

**CISC 603-51- A-2021/SUMMER - THEORY OF COMPUTATION**

**Assignment - 6**

**Computational Complexity**

**P, NP, and NP-Complete Problems**

**By,**

**Satheesh Swaminathan**

**Student ID:** **266582**

**14.1 #1**

A. Algorithm to check if there are any duplicates, consider the first element of the input string x1 if x1 is found on the input tape, move q0 to q1 else stay on q0.

For the next input x2 if found on tape move form q1 to q2 else loop on the same state for the input corresponding to state q1.

If any duplicates are found move back to the initial state q0, repeat this algortihtm until the end of test string.

If there are no duplications, we move to the final state lets say qn

**Time complexity:**

**Worst case** is a successful string read, which is O(n\*n) since we have to read through entire string and compare each with other.

**Best case** would be to find the duplicate at the very beginning making it O(1)

Thus overall complexity is O(n2)

B. This algorithm accepts strings

1234 – no duplicates

1213 – duplicate found, but algorithm/turing machine handles it correctly.

**14.1 #2**

Consider xi, if xi is found on input tape move from q0 state to q1

If the next input is not from xi then loop on the same state q1.

If the inout is same as xi move from q1 to q2 (since its not a triplicate yet) and q2 is not final state.

If input is blank, reset to initial state q0

If input is same as xi move from q2 to q3 which is the final state q4.

Repeat previous steps to completely read the test string.

Time complexity is O(n2) since the algorithms has to read through the entire test string and compare each one with other.

**14.2 #1**

Using two tape turing machine

If the total number of elements in test string is even, length of first tape is n/2 and n/2 for the second state.

If number of elements is odd, then first tape length will be (n-1)/2 and the rest will be on tape 2

Read test strings and move from q0 to q1, compare elements from both tapes

If both tapes has same values as the test string, move to final state q2.

Else loop on the same state q1 until the algorithm exits.

For 2 tape turing machine, for this scenario the time complexity would be O(n) making this a linear search

**14.2 #2**

Single tape, off-line turing machine has 2 way input tape, so it can read from either directions left and right

single-tape off-line Turing machine test string will be read from left to right takes n moves, and from right to left it takes another n moves making the complexity O(T(n)).

Even the standard turnig machine performs the same way providing complexity O(T(n)). It mainly depends on the way the input test strings are passed to the turing machine.

**14.2 #3**

The semi infitie tape divides the tape in two parts making this similar to the 2-tape turing machine allowing it to move in either directions.

Standard turing machine moves from start to end and takes n moves making complexity O(T(n))

For semi infinite tape, takes n/2 the next move in other half of n/2. Making the time complexity O(T(n)).

**14.2 #4**

Chomsky Normal Form is expression

For variables x1,x2,3.....xn

CNF for (x1 ∧ x2) ∨ x3 is

(x1 ∨ x3) ∧ (x2 ∨ x3)

**14.3 #2**

Since the turing machine can read inputs on either directions, Input symbols into first tape, reverse of input symbols into second tape and the outputs are sent into 3rd tape.

Copy input string into tape 1, using two-tape deterministic machine Reverse input string is written into tape 2.

Separate input string in tape 1 by special symbol %, and compare input string before and after % one by one.

Once compare is done, then deterministic Turing machine stop at final state.

Time complexity is O(T(n)) time to reach final.

**14.3 #3**

Using only 2 tapes

Input is written on tape 1 and output Is written on tape 2, using 2 tape determnisitic turning machine, write input on tape 1 seperated by special symbol % , #

Now the turing machine reaches final state

It would still make the complexity O(T(n)) which makes it DTIME(n)

**14.5 #1**

For m < n in the CNF of length n and m literals.

Subscript cannot be greater than m making complexity log2m

Best case provides complexity O(nlogn).

Using Non deterministic turning machine gives complexity of O(n).

Therefore 2n can be generated.

**14.5 #2**

CNFwith length n2 and m literals, subscript cannot be greater than m, best case complexity will be log2m and n symbols O(n2log n)

2 tape non deterministic turing machine the same complexity can be achieved

Therefore O(n2logn) complexity

**14.6 #2**

5 literal CNF canbe reduced to 3 SAT form.

If SAT is satisfiabe 3SAT represenedt by E2 will also be satisfiable and vice versa

Thus SAT can be reduced to 3SAT in polynomial in time.

**14.7 #1**

Hamilton path go through each vertex only once. If the graph has e edges, the Hamilton path can move to each edge can take different paths other than edges or sides.

The sum of weights of edges lets say l, the weight of Hamilton path cannot be greater than l, thus we can say that TSP can be preformed in polynomial time.

**14.7 #2**

Euler circuit includes all edges. If the graph contains v vertex and e edges, takes a total of e moves to pass through each edge, but it is possible to navigate to an edge using sides too therefore making the ciruict decidable.

Cannot be represented as polynomial in time, hence it not np-complete.

**14.7 #3**

HAMPATH, SAT, 3SAT and CLIQ are NP-complete.