Anders Maraviglia

Intro to Algorithms, 5/4/2014

Homework 3

6.4)a) procedure canBeReconstituted(string s[n]){

Foreach character position p in s{

If(dict(s[0, p]) is true):

If(p = n):

Return true

return canBeReconstituted(s[p, n])

}

Return false

}

b) procedure reconstituteString(string s[n]){

if(!canBeReconstituted(s)):

return false

let startPos = 0

foreach character position p in s{

if(dict(s[startPos, p])){

print s[startPos, p] and “ “ //print the word and a space

startPos = p+1

} } }

6.5)a) There are 3 legal patterns that can occur in any column, where pebbles occupy the first and third rows, the second and forth rows, and the first and last rows.

b) procedure optimalPlacement(board[n]){

let prevType start as the optimal pattern for the first column in board

let currType = NULL

foreach column col in board from 1 to n{

set currType to the optimal pattern compatible with prevType

set prevType to currType

} }

6.10) procedure countingHeads(n, k, p[n]){

Let headCount = 0

Foreach probability prob in p:

If(prob == 1):

Increment headCount

Return 1 -

}

6.17) procedure makeChangeP(x[n], v){

Let greatestDenomination = 0

Foreach denomination d in set x[n]:

If(d > greatestDenomination and v – d > = 0):

greatestDenomination = d

if(v – greatestDenomation > 0):

return makeChange(x[n], v – greatestDenomination)

else if(v – greatestDenomation == 0):

return true

return false

}

8.3) STINGY SAT is a generalization of the SAT problem, since if you’re given one of those formulas with a finite number of variables x, the formula and x are an instance of STINGY SAT if and only if that original SAT formula satisfied the assignment.

8.4) a) When you are given a graph with a clique in it, verification in polynomial time that there is an edge between every vertex is simple. Thus CLIQUE-3 is verifiable in polynomial time.

b) CLIQUE must first be reduced to CLIQUE-3 in order to show that the latter is at least as difficult to solve as the former, thus showing that the reduction is in the wrong direction.

c) C is only a vertex cover if V – C is an independent set in G, meaning the stance made in the reduction that “a subset C in V is a vertex cover in G if and only if the complimentary set VC is a clique in G” is wrong.

d) Every vertex in a clique of size k must have a degree of k – 1, thus the largest clique in G must be at least 4. Therefore there are no solutions for any k > 4, and for any k less than this we can check every k-tuple of vertices in O(V4) time.