IS52038B: Algorithms and Data Structures

Volkan Kunduru

March 23, 2018

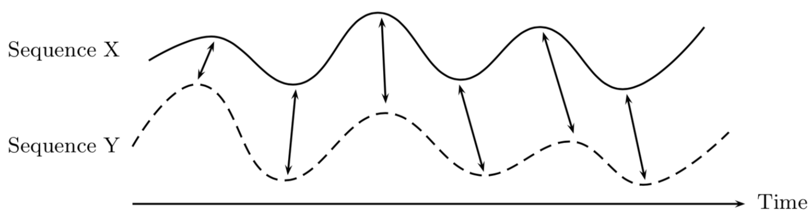
1 Dynamic Time Warping

1.1 What Is It?

Dynamic Time Warping is used to measure sequences that vary in speed which happen in the space of time.(Müller, 2007) This algorithm exploits distortions between two sequences, it checks the alignment in between the two sequences and works out the most cost-efficient sequential distance it takes from start to finish. The main goal of DTW(Dynamic Time Warping) is to compare the sequences by looking for the best alignment using a distance matrix. (Akshaya, Akshaya, Gayathri, & Saravanan, 2016)The working complexity of this algorithm is $O(N^2)$.

(Müller, 2007)

Figure 1: Comparing two sequences in real time, the arrows show the points of alignment.



””

1.2 How Does It Work?

Given the sequences Xi(length of N) and Yj(length of M), that i,j are indexes of integers for sequences X, Y; the shortcut of the formulae is denoted as (Kaleka, n.d.):

$$d(i, j) = f(xi, yj) \geq 0$$

Given that d the distance matrix between X and Y is the only input to the algorithm and the indexes i and j only enter through the function f(Giorgino et al., 2009).

$$|Xi - Yj| + minimum[i-1, j-1][i-1, j][i, j-1]$$

To work out the matrix distance, you first work out the sum of subtracting the two indexes of sequences X,Y and then find the previous inputs and compare them by size; whichever one is the *lowest cost* is always the most preferred. The aim is to keep the overall cost as low as possible(Vasimalla, Challa, & Naik, 2016).

1.3 Where Could It Be Applied To?

DTW is widely applied to is in Speech Recognition: Pattern matching. Pattern matching works by comparing incoming speech patterns with existing patterns already stored in memory. The variation of pattern matching that implements DTW is called word-by-word pattern matching. This is because DTW is widely used in isolated word speech recognition systems. DTW is said to be the most successful algorithm for speech recognition due to the fact that it looks for the best optical alignment between sequences.

1.4 Related Algorithm

An algorithm DTW relates to that we learnt about this module is Dynamic Programming(DP):Minimum Edit Distance. Minimum Edit Distance is the process of transforming one string into another. To find the Minimum Edit Distance you compare the letters in a distance matrix, how ever many operations it takes to get from one letter to another is recorded on the matrix and once the process is completed you trace your steps diagonally from the end to the beginning converting the initial string to the goal string. This relates to DTW because both algorithms use a matrix to compare two sequences and both work with the distance from one sequence to another. (Bille, 2003)

1.5 Quiz Style Question

Computing Abstract Dynamic Time Warping requires the working complexity of:

(a) $O(N^2)$

(b) $O(N^3)$

(c) $O(Nlog(N))$

(d) $O(1)$

Answer: $O(N^2), O(N^3)$

2 Data Structure - Rope

2.1 What Is It?

The Rope data structure was introduced in 1999 as an alternative to a traditional String. Rope is composed of smaller strings and is used to store and manipulate a very long string. Strings operate using concatenation, appending to the end of a string takes time in $O(N)$ whereas with Ropes you are able to reduce the time taken to append $O(log(N))$. (Boehm, Atkinson, & Plass, 1995) Ropes doesn't require additional memory to perform operations such as insertion/deletion/searching/split which makes it more cost efficient. If the user wants to undue the last concatenation made it would simply take $O(1)$ time just by removing the root node from the tree.

2.2 How do we Construct the data structure?

A Rope is a binary tree structure where each node minus the leaf node contains a short string. The idea behind it is to store the number of characters to the left to decrease the cost of finding the character present at the ith position. Implementing Rope is simple; Firstly, you create a new node that stores the root of the string then you mark the left and right children of this node. The left-most is made the root of the string that appears first and the right-most node contains the root of the string that appears second.

