Divide And Conquer Paradigm

Angel Noé Martínez González

June 25, 2015

The Paradigm I

This paradigm solves a problem **recursively** applying three steps at each level of the recursion

- **DIVIDE** the problem into a smaller sub-problems.
- **CONQUER** via recursive calls.
- **COMBINE** solutions of sub-problems into one of the original problem.

Consider also a **base case** for small enough sub-problems.

The Paradigm II

Insights and Hints

- Sub-problems can be any size: 1/2, 1/3, etc.
- Generally, third step is the key to achieve good performance.
- Base case is often too ingnue.

THE PARADIGM III

MERGE SORT (RETAKE)

MERGE-SORT(A, p, r)

if $p < r$ **then**
$q = \lfloor (p + r)/2 \rfloor$
MERGE-SORT(A, p, q)
MERGE-SORT(A, q, r)
MERGE(A, p, q, r)
end if

MERGE(A, p, q, r)

$B = 1^{st}$ part of array.
$C = 2^{nd}$ part of array.
$i = 1, j = 1$
for $k = 1$ to n **do**
if $B[i] < C[j]$ **then**
$A[k] = B[i++]$
else
$A[k] = C[j++]$
end if
end for

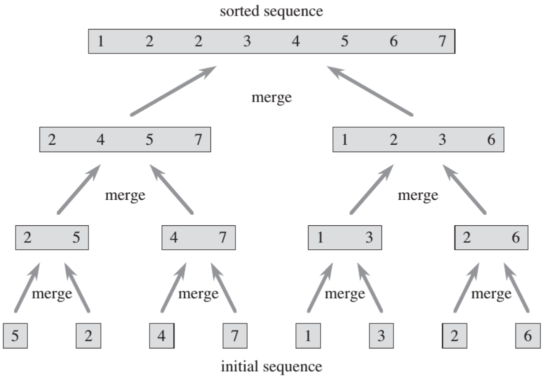
Counting Inversions I

The Maximum Subarray I

BINARY SEARCH I

Recurrence Tree

Merge sort recursion tree



At each level $j = 0, 1, ..., log_2(n)$, there are 2^j subproblems of size $n/2^j$.

The Master Method I

A black box