Department of Computational and Data Sciences
DS226: Introduction to Computing for Artificial Intelligence and Machine Learning
September 1, 2022

Assignment 1

Instructions:

- **Submit a typed report in PDF format on Moodle. Use LaTeXpreferably. Handwritten reports will not be accepted.**
- **Solutions should be in the same order as the questions.**
- **Discussion is encouraged, but answers should be your own. Plagiarism will be penalised severely.**
- **The deadline is strict. For late submissions, the following late submission policy:**

Delay	% of Credit
0-72 hours	50
72 hours - 1 week	25
Beyond 1 week	No credit

Deadline: 9th September 2022, 11:59 AM
Total Credits: 50

Question 1. (10)
Suppose $D = \{0, 1\}$ is a set, and define $\mathbb{X} := D^w$, where w is a positive integer. Let $\mathbf{x}(x_{w-1}, x_{w-2}, \ldots, x_0) \in \mathbb{X}$ be a w-bit integer data representation of an integer with x_{w-1} and x_0 being the most and least significant bits, respectively. Let $\mathbb{Y}$ be a set consisting of 0 and all positive integers up to U_w^{Max}, where U_w^{Max} is the largest integer that can be represented by the w-bit unsigned data representation. Further, the unsigned integer data representation can be defined as a function

$$U_w : \mathbb{X} \to \mathbb{Y}, \qquad U_w(\mathbf{x}) \doteq \sum_{i=0}^{w-1} x_i 2^i.$$

Recall that $\mathbf{x} \in \mathbb{X}$ can also be used to represent a negative integer (signed) data type using *Two's-complement encoding*, defined by the function

$$ST_w : \mathbb{X} \to \mathbb{Z}, \qquad ST_w(\overrightarrow{\mathbf{x}}) \doteq -x_{w-1}2^{w-1} + \sum_{i=0}^{w-2} x_i 2^i,$$

where $\mathbb{Z} := \{ST_w^{Min}, \ldots, ST_w^{Max}\} \subset \mathbb{I}$. Here, ST_w^{Min} and ST_w^{Max} are the smallest and largest integer, respectively, that can be represented by 2^w-bit signed data representation. Given the above definitions, answer the following questions.

a. Define the function of the mapping

$$UtoST_w : \mathbb{Y} \to \mathbb{Z},$$

where $UtoST_w(\cdot)$ maps an unsigned integer to a signed two's-complement integer. (5)

b. Define the function of the mapping

$$STtoU_w : \mathbb{Z} \to \mathbb{Y},$$

where $STtoU_w(\cdot)$ maps a *Two's-complement* integer to an unsigned integer. (5)

Hint: Since both functions, $U_w(\cdot)$ and $ST_w(\cdot)$, are bijective, inverse functions $U_w^{-1}(\cdot)$ and $ST_w^{-1}(\cdot)$ exist.

Question 2. (10)
The IEEE single-precision float representation is a standard for representing floating point numbers established in 1985 by the Institute of Electrical and Electronics Engineers(IEEE). It uses a base of two and consists of a sign bit (0 representing a positive number and 1 representing a negative number), a mantissa with 23 bits, and an 8-bit biased exponent (32 bits in total). The stored exponent is obtained by adding a bias value to the actual exponent used for representing floating point numbers on a computer.

a. Convert the numbers given below to IEEE single precision float format. Also, convert each of them to hexadecimal: (3)
   - 86.125
   - 0.523
   - -0

b. How many numbers lie in between -2^{-12} and -2^{-11} (excluding these two numbers) when we represent the numbers using the IEEE single precision floating point representation? Does your answer change for numbers between -2^{-13} and -2^{-12} (excluding both)? (3)

c. Define $\mathbb{S} := \{1, 2, 3, 4, 5, 6, 7, \ldots, n\} \subset \mathbb{N}$, where $\mathbb{N}$ denotes natural numbers. Suppose the elements of $\mathbb{S}$ are represented by 32-bit IEEE single-precision float representation. Find the smallest value of n that cannot be represented using the IEEE format.
   Hint: Not all numbers can be represented exactly, even if those numbers lie within the range of the 32-bit IEEE single-precision float representation.

   Further, what would be the values of n if 32-bit signed and 32-bit unsigned integer representations are used?

