# Problem Set 9

In this final problem set of the quarter, you will explore the limits of what can be computed efficiently. What problems do we believe are computationally intractable? What do they look like? And are they purely theoretical, or might you might bump into one some day?

As always, please feel free to drop by office hours or send us emails if you have any questions. We'd be happy to help out.

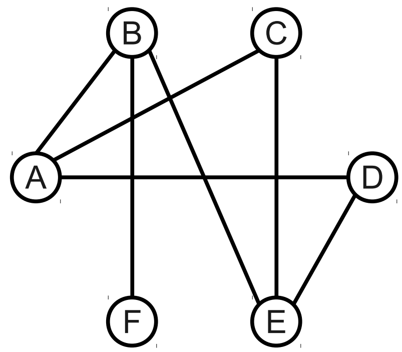
This problem set has 105 possible points. It is weighted at 5% of your total grade (note that this is slightly less than usual).

Good luck, and have fun!

**Due Friday, March 15th at 12:50 PM.**
**Due Monday, March 18th at 11:30AM with a late day.**
**No submissions accepted after March 18th at 11:30AM.**

**Problem One: The Long Path Problem (35 Points)**

Given an undirected graph $G = (V, E)$, a *simple path* in a $G$ is a path between two nodes $u, v \in V$ such that no node is repeated on the path. For example, given this graph:




A → C → E is a simple path from A to E, but A → B → E → C → A → D is not a simple path because node A is visited twice.*

Consider the following language:

*ULONGPATH* = { ⟨*G*, *u*, *v*, *k*⟩ | *G* is an undirected graph,
*u* and *v* are nodes in the graph, and
there exists a simple path from *u* to *v* containing *k* nodes. }

For example, if *G* is the above graph, then ⟨*G*, D, F, 6⟩ ∈ *ULONGPATH* because there is a simple path of six nodes from D to F (namely, D→A→C→E→B→F), but ⟨*G*, A, C, 5⟩ ∉ *ULONGPATH* because there is no simple path of 5 nodes from A to C.

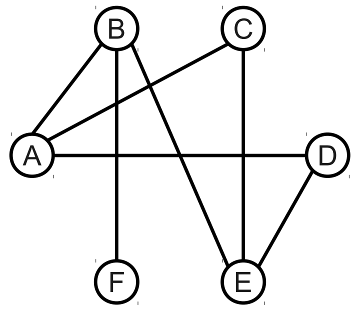
i. Show that *ULONGPATH* ∈ **NP** by designing a polynomial-time verifier for it. You do not need to formally prove that it is correct or that it runs in polynomial time, but you should justify why your answer is correct (one paragraph each for correctness and polynomial time should be sufficient.)

ii. Show that *ULONGPATH* ∈ **NP** by designing a polynomial-time NTM for it. You do not need to formally prove that it is correct or that it runs in polynomial time, but you should justify why your answer is correct (one paragraph each for correctness and polynomial time should be sufficient.)

*(continued on the next page)*

* Although we have defined a path in a graph as a series of edges, it is often easier to reason about the path as the series of nodes that it passes through. We will adopt this convention convention for this problem.

In an undirected graph $G = (V, E)$, a *Hamiltonian path* is a simple path between two nodes u and v that visits every node in the graph exactly once. For example, in this graph:




The path F → B → E → C → A → D is a Hamiltonian path from F to D, but F → B → E → D is not (because it doesn't visit every node), nor is F → B → E → D → A → C → E → D (because it is not a simple path).

The language *UHAMPATH* is defined as follows:

$$UHAMPATH = \{\langle G, u, v\rangle \mid G \text{ is an undirected graph and there is a Hamiltonian path from } u \text{ to } v.\}$$

*UHAMPATH* is known to be **NP**-complete by a fairly clever series of reductions from SAT; see Sipser, page 291 (second edition) or page 319 (third edition) for more details.

iii. Prove that *ULONGPATH* ∈ **NPC** by showing that $UHAMPATH \leq_P ULONGPATH$. Prove your reduction is correct (i.e., $\langle G, u, v\rangle \in UHAMPATH$ iff $f(\langle G, u, v\rangle) \in ULONGPATH$), but feel free to justify informally why your reduction works in polynomial time.

## Problem Two: Resolving $\mathbf{P} \stackrel{?}{=} \mathbf{NP}$ (30 Points)

This problem explores the question

**What would it take to prove whether or not P = NP?**

For each statement below, decide whether the statement would definitely prove **P** = **NP**, definitely prove **P** ≠ **NP**, or would do neither. Write "**P** = **NP**," "**P** ≠ **NP**," or "neither" as your answer to each question – we will not award any credit if you write "true" or "false," since there are three possibilities for each statement. No justification is necessary.

