CS6160 Theory of Computation
Problem Set 8

Department of Computer Science, University of Virginia

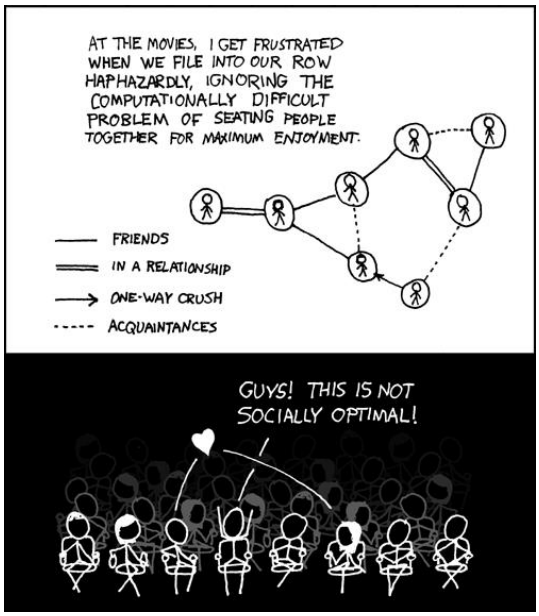
Gabriel Robins

Please start solving these problems immediately, don't procrastinate, and work in study groups. Please prove all your answers; informal arguments are acceptable, but please make them precise / detailed / convincing enough so that they can be easily made rigorous if necessary. To review notation and definitions, please read the "Basic Concepts" summary posted on the class Web site, and also read the corresponding chapters from the Sipser textbook and Polya's "How to Solve It".

Please do not simply copy answers that you do not fully understand; on homeworks and on exams we reserve the right to ask you to explain any of your answers verbally in person (and we have exercised this option in the past). Please familiarize yourself with the UVa Honor Code as well as with the course Cheating Policy summarized on page 3 of the Course Syllabus. To fully understand and master the material of this course typically requires an average effort of at least six to ten hours per week, as well as regular meetings with the TAs and attendance of the weekly problem-solving sessions.

This is not a "due homework", but rather a "pool of problems" meant to calibrate the scope and depth of the knowledge / skills in CS theory that you (eventually) need to have for the course exams, becoming a better problem-solver, be able to think more abstractly, and growing into a more effective computer scientist. You don't necessarily have to completely solve every last question in this problem set (although it would be great if you did!). Rather, please solve as many of these problems as you can, and use this problem set as a resource to improve your problem-solving skills, hone your abstract thinking, and to find out what topics you need to further focus on and learn more deeply. Recall that most of the midterm and final exam questions in this course will come from these problem sets, so your best strategy of studying for the exams in this course is to solve (including in study groups) as many of these problems as possible, and the sooner the better!

Advice: Please try to solve the easier problems first (where the meta-problem here is to figure out which are the easier ones ☺). Don't spend too long on any single problem without also attempting (in parallel) to solve other problems as well. This way, solutions to the easier problems (at least easier for you) will reveal themselves much sooner (think about this as a "hedging strategy" or "dovetailing strategy").



1. True or false: If S is an infinite set of Boolean expressions, and every finite subset of S is satisfiable, then S itself is satisfiable.

2. Prove that NP is not equal to DSPACE(N). (Note: it is still an open question whether either one contains the other, although we know that they are not equal.)

3. Define the set of all prefixes of L as PREFIX(L) = $\{w \mid wy \in L \text{ for some } w,y \in \Sigma^*\}$. Does PREFIX preserve regularity? Context-freeness? Decidability? Turing-recognizability?

4. Define the set of all suffixes of L as SUFFIX(L) = $\{w \mid yw \in L \text{ for some } y,w \in \Sigma^*\}$. Does SUFFIX preserve regularity? Context-freeness? Decidability? Turing-recognizability?

