Recursion, Divide and Conquer

whh
29 Jan 2011
Definition of recursion
Definition of recursion

- definition: please see the definition of recursion.
- Recursion is the process of repeating items in a self-similar way.
Writing Recursion

Find the following instance of the recursion:

- Recurrence relation(s) (by how f relates to itself)
- Base case(s) (by when should f not to recur)
Example 1: Factorial (N!)

N! = N * (N-1) * (N-2) * ... * 1

Problem:
Use recursion to find the value of N!
Example 1: Factorial (N!)

Let f(x) = N!, then

Recurrence relation:

\[ f(x) = f(x-1) \times x \]

Base Case:

\[ f(1) = 1 \]

sample code:

```c
int f(int x){
    if (x==1) return 1;
    else return f(x-1) * x;
}
```
Example 2 : Fibonacci sequence

Fibonacci sequence :
1, 1, 2, 3, 5, 8, 13, 21, 34, ......

Problem :
Use recursion to find the k-th number in the Fibonacci sequence
Example 2: Fibonacci sequence

Let \( f(x) \) be the \( x \)-th number in the Fibonacci sequence.

Recurrence relation:
\[
    f(x) = f(x-1) + f(x-2)
\]

Base Case:
\[
    f(1) = 1, \quad f(2) = 1
\]

Sample code:
```c
int fibonacci(int x){
    if (x==1 || x==2) return 1;
    else return fibonacci(x-1) + fibonacci(x-2);
}
```
Little conclusion: why Recursion?

Fact:
1. recursion is slower than iteration.
2. careless recursion will cause stack overflow (Runtime Error).
Little conclusion: why Recursion?

Fact:
1. recursion is slower than iteration.
2. careless recursion will cause stack overflow (Runtime Error).

Reason:
1. Usually a (nice and) well defined function can be easily implemented by recursion
   - Shorter and Clearer code
2. Solving Divide and Conquer Problem!
3. Exhaustion and Searching!
What is Divide and Conquer?

1. **Divide** a problem into small subproblems
2. **Conquer** each subproblems recursively and independently
3. **Combine** solutions of those subproblems to obtain a solution of the original problem.
Example 4: Merge Sort, Quick Sort

Review of the training last week ......
Example 5: Tower of Hanoi

How to move N stacks from Pag0 to Pag2?
Example 5: Tower of Hanoi

Problem:
Move N stacks from Pag0 to Pag2.

Subproblem:
Move N-1 stacks from Pag i to Pag j

Combine:
1. Move N-1 stacks from Pag0 to Pag1 (by sol. of Subproblem)
2. Move 1 stacks from Pag0 to Pag2
3. Move N-1 stacks from Pag1 to Pag2 (by sol. of Subproblem)
Example 6 : Big Mod

Find the value of

\[ R = X^p \mod M \]
Recursion!

Recurrence Relation:
\[ r(x, p, m) = (r(x, p-1, m) \times x) \mod m \]

Base Case:
\[ r(x, 0, m) = 1 \]
Example 6: Big Mod

Divide and Conquer + Recurrence

Recurrence Relation:
\[ \text{temp} = r(x, p/2, m) \]
\[ r(x, p, m) = (\text{temp} \times \text{temp}) \mod m \quad \text{if } p \% 2 == 1 \]
\[ r(x, p, m) = (\text{temp} \times \text{temp} \times x) \mod m \quad \text{if } p \% 2 == 0 \]

Base Case:
\[ r(x, 0, m) = 1 \]
Example 6: Big Mod

Complexity Analysis:

\[ O(P) \text{ vs } O(\lg P) \]
Example 7: L-pieces

How to use L-piece to cover a $2^n \times 2^n$ board?
Example 7 : L-pieces
Example 8: Maximum Sum

Given a sequence of integer:

3, 4, -6, 3, -8, 1, -3, 9, -4

Find maximum sum of all subsequence.
Example 8: Maximum Sum

Problem:
The maximum sum of sequence $a[1..N]$

Subproblem:
The maximum sum of the left half part. $L$
The maximum sum of the right half part. $R$
The maximum sum of the left half part using $a[N/2]$. $LR$
The maximum sum of the right half part using $a[N/2+1]$. $RL$

Combine:
Max Sum = Max{ $L$, $R$, $LR + RL$ }

Return its $LR$ and $RL$ to the parent problem.
Example 8: Maximum Sum

Faster ==> DP method ===> O(N)
Example 9 : Average

Given N

Output permutation \( \{a_1, a_2, \ldots, a_N\} \) of 1, 2, 3, ..., N such that

if \( a_i + a_j = 2a_k \)
then \( a_k \) is not between \( a_i \) and \( a_j \)

Example :
\( N = 5 \)
1 2 5 4 3
Example 9 : Average

\[ N = 4 \quad \begin{bmatrix} 1 & 3 & 2 & 4 \end{bmatrix} \]

\[ N = 5 \quad \begin{bmatrix} 1 & 5 & 3 & 2 & 4 \end{bmatrix} \]

\[ N = 9 \quad \begin{bmatrix} 2 & 6 & 4 & 8 \end{bmatrix} \quad \begin{bmatrix} 1 & 9 & 5 & 3 & 7 \end{bmatrix} \]

\[ N = 8 \quad \begin{bmatrix} 1 & 3 & 2 & 4 \end{bmatrix} \quad \begin{bmatrix} 2 & 6 & 4 & 8 \end{bmatrix} \quad \begin{bmatrix} 1 & 5 & 3 & 7 \end{bmatrix} \]