CS21
Decidability and Tractability

Lecture 10

January 29, 2014

Problem Set + grading

- 3 points for each part of each problem
- PS1: 24 points total
  - mean: 17.2 median: 19.5

    2013: 19.5, 20
    2012: 19.6, 21
    2011: 18.7, 19
    2010: 19.3, 20
    2009: 20.0, 21
    2008: 20.6, 21

Problem set + grading

- An idea of eventual scale:

  2012: mean 79.9; median 79.6
  2011: mean 75.9; median 76.4
  2010: mean 74.7; median 76.0
  2009: mean 84.8; median 85.5

2012		2011		2010		2009	
98-100	A+	97-100	A+	97-100	A+	97-100	A+
93-97	A	91-96	A	91-96	A	93-96	A
87-92	A-	85-90	A-	87-90	A-	89-92	A-
82-86	B+	80-84	B+	81-86	B+	85-88	B+
77-81	B	75-79	B	75-80	B	80-84	B
74-76	B-	71-74	B-	72-74	B-	77-79	B-
70-73	C+	68-70	C+	68-71	C+	73-76	C+
66-69	C	64-67	C	64-67	C	69-72	C
63-65	C-	57-63	C-	61-63	C-	64-68	C-
57-62	D+	52-56	D+	57-60	D+	61-63	D+
52-56	D	48-51	D	53-56	D	55-60	D
<52	E/F	< 48	E/F	<52	E/F	<55	E/F

Outline

- Church-Turing Thesis
- decidable, RE, co-RE languages
- the Halting Problem
- reductions

Examples of basic operations

- Convince yourself that the following types of operations are easy to implement as part of TM "program"

  (but perhaps tedious to write out…)
  - copying
  - moving
  - incrementing/decrementing
  - arithmetic operations +, -, *, /

Universal TMs and encoding

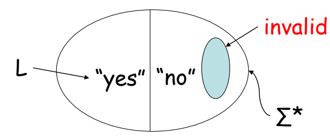
- the input to a TM is always a string in Σ^*
- often we want to interpret the input as representing another object
- examples:
  - tuple of strings (x, y, z)
  - 0/1 matrix
  - graph in adjacency-list format
  - Context-Free Grammar

Universal TMs and encoding

- the input to a TM is always a string in Σ^*
- we must encode our input as such a string
- examples:
  - tuples separated by #: #x#y#z
  - 0/1 matrix given by: #n#x# where $x \in \{0,1\}^{n^2}$
- any reasonable encoding is OK
- emphasize "encoding of X" by writing <X>

Universal TMs and encoding

- some strings not valid encodings and these are not in the language



make sure TM can recognize invalid encodings and reject them

Universal TMs and encoding

- We can easily construct a Universal TM that recognizes the language:

  A_{TM} = {<M, w> : M is a TM and M accepts w}
  - how?
- this is a remarkable feature of TMs (not possessed by FA or NPDAs…)
- means there is a general purpose TM whose input can be a "program" to run

Church-Turing Thesis

- many other models of computation
  - we saw multitape TM, nondeterministic TM
  - others don't resemble TM at all
  - common features:
    - unrestricted access to unlimited memory
    - finite amount of work in a single step
- every single one can be simulated by TM
- many are equivalent to a TM
- problems that can be solved by computer does not depend on details of model!

Church-Turing Thesis

- the belief that TMs formalize our intuitive notion of an algorithm is:

  The Church-Turing Thesis

  everything we can compute on a physical computer

  can be computed on a Turing Machine
- Note: this is a belief, not a theorem.

Recursive Enumerability

- Why is "Turing-recognizable" called RE?
- Definition: a language $L \subset \Sigma^*$ is recursively enumerable if there is exists a TM (an "enumerator") that writes on its output tape

  $\#x_1\#x_2\#x_3\#\ldots$

  and $L = \{x_1, x_2, x_3, \ldots\}$.
- The output may be infinite

Recursive Enumerability

Theorem: A language is Turing-recognizable iff some enumerator enumerates it.

Proof:

($\Leftarrow$) Let E be the enumerator. On input w:

- Simulate E. Compare each string it outputs with w.
- If w matches a string output by E, accept.

