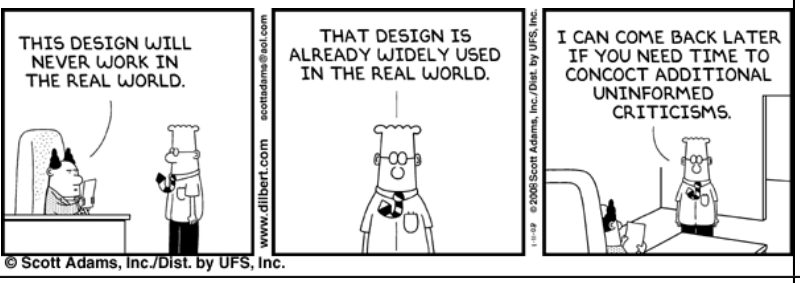
A Universal Computer



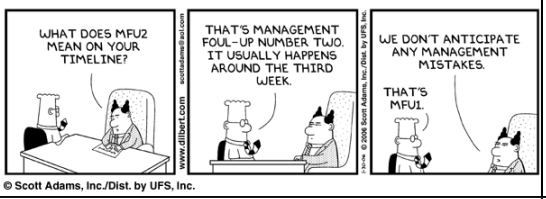
One-Slide Summary

- The **Turing machine** is a fundamental model of computation. It models input, output, processing and memory. A Turing machine has a **finite state machine controller** as well as an **infinite tape**. At each step it reads the current tape symbol, writes a new tape symbol, moves the tape head left or right one square, and moves to a new state in the finite state machine controller. Turing machines are **universal**: they are just as powerful as Scheme, Python, C, or Java.
- The **lambda calculus** is also a universal, fundamental model of computation. You can view it as "the essence of Scheme".

Where Are We?

- Last Time: Passwords, Security, etc.

- PS9 Meetings



Finite State Machine

- There are lots of things we can't compute with only a finite number of states
- Solutions:
  - Infinite State Machine
    - Hard to describe and draw
  - **Add an infinite tape to the Finite State Machine**
    - We'll do this instead.

Turing's Explanation

"We have said that the computable numbers are those whose decimals are calculable by **finite** means. ... For the present I shall only say that the justification lies in the fact that the human memory is necessarily limited."

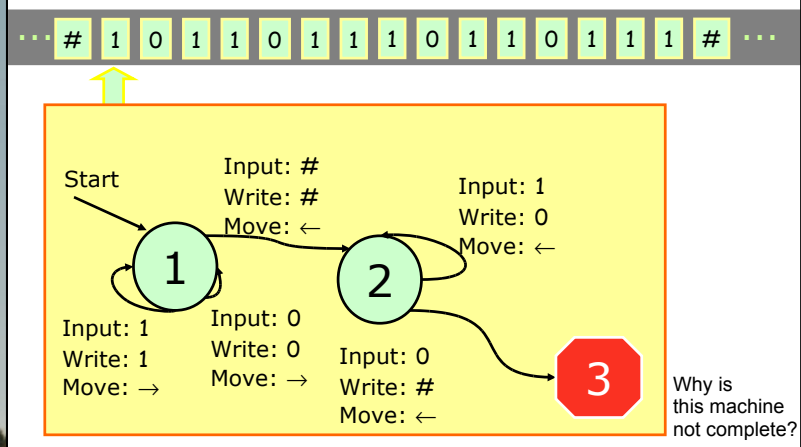
FSM + Infinite Tape

- Start:
  - FSM in Start State
  - Input on Infinite Tape
  - Pointer (= read/write head) to start of input
- Step (4 sub-steps each time):
  - Read one input symbol from tape
  - Write symbol on tape, and move L or R one square
  - Follow transition rule from current state
- Finish:
  - Transition to halt state

Turing Machine Hardware

- **Infinite** ("as much as you want") Paper Tape
  - can read from and write to
  - but only **finite** time ...
- **Finite** Brain (set of rules)
  - modeling "Computers" (People)

