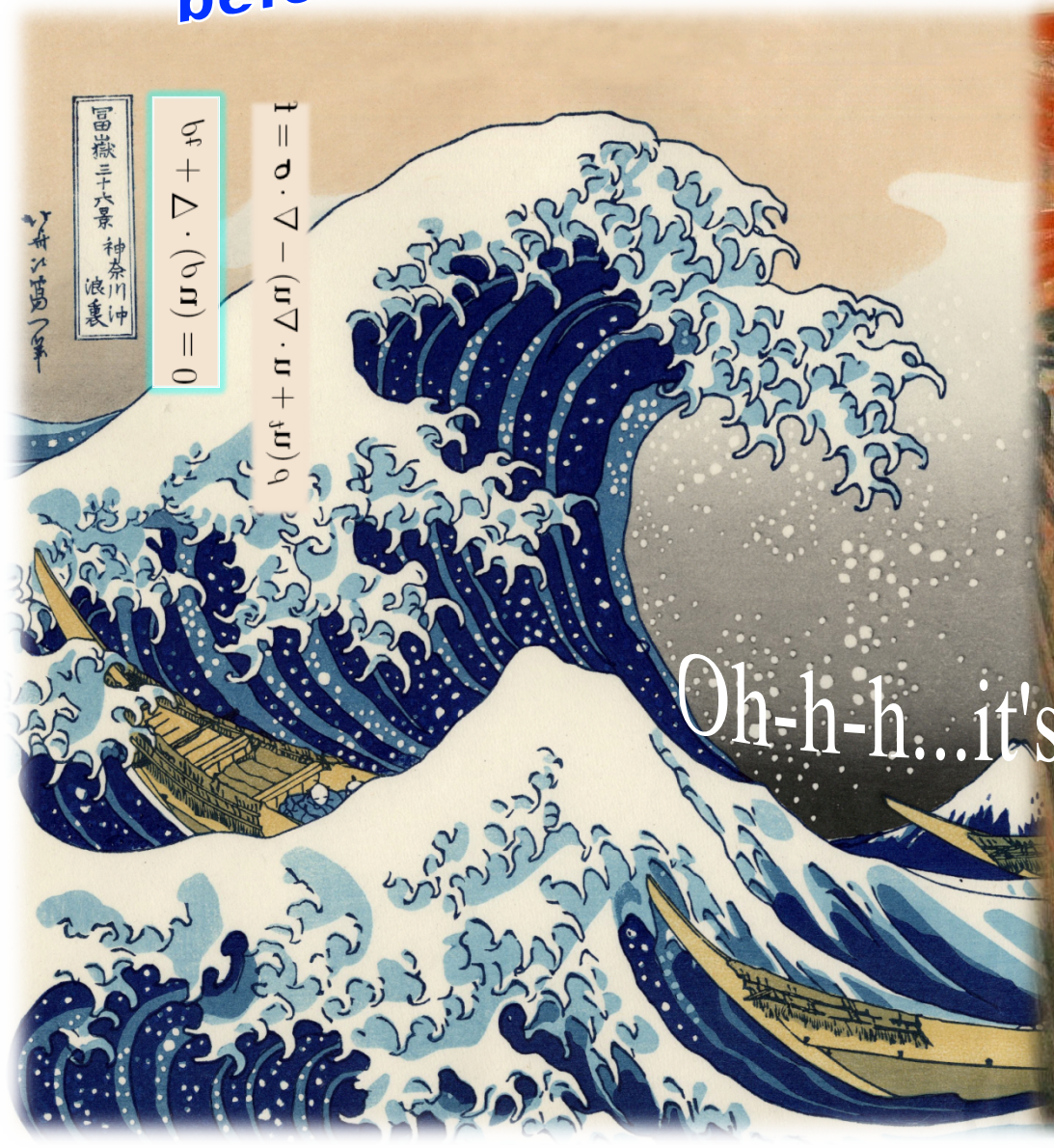
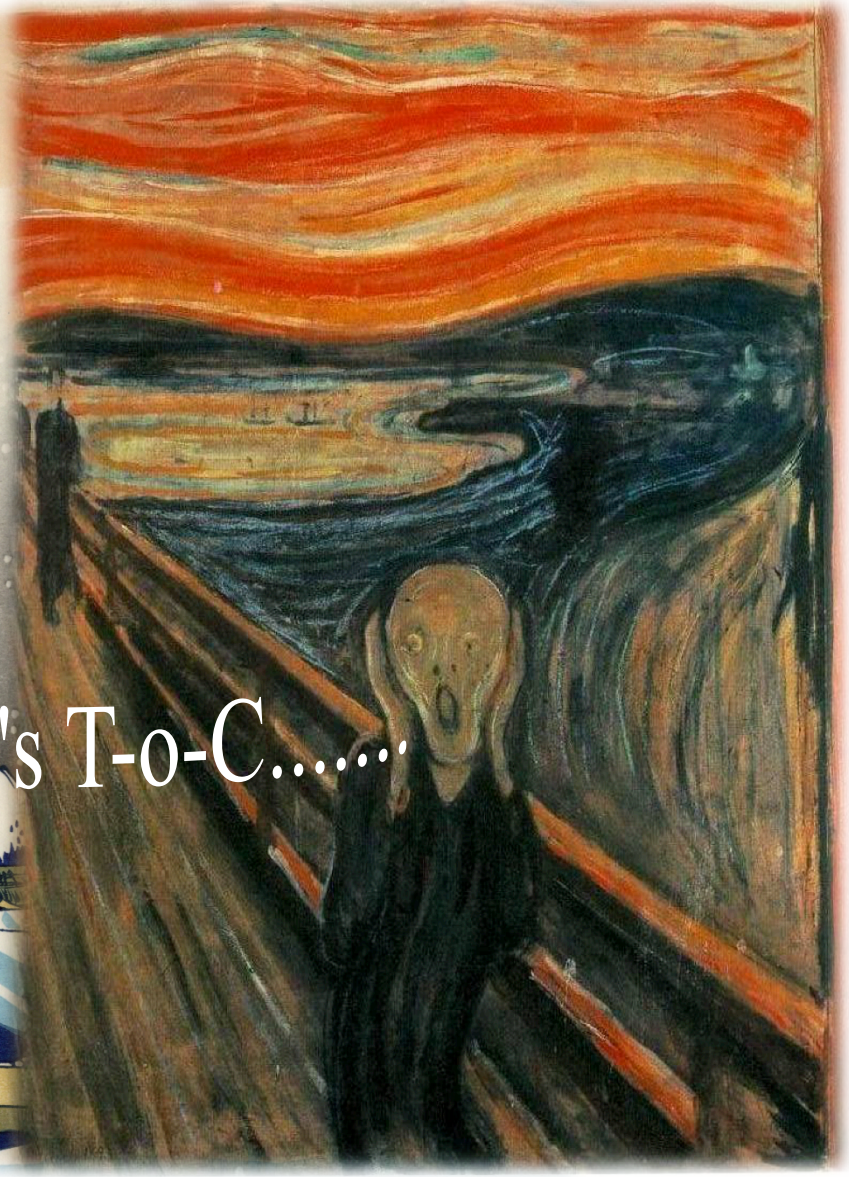
All you need to know about CS before you graduate

