* Activity Selection Problem

* Output an actual LCS

$n = r - p + 1$
$n_1 = q - p + 1$
$n_2 = r - q$

$n_1 + n_2 = n$

RT analysis: Merge algorithm

* counting sort → value of k?

Algorithm $Merge(A, p, q, r)$

```
n1   Copy A[p..q] into L[1..q-p+1]
n2   Copy A[q+1..r] into R[1..r-q]
1    Set sentinels L[q-p+2] = R[r-q+1] = ∞
2    i ← 1;   j ← 1
     for k ← p to r do
             if L[i] ≤ R[j] then
                     A[k] ← L[i];   i ← i+1
             else
                     A[k] ← R[j];   j ← j+1
```

$4n$, 3, 1, 2

$n_1 + n_2 + 3 + 4n$
$= 3 + 5n$
$O(n)$.

$n = r - p + 1$

RT analysis: Partition algorithm

```
?    i ← Random(p, r)
3    Exchange A[i] and A[r]
1    x ← A[r]
1    i ← p - 1
     for j ← p to r - 1 do
1            if A[j] ≤ x then
1                    i ← i + 1
3                    Exchange A[i] and A[j]
3    Exchange A[i+1] and A[r]
1    return i + 1
```

$5(n-1)$, 5

$O(n)$?

$n = r - p + 1$

halted everytime

RT analysis: Binary search

```
BSearch(A, p, r, x) // Search for x in sorted array A[p..r]
while p ≤ r do
    q ← (p + r)/2
    if x < A[q] then
        r ← q − 1
    else if x > A[q] then
        p ← q + 1
    else
        return q // index of x in A
return -1 // return -1 if x is not in A
```

$8 \log_2 n$

$O(\log n)$

$n/2$ $(n/2)/2$ ... $\log n$

$8 \times 8 \times 8$

$2n - 1$

$n = n$

RT analysis: Bubble sort

```
BubbleSort(A, n) // Sort A[1..n]
for i ← 1 to n do
    for j ← n downto i + 1 do
        if A[j] < A[j − 1] then
            Exchange A[j] and A[j − 1]
```

$(n-i)5$

$5(n-i)$

for $k=1$ to n x

$\sum_{k=1}^{n} x = nx$

$O(n)$

$\sum_{i=1}^{n} 1 + 5(n-i)$

$= n + 5n^2 - 5\sum_{i=1}^{n} i$

$= n + 5n^2 - 5n$

$O(n^2)$

$\sum_{i=1}^{n} 5(n-i)$

$= \sum_{i}^{n} 5n - \sum_{i=1}^{n} i$

$= 5\left(\frac{n(n+1)}{2}\right)$

$-5(1+2+\dots)$, $n(n+1)/2$

$5n - 5 + 5n - 5 \cdot 2$

$\sum_{i=1}^{n} 5n - 5i$

$5n \cdot n + c$

$5n^2 + c$

$5n^2 - 5[1+2+3 \dots + n]$

RT analysis: Counting sort

n=n
? key - k

Sort $A[1..n]$ with integer field "key" in range $1..k$:

? Value of k?

```
for j ← 1 to k do
    C[j] ← 0
for i ← 1 to n do
    C[A[i].key]++
for j ← 2 to k do
    C[j] ← C[j] + C[j − 1]
for i ← n downto 1 do
    j ← A[i].key
    B[C[j] − −] ← A[i]
return B
```

2k [1, 1]
3n [1, 2]
3(k-1) [1, 2]
n.5 [1, 2, 2]

$O(k+n)$

RT analysis: Rod-cutting [$(0-\infty)$-knapsack] problem

```
R[0] ← 0
for j ← 1 to n do
    q ← P[j]
    S[j] ← j
    for i ← 1 to j − 1 do
        if P[i] + R[j − i] > q then
            q ← P[i] + R[j − i]; S[j] ← i
    R[j] ← q
return R, S
```

