1 Prelab: Coding Theory

Coding theory is the study of how to transmit data reliably. Think about the game of telephone in which one person whispers a message to another, who then passes that message on to the next person and so on and so forth. The “fun” in the game arises when the message makes it around the room back to the original sender now changed in some (humorous) way from the actual message, presumably having been altered in transmission somewhere along the way. Messages such as email or phone conversations, which are sent on computer networks are subject to these sorts of transmission errors. The two biggest categories of errors are substitution errors, and transposition errors.

TODO (pre-lab): Define these two terms and give an example of each. Also define a checksum.

substitution error: substitution of a symbol in a string by another symbol

transposition error: unintentional exchange of two (or more) elements of an ordered list with all others staying the same (http://www.ventureline.com/accounting-glossary/T/transposition-error-definition/)

checksum: a sum used to check if 2 sets of data are the same. Verifies that data has not been altered or corrupted. (techterms.com)

If you have ever mistyped your credit card number when making an online payment, you probably were alerted by the system to the error. Credit cards choose numbers according to error-checking schemes, to help catch if data has been mistyped or corrupted.

1.1 Pre-lab: design

TODO (pre-lab): Design and research an error-checking circuit that can identify substitution errors or transposition errors. Provide a circuit diagram for your design. Examples of these methods to help get you started include parity checking, repetition codes, and CRC checks. You are welcome to use any of these for your circuit. If you use the parity check your circuit must be able to identify errors in a message of at least four parallel bits, where one of the bits is the checksum. For all other designs, aim for a circuit that can handle three serial bits. Cite any sources you use.
TODO (pre-lab): Hypothesize whether you think this circuit will detect transposition or substitution errors. Again, cite any sources you use.

1.2 Pre-lab: methods

TODO (pre-lab): Determine a method to test each of the following, including the test cases to use on your circuit:

1. Will the physical implementation of your circuit design behave as expected?
2. Will this circuit catch substitution errors?
3. Will this circuit catch transposition errors?
1.3 In-lab: your circuit

TODO: Implement your circuit on your choice of The Magic Box or Logisim, and then test it using your methods from the previous section. Record your results/observations.

TODO: Scale your circuit up: modify it so it can detect errors on messages one bit larger than your current implementation. Then test your circuit again using your own methods. Record your results/observations. Show your circuit to your TA.

TODO: What errors did you find your circuit could catch? Do you confirm or reject your hypothesis? How well did your circuit scale up? Write down your answers and show them to your TA.
1.4 Real-world applications

Credit cards use an error-detection scheme known as the Luhn algorithm. **TODO: Look up this scheme. How does it work? What types of errors will it detect? For what else, other than credit cards, is this algorithm used?** Write down your findings.

The algorithm first doubles every second digit, starting from the second-from-right. For any products with double digits, each digit is considered a single number, and then all of these digits are added to the ones that weren't doubled. If then umber is valid, the sum must end in 0: the rightmost digit is the checksum. It will catch all single-digit errors, and transposition errors (unless even or odd digits are transposed with one another). Used for US & CA SIN #s, insurance #s, Swedish birth #s.

The codes used for identifying books, ISBN (International Standard Book Number) also uses a checksum-based scheme. **TODO: Look up the ISBN’s error detection scheme. How does it compare to the Luhn algorithm? Give an example of an error the ISBN scheme would not catch that the Luhn algorithm would catch, and vice versa.** Write down your findings and show them to your TA.

ISBN should be better at detecting errors than the Luhn algorithm. An ISBN is composed of 10 digits, with the rightmost digit as a checksum. Each of the 9 leftmost digits is multiplied by a different value, from 10 to 2. These products are then summed, along with the checksum. The resulting value must be divisible by 11.

ie:

```
0 8 4 3 6 1 0 7 2 7 ISBN #
x 10 9 8 7 6 5 4 3 2 = 198
```

The following error would be caught by the ISBN algorithm, but not by the Luhn algorithm:

0843610727 (correct)
0843612707 (2nd and 4th digits from the right transposed)

The following error would be detected by the Luhn algorithm, but not by ISBN:

0143310623 (correct)
0243310323 (substitution errors on the 3rd and 9th digits from the right)

Source: Chapter 4 of ISBN user's manual online:
1.5 In-lab: analysis and discussion

TODO (further analysis): In 2003, NASA began a robotic space mission to explore Mars. Two unmanned semi-autonomous rovers, Spirit and Opportunity are currently deployed on the surface of Mars. While the rovers can operate on their own, occasionally they require human control which needs to be communicated via radio signals from Earth.

Radio signals travel at the speed of light \( c = 3 \cdot 10^8 \text{ m/s} \) and Earth and Mars are on average about \( 225\cdot 10^6 \text{ km} \) apart from each other. How long would it take to send a control signal from Earth to Mars? How would this influence our design of how data is transmitted to the robots, in terms of error checking and correction?

12.5 minutes for a signal to go from Earth to Mars (or vice

2 End of Lab Survey

TODO: To help us improve these labs both this term and for future offerings, complete the survey at http://www.tinyurl.com/cs121labs.
A  Challenge problem

TODO (challenge): Implement a small circuit that corrects errors upon detecting them. You may want to first read up on forward error coding, Reed-Solomon codes, or Hamming codes.

B  Marking scheme

All labs are out of ten marks, with two marks for pre-labs, and eight marks for in-lab work. In more detail:

- Two marks - Pre-lab questions
- Five marks - In-lab questions. It is one mark for your results, one for scaling up, one for your findings; and two for the real world applications.
- Two marks - For detailed analysis or insight in approaching assigned work, such as by answering the questions labeled (further analysis) with more than one sentence of work. (TAs at their discretion may occasionally award these marks for detailed analysis seen in other parts of the lab.)
- One mark - End of lab survey.

TAs may at their discretion award one bonus mark, such as for completing a challenge problem. It is expected that most students will achieve 6-8. If you feel you’re heading for 0-5, get immediate help from the TAs!