Intro to Theory of Computation

www. inf.unisi.ch\faculty\sharygina





Oh-h-h...it's T-o-C.....

Today's lecture

- Turing machine as a universal model of computation
  - Alan Turing's bio
  - Formalism
  - Demo

Turing machine

- Fundamental concept in computer science

- Invented by Alan Turing (1912 – 1954) in 1936

Theory of Computation

- Computability Theory:

  What problems can(not) computers solve?

- Complexity Theory:

  What makes some problems computationally hard and others easy?

Theory of Computation

- Computability Theory:

  What problems can(not) computers solve?

- Complexity Theory:

  What makes some problems computationally hard and others easy?

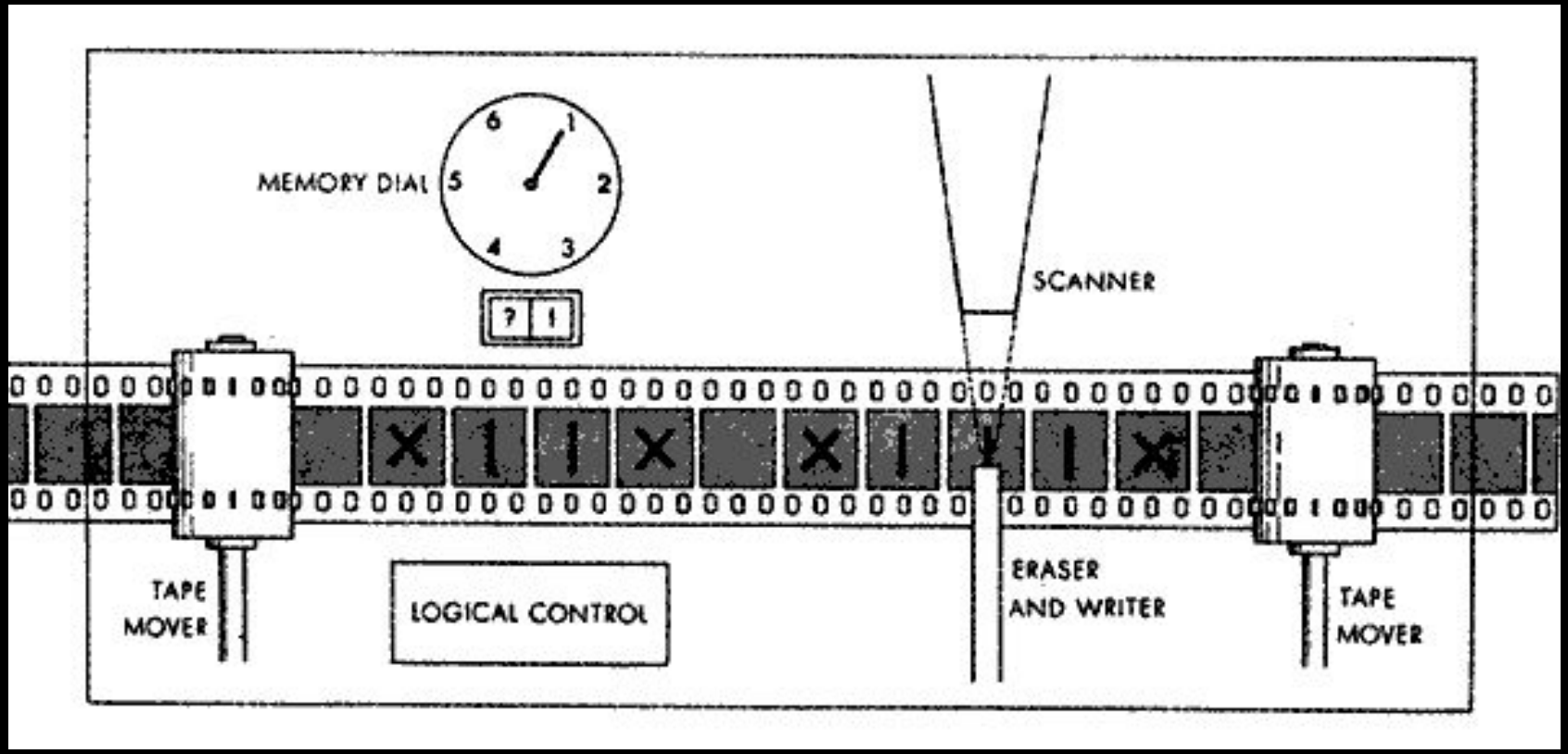
What is role of Turing and the purpose of Turing machine for above?