Turing Machine

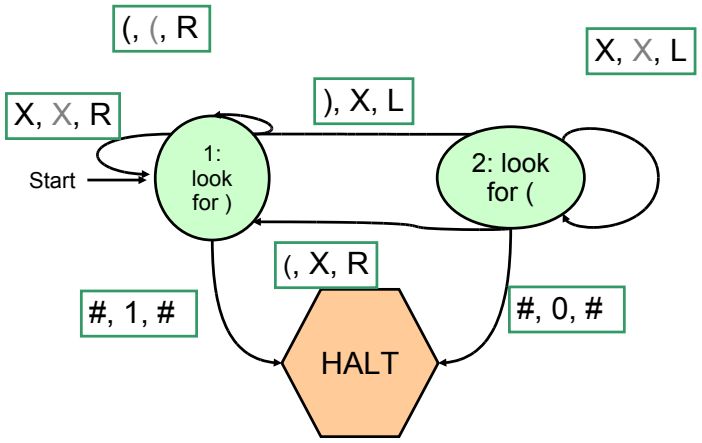


Why is this machine not complete?

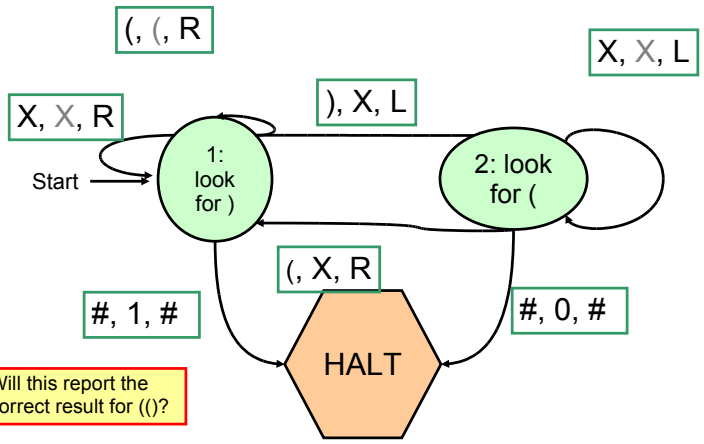
Matching Parentheses

- Find the leftmost)
  - If you don't find one, the parentheses match, write a 1 at the tape head and halt.
- Replace it with an X
- Look left for the first (
  - If you find it, replace it with an X (they matched)
  - If you don't find it, the parentheses didn't match - end write a 0 at the tape head and halt

Matching Parentheses

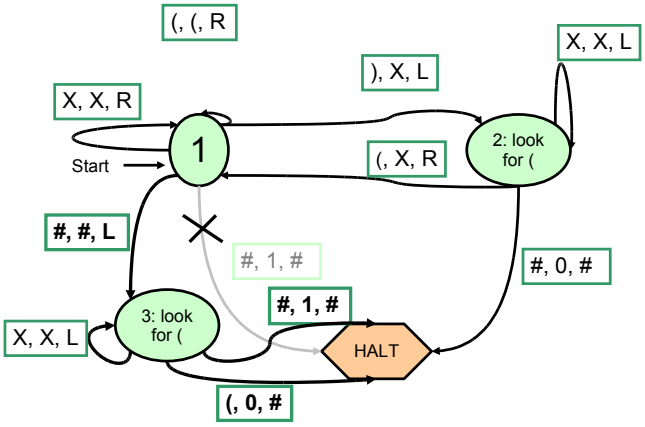


Matching Parentheses



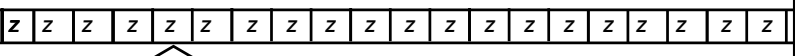
Will this report the correct result for (()?

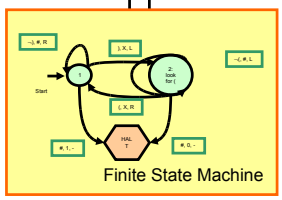
Matching Parentheses



Turing Machine

Infinite Tape





TuringMachine ::= < *Alphabet*, *Tape*, ***FSM*** >
Alphabet ::= { *Symbol** }
Tape ::= < *LeftSide*, *Current*, *RightSide* >
OneSquare ::= *Symbol* | #
Current ::= *OneSquare*
LeftSide ::= [*Square**]
RightSide ::= [*Square**]

Everything to left of *LeftSide* is **#**.
Everything to right of *RightSide* is **#**.

