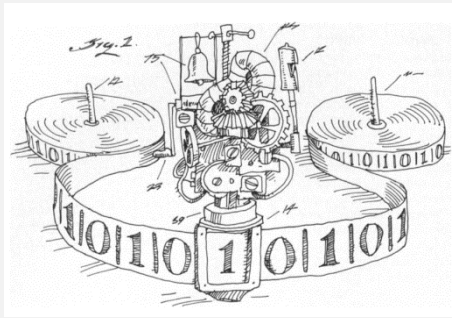
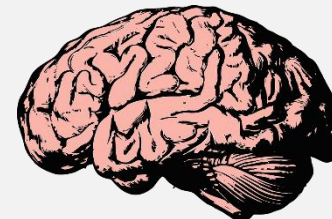
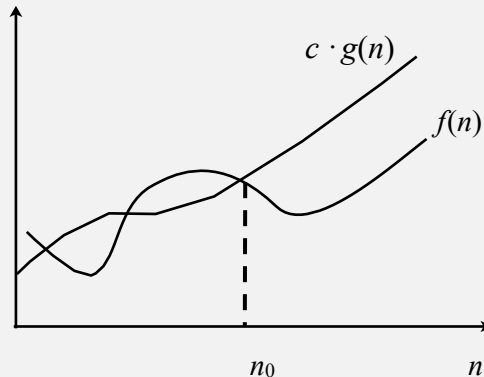


COMP 3602

Intro to Theory of Computation

Lecture 9



Logistics

- Test 1 day/time: Mon Feb 9
 - Written test on Turing machines, automata/regex, PDA/CFGs, pumping lemmas
 - Practice problems are up
 - One **letter-size double-sided handwritten** cheatsheet allowed
- First oral exam: as soon as we finish the next topic
- Presentation proposals?
- Drills are up, some due Feb 9th .
 - Do the sample drills first!
 - You can do the drills multiple times, but the grade is an average of all attempts
 - So if you open it and let it time out, it will add a 0 to your average!

Practice problems questions?

1. Give regular expressions and draw automata (DFAs or NFAs) for the following languages:
 - a. $\{w \in \{0,1\}^* \mid w \text{ consists only of the string } 010\}$
 - b. $\{w \in \{0,1\}^* \mid w \text{ contains a substring } 010\}$
 - c. $\{w \in \{0,1\}^* \mid w \text{ contains a substring } 010, \text{ starts with } 0 \text{ and ends with } 1\}$
 - d. $\{w \in \{0,1\}^* \mid w \text{ does not contain a substring } 010\}$
 - e. $\{w \in \{a,b,c\}^* \mid w \text{ contains exactly 2 "c" and at least one "a" }\}$
 - f. $\{w \in \{a,b,c\}^* \mid \text{length of } w \text{ is divisible by 5 and it ends at "a" }\}$
2. Give NFAs with the specified number of states for the following languages, then use the subset construction to get DFAs out of them.
 - a. $L = 1^*$, one state.
 - b. Strings with all 1s (if any) in the same substring (That is, strings without a substring 100^*1). Three states.
 - c. $0^*(1000^*)^*$ with three states (here, "1000*" means 100 followed by 0^*)
3. Give a pushdown automaton and a context-free grammar for the following languages (for extra practice, try writing grammars for languages above)
 - a. $\{w \in \{0,1\}^* \mid \text{the length of } w \text{ is odd}\}$
 - b. $\{w \in \{a,b,c\}^* \mid w \text{ contains more c's than a's}\}$
 - c. $\{w \in \{0,1\}^* \mid w \text{ is a palindrome or } w=0^n u 0^n \text{ for some } n, \text{ where } u \in \{0,1\}^* \}$
4. Show using the pumping lemmas that
 - a. $\{a^n b \{a,b,c,d\}^* c d^{2n}\}$ is not regular.
 - b. $\{w \in \{0,1\}^* \mid \text{contains a substring } 10 \text{ or } w=0^n 1^m \text{ where } n \neq m \}$ is not regular. Hint: use closure properties of regular languages.
 - c. $\{ww^r w \mid w = w_1 \dots w_n \in \{0,1\}^* \text{ where } w^r = w_n \dots w_1 \text{ is the reverse of } w \}$ is neither regular nor context-free.
 - d. $\{1^{n^2}\}$ is neither regular nor context-free.