We need a device that could be used to compare problems and their solutions – Turing machine is this device in computing

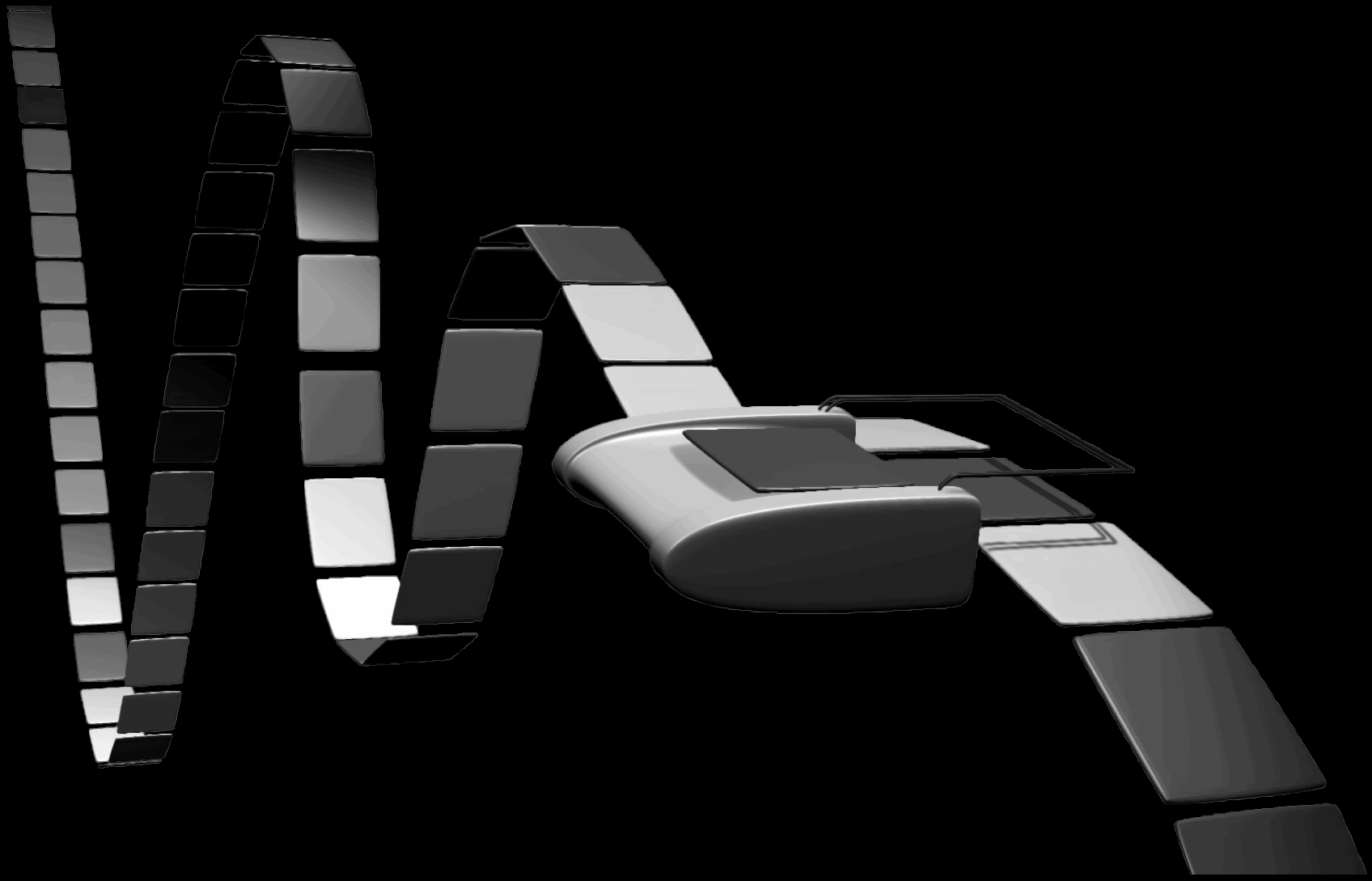
Turing Machine

- A Turing machine is a much more accurate model of a general purpose computer than FA or PDA
- A Turing machine can do everything that a real computer can do
- Even a Turing machine can not solve certain problems: in a very real sense, these problems are beyond the theoretical concepts of computation

Turing Machine



Turing Machine



What is different from FA?

