Universal Turing Machines and Diagonalization

Mark Greenstreet, CpSc 421, Term 1, 2006/07

- Universal Turing Machines
  - A Turing Machine that can be $programmed$ to simulate any other Turing Machine.
- Diagonalization
  - A way to show compare the sizes of infinite sets.
  - On Monday, we'll use it to give a formal proof that the Halting Problem is undecidable.

Some "Universal" Languages

- $A_R = \{D\#w \mid D$ describes a DFA that accepts string $w\}$
  - This is the "universal" language for Regular Languages.
  - We described a Turing Machine for A_R in the Nov. 1 lecture.
- $A_{CFL} = \{G\#w \mid G$ describes a CFG that generates string $w\}$
  - This is the "universal" language for Context-Free Languages.
  - We described a Turing Machine for A_{CFL} in the Nov. 1 lecture.
- $A_{TM} = \{M\#w \mid M$ describes a TM that accepts string $w\}$
  - This is the "universal" language for Turing Recognizable Languages.
  - We'll described a Turing Machine for A_{TM} now.

A Universal Turing Machine

$A_{TM} = \{M\#w \mid M$ describes a TM that accepts string $w\}$

We'll define a Turing Machine, U, that recognizes A_{TM}.

Σ_U: $\{\mathtt{0}, \mathtt{1}, \mathtt{(}, \mathtt{,}, \mathtt{)}, \mathtt{\#}\}$

Γ_U: $\Sigma \cup \{\Box, \ldots\}$

w: The format for the input tape is described on the next slide.

Tapes: We'll use six tapes:

$input$	$=$	The input string, $M\#w$ is written here.
δ_M	$=$	A list of tuples representing the transition function of M is written here.
q_M	$=$	The current state of M is written here.
c_M	$=$	The current tape symbol of M is written here.
$tape_M$	$=$	The current tape contents for M.
$scratch$	$=$	A scratch tape.

Input Tape Format for U

$|Q_M|\mathtt{,}|\Sigma_M|\mathtt{,}|\Gamma_M|\delta_M\#w$ where

$|Q_M|$: Binary representation of the number of states of M.

$|\Sigma_M|$: Binary representation of the number of symbols in the input alphabet of M.

$|\Gamma_M|$: Binary representation of the number of symbols in the tape alphabet of M.

δ_M: A list of tuples for the transition function for M. Each tuple has the form: (q, c, q', c', d) where $\delta_M(q, c) = (q', c', d)$. In other words, when M is in state q and reads c, it transitions to state q', writes a c' on the tape and moves one square in direction d, $d \in \{\mathtt{0}, \mathtt{1}\}$, where $\mathtt{0}$ denotes a left move and $\mathtt{1}$ denotes a right move.

q_0, $accept$, and $reject$: we assume that these special states are represented by 0, 1, and 2 respectively.

w: The input string: binary numbers separated by commas. We assume that each symbol in Γ is encoded using the same number of bits, $\lceil \log_2 |\Gamma| \rceil$.

Operation of U (1/2)

Make sure the input is valid:

- Check that the tape has the form B^*,B^*,B^*C^*#B^*(,B^*)* where

  $B = \{0,1\}$

  $C = (B^*,B^*,B^*,B^*,B^*)$

  Note: This format requirement is a regular language. U can check this by scanning the tape from left-to-right using its finite states to implement a DFA.
- Read $|Q_D|$, $|\Sigma_D|$ and $|\Gamma_D|$.
- Copy δ_M onto the δ_M tape.
- Make sure that each tuple, (q, c, q', c', d) for δ_M has $q, q' \in 0 \ldots (|Q_D| - 1)$, $c, c' \in 0 \ldots (|\Gamma_D| - 1)$, $d, \in B$. Make sure all combinations for q and c are covered.
- Copy w onto the $tape_M$ tape —
  write the binary string for M's blank if $w = \epsilon$.
- Make sure that each symbol for w is in Σ_D.