How well does this model your computer?

Turing Machine: FSM + Infinite Tape

- Start:
  - FSM in Start State
  - Input on Infinite Tape
  - Tape head at start of input
- Step (4 sub-steps):
  - Read current input symbol from tape
  - Follow transition rule from current state on input
    - Write symbol on tape
    - Move L or R one square
    - Update FSM state
- Finish: Transition to halt state

Liberal Arts Trivia: Politics

- This military alliance, established by the North Atlantic Treaty in 1949, provides for a system of collective defense whereby its member states agree to help each other in response to an attack by an external party. An infamous initial goal was “to keep the Russians out, the Americans in, and the Germans down.” The combined military spending of its members accounts for over 70% of the world's total defense spending.

Liberal Arts Trivia: Chemistry

- Diacetylmorphine was first synthesized in 1874. It was later commercialized by (the company that would become) Bayer in a failed effort to produce codeine. From 1898 to 1910 it was marketed as a non-addictive morphine substitute and cough suppressant, and as a cure for morphine addiction. It was quickly discovered that it rapidly metabolized into morphine, and, as such, was essentially just a quicker form of morphine. Give today's name for this drug, which made field subjects feel heroic.

A function $f(w)$ has:

Domain D $w \in D$

Result Region S $f(w) \in S$

Integer Domain:

Unary: 11111

Binary: 101

Decimal: 5

We prefer Unary representation:

Easier to manipulate

A function may have many parameters:

Example:

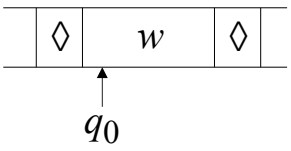
Addition function

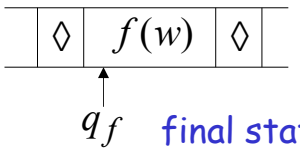
$f(x,y)=x+y$

x, y are integers

Definition:

A function f is computable if there is a Turing Machine M such that:





final state

Initial Configuration

Final configuration

$w \in D$ Domain

Example

The function $f(x,y)=x+y$ is computable

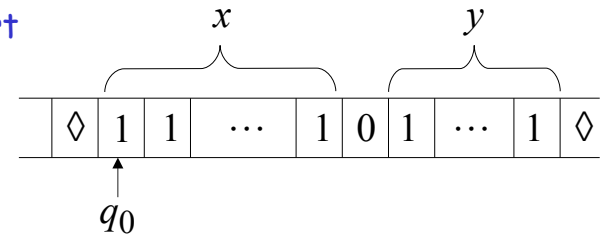
x, y are integers

Turing Machine:

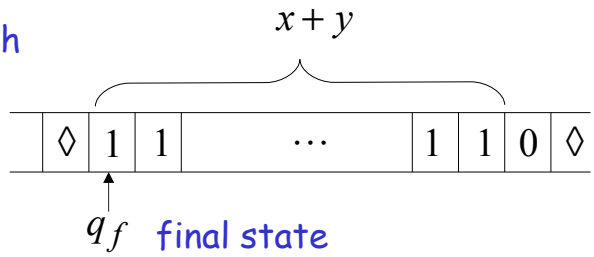
Input string: $x0y$ unary

Output string: $xy0$ unary

Start

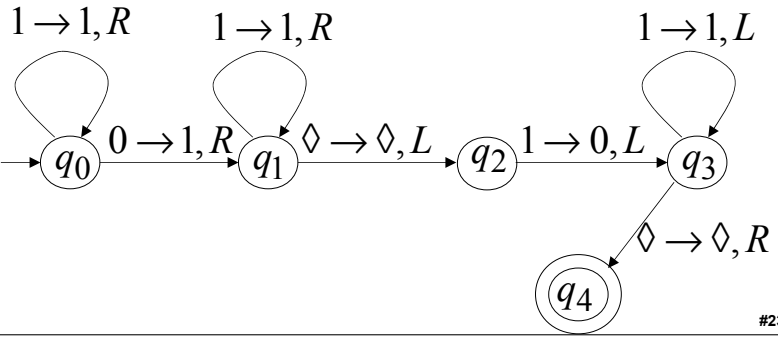


Finish



final state

Turing machine for function $f(x,y)=x+y$

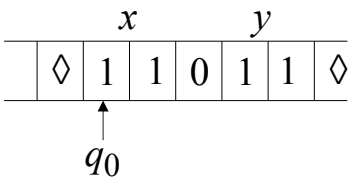


Execution Example:

$x=11$ (2)

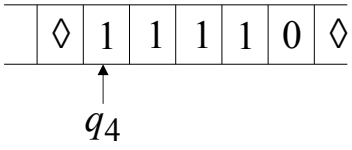
$y=11$ (2)