1, 1, 1, 2, 3, 1

$\sum_{j=1}^{n} 2 + 5(j-1)$

$(j-1).5$

$2n + 5\frac{n(n-1)}{2} - 5n$

$O(n^2)$

RT analysis: Activity selection problem

$ASP[0] \leftarrow 0$
for $i \leftarrow 1$ **to** n **do**
  $j \leftarrow i-1$
  while $f_j > s_i$ **do**
    $j \leftarrow j-1$
  if $ASP[i-1] \geq ASP[j] + P_i$ **then**
    $ASP[i] \leftarrow ASP[i-1]$; $Use[i] \leftarrow$ 'N'
  else
    $ASP[i] \leftarrow ASP[j] + P_i$; $Use[i] \leftarrow$ 'Y'; $Pre[i] \leftarrow j$
return ASP, Use, Pre

m = length of first; n = length of second

RT analysis: Longest common subsequence problem

$O(m)$
for $i \leftarrow 0$ **to** m **do**
  $C[i,0] \leftarrow 0$
$O(n)$
for $j \leftarrow 0$ **to** n **do**
  $C[0,j] \leftarrow 0$
$(m+1)m$
for $i \leftarrow 1$ **to** m **do**
  for $j \leftarrow 1$ **to** n **do**
    if $x[i] = y[j]$ **then**
      $C[i,j] \leftarrow C[i-1,j-1]+1$; $B[i,j] \leftarrow$ 'D'
    else if $C[i,j-1] > C[i-1,j]$ **then**
      $C[i,j] \leftarrow C[i,j-1]$; $B[i,j] \leftarrow$ 'L'
    else
      $C[i,j] \leftarrow C[i-1,j]$; $B[i,j] \leftarrow$ 'U'
return C, B

$O(mn)$

m = first length n = second length

RT analysis: Output an actual LCS

```
i ← m
j ← n
k ← C[m, n]
while k > 0 do
    if B[i, j] = 'D' then
        z[k] ← x[i]
        k ← k − 1 ;   i ← i − 1 ;   j ← j − 1
    else if B[i, j] = 'L' then
        j ← j − 1
    else
        i ← i − 1
return z
```

k = C[m, n] = Value at

O(k)

~ linear?

RT analysis: Huffman coding

Input: Alphabet Σ, frequency of each character. Let $n = |\Sigma|$

1. For each character $c \in \Sigma$, create a tree with a single node containing element c, with priority $f(c)$
2. Create a priority queue with these n trees
3. While the priority queue has more than one tree, remove two least fequent trees x and y, merge them together, and insert the resulting tree into the priority queue, with priority $f(x) + f(y)$
4. Return the single tree in the priority queue

O(n) O(n) O(n log n) n O(log n) O(1)

O(n log n)

$n = |V|$

RT analysis: APSP by extension

for $u \in V$ **do**
    for $v \in V$ **do**
        $D[u,v] \leftarrow w(u,v)$
$k \leftarrow 1$
while $k < |V| - 1$ **do**
    for $u \in V$ **do**
        for $v \in V$ **do**
            $T[u,v] \leftarrow D[u,v]$
            for $m \in V$ **do**
                if $T[u,v] > D[u,m] + D[m,v]$ **then**
                    $T[u,v] \leftarrow D[u,m] + D[m,v]$
    $D \leftarrow T$
    $k \leftarrow 2 * k$ $-(\log n)$
return D

$O(n^2)$ $O(n^3 \log n)$ $O(n^3)$ $O(n)$

$O(n^3 \log n)$

$n = |V|$

RT analysis: Floyd-Warshall's algorithm for APSP

for $u \in V$ **do**
    for $v \in V$ **do**
        $D[u,v] \leftarrow w(u,v)$
for $k \leftarrow 1$ **to** $|V|$ **do**
    for $u \in V$ **do**
        for $v \in V$ **do**
            $D[u,v] \leftarrow \min(D[u,v], D[u,k] + D[k,v])$

n^2 n $3n^3$ $3n^2$ $3n$

$3n^3 + n^2$

$O(n^3)$

RT analysis: KMP algorithm for string matching

$\pi[1] \leftarrow 0$
$k \leftarrow 0$
for $q \leftarrow 2$ **to** m **do**
    while $k > 0$ **and** $P[k+1] \neq P[q]$ **do**
        $k \leftarrow \pi[k]$
    if $P[k+1] = P[q]$ **then**
        $k \leftarrow k+1$
    $\pi[q] \leftarrow k$
return π

$\pi[k]$: ? order

$0 < \pi[k] < K$

$O(km)$

RT analysis: Counting sort

```
for(q=1; q<=k; q++) {
   L[q] = new LinkedList();
}
for(i=1; i<=n; i++) {
   L[A[i].key].add(A[i]);
}
k = 1;
for(q=1; q<=k; q++) {
   for(x: L[q]) {
      A[k] = x;
      k++;
   }
}
```

$O(k)$
$O(n)$
$O(k^2)$
$O(k^2+n)$