Recursive Enumerability

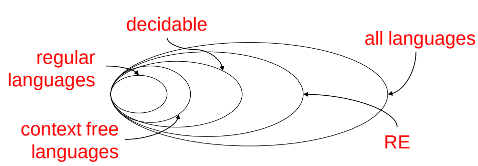
Theorem: A language is Turing-recognizable iff some enumerator enumerates it.

Proof:

($\Rightarrow$) Let M recognize language $L \subset \Sigma^*$.

- let $s_1, s_2, s_3, \ldots$ be enumeration of Σ^* in lexicographic order.
- for i = 1,2,3,4,…
  - simulate M for i steps on $s_1, s_2, s_3, \ldots, s_i$
- if any simulation accepts, print out that s_j

Undecidability



decidable $\subset$ RE $\subset$ all languages

our goal: prove these containments proper

Countable and Uncountable Sets

- the natural numbers **N** = {1,2,3,…} are countable
- Definition: a set S is countable if it is finite, or it is infinite and there is a bijection

  f: **N** $\rightarrow$ S

Countable and Uncountable Sets

- Theorem: the positive rational numbers Q = {m/n : m, n $\in$ **N** } are countable.
- Proof:

1/1	1/2	1/3	1/4	1/5	1/6	…
2/1	2/2	2/3	2/4	2/5	2/6	…
3/1	3/2	3/3	3/4	3/5	3/6	…
4/1	4/2	4/3	4/4	4/5	4/6	…
5/1	…					

Countable and Uncountable Sets

Theorem: the real numbers **R** are NOT countable (they are "uncountable").

- How do you prove such a statement?
  - assume countable (so there exists bijection f)
  - derive contradiction (some element not mapped to by f)
  - technique is called diagonalization (Cantor)

Countable and Uncountable Sets

- Proof:
  - suppose **R** is countable
  - list **R** according to the bijection f:

n	f(n)
1	3.14159…
2	5.55555…
3	0.12345…
4	0.50000…
…	

Countable and Uncountable Sets

- Proof:
  - suppose **R** is countable
  - list **R** according to the bijection f:

n	f(n)
1	3.14159…
2	5.55555…
3	0.12345…
4	0.50000…
…	

set $x = 0.a_1a_2a_3a_4\ldots$

where digit $a_i \neq i^{th}$ digit after decimal point of f(i) (not 0, 9)

e.g. x = 0.2312…

x cannot be in the list!

non-RE languages

Theorem: there exist languages that are not Recursively Enumerable.

Proof outline:
- the set of all TMs is countable
- the set of all languages is uncountable
- the function L:{TMs} →{languages} cannot be onto

non-RE languages

- Lemma: the set of all TMs is countable.
- Proof:
  - each TM M can be described by a finite-length string <M>
  - can enumerate these strings, and give the natural bijection with **N**

non-RE languages

- Lemma: the set of all languages is uncountable
- Proof:
  - fix an enumeration of all strings $s_1, s_2, s_3, \ldots$ (for example, lexicographic order)
  - a language L is described by its characteristic vector χ_L whose i^{th} element is 0 if s_i is not in L and 1 if s_i is in L

non-RE languages

- suppose the set of all languages is countable
- list characteristic vectors of all languages according to the bijection f:

n	f(n)
1	0101010…
2	1010011…
3	1110001…
4	0100011…
…	

non-RE languages

– suppose the set of all languages is countable

– list characteristic vectors of all languages according to the bijection f:

n	f(n)
1	0101010…
2	1010011…
3	1110001…
4	0100011…
…	

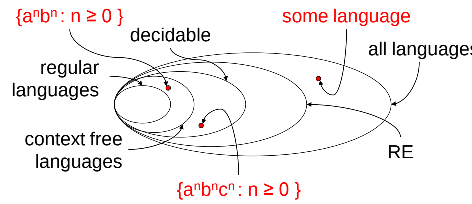
set x = 1101…

where i^{th} digit ≠ i^{th} digit of f(i)

x cannot be in the list!

therefore, the language with characteristic vector x is not in the list

So far…



- This language might be an esoteric, artificially constructed one. Do we care?
- We will show a natural undecidable L next.

The Halting Problem

- Definition of the “Halting Problem”:
  HALT = { <M, x> : TM M halts on input x }
- HALT is recursively enumerable.
  – proof?
- Is HALT decidable?