5. Define the set of all subsequences of L as SUBSEQ(L) = $\{w_1w_2w_3\ldots w_k \mid \exists\, k \in \mathbb{N}, \exists\, w_i \in \Sigma^* \text{ for } 1 \le i \le k, \text{ and } \exists\, x_j \in \Sigma^* \text{ for } 0 \le j \le k \text{ such that } x_0w_1x_1w_2x_2w_3x_3\ldots w_kx_k \in L\}$. Does SUBSEQ preserve regularity? Context-freeness? Decidability? Turing-recognizability?

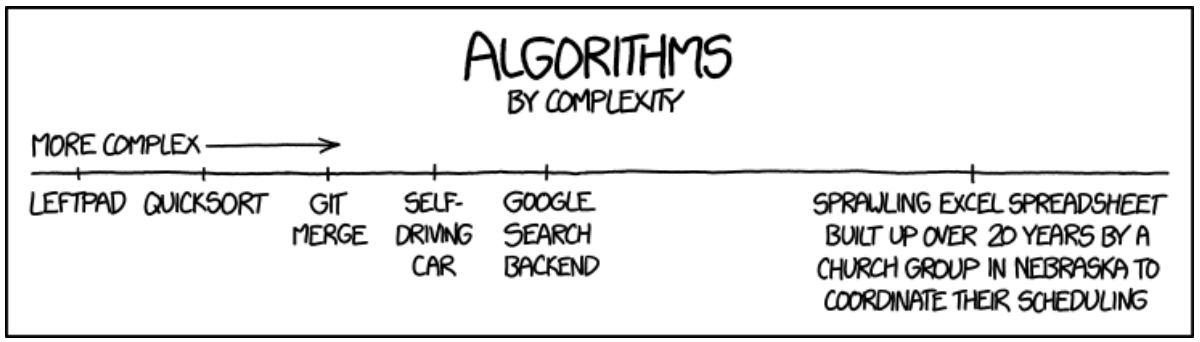
6. Define the set of all supersequences of L as SUPERSEQ(L) = $\{ x_0w_1x_1w_2x_2w_3x_3\ldots w_kx_k \mid \exists\, k \in \mathbb{N}, \exists\, w_i \in \Sigma^* \text{ for } 1 \le i \le k, \text{ and } \exists\, x_j \in \Sigma^* \text{ for } 0 \le j \le k \text{ such that } w_1w_2w_3\ldots w_k \in L\}$. Does SUPERSEQ preserve regularity? Context-freeness? Decidability? Turing-recognizability?

7. A language L is said to be "definite" if there exists some fixed integer k such that for any string w, whether $w \in L$ depends only on the last k (or less) symbols of w. (a) State this definition more formally. (b) Is a definite language necessarily regular? (c) Is the set of definite languages closed under union? Intersection? Complementation? Concatenation? Kleene closure?

8. Define the density of a language to be the function $D_L(n) = |\{w \mid w \in L \text{ and } |w| \le n\}|$. What is the density of $(a+b)^*$? What is the density of a^*b^* ? Show that the density of a regular language is either bounded from above by a polynomial, or bounded from below by an exponential (i.e., a function of the form 2^{cn} for some constant c). In other words, densities of regular languages can not be functions of intermediate growth such as $n^{\log n}$.

9. True or false: the densities of the decidable languages can be any computable function.

10. A string w is square-free if it can not be written in the form $w=xy^2z$ for some $x,z \in \Sigma^*$ and $y \in \Sigma^+$. Are there arbitrarily long square-free strings on a two-letter alphabet? How about a three-letter alphabet?



11. A string w is cube-free if it can not be written in the form $w=xy^3z$ for some $x,z\in\Sigma^*$ and $y\in\Sigma^+$. Are there arbitrarily long cube-free strings on a two-letter alphabet?

12. True or false: if $|\Sigma|=1$ then the set of all cube-free strings in Σ^* is regular.
True or false: if $|\Sigma|=2$ then the set of all square-free strings in Σ^* is regular.
True or false: if $|\Sigma|>2$ then the set of all square-free strings in Σ^* is not regular.

