CSCI432 HW5

Brock Ellefson

October 25, 2017

1 Prove that the Frechet distance is a distance metric

Prove that the Frechet distance is a distance metric.

To be a distance metric, you must satisfy 4 requirements:
Let X = discrete space
A metric is a function d: X x X $\rightarrow \mathbb{R}$ such that:

1. d(x,y) = d(y,x)
2. d(x,y) = 0 $\Leftrightarrow$ x = y
3. d(x,y) + d(y,z) $\geq$ d(x,z)
4. d(x,y) $\geq$ 0

So,
Let A, B be two curves in a metric space, and t as a parameter in time. A and B are the infimum of all α and β of [0,1] where t $\in$ [0,1]
Then the Frechet Distance is:
$F(A,B) = inf_{\alpha,\beta} max_{t\in[0,1]}\{d(A(\alpha(t)), B(\beta(t)))\}$.

So,
$F(A,B) = inf_{\alpha,\beta} max_{t\in[0,1]}\{d(A(\alpha(t)), B(\beta(t)))\}$, and
$F(B,A) = inf_{\beta,\alpha} max_{t\in[0,1]}\{d(B(\beta(t)), A(\alpha(t)))\}$.

2 Recurrence Relations

2.a T(n) = 2T(n/4) + n^2

Master's Theorem:
T(n) = aT($\frac{n}{b}$) + f(n)
Where, a $\geq$ 1, b > 1, f(n) is asymptotically positive

T(n) = 2T(n/4) + n^2

a = 2, b = 4, f(n) = n^2
$n^{log_b a} \Rightarrow n^{log_4 2} \Rightarrow n^{1/2}$
Case 3:
if f(n) is $\Omega(n^{log_b a+\epsilon})$ for some $\epsilon > 0$ and if f(n/b) $\leq$ f(n), then T(n) = $\theta(f(n))$
Therefore, T(n) = $\theta(n^2)$

2.b T(n) = 4T(n/2) + n

Master's Theorem:
T(n) = aT($\frac{n}{b}$) + f(n)
Where, a $\geq$ 1, b > 1, f(n) is asymptotically positive

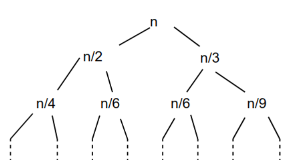
T(n) = 4T(n/2) + n
a = 4, b = 2, f(n) = n
$n^{log_b a} \Rightarrow n^{log_2 4} \Rightarrow n^2$
Case 1:
if f(n) = $O(n^{log_b a-\epsilon})$ for some $\epsilon > 0$ then T(n) = $\theta(n^2)$.
Therefore, T(n) = $\theta(n^2)$

2.c T(n) = 3T(2n/3) + 4n

Master's Theorem:
T(n) = aT($\frac{n}{b}$) + f(n)
Where, a $\geq$ 1, b > 1, f(n) is asymptotically positive

T(n) = 3T(2n/3) + 4n
a = 3, b = 2/3, f(n) = 4n
$n^{log_b a} \Rightarrow n^{log_{3/2} 3} \Rightarrow n^{2.71}$
Case 1:
if f(n) = $O(n^{log_b a-\epsilon})$ for some $\epsilon > 0$ then T(n) = $\theta(n^{2.71})$.
Therefore, T(n) = $\theta(n^{2.71})$

2.d T(n) = T(n/2) + T(n/3)



Our longest path in this tree is the leftmost path, following a sequence: $log_2 n$.
So our initial guess is for this recurrence is $O(nlogn)$.

2.e 2T(n/2) + O(log n)

3 Climbing Stairs Problem

Algorithm 1 Climbing Stairs

Input: n (number of stairs), k (max steps can advance at once)
Output: Integer (Numbers of ways to reach destination)
Procedure: $CountingStairs(n, k)$
count $\leftarrow$ 0
if n $\leq$ 1 **then**
    return 1
end if
for $i \leftarrow 1...(n)$ **do**
    $count \leftarrow COUNTINGSTAIRS(n - i, k)$
end for
return count