{22} = P_1+P_5-P_3-P_7$
13. return C

End if

Strassen's Matrix Multiplication Algorithm

Time complexity of this algorithm is computed as following:-

$$T(n) = \begin{cases} \theta(1) & \text{if } n=1 \\ 7T(n/2) + \theta(n^2) & \text{if } n > 1 \end{cases}$$

After solving this recurrence relation, we get

$$\mathbf{T(n) = \theta(n^{2.8})}$$