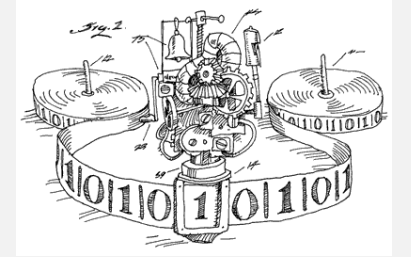
5. Describe Turing machines for the following languages. Use the definition of Turing machines with a single 2-way infinite tape, and movement directions $\{L,R\}$. Make sure to include both the informal outline of the algorithm, and the full description, that is, Σ , Γ , Q , q_0 , q_a , q_r and δ . Do not draw diagrams for Turing machines! Hint: first look at a few examples of strings that are in the language and strings that are not in the language, and don't forget the empty string.
 - $L_1 = \{w \in \{b,c,d\}^* \mid \text{non-empty strings of length at least 3 with the last three symbols all different.}\}$
 - $L_2 = \{w \in \{0,1\}^* \mid w=0^n1^n0^{2n} \text{ for } n \geq 0 \}$. That is, L_2 consists of all strings of length divisible by 4 (including length 0), where the first quarter of the string is all 0s, second quarter is all 1s, and the second half all 0s again.
6. Describe a Turing machine that decrements by 2 a natural number ≥ 2 written on its tape in binary. It should end in q_a pointing to the first symbol of the output if the tape contained a string starting with 1 encoding a natural number $x \geq 2$, and in q_r otherwise. Give both an informal algorithm and a full description of the TM.
7. Consider a TM where instead of "move left one cell" and "move right one cell" the only available moving operations are "jump left 11 steps", "jump right 5 steps" and "stay at the same place"; let's call this model 11L5RS Turing machines. Just as in the model of TM we defined in class, 11L5RS TMs have one two-way-infinite tape, the only difference is the movement operations.
 - Show that 11L5RS TM model is equivalent to the model we defined in class. Do both directions, with constructions (from M construct M'), and an outline of a proof why your construction works.
 - Show that if instead it were "jump left 10 steps" "jump right 5 steps", then this 10L5RS TM model would not be able to simulate the standard TM.
8. Show that single-tape Turing machines that only move right recognize exactly the regular languages. Hint: do both directions of the equivalence, and remember that the Turing machine is not guaranteed to stop.
9. Show that finite automata with two stacks can simulate Turing machines.

Do not draw diagrams for Turing machines!

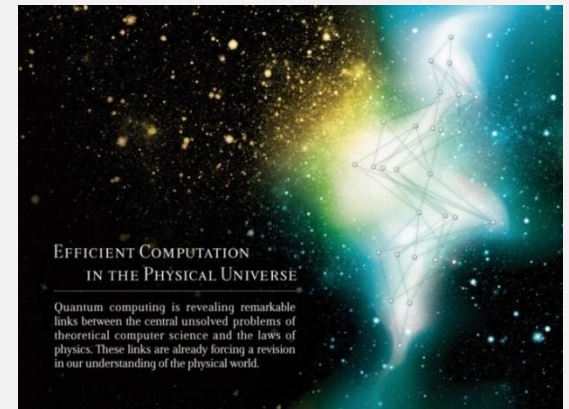
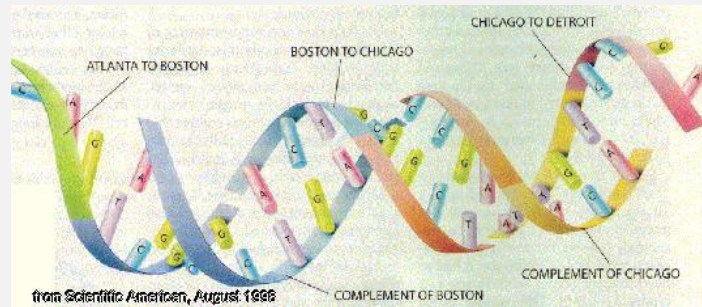
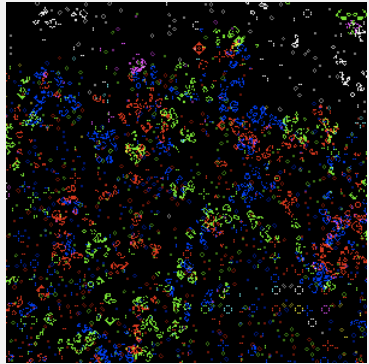
Questions/comments?

