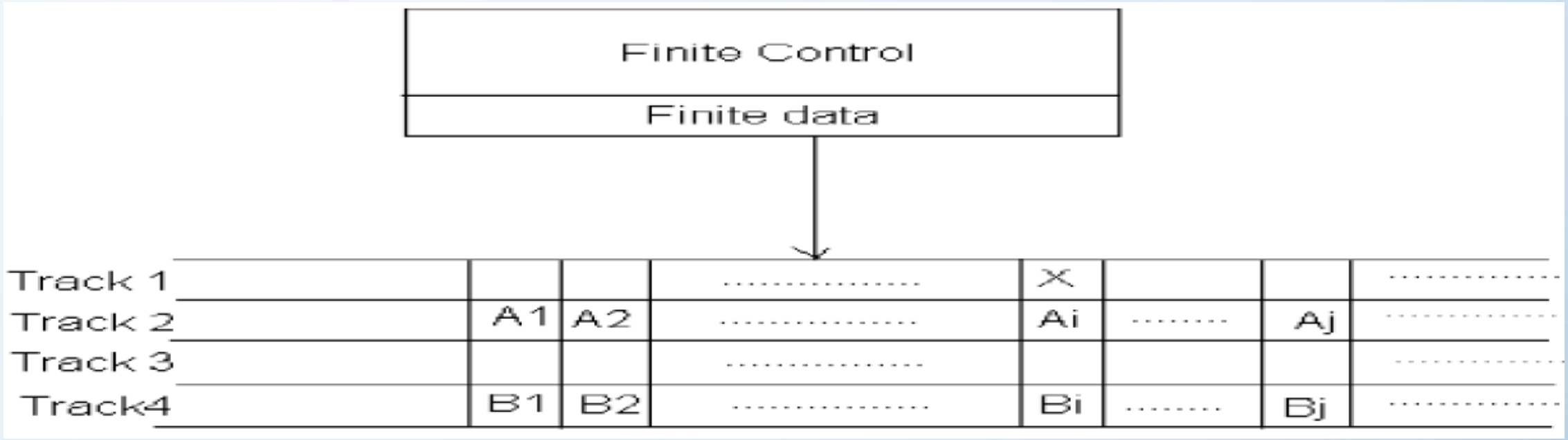
# CSC-257
# Theory Of Computation
## (BSc CSIT, TU)

Ganesh Khatri
kh6ganesh@gmail.com

# Equivalence of one-tape and multi-tape TM

- **Theorem :** Every language accepted by a Multi-tape TM is recursively enumerable. Or, Any languages that are accepted by a multi-tape TM are also accepted by one tape Turing Machine.
- **Proof :** Let L is a language accepted by a n-tape TM, $T_1$= (Q1, Σ, Γ1, δ1, q1, B, F1).
- Now, we have to simulate $T_1$ with a one-tape TM, $T_2$ = (Q2, Σ, Γ2, δ2, q2, B, F2) considering there are 2n tracks in the tape of $T_2$.
- For simplicity, let us assume n = 2, then for n > 2 is the generalization of this case.
- Then total number of tracks in $T_2$ will be 4.
- The second and fourth tracks of $T_2$ hold the contents of first and second tapes of $T_1$.
- The track1 in $T_2$ holds head position of tape 1 of $T_1$ and track3 in $T_2$ holds head position of tape2 of $T_1$

# Equivalence of one-tape and multi-tape TM



# Equivalence of one-tape and multi-tape TM

- Now, to simulate a move of $T_1$ with $T_2$,
  - $T_2$'s head must visit the n-head markers so that it must remember how many head markers are to its left at all times. That count is stored as a component of $T_2$'s finite control
  - After visiting each head marker and storing the scanned symbol in a component of its finite control, $T_2$ knows what tape symbols are being scanned by each of $T_1$'s head
  - $T_2$ also knows the state of $T_1$, which it stores in $T_2$'s own finite control. Thus $T_2$ knows what move $T_1$ will make
  - $T_2$ now revisits each of the head markers on its tape, changes the symbol in track representing corresponding tapes $T_1$ and moves the head marker left or right, if necessary.
  - Finally, $T_2$ changes the state of $T_1$ as recorded in its own finite control.
  - Hence $T_2$ has simulated one move of $T_1$.
  - We select $T_2$'s accepting states, all those states that record $T_1$'s state as one of the accepting a state of $T_1$. Hence, whatever $T_1$ accepts $T_2$ also accepts.

