TRANSFORM – CONQUER

Thursday, October 4, 2018 5:13 PM

IDEA: "massage" problem into a known, solved problem.

HOW?

1) INSTANCE simplification: reduce a given instance to a simpler instance that is easy to solve

Ex: input: A[lo..hi]
output: true if all elements in A[lo..hi] are different.

brute:
```
for (i = lo; i <= hi; ++i)
  for (j = i+1; j <= hi; ++j)
    if A[i] == A[j]
      return false;
return true;
```
$\Theta(n^2)$

transform: this problem can be solved in linear time IF the array is sorted:
then are duplicates $\Leftrightarrow$ A[i] == A[i+1] for some i

```
sort(A[lo..hi]);                 Θ(n lg n)
for (i = lo; i < hi; ++i)        Θ(n)
  if A[i] == A[i+1]
    return false.
return true;                     Θ(n lg n)
```

Ex: INPUT: A[lo..hi], x
OUTPUT: true if x == A[j] some $lo \le j \le hi$

brute: scan the array $\Theta(n)$

transform: if A[lo..hi] is sorted, then binary search takes $\Theta(\lg n)$

Ex: RACE & CARE are ANAGRAMS because they are composed of the same set of letters

INPUT: a dictionary

INPUT: a dictionary

OUTPUT: partition the dictionary into sets of anagrams

Brute:

```
for each word w in Dictionary
    for each word x > w
        if x == permutation of w
            insert x into S[w]
```

Transform:

```
for each w in Dict
    compute SIGNATURE
```

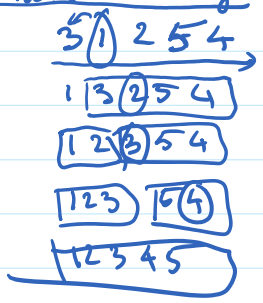
CARE → ACER

⋮

RACE → ACER

sort dictionary by signature (anagrams form blocks)

2) Representation change:



selection $\Theta(n^2)$

Heap Sort: selection sort where unsorted array is organized into a min-heap

make-heap: $\Theta(n)$

find-min: $\Theta(1)$

delete-min: $\Theta(n)$

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

$$= \Theta(n \lg n)$$

3) Reduction: change problem to a DIFFERENT solved problem

Ex: LCM

input: 2 nonnegative integers a, b

OUTPUT: smallest common multiple of a & b

Ex:

lcm(2, 3) = 6

lcm(4, 14) = 28

Reduction: $ab = gcd(a,b) * lcm(a,b)$

$$\frac{ab}{\gcd(a,b)} = \operatorname{lcm}(a,b)$$