Lecture 14: Church-Turing Thesis

Alonzo Church (1903-1995)

Alan Turing (1912-1954)

Reminder: PS4 is due Tuesday

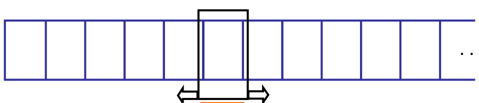
cs302: Theory of Computation
University of Virginia
Computer Science

David Evans
http://www.cs.virginia.edu/evans

Menu

- Finish computing model for TM
- The Most Bogus Sentence
- Robustness of TM Model (Church-Turing Thesis)

Turing Machine

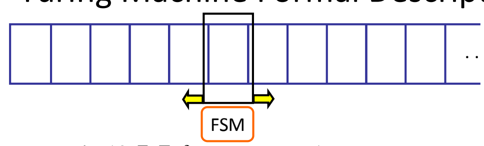


Infinite tape: Γ^*

Tape head: read current square on tape, write into current square, move one square **left** or **right**

FSM: like PDA, except:
transitions also include direction (**left**/**right**)
final accepting and rejecting states

Turing Machine Formal Description

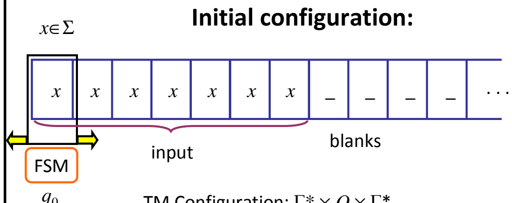


7-tuple: $(Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$
Q: finite set of states
Σ: input alphabet (cannot include blank symbol, _)
Γ: tape alphabet, includes Σ and _
δ: transition function: $Q \times \Gamma \rightarrow Q \times \Gamma \times \{\mathbf{L}, \mathbf{R}\}$
q_0: start state, $q_0 \in Q$
q_{accept}: accepting state, $q_{accept} \in Q$
q_{reject}: rejecting state, $q_{reject} \in Q$

(Sipser's notation)

Turing Machine Computing Model

Initial configuration:



TM Configuration: $\Gamma^* \times Q \times \Gamma^*$

tape contents left of head | current FSM state | tape contents head and right

TM Computing Model

$$\delta^*: \Gamma^* \times Q \times \Gamma^* \rightarrow \Gamma^* \times Q \times \Gamma^*$$

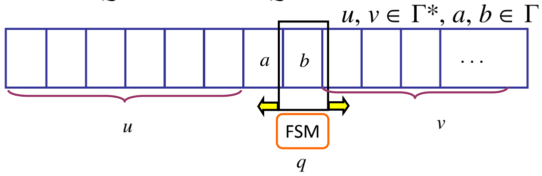
The q_{accept} and q_{reject} states are final:

$$\delta^*(L, q_{accept}, R) \rightarrow (L, q_{accept}, R)$$
$$\delta^*(L, q_{reject}, R) \rightarrow (L, q_{reject}, R)$$

TM Computing Model

$\delta^*: \Gamma^* \times Q \times \Gamma^* \rightarrow \Gamma^* \times Q \times \Gamma^*$

$u, v \in \Gamma^*, a, b \in \Gamma$



$\delta^*(ua, q, bv) = \delta^*(uac, q_r, v)$ if $\delta(q, b) = (q_r, c, \mathbf{R})$

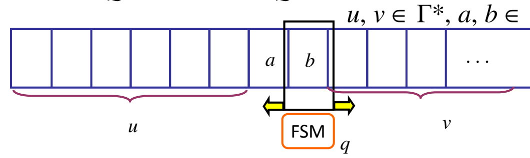
$\delta^*(ua, q, bv) = \delta^*(u, q_r, acv)$ if $\delta(q, b) = (q_r, c, \mathbf{L})$

Also: need a rule to cover what happens at left edge of tape

TM Computing Model

$\delta^*: \Gamma^* \times Q \times \Gamma^* \rightarrow \Gamma^* \times Q \times \Gamma^*$

$u, v \in \Gamma^*, a, b \in \Gamma$



$\delta^*(ua, q, bv) = \delta^*(uac, q_r, v)$ if $\delta(q, b) = (q_r, c, \mathbf{R})$

$\delta^*(ua, q, bv) = \delta^*(u, q_r, acv)$ if $\delta(q, b) = (q_r, c, \mathbf{L})$

$\delta^*(\varepsilon, q, bv) = \delta^*(\varepsilon, q_r, cv)$ if $\delta(q, b) = (q_r, c, \mathbf{L})$

Do we need a rule for the right edge of the tape?

TM Computing Model

$\delta^*: \Gamma^* \times Q \times \Gamma^* \rightarrow \Gamma^* \times Q \times \Gamma^*$

A string w is in the language of Turing Machine T **if**

$\delta^*(\varepsilon, q_0, w) = (\alpha, q_{accept}, \beta)$

A string w is not in the language of Turing Machine T **if**

$\delta^*(\varepsilon, q_0, w) = (\alpha, q_{reject}, \beta)$

Does this cover all possibilities?

Termination

- DFAs, DPDAs:
  - Consume one input symbol each step
  - Must terminate
- NFAs:
  - Equivalent to DFA: must terminate
- Turing Machine:
  - Can move left and right: no "progress" guarantee

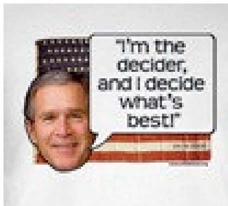
Possible Outcomes

1. Running TM M on input w eventually leads to q_{accept}.
2. Running TM M on input w eventually leads to q_{reject}.
3. Running TM M on input w runs forever (never terminates).