Time 0

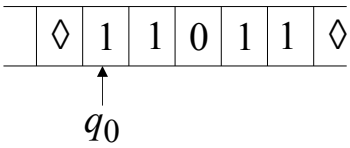


Final Result

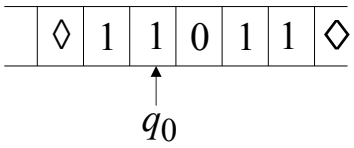
$x+y$

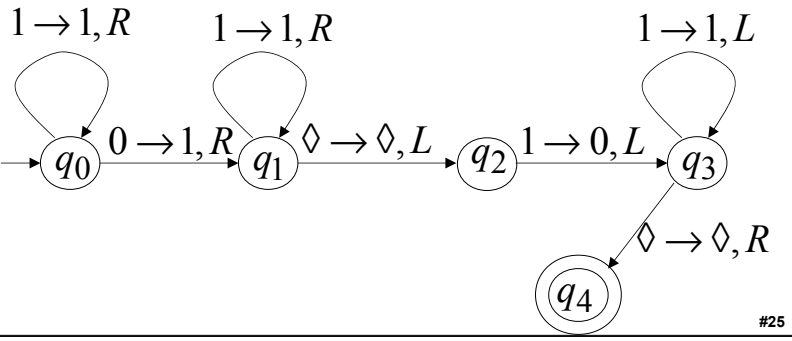


Time 0



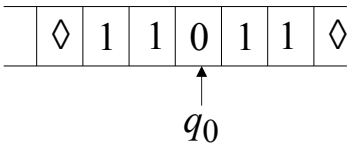
Time 1



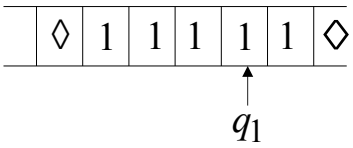


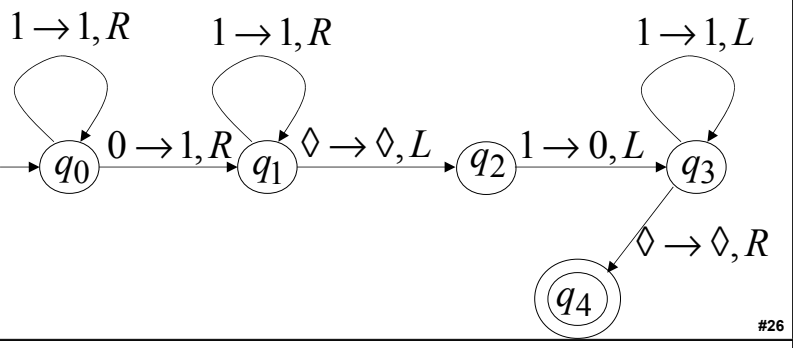
#25

Time 2



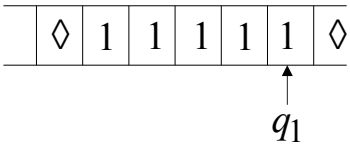
Time 3



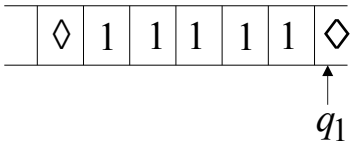


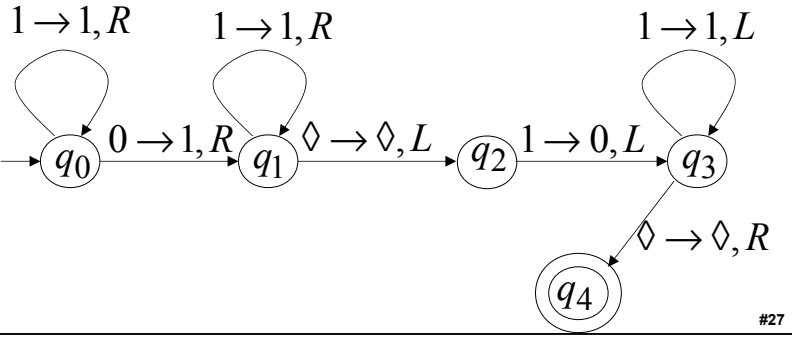
#26

Time 4



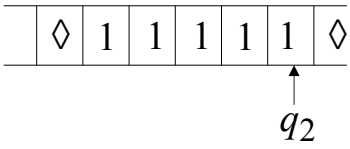
Time 5



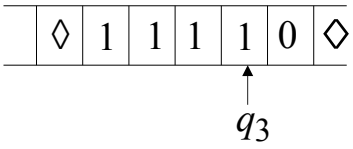


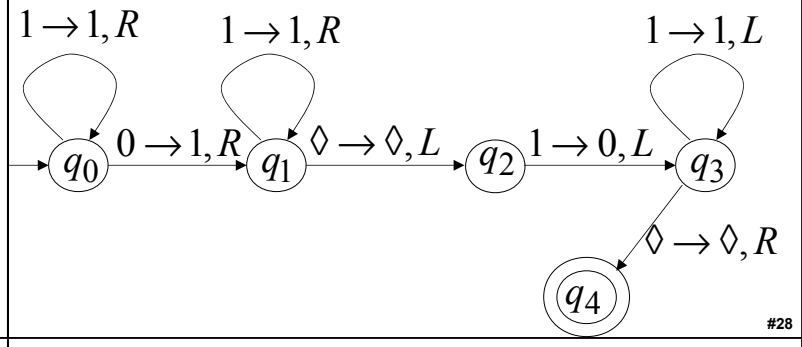
#27

Time 6



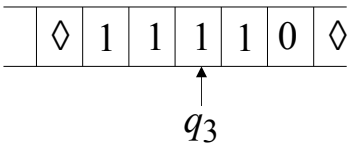
Time 7



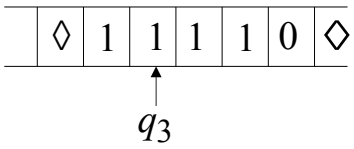


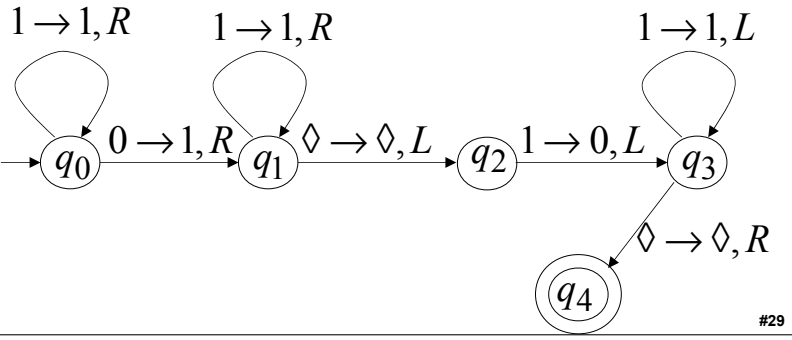
#28

Time 8



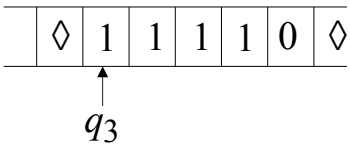
Time 9



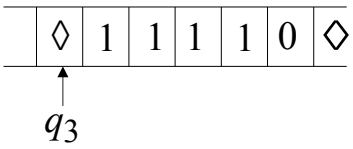


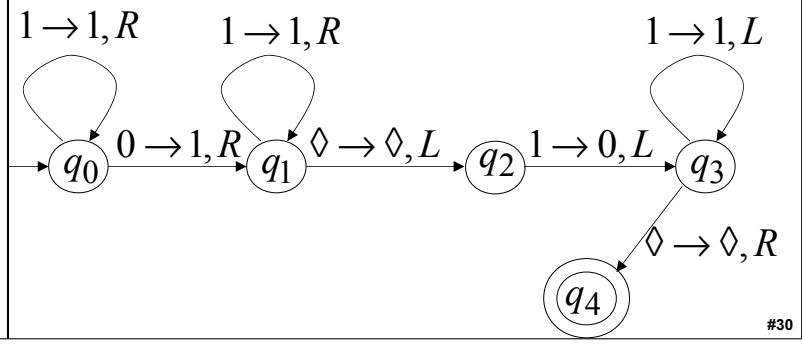
#29

Time 10



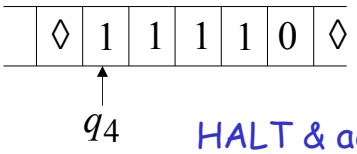
Time 11



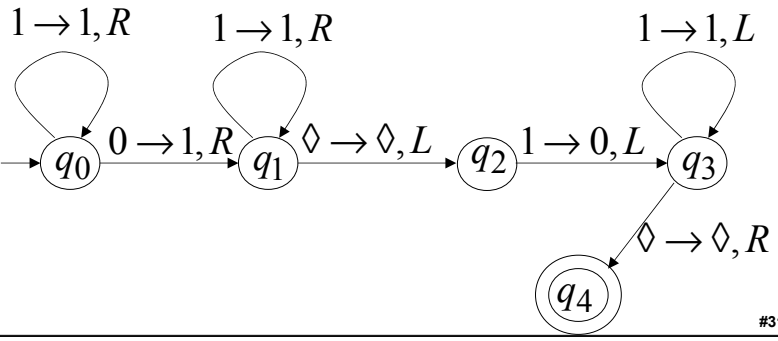