1. There is a **P** language that can be decided in polynomial time.
2. There is an **NP** language that can be decided in polynomial time.
3. There is an **NP-*complete*** language that can be decided in polynomial time.
4. There is an **NP**-***hard*** language that can be decided in polynomial time.
5. There is an **NP** language that ***cannot*** be decided in polynomial time.
6. There is an **NP-*complete*** language that ***cannot*** be decided in polynomial time.
7. There is an **NP**-***hard*** language that ***cannot*** be decided in polynomial time.
8. There is ***some*** **NP**-complete language that can be decided in $O(2^n)$ time.
9. There is ***no*** **NP**-complete language that can be decided in $O(2^n)$ time.
10. There is a polynomial-time ***verifier*** for every language in **NP**.
11. There is a polynomial-time ***decider*** for every language in **NP**.
12. There is a language $L \in \mathbf{P}$ where $L \leq_P$ 3SAT.
13. There is a language $L \in \mathbf{NP}$ where $L \leq_P$ 3SAT.
14. There is a language $L \in \mathbf{NPC}$ where $L \leq_P$ 3SAT.
15. There is a language $L \in \mathbf{P}$ where 3SAT $\leq_P L$.
16. There is a language $L \in \mathbf{P}$ where 3SAT $\leq_M L$.
17. All languages in **P** are decidable.
18. All languages in **NP** are decidable.
19. There is a polynomial-time algorithm that correctly decides SAT for all strings of length ***at most*** $10^{100}$, but that might give incorrect answers for longer strings.
20. There is a polynomial-time algorithm that correctly decides SAT for all strings of length ***at least*** $10^{100}$, but that might give incorrect answers for shorter strings.

## Problem Three: The Big Picture (35 Points)

We have covered a *lot* of ground in this course throughout our whirlwind tour of computability and complexity theory. This last question surveys what we have covered so far by asking you to see how everything we have covered relates.

Take a minute to review the hierarchy of languages we explored:

$$\mathbf{REG} \subset \mathbf{DCFL} \subset \mathbf{CFL} \subset \mathbf{P} \subseteq \mathbf{NP} \subset \mathbf{R} \subset \mathbf{RE} \subset \mathbf{ALL}$$

The following questions ask you to provide examples of languages at different spots within this hierarchy. In each case, you should provide an example of a language, but you don't need to formally prove that it has the properties required. Instead, describe a proof technique you could use to show that the language has the required properties. There are many correct answers to these problems, and we'll accept any of them.

i. Give an example of a regular language. How might you prove that it is regular?

ii. Give an example of a context-free language is not regular. How might you prove that it is context-free? How might you prove that it is not regular?

iii. Give an example of a language in **P** that is not context-free. How might you prove that it is in **P**? How might you prove that it is not context-free?

iv. Give an example of a language in **NP** suspected not to be in **P**. How might you prove that it is in **NP**? Why do we think that it is not contained in **P**?

v. Give an example of a language in **RE** not contained in **R**. How might you prove that it is **RE**? How might you prove that it is not contained in **R**?

vi. Give an example of a language in co-**RE** not contained in **R**. How might you prove that it is co-**RE**? How might you prove that it is not contained in **R**?

vii. Give an example of a language that is neither **RE** nor co-**RE**. How might you prove it is not contained in **RE**? How might you prove it is not contained in co-**RE**?

## Problem Four: Course Feedback (5 Points)

We want this course to be as good as it can be, and we'd really appreciate your feedback on how we're doing. For a free five points, please answer the following questions. We'll give you full credit no matter what you write (as long as you write something!), but we'd appreciate it if you're honest about how we're doing.

i. **If we should keep any one thing about this course the same in future offerings, what would it be?**

ii. **If you could change any one thing about this course, what would it be?**

iii. **What topic did you think was the most interesting? What topic did you think was the least interesting?**

## Submission instructions

There are three ways to submit this assignment:

1. Hand in a physical copy of your answers at the start of class. This is probably the easiest way to submit if you are on campus.
2. Submit a physical copy of your answers in the filing cabinet in the open space near the handout hangout in the Gates building. If you haven't been there before, it's right inside the entrance labeled "Stanford Engineering Venture Fund Laboratories." There will be a clearly-labeled filing cabinet where you can submit your solutions.
3. Send an email with an electronic copy of your answers to the submission mailing list (cs103-win1213-submissions@lists.stanford.edu) with the string "[PS9]" somewhere in the subject line. If you do submit electronically, please submit your assignment as a single PDF if at all possible. Sending multiple files makes it harder to print out and grade your submission.

## Extra Credit Problem: P $\stackrel{?}{=}$ NP (Worth an A+, $1,000,000, and a Stanford Ph.D)

Prove or disprove: **P** = **NP**.