Operation of U (2/2)

- Simulate M.

```
q ← 0
while(q ∉ {1,2}) {
  c ← string in B* under head on tape_M.
    (if there is a blank under the head, write a com
     and the binary string for M's blank)
  scan δ_M tape to find entry for (q,c),
    let this be (q,c,q',c',d)
  copy q' onto the q tape.
  copy c' onto the tape_M tape.
  move head for tape_M according to d.
}
if(q == 1) accept;
else reject.
```

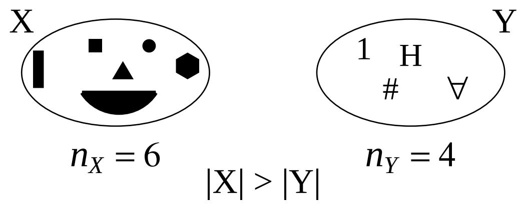
Observations

- If M accepts w, then U accepts $M\#w$.
- If M rejects w, then U rejects $M\#w$.
- If M loops on w, then U loops on $M\#w$.
- $\therefore U$ recognizes A_{TM}.
- U is universal:
  - One machine U works with any input $M\#w$.
    In other words, U can simulate any Turing machine, M.
  - You can think of the M part of $M\#w$ as a program, and the w part as the input data for the program.
  - U is a programmable machine. Rather than building a new TM for each problem, we just program U to simulate whatever TM we want.

Halting for Turing Machines

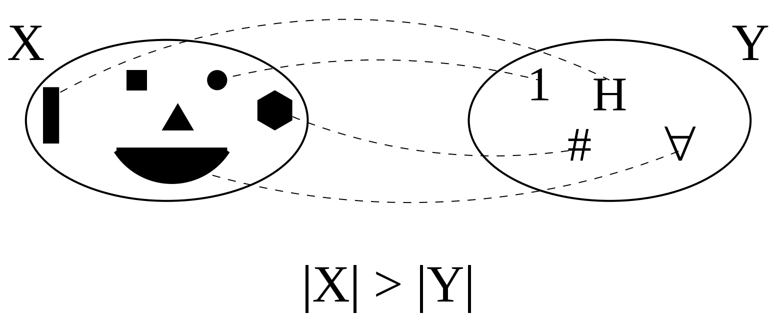
- From the previous slide, U loops on input $M\#w$ iff M loops on input w.
- We've shown that U *recognizes* A_{TM}, but it doesn't *decide* A_{TM}.
- Could we build some other machine, U' that can determine when a machine M loops on its given input? If so, then U' would decide A_{TM}.
  - This would require solving the Halting Problem for Turing Machines.begin
  - We gave an informal argument (see the Oct. 23 slides) that the Halting Problem for Java^{TM} programs is undecidable (by Java programs). On Monday (Nov. 6), we'll show that the Halting Problem for Turing Machines is undecidable.
  - First, we'll look at "diagonalization", the main mathematical concept that we'll need for the proof.

Which Set is Bigger?



- Let X and Y be sets.
- Is $|X| > |Y|$?
- Solution by counting:
  - Count each element in X. Let n_X be the number.
  - Count each element in Y. Let n_Y be the number.
  - If $n_X > n_Y$, then $|X| > |Y|$.

Comparing by Pairing



- If there is an onto function, $f : X \to Y$, then $|X| \geq |Y|$.
- If there are onto functions $f : X \to Y$ and $g : Y \to X$, then $|X| \geq |Y|$ and $|Y| \geq |X|$, Thus, $|X| = |Y|$.
- Note that if $f : X \to Y$ is one-to-one and onto, then f^{-1} exists and is one-to-one, and onto as well. Thus, if there is a one-to-one and onto function, $f : X \to Y$, then $|X| = |Y|$.
- If there is no onto function $g : Y \to X$, then $|X| > |Y|$.

Even Integers vs. All Integers

- Let $\mathbb{Z}$ be the set of all integers, and $\mathbb{E}$ be the set of all even integers.
  - Let $f : \mathbb{Z} \rightarrow \mathbb{E}$ be the function $f(x) = 2x$.
  - f is one-to-one: If $f(x) = f(y)$, then $2x = 2y$, and $x = y$.
  - f is onto: If $y \in \mathbb{E}$, then $y/2 \in \mathbb{Z}$, and $f(y/2) = y$.
  - $\therefore \mathbb{E} = \mathbb{Z}$.
    In English, this says that the number of even integers is equal to the number of all integers!
- A similar argument shows that $|\mathbb{N}| = |\mathbb{Z}|$.

