Com S 311 Section B
Introduction to the Design and Analysis of Algorithms

Xiaoqiu (See-ow-chew) Huang

Iowa State University

January 26, 2021

Instructor's Teaching and Research

Taught 228 nine times and bioinformatics courses several times.

Developed several algorithms and programs for reconstruction and analysis of genome sequence data.

One of them is Global Alignment Program 3 (GAP3).

GAP3 takes (as input) two DNA sequences and produces (as output) an ordered list of similarity blocks separated by different sequence regions, with each block showing similar regions.

Algorithm design techniques such as dynamic programming and divide-and-conquer are used in GAP3 to compute an optimal solution in $O(n^2)$ time and $O(n)$ space.

Algorithms are Useful in Data Science

Consider applications in reconstruction and analysis of fungal genome sequence data.

Raw datasets of short reads (each 1-20 GB in size) are generated for a number of isolates in the fungal species complex *Fusarium oxysporum*.

Each dataset is assembled by a computer program into a file of long sequences (50 MB in size) for each isolate on a compute node with 100 CPUs.

The files of long sequences for these isolates are analyzed by computer programs to understand the evolutionary histories of and genetic variations in these isolates.

These computer programs are developed by using algorithm design techniques in Com S 311.

Syllabus

The course syllabus contains a lot of information about this course.

Please read it carefully.

Insertion Sort

Input: An array A of n items that can be compared.

Output: An array A of items in ascending order,

$A[1] \leq A[2] \leq A[3] \leq ... \leq A[n]$.

Idea: For each j from 2 to n, place $A[j]$ at a proper place in $A[1]$ through $A[j]$.

```
INSERTION-SORT(A)

0      n = A.length
1      for j = 2 to n  // j goes from 2, 3, ... up to n
2        key = A[j]
3        // Insert A[j] into the sorted sequence A[1..j-1].
4        i = j - 1
5        while  i > 0 and A[i] > key
6           A[i+1] = A[i];
7           i = i - 1
8        A[i+1] = key;

    A:    8 | 5   1   3   6
index:    1   2   3   4   5
     j        2

    A:    5   8 | 1   3   6
index:    1   2   3   4   5
     j            3
```

```
    A:   1   5   8 | 3   6
index:   1   2   3   4   5
     j               4

    A:   1   3   5   8 | 6
index:   1   2   3   4   5
     j                   5

    A:   1   3   5   6   8 |
index:   1   2   3   4   5
     j                       6
```

Loop Invariant

Loop invariants are used to show that algorithms are correct.

A loop invariant for Insertion Sort:

At the start of each iteration of the outer loop, the subarray $A[1]$ through $A[j-1]$ consists of the elements originally in the subarray, but in sorted order.

Justification

Initialization: It is true when $j = 2$, since the subarray $A[1..j-1]$ consists of the only element $A[1]$, which is sorted.

Maintenance: Assume that the loop invariant is true at the start of an iteration of the outer loop. We want to show that it is true at the start of the next iteration.

By the assumption, $A[1..j-1]$ is sorted. Lines 4-8 inserts $A[j]$ into the sorted sequence $A[1..j-1]$.

Justification

If $A[j] \geq A[j-1]$, then by the assumption that $A[1..j-1]$ is sorted, we conclude that $A[j]$ is equal to or larger than every element in $A[1..j-1]$. So line 5 is false with $i = j-1$, and $A[j]$ is placed at position $i+1 = j-1+1 = j$.

If $A[j] < A[1]$, then by the assumption that $A[1..j-1]$ is sorted, $A[j]$ is smaller than every element in $A[1..j-1]$. So line 5 is true for each $i > 0$, and $A[j]$ is placed at position $i+1 = 0+1 = 1$, with the previous $A[1..j-1]$ moved to $A[2..j]$.

Otherwise, $A[j] \geq A[k]$ and $A[j] < A[k+1]$ for some k, $1 \leq k \leq j-2$. By the assumption that $A[1..j-1]$ is sorted, $A[j] \geq A[h]$ for each h, $1 \leq h \leq k$, and $A[j] < A[h]$ for each h, $k+1 \leq h \leq j-1$. So $A[j]$ is placed at position $i+1 = k+1$, with the previous $A[k+1..j-1]$ moved to $A[k+2..j]$.

Termination: At the end of last iteration with $j = n$, $A[1..n]$ consists of the elements in the array, but in sorted order.