Recognizing vs. Deciding

- **Turing-recognizable**: A language L is "Turing-recognizable" if there exists a TM M such that for all strings w:
  - If $w \in L$ eventually M enters q_{accept}
  - If $w \notin L$ either M enters q_{reject} ***or*** M never terminates
- **Turing-decidable**: A language L is "decidable" if there exists a TM M such that for all strings w:
  - If $w \in L$, M enters q_{accept}.
  - If $w \notin L$, M enters q_{reject}.

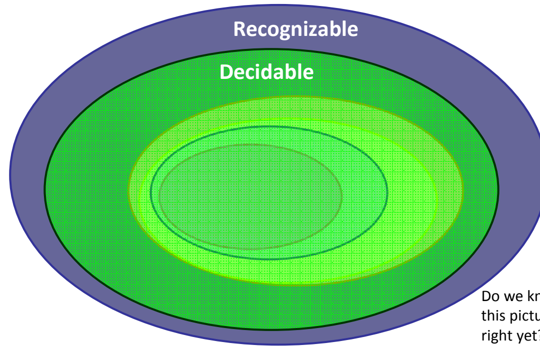
Decider vs. Recognizer?



Deciders *always* terminate.

Recognizers can run forever without deciding.

Decidable and Recognizable Languages



Do we know this picture is right yet?

The Most Bogus Sentence Guesses

- "Intuitive notion of algorithms equals Turing machine algorithms."
- "Some of these models are very much like Turing machines, but others are quite different."
- "Think of these as 'virtual' tapes and heads," on page 149. The quotation marks around virtual imply that the tapes and heads are not virtual, which is false.
- "If you feel the need to review nondeterminism, turn to Section 1.2 (page 47)." (By this point, one should have a firm grasp of nondeterminism.)
- "Proving an algorithm doesn't exist requires having a clear definition of algorithm."
- "For mathematicians of that period to come to this conclusion [(Hilbert's 10th Problem's accepted solution)] with their intuitive concept of algorithm would have been virtually impossible."

I don't find any of these statement bogus.

A bogus sentence (but not the one I had in mind)

- "To show that two models are equivalent we simply need to show that we can simulate one by the other."

Winner: David Horres

A B

For set equivalence, need to show A $\subseteq$ B and B $\subseteq$ A.
For machine equivalence, need to show
A can simulate B and B can simulate A.

The Most Bogus Sentence

"A Turning machine can do everything a real computer can do."

On the first page!

Winners: Erin Carson, Emily Lam, Ruixin Yang,

Things Real Computers Can Do

Generate Heat

Stop a Door

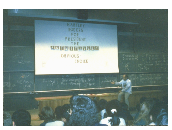
Provide an adequate habitat for fish

Computational Thing Most Real Computers Can Do (that Turing Machines can't)

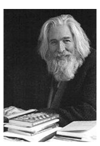
Generate randomness

Church-Turing Thesis

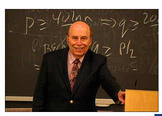
Alonzo Church's "Less Successful" PhD Students



Hartley Rogers



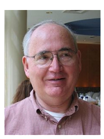
Raymond Smullyan



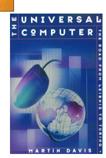
Michael Rabin

John Kemeny

Stephen Kleene



Dana Scott



Martin Davis

See http://www.genealogy.ams.org/id.php?id=8011 for full list

Alan Turing (1912-1954)

- Published *On Computable Numbers, with an Application to the Entscheidungsproblem* (1936)
  - Introduced the Halting Problem
  - Formal model of computation (now known as "Turing Machine")
- Codebreaker at Bletchley Park
  - Involved in breaking Enigma Cipher
- After the war: convicted of homosexuality (then a crime in Britain), committed suicide eating cyanide apple

Church-Turing Thesis

- As stated by Kleene:

  Every effectively calculable function (effectively decidable predicate) is general recursive.

  "Since a precise mathematical definition of the term effectively calculable (effectively decidable) has been wanting, we can take this thesis … as a definition of it…"

Yes, this is circular: everything calculable can be computed by a TM, and we define what is calculable as what can be computed by a TM.

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 model any mechanical computer with a TM
- The set of languages that can be decided by a TM is identical to the set of languages that can be decided by any mechanical computing machine
- If there is no TM that decides problem P, there is no algorithm that solves problem P.

All of these statements are implied by the Church-Turing thesis

Examples

- [Last class and PS4] Equivalence of TM and 2-stack deterministic PDA + ε-transitions
- [PS4] Making the tape infinite in both directions adds no power
- [Soon] Adding a second tape adds no power
- [Church] Lambda Calculus is equivalent to TM
- [Chomsky] Unrestricted replacement grammars are equivalent to TM
- [Takahara and Yokomori] DNA is at least as powerful as a TM
- [Hotly Debated] Is the human brain equivalent to a TM?

"Some of these models are very much like Turing machines, but others are quite different." (not such a bogus sentence)

Charge

- Next week: what languages cannot be recognized by a TM?
- Read Chapter 4: Decidability
  - I don't think it has any extremely bogus sentences, but if you find one send it to me…