Algorithms Design and Analysis [ETCS-301]

Dr. A K Yadav
Amity School of Engineering and Technology
(affiliated to GGSIPU, Delhi)
akyadav1@amity.edu
akyadav@akyadav.in
www.akyadav.in
+91 9911375598

August 20, 2019

Complexity analysis: Quick Sort I

```
QUICK-SORT(A, p, r)
  if p < r
      q=PARTITION(A,p,r)
      QUICK-SORT(A, p, q-1)
      QUICK-SORT(A, q+1, r)
PARTITION(A, p, r)
  x = A[r]
  i = p
  for j = p to r − 1
      if A[j] ≤ x
          if(i ≠ j) swap(A[i], A[j])
          i = i + 1
  swap(A[i], A[r])
  return i
```

Analysis:

Complexity analysis: Quick Sort II

- The number of calls of PARTITION depends on QUICK-SORT
- The number of calls of QUICK-SORT depends on q
- If PARTITION returns q as middle value for every call
- Then array will be divided in two half every time
- So after $\lg n$ times, $p = r$ and process will terminate.
- This will be **Best Case** of the algorithm.
- $\therefore T(n) = O(n \lg n)$ for best case
- But if PARTITION find $A[r]$ fit at its current location after checking p to r-1 elements then it return q as an end position and making partition of size 0 and $n - 1$.
- So every time the array size is reduced by only 1.
- The process will terminate after n operations.

Complexity analysis: Quick Sort III

- This will be **Worst Case** of the algorithm and happens when input is already sorted.
- Step 7 will be executed n times for every value of q in step 2.
- $\therefore T(n) = O(n^2)$ for worst case

Randomized Quick Sort I

- To avoid the worst performance of the Quick Sort for sorted input, we use Randomized Quick Sort.
- In this we choose pivot element randomly.

Algorithm:

RANDOMIZED-QUICK-SORT(A, p, r)

if $p < r$

  q=RANDOMIZED-PARTITION(A,p,r)

  RANDOMIZED-QUICK-SORT(A, p, q-1)

  RANDOMIZED-QUICK-SORT(A, q+1, r)

RANDOMIZED-PARTITION(A, p, r)

$i = Random(p, r)$

$swap(A[i], A[r])$

$x = A[r]$

$i = p$

Randomized Quick Sort II

for $j = p$ to $r - 1$
    if $A[j] \leq x$
        if$(i \neq j)$ swap$(A[i], A[j])$
        $i = i + 1$
swap$(A[i], A[r])$
return i

This algorithm performs worst case only when $Random(p, r)$ always gives the location of the largest/smallest number which is very rare.

Strassen's algorithm for Matrix Multiplications I

- if we multiply two matrices $C_{n\times m} = A_{n\times l}B_{l\times m}$
- Total number of scaler multiplications will be $n \times l \times m$
- If we take matrices of size $n \times n$ where n is power of 2 i.e $n = 2^i$
- Total number of scaler multiplications will be n^3
- If we divide size by 2 then the matrices will be

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}$$

$$B = \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix}$$

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

Strassen's algorithm for Matrix Multiplications II

- $$\begin{pmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix}$$
- $$C_{11} = A_{11}.B_{11} + A_{12}.B_{21}$$
  $$C_{12} = A_{11}.B_{12} + A_{12}.B_{22}$$
  $$C_{21} = A_{21}.B_{11} + A_{22}.B_{21}$$
  $$C_{22} = A_{21}.B_{12} + A_{22}.B_{22}$$
- So there are 8 multiplication and 4 submissions of size $n/2$

- $$T(n) = 8T\left(\frac{n}{2}\right) + 4(n/2)^2$$

- Solution of the recurrence will be $\Theta(n^3)$ using master theorem. So no benefit of dividing the main problem in subproblems.
- Strassen's gives 18 sum and 7 products of size $n/2$ as follows:

$$S_1 = B_{12} - B_{22}$$

$$S_2 = A_{11} + A_{12}$$

$$S_3 = A_{21} + A_{22}$$

$$S_4 = B_{21} - B_{11}$$

$$S_5 = A_{11} + A_{22}$$

Strassen's algorithm for Matrix Multiplications IV

$$S_6 = B_{11} + B_{22}$$

$$S_7 = A_{12} - A_{22}$$

$$S_8 = B_{21} + B_{22}$$

$$S_9 = A_{11} - A_{21}$$

$$S_{10} = B_{11} + B_{12}$$

- 7 products will be

$$P_1 = A_{11}.S_1$$

$$P_2 = S_2.B_{22}$$

$$P_3 = S_3.B_{11}$$

$$P_4 = A_{22}.S_4$$

$$P_5 = S_5 . S_6$$

$$P_6 = S_7 . S_8$$

$$P_7 = S_9 . S_{10}$$

- Now the the matrix will be

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

$$C_{12} = P_1 + P_2$$

$$C_{21} = P_3 + P_4$$

$$C_{22} = P_5 + P_1 - P_3 - P_7$$

- There are 7 multiplication and 18 submissions of size $n/2$

- $$T(n) = 7T\left(\frac{n}{2}\right) + 18(n/2)^2$$
- The solution of the recurrence will be $\Theta(n^{\lg 7})$ using master theorem which is less then $\Theta(n^3)$
- So Strassen's algorithm is faster then divide and conquer.
- Ques.1 Find the product of the following matrices using Strassen's algorithm.

$$\begin{pmatrix} 1 & 3 \\ 7 & 5 \end{pmatrix} \begin{pmatrix} 6 & 8 \\ 4 & 2 \end{pmatrix}$$

Thank you

Please send your feedback or any queries to akyadav1@amity.edu, akyadav@akyadav.in or contact me on +91 9911375598