Naturals vs. Rationals (1/2)

- Let $\mathbb{Q}^+$ be the set of all strictly-positive rational numbers, and $\mathbb{N}^+$ be the strictly-positive naturals.
- Let $f : \mathbb{Q}^+ \rightarrow \mathbb{N}^+$ with $f(x) = \lceil x \rceil$. Clearly, f is onto, thus $|\mathbb{Q}^+| \geq |\mathbb{N}^+|$ — there are at least as many positive rational numbers as positive naturals.
- Let $g : \mathbb{Q}^+ \rightarrow \mathbb{N}^+$ with

$$\begin{aligned} g(n) &= \frac{x(n)+1-zn}{z(n)} \\ x(n) &= \left\lfloor \frac{1}{2}(\sqrt{8n-7}\ +\ 1) \right\rfloor \\ y(n) &= \frac{1}{2}(x(n)^2 - x(n)) \\ z(n) &= n - y(n) \end{aligned}$$

For example:

n	1	2	3	4	5	6	7	8	9	10	11	...
$x(n)$	1	2	2	3	3	3	4	4	4	4	5	...
$y(n)$	0	1	1	3	3	3	6	6	6	6	10	...
$z(n)$	1	1	2	1	2	3	1	2	3	4	1	...
$g(n)$	$\frac{1}{1}$	$\frac{2}{1}$	$\frac{1}{2}$	$\frac{3}{1}$	$\frac{2}{2}$	$\frac{1}{3}$	$\frac{4}{1}$	$\frac{3}{2}$	$\frac{2}{3}$	$\frac{1}{4}$	$\frac{5}{1}$	...

Naturals vs. Rationals (2/2)

- Visualizing $g(n)$.

$$\begin{array}{cccccc} \frac{1}{1} & \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \cdots \\ \frac{2}{1} & \frac{2}{2} & \frac{2}{3} & \frac{2}{4} & \cdots & \\ \frac{3}{1} & \frac{3}{2} & \frac{3}{3} & \cdots & & \\ \frac{4}{1} & \frac{4}{2} & \cdots & & & \\ \frac{5}{1} & \cdots & & & & \\ \vdots & \cdots & & & & \end{array}$$

Naturals vs. the Reals

- Let $V = [0, 1)$ be a half-open, interval of real numbers.
- We'll show that $|V| > |\mathbb{N}|$. Clearly $|V| \leq |\mathbb{R}|$ (in fact, $|V| = |\mathbb{R}|$). Thus, this will show that $|\mathbb{R}| > |\mathbb{N}|$.
- The proof is by contradiction.
  - Assume that $|\mathbb{R}| \leq |\mathbb{N}|$.
  - This means that there exists an onto function $g : \mathbb{N} \to \mathbb{R}$.
  - On the next slide, we'll show that this leads to a contradiction. The argument we use is called a $diagonalization$ argument.
  - g is not onto, a contradiction. This shows that g cannot exist.
  - $\therefore$, $|[0, 1)| > |\mathbb{N}|$. which implies $|\mathbb{R}| > |\mathbb{N}|$.

Diagonalization

- Let $digit(x, k)$ denote the k^{th} digit after the decimal point of x. For example, $digit(0.707106, 4) = 1$, and $digit(\sqrt{\frac{1}{2}}, 40) = 8$.
- Let $y = \sum_{m=1}^{\infty} ((digit(g(m), m) \bmod 8) + 1) \times 10^{-m}$.
  This choice of digits has two handy properties:
  - For all m, $digit(y(m), m) \neq digit(g(m), m)$.
  - All digits are in $\{1, 2, 3, 4, 5, 6, 7, 8\}$. This avoids having to deal with problematic valus for y such as $0.19999999999\ldots$ which is equal to 0.2, or $0.999999999999\ldots$ which is not in $[0, 1)$.
- $y \in [0, 1)$, and $\forall m.\ y \neq g(m)$.
- g is not onto, a contradiction. This shows that g cannot exist.

Diagonalization (2/2)

- Consider the following example of a possible function for g:

m	$g(m)$
0	0.<u>9</u>50129285147175
1	0.2<u>3</u>1138513574288
2	0.60<u>6</u>842583541787
3	0.485<u>7</u>82468709300
4	0.8912<u>8</u>8966148902
5	0.76209<u>6</u>833027395
6	0.456467<u>4</u>65168341
7	0.0185036<u>4</u>3248224
8	0.82140716<u>4</u>295253
9	0.444703364<u>3</u>53194
⋮	⋮

- Then y constructed as described on the previous slide will be $0.2378175554\ldots$. Note that for each m, the m^{th} digit of y is different than the m^{th} digit of $g(m)$. Thus, y is guaranteed *not* to appear on the list.

Reading List:

- Today: Sipser, 4.2
- Nov. 6: Sipser, 4.2 (cont., midterm 2 cutoff)
- Nov. 8: Sipser, 5.1
- Nov. 10: Sipser, 5.1 (cont.)
- Nov. 13: Remembrance Day (no lecture)
- Nov. 15: Midterm 2
- Nov. 17: Sipser, 5.2
- Nov. 20: Sipser, 5.3