#30

Time 12



q_4 HALT & accept



Liberal Arts Trivia: American Lit

- This early-20th-century American author invited and wrote about cosmic horror, the idea that life is incomprehensible to human minds and that the universe is fundamentally alien. Thus, those who genuinely reason gamble with insanity. He was little read during life but is now regarded with Edgar Allen Poe as one of the most influential horror writers of the 20th century.

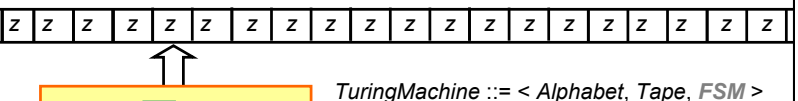
Liberal Arts Trivia: Chemistry

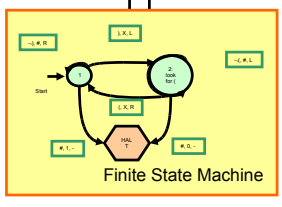
- This highly flammable compound is formed by nitrating cellulose through exposure to nitric acid. It can be used as a propellant or low-order explosive. French chemist Paul Vieille invited the first practical smokeless powder for firearms and artillery ammunition from this; is has also been used as a film base for medical X-rays. It is commonly produced in introductory chemistry classes by treating cotton balls with nitric acid.