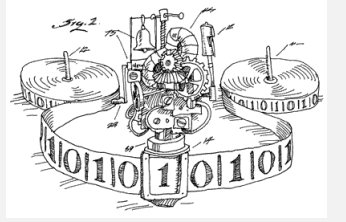


Church-Turing thesis



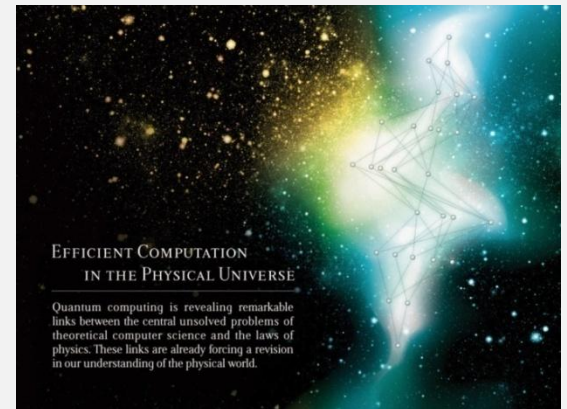
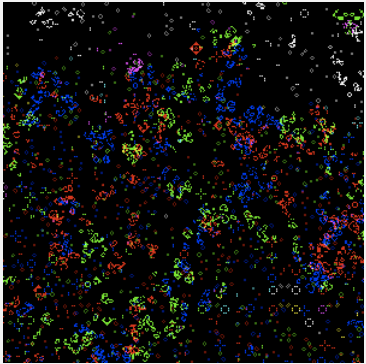
Everything we can call “computable” in any sense of this word is computable by a Turing machine.





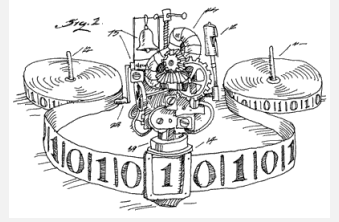
Everything we can call “computable” in any sense of this word is computable by a Turing machine.

- Is there anything not computable by Turing machines?
- And if so, what do these languages look like?





Turing-recognizable and decidable languages



- A TM M *accepts (recognizes) a language* L iff it stops in the state q_{accept} for all and only strings in L
 - It does not have to stop for strings not in L .
 - If it also stops and rejects for all strings not in L , then M *decides* L .
- A language L is *Turing-recognizable* iff there exists a Turing machine M recognizing it, that is, $\mathcal{L}(M) = L$
 - Turing-recognizable is also called recursively enumerable, r.e., semi-decidable
- A language L is *decidable* iff there exists a Turing machine M **which halts on all inputs** such that $\mathcal{L}(M) = L$
 - Decidable is also called recursive.



Review: bijections and countability

- A function $f: A \rightarrow B$ is a *bijection* iff it is total, one-to-one and onto
 - Total: $\forall x \in A \exists y \in B f(x) = y$
 - One-to-one: $\forall x_1, x_2 \in A f(x_1) = f(x_2) \rightarrow x_1 = x_2$
 - Onto: $\forall y \in Y \exists x \in X f(x) = y$
 - Bijection: $(\forall x \in A \exists! y \in B f(x) = y) \wedge (\forall y \in B \exists! x \in A f(x) = y)$
- A set S is countable (countably infinite) iff there is a bijection $f: S \rightarrow \mathbb{N}$
 - An infinite set which is not countable is called uncountable.
- Theorem: An infinite subset of a countable set is countable.
 - That is, if $image(f) \subseteq \mathbb{N}$ is infinite, then S is countable.
 - Only need to show that f is total and one-to-one.