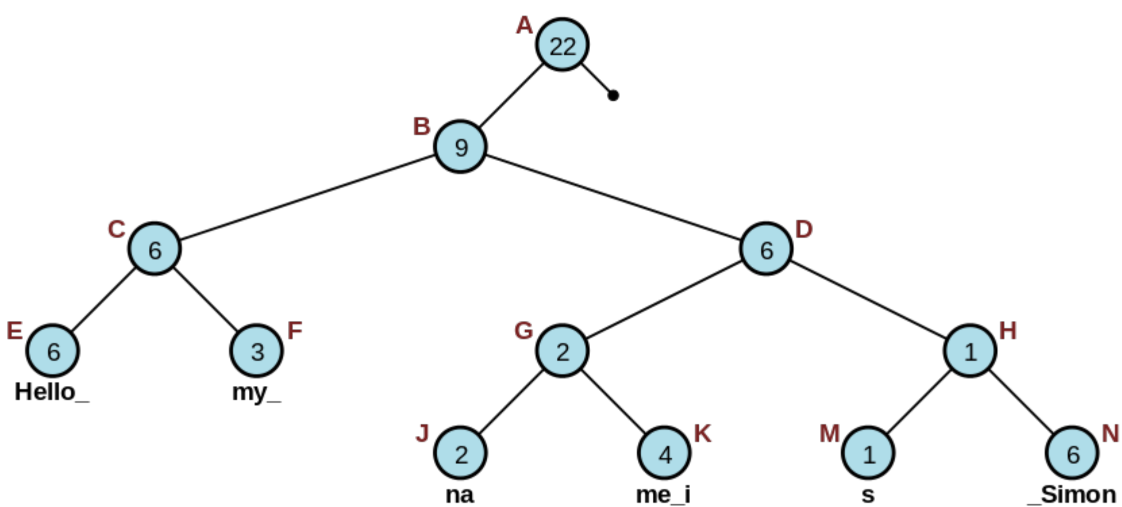
The formula:

$$size(v) = \sum_{u \in T(v)} weight(u) \tag{1}$$

T is a binary tree in which each node v stores two additional values of weight(v) and size (v).Weight(v) is assigned to a node as a number when it is given a value. It can be modified at any time. size(v) is the sum of all weight values in the subtree at root (v).

(Geeks, n.d.)

Figure 2: A simple rope built on the string of "Hello my name is Simon".



2.3 Where could this be applicable?

The Rope data structure is used mostly in text editing applications due to the fact that it allows for operations such as insertion, deletion and random access on large strings efficiently $\theta(1)$. Rope doesn't require $O(n)$ extra memory when operated upon.

2.4 Related Data Structure

The most closely related Data Structure that relates to Rope that i have learnt about this year is Strings. A String is a linear collection of text. How Rope works is by operating on strings to allow for a large string to be manipulated upon.(Teachings, n.d.)

2.5 Quiz Style question

Given the operations Concat, Split, Insert, Delete and Report. What is the time complexity of the operation Concat?

(a) $O(logN)$
(b) $O(N)$
(c)$O(j + logN)$
(d)$O(1)$
(e)$O(N^2)$
(f)$none of the above.$
Answer: $O(1), O(logN)$

References

Akshaya, B., Akshaya, S., Gayathri, S., & Saravanan, P. (2016). Investigation of bi-max algorithm for online purchase recommender system using social networks. *Indian Journal of Science and Technology*, *9*(44).

Bille, P. (2003). *Tree edit distance, alignment distance and inclusion* (Tech. Rep.). Citeseer.

Boehm, H.-J., Atkinson, R., & Plass, M. (1995). Ropes: an alternative to strings. *Software: Practice and Experience*, *25*(12), 1315–1330.

Geeks, G. F. (n.d.). *Ropes data structure (fast string concatenation).* Retrieved from `https://www.geeksforgeeks.org/ropes-data-structure-fast-string-concatenation/`

Giorgino, T., et al. (2009). Computing and visualizing dynamic time warping alignments in r: the dtw package. *Journal of statistical Software*, *31*(7), 1–24.

Kaleka, J. S. (n.d.). Isolated word recognition using dynamic time warping.

Müller, M. (2007). Dynamic time warping. *Information retrieval for music and motion*, 69–84.

Teachings, M. (n.d.). *Data structures for strings.* Retrieved from `http://cglab.ca/ morin/teaching/5408/notes/strings.pdf`

Vasimalla, K., Challa, N., & Naik, S. M. (2016). Efficient dynamic time warping for time series classification. *Indian Journal of Science and Technology*, *9*(21).