- Input tape is infinite
- Input head can both READ and WRITE
- Input head can move LEFT and RIGHT
- Has both ACCEPT and REJECT states ***and accepts/rejects right away*** (does not need to reach the end of input)

Designing TM

- Let $L = \{ w\#w \mid w \in \{0,1\}^* \}$

Give a high-level description of a Turing Machine that accepts an input if it is in L, and rejects if not.

Let's try designing it all together

See solution in Section 3,1 of M. Sipser's book

Formal Definition of TM

Turing Machine: $(Q, S, G, d, q_o, q_{accept}, q_{reject})$

Main points:

– Q is a set of states

– $S \subset G$, S is the input alphabet, G is the tape alphabet

– Blank symbol is in G, but not in S

– $d : Q \times G \rightarrow Q \times G \times \{L, R\}$ is the transition function

– Special case: when rd/wr head is at left end of the input tape

TM Configuration

u q v -- TM is in state *q* with strings *u and v* on the tape and the head on the first symbol of *v*

C_i *yields* C_j if the Turing machine can legally go from config C_i to C_j *in a single step*

TM Configuration

u q v -- TM is in state *q* with *uv* on the tape and the head on the first symbol of *v*

Ci *yields* Cj if the Turing machine can legally go from config Ci to Cj *in a single step*

Examples:
Given: $a,b,c \in G$, $u,v \in G^*$, $q_i,q_j \in Q$

Configurations: 1) $ua\ q_i\ bv$; 2) $u\ q_j\ acv$

Say that $ua\ q_i\ bv$ yeilds $u\ q_j\ acv$ if the transition function $d(q_i,b) = (q_j,c,L)$: *Turing machines moves leftwards*

Give an example for a move rightwards

TM Configuration

Start Configuration of M on ω is $q_0\omega$, which indicates that the machine is in the start state q_0 with its head at the leftmost position

Accepting Configuration and *Rejecting Configuration* are the halting configurations and do not yield further configurations.

TM Acceptance

Turing Machine accepts w if $\exists$ sequence of configurations $C_1, C_2, \ldots, C_k$ where

1. C_1 is the start configuration
2. C_i yields C_{i+1} $(1 \leq i \leq k\text{-}1)$
3. C_k is an accepting configuration

Turing machine language

The collection of strings that TM accepts is the language of TM

A TM recognizes a language if it *accepts* all and only those strings in the language

A language is called Turing-recognizable or

recursively enumerable if some TM recognizes it

Turing-decidable languages

While possible outcomes are *accept*, *reject* and *loop*, a TM decides a language if it *accepts* all strings in the language and *rejects* all strings not in the language

A language is called decidable or recursive

if some TM decides it

Examples

We will consider (later in the course) examples of various decidable languages

Every decidable language is Turing-recognizable

We will also consider examples of languages that are not Turing-recognizable (these are the languages for which TM loops forever). We will do it after we develop a technique for proving undecidability

