CS350 homework 1

Yuchen Hou

September 9, 2015

1 Language

1.1

{'(n, p, q)' | n.getType() == Number and p.getType() == Prime and q.getType() == Prime and n == pq}

1.2

{'n' | ∃(p, q) n == pq and p.getType() == Prime and q.getType() == Prime}

1.3

{'(A, w)' | A.getType() == NFA and w.getType() == Word and A.accept(w)}

1.4

{'A' | ∃w A.getType() == NFA and w.getType() == Word and A.accept(w)}

2 Big O

2.1 $f(n) = 2n^3 - 18n$

$$\begin{cases} 2n^3 = O(n^3) \\ -18n = O(n) \end{cases} \implies f(n) = O(n^3) \implies f(n) = O(n^4)$$

$O(log(n))$ grows exponetially slower than $O(n)$
$\implies O(n^2 log(n))$ grows exponentially slower than $O(n^3)$
$\implies f(n)! = O(n^2 log(n))$

2.2 $f(n) = 3n^2 2^{2n}$

a.e. $f(n) < 2^n 2^{2n} = 2^{3n} \implies f(n) = 2^{O(n)}$

3 Probelm definition

A map can be an instance of a graph where a road is an edge and a junction is a node; a molecular structure can be an instance of a graph where a bond is an edge and an atom is a node. Then the similarity measure for maps or molecular structures is implemented as a similarity function of the graph class. One of the most important attribute of a graph is its shortest paths between all node pairs, so the similarity of 2 graphs can be implemented as the similarity of their shortest paths. The shortest paths of a graph can be represented as a dictionary of dictionary of lists of nodes. The key to the first dictionary is the source node, the key to the second dictionary is the destination node, the list is the path(the node sequence) from source to destination. So the similarity of the shortest paths can be implemented as the similarity of the lists. The lists of nodes can be represented as strings. Then the similarity of the lists can be implemented as similarity of the strings, which has many existing algorithms, e.g., Hamming distance. In this way, the measure of graph similarity is reduced to string similarity. Therefore, the measure can be defined as the normalized sum of shortest path string similarities.