Liberal Arts Trivia: Taoism

- Name the group of legendary, undying *xian* from Chinese mythology. Each member's power can be transferred to a tool that can give life or destroy evil. They are revered by Taoists, and include Royal Uncle *Cao*, Iron Crutch *Li*, Elder *Zhang Guo*, and *Lu Tung-Pin*. Many of them were said to practice *neidan*, or internal alchemy.

Turing Machine





TuringMachine ::= < *Alphabet*, *Tape*, ***FSM*** >
Alphabet ::= { *Symbol** }
Tape ::= < *LeftSide*, *Current*, *RightSide* >
OneSquare ::= *Symbol* | #
Current ::= *OneSquare*
LeftSide ::= [*Square**]
RightSide ::= [*Square**]

Everything to left of *LeftSide* is **#**.
Everything to right of *RightSide* is **#**.

Describing Finite State Machines

TuringMachine ::= < *Alphabet*, *Tape*, ***FSM*** >
FSM ::= < *States*, *TransitionRules*, *InitialState*, *HaltingStates* >
States ::= { *StateName** }
InitialState ::= *StateName* must be element of *States*
HaltingStates ::= { *StateName** } all must be elements of *States*
TransitionRules ::= { *TransitionRule** }
TransitionRule ::=
< *StateName*, ;; Current State
OneSquare, ;; Current square
StateName, ;; Next State
OneSquare, ;; Write on tape
Direction > ;; Move tape
Direction ::= L, R, #

Transition Rule is a procedure:
Inputs: *StateName, OneSquare*
Outputs: *StateName*, *OneSquare*, *Direction*

Enumerating Turing Machines

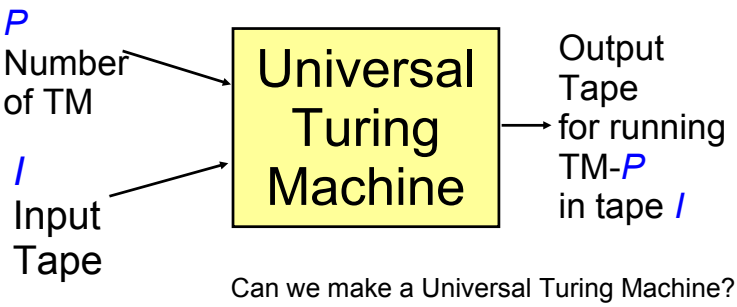
- Now that we've decided how to describe Turing Machines, we can number them
- TM-5023582376 = balancing parens
- TM-57239683 = even number of 1s
- TM-352379683472103829673825987 = Photomosaic Program
- TM-3672349872381692309875823987609823712347823 = Windows 7

Enumerating Turing Machines

- Now that we've decided how to describe Turing Machines, we can number them
- TM-5023582376 = balancing parens
- TM-57239683 = even number of 1s
- TM-352379683472103829673825987 = Photomosaic Program
- TM-3672349872381692309875823987609823712347823 = Windows 7

Not the real numbers – they would be much bigger!

Universal Turing Machine

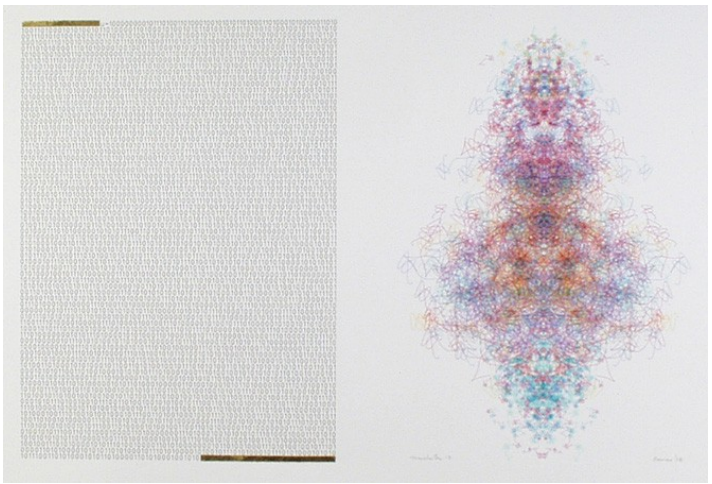


also, just a number!