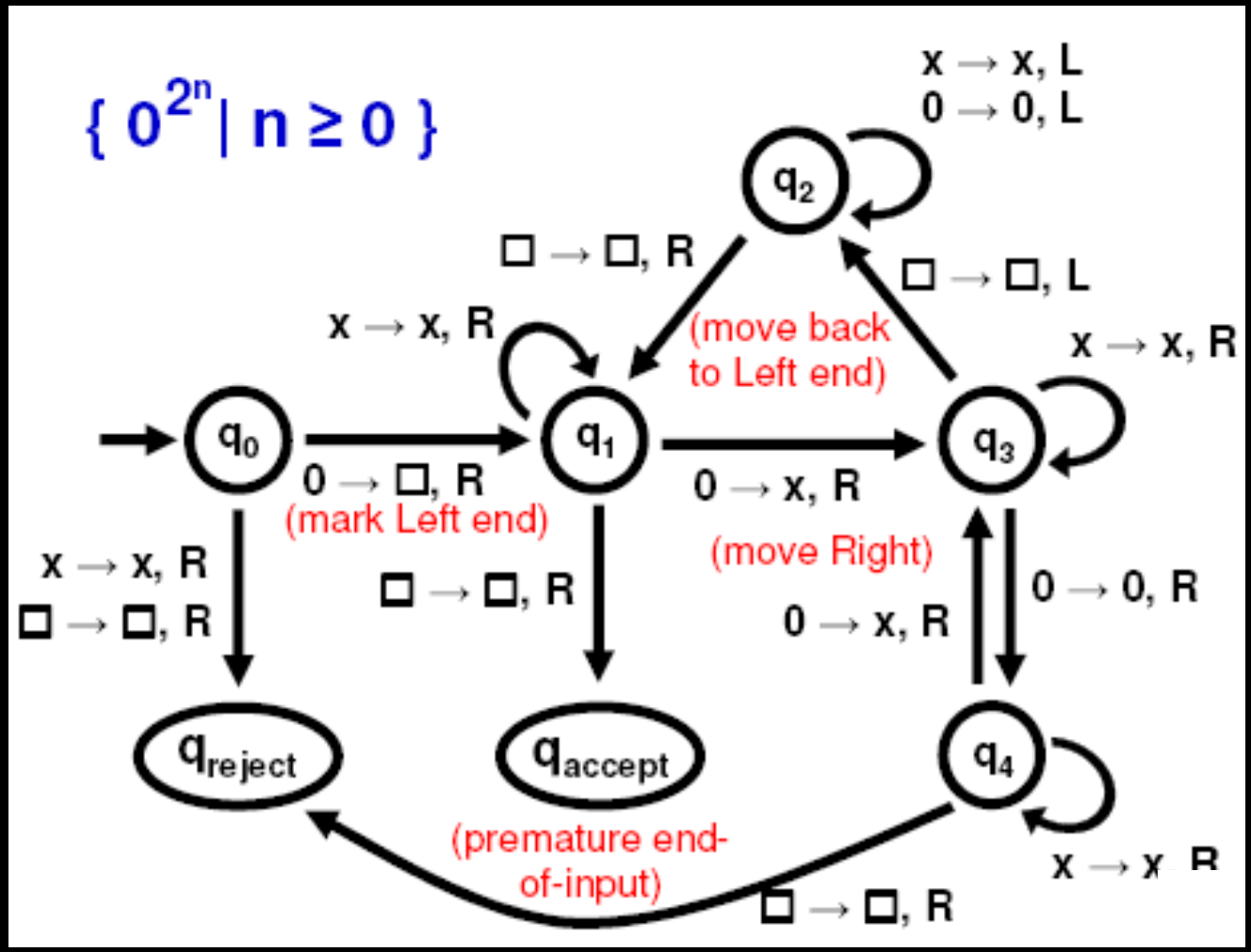
Design a TM that decides $L = \{ 0^{2^n} \mid n \geq 0 \}$

The language L consists of all strings of 0s whose length is a power of 2

Main idea to design: "repeated division by 2"

See solution in M. Sipser's book, Ex. 3.7 of Section 3.1

Design a TM that decides $L = \{ 0^{2^n} \mid n \geq 0 \}$



Summary on Turing Machines

- A Turing machine is a much more accurate model of a general purpose computer than FA or PDA

- Even a Turing machine can not solve certain problems: in a very real sense, these problems are beyond the theoretical concepts of computation

Next steps

HW 1

Variants of Turing Machines

- reading assignment and presentations in class