MergeSort

Ignacio Briones Santander, Ronald Verdugo

Universidad Tecnologica Metropolitana de Chile

26 de Mayo, 2013

Mergesort

- El algoritmo de ordenamiento por mezcla (merge sort en inglés) es un algoritmo de ordenamiento externo estable basado en la técnica divide y vencerás. Es de complejidad O(n log n).
- En las ciencias de la computación, el término divide y vencerás (DYV) hace referencia a uno de los más importantes paradigmas de diseño algorítmico.

Mergesort

- Este método está basado en la resolución recursiva de un problema dividiéndolo en dos o más subproblemas de igual tipo o similar. El proceso continúa hasta que éstos llegan a ser lo suficientemente sencillos como para que se resuelvan directamente.

Ordenamiento externo

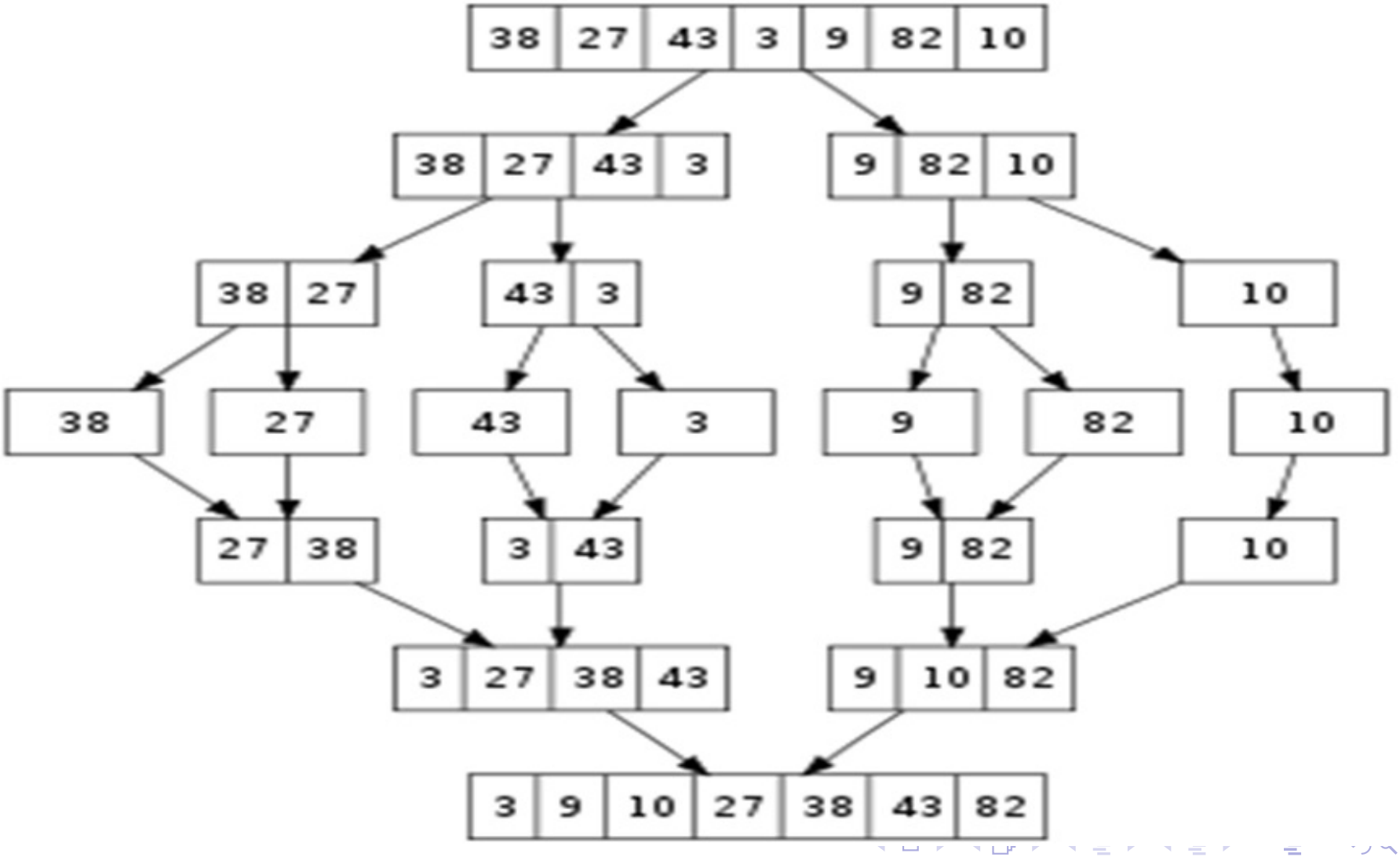
- Ordenamiento externo es un término genérico para los algoritmos de ordenamiento que pueden manejar grandes cantidades de información. El ordenamiento externo se requiere cuando la información que se tiene que ordenar no cabe en la memoria principal de una computadora (típicamente la RAM) y un tipo de memoria más lenta (típicamente un disco duro) tiene que utilizarse en el proceso.

Mergesort y su complejidad O(n log n)

- La Teoría de la Complejidad Computacional es una rama de la teoría de la computación que se centra en la clasificación de los problemas computacionales de acuerdo a su dificultad inherente, y en la relación entre dichas clases de complejidad.

- Un problema se cataloga como "inherentemente difícil" si su solución requiere de una cantidad significativa de recursos computacionales, sin importar el algoritmo utilizado. La teoría de la complejidad computacional formaliza dicha aseveración, introduciendo modelos de cómputomatemáticos para el estudio de estos problemas y la cuantificación de la cantidad de recursos necesarios para resolverlos, como tiempo y memoria.

Algoritmo Mergesort



Complejidad

- La función de complejidad T(n) para el algoritmo mergeSort viene dada por una definición recursiva. En particular, se distingue el tratamiento de listas unitarias y el tratamiento de listas de tamaño mayor o igual que dos.

$$T(n) = \begin{cases} c_1 & si \quad n = 1 \\ 2.T(n/2) + c_2.n & si \quad n > 1 \end{cases}$$

$$(n = 2^k,\ k \geq 0)$$

Algoritmo (pseudocodigo)

```
function mergesort(array A[x..y])
begin
  if ((y-x) > 0):
    array A1 := mergesort(A[x..(int( x+y / 2))])
    array A2 := mergesort(A[int(1+(x+y / 2))..y])
    return merge(A1, A2)
  else:
    return A
end

function merge(array A1[0..n1], array A2[0..n2])
begin
  integer p1 := 0
  integer p2 := 0
  array R[0..(n1 + n2 + 2)]//suponiendo que n1 y n2 son las posiciones
  //del array y no el length de este mismo, de otro modo seria (n1 + n2)
  while (p1 <= n1 and p2 <= n2):
    if (p1 <= n1 and A1[p1] <= A2[p2]):
      R[p1 + p2] := A1[p1]
      p1 := p1 + 1

    else
      if (p2 <= n2 and A1[p1] > A2[p2]):
        R[p1 + p2] := A2[p2]
        p2 := p2 + 1
  return R
end
```