Can we make a Universal Turing Machine?

Yes!

- People have designed Universal Turing Machines with
  - 4 symbols, 7 states (Marvin Minsky)
  - 4 symbols, 5 states
  - 2 symbols, 22 states
  - 18 symbols, 2 states
  - 2 states, 5 symbols (Stephen Wolfram)
- No one knows what the smallest possible UTM is



Manchester Illuminated Universal Turing Machine, #9 from http://www.verostko.com/manchester/manchester.html

Church-Turing Thesis

- **Any mechanical computation can be performed by a Turing Machine**
- There is a TM-*n* corresponding to every computable problem
- We can any "normal" (classical mechanics) computer with a TM
  - If a problem is in polynomial time on a TM, it is in polynomial time on an iMac, Cray, Palm, etc.
  - But maybe not a quantum computer! (later class)

Universal Language

- Is Scheme/Charme/Python as powerful as a Universal Turing Machine?
- Is a Universal Turing Machine as powerful as Scheme/Charme/Python?

Universal Language

- Is Scheme/Charme/Python as powerful as a Universal Turing Machine?

Yes: show we can simulate a UTM with a Scheme program

- Is a Universal Turing Machine as powerful as Scheme/Charme/Python?

Can we simulate a Scheme interpreter with a TM?

Complexity in Scheme

- Special Forms
  - **if**, **cond**, **define**, etc.
- Primitives
  - Numbers (infinitely many)
  - Booleans: **#t**, **#f**
  - Functions (**+**, **-**, **and**, **or**, etc.)
- Evaluation Complexity
  - Environments (more than ½ of our eval code)

If we have lazy evaluation and don't care about abstraction, we don't need these.

Hard to get rid of?

Can we get rid of all this and still have a useful language?

λ-calculus

Alonzo Church, 1940

(LISP was developed from λ-calculus, not the other way round.)

$$\begin{aligned} term = \; & variable \\ \mid \; & term\ term \\ \mid \; & (term) \\ \mid \; & \lambda\ variable\ .\ term \end{aligned}$$

What is Calculus?

- In High School:

$d/dx\ x^n = nx^{n-1}$ [Power Rule]

$d/dx\ (f + g) = d/dx\ f + d/dx\ g$ [Sum Rule]

Calculus is a branch of mathematics that deals with limits and the differentiation and integration of functions of one or more variables...

Surprise Liberal Arts Trivia

- This branch of mathematics involving symbolic expressions manipulated according to fixed rules takes its name from the diminutive form of calx/calcis, the latin word for rock or limestone. The diminutive word thus means "pebble": in ancient times pebbles were placed in sand and used for counting using techniques akin to those of the abacus.

Real Definition

- A **calculus** is just a bunch of rules for manipulating symbols.
  - Latin word calx meaning pebble ...
- People can give meaning to those symbols, but that's not part of the calculus.
- Differential calculus is a bunch of rules for manipulating symbols. There is an interpretation of those symbols corresponds with physics, slopes, etc.

Lambda Calculus

- Rules for manipulating strings of symbols in the language:

$$\begin{aligned} term = \;& variable \\ & \mid term\ term \\ & \mid (term) \\ & \mid \lambda\ variable\ .\ term \end{aligned}$$

- Humans can give meaning to those symbols in a way that corresponds to computations.

Why?

- Once we have precise and formal rules for manipulating symbols, we can use it to reason with.
- Since we can interpret the symbols as representing computations, we can use it to reason about programs.

Evaluation Rules

α-reduction (renaming)

$$\lambda y.\ M \Rightarrow_{\alpha} \lambda v.\ (M\ [\text{each } y \text{ replaced by } v])$$

where v does not occur in M.

β-reduction (substitution)

$$(\lambda x.\ M)N \Rightarrow_{\beta} M\ [\text{each } x \text{ replaced by } N\ ]$$

Homework

- Exam 2 Out Today
- PS 9