This assignment is **due in Week 10** and should be submitted to Gradescope.

All work must be **done individually** without consulting anyone else's solutions in accordance with the University's "Academic Dishonesty and Plagiarism" policies.

Go to the last page of this document and read the **Submission Instructions**. For clarifications and updates, monitor "Assignment FAQ".

Problem 1. (10 marks) Consider the following deterministic Turing Machine M over input alphabet $\Sigma = \{a, b\}$:

```
0 _ _ L 1
0 * * R 0

1 b _ L 2
2 a _ L 3

1 _ _ * halt_accept

3 _ _ R 0
3 * * L 3
```

1. (5 marks) State five strings that are in $L(M)$, and five that are not. The strings should be over Σ.
2. (5 marks) Provide a low level description in Morphett notation of a (1-tape deterministic) Turing Machine for the language that has time complexity at most $5n + 5$.

Problem 2. (10 marks) Consider the following nondeterministic Turing Machine N over input alphabet $\Sigma = \{a, b\}$:

```
0 _ _ * halt-reject
0 a a r 0
0 b b r 0
0 b x l 1

1 x x l 1
1 a x r 2
1 b x r 2
1 _ _ r 4
```

```
2 x x r 2
2 a x r 3
2 b x r 3
2 _ _ * halt-reject


3 x x r 3
3 a x l 1
3 b x l 1
3 _ _ * halt-reject

4 x x r 4
4 a a * halt-reject
4 b b * halt-reject
4 _ _ * halt-accept
```

1. (5 marks) State five strings that are in $L(N)$, and five that are not. The strings should be over Σ.
2. (5 marks) Provide a low level description in Morphett notation of a (1-tape deterministic) Turing Machine for the language.

Note: Morphett's simulator of nondeterministic TMs uses randomness to resolve nondeterminism. This is not the semantics of NTMs.

Problem 3. (30 marks) For each of the following languages over the input alphabet $\Sigma = \{a, b, c\}$, provide a low level description in Morphett notation of a (1-tape deterministic) TM for the language.

1. The language of non-empty strings where the final character appears at most 3 times in the string (including the final character).

   E.g., *abccaba* is in the language, while *abcbcbab* is not.
2. The language of strings of the form $a^n b^n c^n a^n$ for $n \geq 1$.

   E.g., *aabbccaa* is in the language, while *abc* is not.
3. The language of strings that can be turned into a palindrome by replacing at most two characters by other characters.

   E.g., *aba* is in the language because it is a palindrome, *abb* is in the language because we can change one character to get a palindrome (e.g., *aba*), and *aabc* is in the language because we can change two characters to get a palindrome (e.g., *aaaa*); however *aabbccc* is not in the language.
4. The language of strings for which the longest substring that matches a^* is longer than the longest substring that matches b^*.

   E.g., *caaaccbbaabaaac*, *baaacbbcaaabb* and *aaaa* are in the language, while *aabbbcacacacaca* is not.

5. The language of strings of the form $uvcvu$ where $u, v \in \{a, b\}^*$.

   E.g., *aabbacbaaab* is in the language (take $u = aab, v = ba$), while *aabbcabab* is not.

6. The language of strings of the form uvw where v is a non-empty string with the same number of as, bs, and cs. E.g., $bb\underline{aabbbccacc}cbc$ is in the language, while *bbaabbbcc* is not.

Problem 4. (5 marks + 5 bonus marks)

Your robot buddy *GNPT-4* has come up with a revolutionary new strategy to prove that it is in fact equal in computational power to its more well-known cousin. It has a simple yet brilliant proof strategy: it will start by proving that P in fact equals the set of Turing-decidable languages, by showing that every decider runs in polynomial time. Once it has done this, it will obtain as a corollary that NP is also equal to this set, and the result will follow. GNPT-4 would like you to check its generated proof, and has generously offered you half of the million dollar bounty for doing so.

Unfortunately, you're starting to have some concerns about the claim that every decider runs in polynomial time. GNPT-4's proof of this claim is 2123 pages long, so you don't really feel like checking it in detail for a flaw. Instead, you have a much better idea: you'll provide an explicit counterexample of a machine that does not run in polynomial time.