# Non-Deterministic Turing Machine

- A non-deterministic Turing Machine (NTM), T = (Q, Σ, Γ, δ, q0, B, F) is defined exactly the same as an ordinary TM, except the value of transition function δ.
- In NTM, the values of the transition function δ are subsets, rather than a single element of the set Q×Γ×{R,L,S}.
- Here, the transition function δ is such that for each state q and tape symbol x, δ(q, x) is a set of triples :
  {(q1, Y1, D1), (q2, Y2, D2), ..... (qk, Yk, Dk) where k is any finite integer
- The NTM can choose, at each step, any of the triples to be the next move.
- It cannot, however pick a state from one, a tape symbol from another, and the direction from yet another.

# Church-Turing Thesis

- The Church-Turing Thesis states that in its most common form that "every computation or algorithm can be carried out by a Turing Machine."
- This statement was first formulated by Alonzo Church.
- It is not a mathematically precise statement so un-provable.
- However, it is now generally assumed to be true.
- The thesis might be replaced as saying that the notation of effective or mathematical method in logic and mathematics is captured by TM.

# Church-Turing Thesis

- It is generally assumed that such methods must satisfy the following requirements :
  - The method consists of a finite set of simple and precise instructions that are described with a finite number of symbols
  - The method will always produce the result in a finite number of steps
  - The method can in principle be carried out by a human being with only paper and pencil.
  - The execution of the method requires no intelligence of the human being except that which is needed to understand and execute the instructions
- The example of such a method is the "Euclidean algorithm" for determining the "Greatest Common Divisor" of two natural numbers

# Church-Turing Thesis

- The invention of TM has accumulated enough evidence to accept this hypothesis.
- Following are the some of evidences :
  - In nature, a human normally works on 2D paper. The transfer of attention is not adjacent block like TM. However, transferring attention from one place to another during computation can be simulated by TM as one human step by multiple tapes
  - Various extensions of TM model have been suggested to make computation efficient like tape with multiple tracks, multi-tape etc.
  - Other theoretical model have been suggested that are closer to modern computers in their operation (e.g. simple programming type languages, grammar and others)
- Thus after adopting Church-Turing Thesis, we are giving a precise meaning of the term : "An algorithm is a procedure that can be executed on a Turing Machine"

# Universal Turing Machine

- In computer science, a universal Turing machine (UTM) is a Turing machine that simulates an arbitrary Turing machine on arbitrary input.
- The universal machine essentially achieves this by reading both the description of the machine to be simulated as well as the input to that machine from its own tape
- A machine that can simulate the behavior of an arbitrary TM is called a Universal Turing Machine
- Thus, we can describe a Universal Turing Machine $T_u$ as a TM, that on input <M, w>; where M is a TM and w is string of input alphabets, simulates the computation of M on input w.
- Specially,
  - $T_u$ accepts <M, w> iff M accepts w.
  - $T_u$ rejects <M, w> iff M rejects w

# Restricted Turing Machines

- **Halting Turing Machine :**
  - A Turing Machine is said to be a halting Turing machine if it always halts for every input string.
  - It can accept the recursive language and is less powerful than Turing machine
- **Linear Bounded Automata :**
  - It behaves as a Turing machine but the storage space of tape is restricted only to the length of the input string.
  - It is less powerful than a Turing machine but more powerful than push down automata.
- **Unidirectional Turing Machine :**
  - The head of this type of Turing machine can move only in one direction.
  - It can accept the only regular language.
  - It has the same power as finite automata but less powerful than push down automata.

# Restricted Turing Machines

- **Read Only Turing Machine :**
  - It is equivalent to finite automata.
  - It contains a read head only which doesn't have written capability.
  - It accepts only regular languages
- **Read Only-Unidirectional Turing Machine :**
  - It is similar to finite automata.
  - It contains a read-only head and can move only in one direction.
  - It accepts a regular language
- **Semi-Infinite Tape Turing Machine :**
  - A Turing Machine with a semi-infinite tape has a left end but no right end.
  - The left end is limited with an end marker