List of questions for INFO-F-408 – Calculability and Complexity

Gilles Geeraerts

2013–2014

Please note: the chapter numbers refer to the chapters of the reference book (Sipser).

1 Chapter 3: The Church-Turing Thesis

1. Explain how decision problems can be formalised by the notion of language of finite words. Explain how the following problem can be formalised this way: 'Determine whether a finite graph has a path from a node s to a node t'.
2. Define *formally* the notion of *configuration* of a Turing machine. Define the notion of *computation* of a deterministic Turing machine M on a word w. When is a computation *accepting* ?
3. What is the relationship between the class of languages that can be decided by a *deterministic* Turing machine, and the class of languages that can be decided by a *non-deterministic* Turing machine ? Prove your answers.
4. Define the following notions: 'recognised language of a Turing machine' and 'decided language of a Turing machine'. What is the fundamental difference between those two notions ? Why do we use deciders to formalise the intuitive notion of algorithm ?
5. What is the Church-Turing thesis ? Give a list of arguments in favor of this thesis. Why is it not a theorem ?
6. Why don't we use the notions of finite automata or pushdown automata to formalise algorithms ? Compare the expressive powers of those two models, and relate their respective expressive power to that of Turing machines.

2 Chapter 4: Decidability

7. Prove that the set of Turing machines is countable.
8. Prove that the set of real numbers is not countable.
9. Prove that the set of rational numbers is countable.
10. Prove that the language $L_0 = \{w_i \mid w_i \notin L(M_i)\}$ is not Turing recognizable.
11. Prove that the language $A_{\mathsf{TM}} = \{(M, w) \mid M \text{ is a TM that accepts } w\}$ is undecidable.
12. Explain why proving that A_{TM} is undecidable establishes that $\mathsf{R} \neq \mathsf{RE}$, where:
    - $A_{\mathsf{TM}} = \{(M, w) \mid M \text{ is a TM that accepts } w\}$,
    - R is the set of decidable languages, also called 'recursive languages', and
    - RE is the set of recognised languages, also called 'recursively enumerable languages'.

13. Prove that a language A is decidable iff A is Turing-recognisable and co-Turing-recognisable (i.e., its complement is recognisable).

14. Prove that all regular languages and all context free languages are decidable.

15. Define the notion of multi-tape Turing machine. Prove that all multitape Turing machines can be turned into an equivalent single tape Turing machine.

3 Chapter 5: Reducibility

16. Prove that the following problem is undecidable:

$$HALT_{TM} = \{(M, w) \mid M \text{ is a TM and halts on input } w\}$$

17. Prove that the following problem is undecidable:

$$REGULAR_{TM} = \{(M) \mid M \text{ is a TM and } L(M) \text{ is a regular language}\}$$

18. Prove that the following problem is undecidable:

$$EQ_{TM} = \{(M_1, M_2) \mid M_1, M_2 \text{ are TMs and } L(M_1) = L(M_2)\}$$

19. Define the notion of *reduction function* $f : \Sigma^* \to \Sigma^*$ from a problem A to a problem B and prove the following theorem: '*If $A \leq_m B$ and B is decidable then A is decidable*' where $A \leq_m B$ reads '*A is reducible to B*'.

4 Chapter 7: Time Complexity

20. Define the notion of time complexity $\mathsf{TIME}(f(n))$. Explain why we can say that this is a worst-case measure of complexity. Why do we use big **O** notation when we reason on the time complexity of a Turing machine ?

21. Define the following notations, where f and g are functions: $f = \mathbf{O}(g)$ and $f = \mathbf{o}(g)$. Explain the intuition behind those notions. Does there exist a function f s.t. $f = \mathbf{o}(f)$? and s.t. $f = \mathbf{O}(f)$?

22. Define the running time of a Turing machine that is a decider. If M is a non deterministic machine that decides the language L in $\mathbf{O}(t(n))$, how can we bound the running time of a deterministic Turing machine that decides L ?

23. Define the class P and explain why it is an important complexity class.

24. Prove that $\mathsf{RELPRIME} \in \mathsf{P}$, where:

$$\mathsf{RELPRIME} = \{(p, q) \mid p \text{ and } q \text{ are relatively prime}\}$$

and two numbers are relatively prime iff their only common divisor is 1.

25. Recall that

$$\mathsf{RELPRIME} = \{(p, q) \mid p \text{ and } q \text{ are relatively prime}\}$$

and two numbers are relatively prime iff their only common divisor is 1. Does the following algorithm prove that $\mathsf{RELPRIME} \in \mathsf{P}$?

```
Let r := min(p,q)
while r > 1
  if r divides p and r divides q
    break
  r := r-1
if r = 1: accept
else: reject
```

26. Prove that CFL $\subseteq$ P.

27. We have given two definitions of the class NP. One uses 'certificates' and one uses non deterministic Turing machines. Recall the two definitions and prove that they are equivalent.

28. Give an example of a problem which (i) is in NP and (ii) is not known to be in P. Prove the membership to NP using the notion of certificate. Say whether you believe that this problem is in P and justify your intuition.

29. Is it the case that P $\subseteq$ NP ? Explain. Is it the case that NP $\subseteq$ P ? Explain.

30. Explain why NP $\subseteq$ ExpTime.

31. Define the notion of *NP complete problem* and explain why this notion is important.

32. Define the notion of '*polynomial time mapping reducible*'. Show that *if A is polynomial time reducible to B and $B \in$ P then $A \in$ P.*

33. Prove the following two statements: (i) if $B \in$ NP-Complete and $B \in$ P, then P=NP, (ii) if $B \in$ NP-Complete and $B \leq_P C$, and $C \in$ NP then C is NP-Complete.

34. State Cook's theorem, and give an outline of the proof.

35. Prove that VERTEX-COVER is NP-complete.

36. Prove that HAMILTONIAN-PATH is NP-complete.

37. Prove that SUBSET-SUM is NP-complete.

38. Prove that CLIQUE is NP-complete.

39. Prove that 3SAT is NP-complete.

5 Chapter 8: Space Complexity

40. Define the notion of space complexity. Illustrate the difference between time complexity and space complexity by showing an example of problem that can be solved in $\mathbf{O}(t(n))$ space but we believe cannot be solved in $\mathbf{O}(t(n))$ time.

41. If a language $L \in$ SPACE($\mathbf{O}(t(n))$ with $t(n) \geq n$, what can we say about the time complexity of L ?

42. State and prove Savitch's theorem.

43. Define the classes PSPACE, NPSPACE, coPSPACE and coNPSPACE. Explain why all those classes are equal.

44. Define the notion of *PSPACE completeness* and give an example of problem that is PSPACE-Complete.

45. Explain why TQBF can be solved in polynomial space.

46. Explain why the proof of Cook-Levin theorem cannot be applied directly to show that TQBF is PSPACE-Complete.

47. What are the known relationships between P, NP, PSPACE, NPSPACE and EXPTIME ? Explain.