Question 3. (10)
(a). Incorrect integer operations have caused bugs in gaming since the days of arcade games. One apocryphal example of this, the stuff of internet legends and countless memes, is that of the 1991 video game - Civilization. In this game, countries were ranked based on their aggression scores between 1 and 10, where one is a peace-loving nation and nations with an aggression score of ten are bombing countries with nuclear weapons. (1.5)

i. Suppose an unsigned integer is used to store the aggression score value; what would be the minimum number of bits, w, required to store the value?

Two points would be deducted from their aggression score when the countries achieve democracy. Imagine a country, X, which started as a peaceful nation with an aggression score of one. We would expect this country to remain a peaceful nation after achieving democracy. However, as soon as X achieved democracy, it started bombing other countries using nuclear weapons.

ii. Starting with the w-bit unsigned representation of 1, explain this counter-intuitive behavior by performing the same bit-wise operations as the game's logic.

iii. What is the technical term for the error causing this counter-intuitive behavior? How would you solve the error without affecting the upper range of the aggression score?

(b). In 2015, the US federal aviation administration ordered operators of the Boeing 787 to restart their electrical systems periodically. The problem was attributed to a timing counter, stored as a w-bit signed integer, which counts the duration for which the generator control units have been running in centiseconds ($\frac{1}{100}^{th}$ of a second). Suppose the system runs continuously for around 249 days, the same error as in part (a). occurs, and all electrical control systems could switch off (even mid-flight), leading to a loss of control and possibly a crash. Working your way backward, find (1.5)

i. the number of bits (w) used to store the timing counter.

What will be the period after which this error will occur if the timing counter is stored using a

ii. w-bit unsigned integer?
iii. $2w$-bit signed integer?

(c). Recall the Ariane 5 disaster discussed in class. The root cause was an erroneous conversion of a 64-bit floating point to a 16-bit integer. Consider a hypothetical scenario where a rocket with a lateral velocity V_H is represented by 16-bit (half precision) floating point values. For each stage of the rocket, convert the floating point binary value first into its decimal representation, then a 16-bit two's complement signed integer (round-to-zero). At which stage does the value of V_H become large enough to throw an error? For a 16-bit floating representation, assume the first bit to be the sign bit, the next 5 bits to be the exponent bits, and the remaining to be the mantissa. For stages with an error, you can set the leading bit to 1 and the remaining bits to 0 to denote an error (3)

Stage	16-bit IEEE floating point	Decimal	16-bit signed integer
I	0110001111111011		
II	0110011111101100		
III	0111001101101101		
IV	0111100000011111		
V	0111101000111111		

(d). Assume that the rocket does not break down after the error and instead starts moving towards a city center. Fortunately, you have an anti-missile rocket system placed in the city. *Unfortunately*, such anti-missile systems have been known to fail due to another floating point bug.
In particular, the anti-missile system contains an internal time counter which increments every 0.1 seconds. The binary representation of $\frac{1}{10}$ is the non-terminating sequence $0.000110011[0011]\ldots_2$, where the portion in the brackets is repeated indefinitely. To determine the time, the internal logic multiplies the value in the counter by a 24-bit fractional binary approximation to $\frac{1}{10}$, denoted by x, such that (4)

$$x = 0.00011001100110011001100_2.$$

i. What is the binary representation of 0.1-x?
ii. Convert this error in representation to an approximate decimal value.
iii. The counter starts at 0 when the system is first powered up and keeps counting from there. Plot a graph between the actual time t (in hours) on the X-axis and the error in time measured Δt in seconds on the Y-axis. What is the value of Δt after 50 and 100 hours of operation, respectively?
iv. For the incoming faulty rocket, now approaching at speed $V = 3000$ m/s, the error in tracking the rocket is given by $x_{err} = V \times \Delta t$. Plot a graph between t in hours on the X-axis and x_{err} in meters on the Y-axis. Suppose the anti-missile system fails to intercept the rocket when x_{err} exceeds 500 meters. How long should the anti-missile system be allowed to operate before it has to be restarted in this scenario?