1. (5 marks) Provide a low level description in Morphett notation of a (1-tape deterministic) TM over input alphabet $\Sigma = \{a\}$ that accepts every string, has at most 20 states, and has time complexity $f(n)$ such that $2^n \leq f(n) \leq 2^{2n+1}$ for all n.
2. (5 bonus marks) Provide a low level description in Morphett notation of a (1-tape deterministic) TM over input alphabet $\Sigma = \{a\}$ that accepts every string, has at most 40 states, and has time complexity *exactly* 2^n.

Problem 5. (15 marks)

You're a budding cartoonist, trying to create the next great TV animation. You've come up with the perfect idea, but now you need to pitch it to the executives. You know from your experience in the industry how the process works: you make a proposal with a string over $\Sigma = \{a, b\}$ and the network runs a Turing machine Q on it. If Q accepts, your show will be ready for broadcast, but if it doesn't, you will be shown the door, filled with eternal regret at what could have been. Of course, as Q is a Turing machine, there is also the possibility that Q will diverge. (For example, this is what happened after season 7 of *Futurama*.)

One of your shady contacts (apparently they're a secret agent who uses finite automata, or something?) has managed to obtain a copy of the network's machine Q for you. You now want to analyse Q to figure out how to pitch your show so it will be accepted. Furthermore, you've heard that it's considered especially fortuitous if Q runs in a number of steps that is a multiple of 77, and such shows will be given air during the network's prime timeslots. So you'd like a machine that will analyse Q and your proposal to see if that will be the case.

1. (5 marks) Prove that the language $\{M, x\colon M$ halts on x in exactly $77n$ steps for some integer $n > 0\}$ is undecidable.

Okay, so that was a bust. You've set your sights lower: at this point you just want any description that will be accepted, and you're willing to retool your proposal to make it work. Rather than focusing on your specific string, you'd like a machine that will analyse just Q, and find some string, any string, that it will accept. There is, however, the possibility that Q doesn't accept any string. (That would explain why there are no decent new shows these days.) In this event, your endeavour is doomed and you don't care about the output, but you'd like the analysing machine to at least halt, so you're not stuck waiting forever.

2. (10 marks) Consider the following specification. The inputs are Turing machines over input alphabet $\Sigma = \{a, b\}$.

   (a) If the input is a Turing machine M that accepts some input, the output should be any string x that M accepts.

   (b) If the input is a Turing machine M that does not accept any input, the output should be any string x. (There still must be an output, ie. the machine satisfying this specification must halt.)

   Prove or disprove whether there exists a Turing Machine that halts on every input and satisfies this specification.

Submission Instructions

You will submit answers to all the problems on Gradescope.

Problems 1, 2, 3 and 4 are autograded.

It is essential that you ensure that your submission is formatted so that the autograder can understand it. Upon submitting your responses, you should **wait** for the autograder to provide feedback on whether your submission format was correct. **An incorrectly formatted submission for a question will receive zero marks for that question.** A scaffold will be provided on Ed with the file names the autograder expects.

Problem 1.1, 2.1 format:

The first line of each answer should contain a comma separated sequence of five strings that are in the language, and the second line should contain a comma separated sequence of five strings that are not in the language. For example, if the language consists of all strings that only contain b's, an example of a correct text file would be:

```
epsilon, b, bb, bbb, bbbb
a, aa, aaa, aaaa, aaaaa
```

Problem 1.2, 2.2, 3, 4 format (TMs):

All TMs that you are required to provide in this assignment are deterministic and have a single tape, and that tape is doubly-infinite. When asked to give a **low-level description** use Morphett's format. The initial state must be 0

Note that your machine should use an explicit transition to halt-reject when rejecting a string. If the machine has no transition on a (state, input) pair, this will be treated as an error, and will *not* be treated as rejecting the string. You may wish to include the following line in your machines, to treat all undefined transitions as rejects: * * * * halt-reject

Problem 5 format:

Problem 5 is handgraded. You will submit a single **typed pdf (no pdf containing text as images, no handwriting)**. Start by typing your student ID at the top of the first page of each pdf. Do **not** type your name. Do not include a cover page. Submit only your answers to the questions. Do **not** copy the questions. Your pdf must be readable by Turnitin.