Lemma: Any Turing machine can be encoded as a binary string

- We will use the notation $\langle M \rangle$ to mean a binary string encoding M
- Proof skipped
 - Any encoding would do, as long as we use the same encoding for all TMs

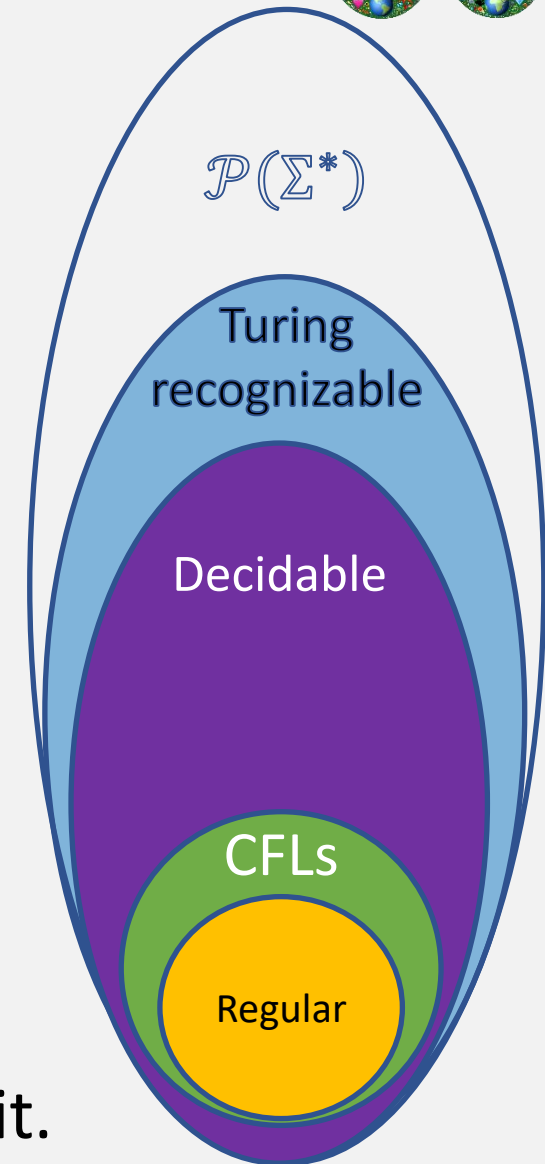
Theorem: Set of all Turing machines is **countable**.

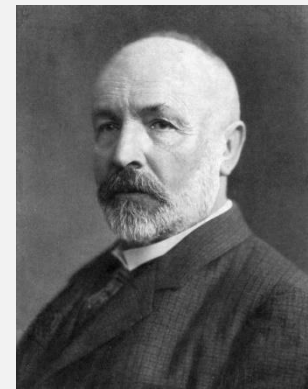
Proof: We will construct a total one-to-one f from the set of all TMs to $\mathbb{N}$.

- For a TM M , define $f(M) = 1\langle M \rangle$
- $1\langle M \rangle$ is a binary string starting with 1, and so represents a natural number.
 - And there are infinitely many TMs: can always add another state.
- f is total: for every M , there exists $\langle M \rangle$ by the lemma above, and so $1\langle M \rangle$
- f is one-to-one: for different TMs, their encodings and so $1\langle M \rangle$ are different.



- *Theorem:* Set of all languages is **uncountable**.
 - Here, we'll prove it for languages over $\{0,1\}^*$.
 - Same argument for any Σ
- Proof: by diagonalization (next slide)
- *Corollary:* There exists a language L not computable by any TM
 - In fact, there are uncountably many such languages!
- *Definition:*
 - A language L is *undecidable* if no Turing machine decides it.
 - It is not Turing-recognizable if no Turing machine recognizes it.