13. Define the operation ♣ on languages as follows: $\clubsuit(L) = \{w \mid \exists\ w \in \Sigma^*, ww^R \in L\}$, where w^R denotes the "reverse" of the string w. Does ♣ preserve context-freeness? Decidability? Recognizability?

14. Is a countably-infinite union of decidable languages necessarily decidable?
Is a countably-infinite intersection of decidable languages necessarily decidable?

15. Is a countably-infinite union of recognizable languages necessarily recognizable?
Is a countably-infinite intersection of recognizable languages necessarily recognizable?

16. Can a countably-infinite union of non-finitely-describable languages be regular?
Is a countably-infinite intersection of non-finitely-describable languages be regular?

17. Let $YESNO(L)=\{xy \mid x\in L \text{ and } y\notin L, x,y\in\Sigma^*\}$. Does YESNO preserve recognizability?

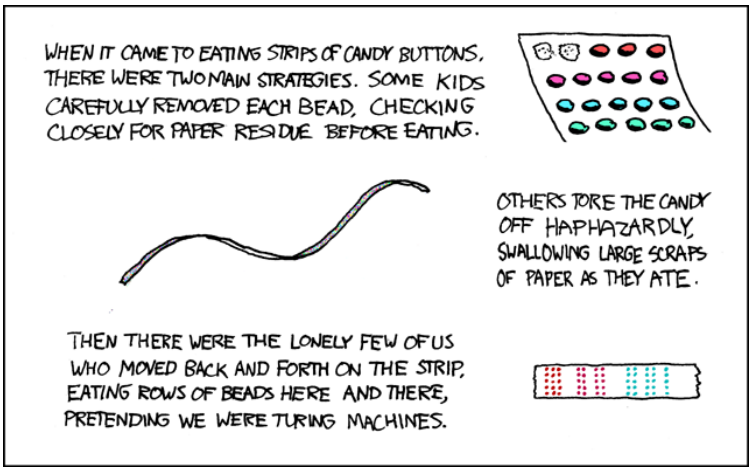
18. Let $PALI(L)=\{w \mid x\in L \text{ and } x^R\in L\}$. Does PALI preserve recognizability?

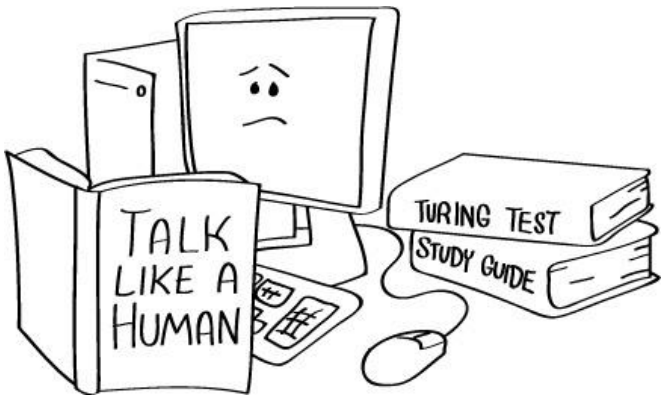
19. Define a DIVISION operator on languages as follows:

$$\frac{L_1}{L_2} = \{w \mid w \in \Sigma^* \text{ and } \exists\ v \in L_2 \ni wv \in L_1\}$$

(a) Does DIVISION preserve decidability?
(b) Does DIVISION preserve recognizability?

20. Give (and prove) several example non-Turing-recognizable languages.





21. Is it decidable whether given a given one-state PDA accepts all input strings?

22. Is it decidable whether given a given PDA accepts all input strings?

23. Is it decidable whether a given context-free grammars generates all strings?

24. Is it decidable whether given PDA has a minimum possible number of states?

25. Is it decidable whether two given PDAs accept the same language?

26. Is it decidable whether two given context-free grammars generate the same language?

27. Is it decidable whether there are any strings accepted by a given Turing machine?

28. Is it decidable whether a given unrestricted grammar accepts a given string?

29. Is it decidable whether there are any strings accepted by a given unrestricted grammar?