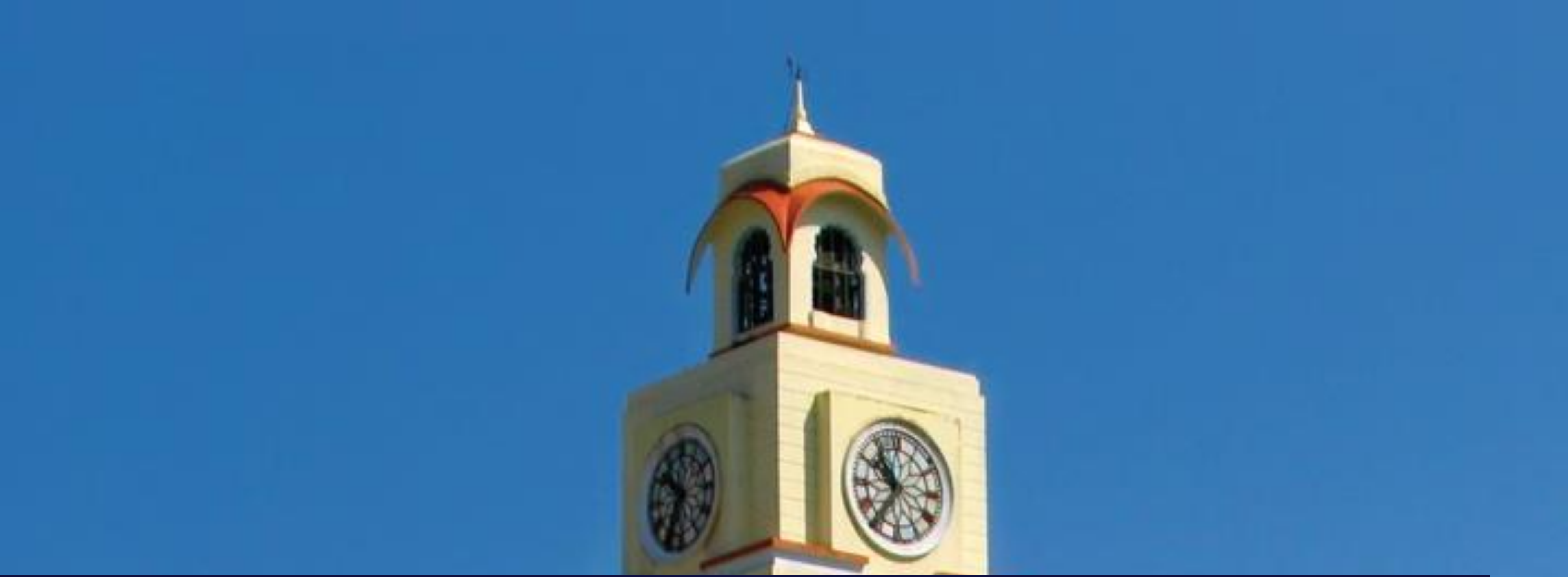


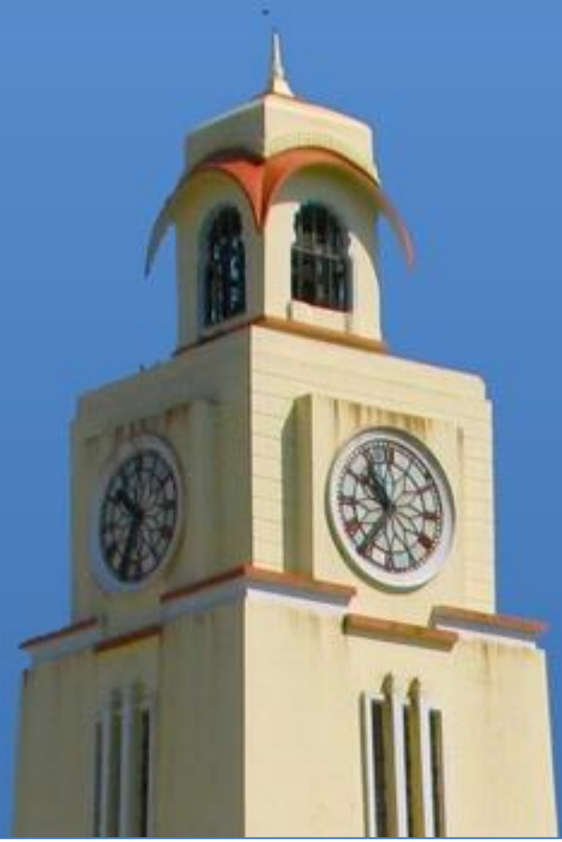
BITS Pilani

Pilani|Dubai|Goa|Hyderabad

Theory of Computation

CS F351

Vishal Gupta
Department of Computer Science and Information Systems
Birla Institute of Technology and Science
Pilani Campus, Pilani



Agenda:

1. **Reducibility**

2. **Undecidability**

Computable function

- A function f: $\Sigma^* \rightarrow \Sigma^*$ is a computable function if some Turing Machine M, on every input w, halts with just f(w) on its tape.
  - For example, all arithmetic operations on integers are computable functions.
- Computable function may be transformations of machine descriptions.

w → **Computable function f**: If w is an encoding of Turing Machine M, then modify M to M’ → f(w) = M’

BITS Pilani, Deemed to be University under Section 3 of UGC Act, 1956

Reducibility

- Language A is reducible to language B, written as **$A \leq_m B$**, if there is a computable function f: $\Sigma^* \rightarrow \Sigma^*$, where for every w,

  $$w \in A \quad \leftrightarrow \quad f(w) \in B.$$

  The function f is called the *reduction* of A to B.

- Being able to reduce problem A to problem B means that a computable function exists that convert instances of A to instances of B.

Reduction

"If A $\leq_m$ B and B is decidable, then A is _*decidable*_"

Proof:

Let M be a TM which decides B and f be the reduction from A to B. We describe a TM N to decide A as follows:

N = "On input w":

1. **Compute f(w)**
2. **Run M on input f(w) and output whatever M outputs.**

[NOTE: If w belongs to A then f(w) belongs to B. Thus M accepts f(w) whenever w belongs to A.]

BITS Pilani, Deemed to be University under Section 3 of UGC Act, 1956

Reduction: Proving Un-decidability

To prove that certain problems are un-decidable, we use the following corollary of previous theorem:

"If $A \leq_m B$ and A is known to be un-decidable, then B is un-decidable too"

BITS Pilani, Deemed to be University under Section 3 of UGC Act, 1956

Example 1

Problem A: A_{TM} = {"M" "w" | M is a TM and M accepts w}

Problem B: $HALT_{TM}$ = {"M" "w" | M is a TM and M halts on w}

Proof: Assume that TM R decides $HALT_{TM}$. Below is the construction of TM S which decides A_{TM}.

S = On input "M" "w":

1. **Run TM R on input "M" "w"**
2. **If R rejects, *Reject***
3. **If R accepts, simulate M on w until it halts.**
4. **If M has accepted, *Accept* ; if M has rejected, *Reject***

BITS Pilani, Deemed to be University under Section 3 of UGC Act, 1956

Example 1

Problem A: A_{TM} = {“M” “w”| M is a TM and M accepts w}

Problem B: $HALT_{TM}$ = {“M” “w” | M is a TM and M halts on w}

So, what is the computable function f here that takes the input of the form “M” “w” and returns output of the form “M_1” “w_1” where:

“M” “w” ϵ A_{TM} if and only if “M_1” “w_1” ϵ $HALT_{TM}$

The following machine F computes a reduction f.

F = On input “M” “w”

1. Construct the following machine M_1:

   M_1 = On input x:

   1. Run M on x.
   2. If M accepts, *accept*.
   3. If M rejects, enter a loop.

2. Output “M1” “w”.

BITS Pilani, Deemed to be University under Section 3 of UGC Act, 1956

Example 4

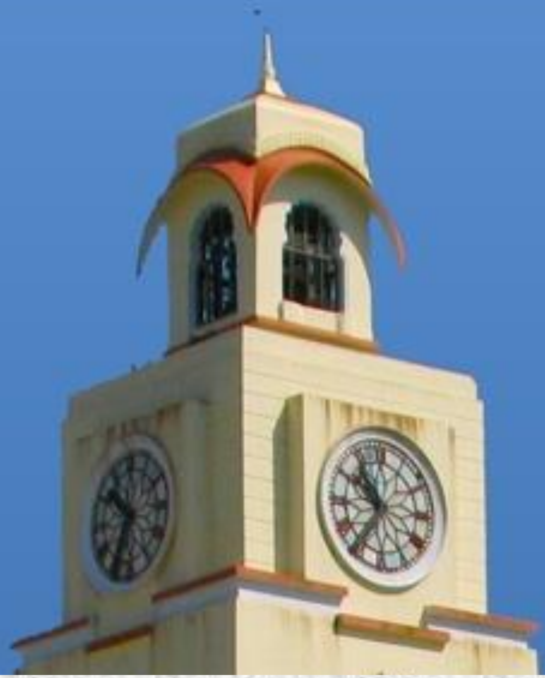
Problem A: E_{TM} = {"M" | M is a TM and L(M) is ϕ}

Problem B: EQ_{TM} = {"M_1" "M_2"|M_1 and M_2 are TM's and L(M_1) = L(M_2)}

Proof: Suppose EQ_{TM} is decidable and TM R decides it. We can construct another TM S which decides E_{TM} as follows:

S = "On input "M":

1. Run R on input "M" "M_1", where M_1 is a TM that rejects all inputs.
2. If R accepts, *accept*; If R rejects, *reject*.



innovate achieve lead

BITS Pilani

Pilani|Dubai|Goa|Hyderabad

Thank You