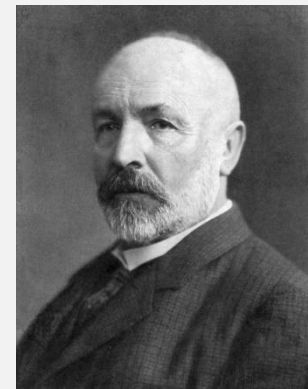




Diagonalization

- Let $A = \{a, b, c\}$. $\mathcal{P}(A) = \{\emptyset, \{a\}, \{b\}, \{c\}, \{a, b\}, \{a, c\}, \{b, c\}, \{a, b, c\}\}$
- Let's show that $|A| < |\mathcal{P}(A)|$
 - Imagine that someone is trying to convince you that $|A| = |\mathcal{P}(A)|$. Then they should give you a bijection $f: A \rightarrow \mathcal{P}(A)$
 - That is, give you $S_1, S_2, S_3 \in \mathcal{P}(A)$ such that $f(a) = S_1, f(b) = S_2, f(c) = S_3$ and there is nothing in $\mathcal{P}(A)$ except S_1, S_2, S_3 .
 - Say they gave you $S_1 = \{a\}$, $S_2 = \{b, c\}$ and $S_3 = \emptyset$. Here is how you can construct a set in $\mathcal{P}(A)$ which is not in the list S_1, S_2, S_3 .
 - Make a table with rows marked by elements of A
 - To represent a set $S \in \mathcal{P}(A)$, use its characteristic string: say for every element of the universe of S , A , if it is in S or not.

	<i>a</i>	<i>b</i>	<i>c</i>
$f(a) = S_1$	1	0	0
$f(b) = S_2$	0	1	1
$f(c) = S_3$	0	0	0



Diagonalization

- Let $A = \{a, b, c\}$. $\mathcal{P}(A) = \{\emptyset, \{a\}, \{b\}, \{c\}, \{a, b\}, \{a, c\}, \{b, c\}, \{a, b, c\}\}$
- Let's show that $|A| < |\mathcal{P}(A)|$
 - Imagine that someone is trying to convince you that $|A| = |\mathcal{P}(A)|$. Then they should give you a bijection $f: A \rightarrow \mathcal{P}(A)$
 - That is, give you $S_1, S_2, S_3 \in \mathcal{P}(A)$ such that $f(a) = S_1, f(b) = S_2, f(c) = S_3$ and there is nothing in $\mathcal{P}(A)$ except S_1, S_2, S_3 .

	<i>a</i>	<i>b</i>	<i>c</i>
$f(a) = S_1$	1	0	0
$f(b) = S_2$	0	1	1
$f(c) = S_3$	0	0	0
D	0	0	1

- Say they gave you $S_1 = \{a\}$, $S_2 = \{b, c\}$ and $S_3 = \emptyset$. Here is how you can construct a set in $\mathcal{P}(A)$ which is not in the list S_1, S_2, S_3 .
 - Make a table with rows marked by elements of A
 - To represent a set $S \in \mathcal{P}(A)$, use its characteristic string: say for every element of the universe of S , A , if it is in S or not.
- Now, the diagonal set is $D = \{c\}$ is in $\mathcal{P}(A)$, but it is not one of S_1, S_2, S_3 .
- Therefore, f cannot be a bijection! And this works for any S_1, S_2, S_3 .



- *Theorem*: Set of all languages is **uncountable**.
 - Here, we'll prove it for languages over $\{0,1\}^*$. Same argument for any Σ
- Proof: by diagonalization
 - Encode each language L by its infinite characteristic string
 - List strings in string order (shortlex)
 - First, order by increasing length
 - Within each length, in lexicographic order
 - Put “yes” if string $s \in L$, “no” if $s \notin L$
 - Let the language D be: $\{s_i \mid s_i \notin L_i\}$
 - If D were in the list, e.g. L_k , contradiction:
 - It would have to differ from itself in k^{th} place.

	λ	0	1	00	01	...	s_k		
L_1	yes	yes	no	yes	yes	...			
L_2	yes	no	yes	no	yes	...			
L_3	no	no	no	no	no	...			
...									
L_k	no	yes	yes	no	yes	...	yes	..	
...									
D	no	yes	yes	...	...	...	no	..	

- *Lemma:* Any Turing machine can be encoded as a binary string
 - We will use the notation $\langle M \rangle$ to mean a binary string encoding M
- *Theorem:* Set of all Turing machines is **countable**.
 - Proof: match encodings of TMs to binary numbers
- *Theorem:* Set of all languages is **uncountable**.
 - Proof: by diagonalization
- *Corollary:* There exists a language not recognizable by any TM
 - In fact, there are uncountably many such languages!
- *Definition:*
 - A language L is *undecidable* if no Turing machine decides it.
 - It is not Turing-recognizable if no Turing machine recognizes it.

