Claim : We have already proved in earlier course that having more tapes does not increase the power of a Turing Machine, i.e. A k-tape Turing Machine may be simulated by a one-tape Turing Machine with some extra polynomial time.

Given that L can be completed in time T(n) where n = |x|.

Let M be a machine running in time T(n) on all inputs of size n such that L = L(n); i.e. whenever input x ϵ L, M says yes and whenever input x !ϵ L, M says no.
(Assumption : M is a sigle tape Turing Machine since multiple tapes don't give extra power).
This existence of M is guranteed because it is given that L can be computed in time T(n) where n = |x| and T(n) is a time-constructible function.

So, we have to come up with with an oblivious Turing Machine N such that L(M)=L(N) and N runs in $O(T(n)^2)$.

The machine N will have 3 tapes; one to simulate tape of M with an additional marker on the tape for the location of the simulated head of M, one counter for how many steps of M have been simulated, and one additional counter. For simulating step i of computation of M, N makes two passes over M tape from location 1 to location i and back. (The additional counter is used to know when to turn around). Since head of M moves only one cell per step (and that too right/left since M is our old (normal)Turing Machine). Hence, upto step i, head of M would have visited atmost 1st i cells; hence it will point to the jth cell, 1<=j<=i. Hence, when N makes two pass over the 1st tape (tape of M), it must observer the head of M twice – once while going forward upto location i and again while coming back to 1.

While moving forward, N sees the current tape symbol under head of M and on the way back, it implements the step of M, i.e. over-write the current tape symbol with the new symbol and move the marker (denoting location of simulated head of M) right or left on the 1st tape as δ demands.

When N has finished the ith step, i.e. Came back to location 1 on tape 1, it increments the counter on the 2nd tape and 3rd tape representing how many steps of M' has been simulated.

Clearly, N will simulate M. After T(n) steps if M is in final state, N will say “Yes” otherwise it will say “No”.

N is oblivious because head movements of N (from location 1 to i and back to 1) are not dependent on actual input (In this implementation, they depend only on i(step number) not even on the input length).

Simulating step i of M takes O(i) time (moves from location 1 to i and back) + O(1) increment counter on 2nd and 3rd tape so, total time to simulate T(n) steps is $O(T(n)^2)$.

The 3 tape machine can be implemented by a single tape in $O((k * T(n))^2)$ (which is same as $O(T(n)^2)$ since k is constant) if one appends the 2nd and 3rd tape are appended after the 1st tape.
Each tape contains only O(T(n)) symbols since :
Tape 1 : Different cells M can visit during its run = T(n)
Tape 2 : Since M runs for T(n) steps hence counter needs to be incremented upto T(n). I denote the counter in unary notation, hence max cells required = T(n)
Tape 3: Same as tape 2

We also need some extra space to store the state of machine after step i (same as step of M after ith step) = log(|Q|) = O(1) cells where Q = Set of states
We also implement the step i of M while simulating it hence we need to store δ on the tape in the form of || (q, a) | (p,b) | R|......|| a state can be encoded in log(|Q|) bits , a tape symbol in log(|T|) bits (T: Set of tape symbols) and Right/Left in 1 bit.

In short these information can be stored in const space (say between tape 1 and tape 2) but it won't affect asymptodic time of simulation since now $O((T(n) + b)^2)$ is same as $O(T(n)^2)$ if b is constant.

Thus we have shown that an oblivious TM can decide L in $O(T(n)^2)$ if it can be decided by a normal (old) TM in T(n) time where T(n) is some time-constructible function

– Shashank Rai (11010162)
Vishal Anand(11010170)
Vishawadeep Mattu(11010171)