Question 4. (10)

a. Suppose we represent numbers with decimal in the form

$$\pm 0.d_1 d_2 \cdots d_k \times 10^n, \quad 1 \le d_1 \le 9, \quad \text{and} \quad 0 \le d_i \le 9,$$

for each $i = 2, 3, \cdots, k$. With a similar representation, a positive real number y is given by,

$$y = 0.d_1 d_2 \cdots d_k d_{k+1} d_{k+2} \cdots \times 10^n$$

Let $\psi_k(y)$ be a k-digit **rounding-off** approximation operator on y such that $\psi_k(y) = 0.d_1 d_2 \cdots d_{k-1} d'_k \times 10^n$. Then prove that,

$$\left| \frac{y - \psi_k(y)}{y} \right| \le 0.5 \times 10^{1-k}.$$

(2)

b. Suppose we represent the numbers with decimals as

$$\pm 0.d_1d_2d_3d_4 \times 10^n, \quad 1 \le d_1 \le 9, \quad \text{and} \quad 0 \le d_i \le 9,$$

for each $i = 2, 3, 4$ and $|n| \le 15$. Find the largest p for which $\dfrac{p!}{3!\,(p-3)!}$ can be computed without causing **overflow**. (4)

c. To compute any consistent average of given two numbers, you happen to choose *exponential average* given as $f_e(a,b) = \dfrac{e^a - e^b}{a-b}$. Further, you have decided to approximate the exponential by $e^z = \dfrac{6+2z}{6-4z+z^2}$ (which is [2/1] *Padè approximation* of e^z). Using 5-digit rounding off, compute $f_e(a,b)$ for $a = 1.0166$ and $b = 1.0116$. Also, compute the absolute and relative round-off errors, and study the behaviour of round-off error. (4)

Question 5. (10)

The number of floating-point operations per second (FLOPs) performed by a program can be used as one of the measures of the computational cost of the implemented algorithm. A FLOP is a basic unit of floating-point computation, and it can either be an addition or subtraction, multiplication or division. As such, calculate the number of floating point operations required for each of the following:

a. Consider a vector space $\mathbb{V} \subset \mathbb{R}^n$ with an the inner product $f : \mathbb{V} \times \mathbb{V} \to \mathbb{R}$ defined as:

$$f(\mathbf{a}, \mathbf{b}) = \sum_{i=1}^{n} a_i b_i, \text{ for } \mathbf{a}, \mathbf{b} \in \mathbb{V}.$$

Given the above definition, we use the following pseudo-code to compute $f(\mathbf{a}, \mathbf{b})$:

```
$f \leftarrow 0$
for i=1 to n do
    $f \leftarrow f + a_i b_i$
end for
```

Find the number of floating point operations performed by the above algorithm in terms of n (1).

b. Consider the matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ and a vector $\mathbf{b} \in \mathbb{R}^n$, the vector $\mathbf{c} \in \mathbb{R}^m$ resulting from the matrix-vector multiplication of $\mathbf{A}$ and $\mathbf{b}$. The elements of the vector $\mathbf{c}$ are found using the following pseudo-code:

```
Require: $A \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^n$
  $c \leftarrow 0, c \in \mathbb{R}^m$
  for i=1 to m do
     for j=1 to n do
       $c_i \leftarrow c_i + a_{ij} b_j$
     end for
  end for
```

Find the number of floating point operations performed by the above algorithm in terms of m and n. (2)

c. Consider two matrices $\mathbf{A} \in \mathbb{R}^{m\times n}$ and $\mathbf{B} \in \mathbb{R}^{n\times r}$ and the matrix $\mathbf{C} \in \mathbb{R}^{m\times r}$ which is a result of the product of the matrices $\mathbf{A}$ and $\mathbf{B}$. The elements of the matrix $\mathbf{C}$ are found using the following pseudo-code:

```
Require: $A \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{n\times r}$
  $C \leftarrow 0, C \in \mathbb{R}^{m\times r}$
  for i=1 to m do
     for j=1 to r do
        for k=1 to n do
           $c_{ij} \leftarrow c_{ij} + a_{ik}b_{kj}$
        end for
     end for
  end for
```

Find the number of floating point operations performed by the above algorithm in terms of m, n and r. (3)

d. Consider the matrices $\mathbf{A} \in \mathbb{R}^{m\times n}$, $\mathbf{B} \in \mathbb{R}^{n\times r}$, $\mathbf{C} \in \mathbb{R}^{r\times p}$ and the vector $\mathbf{x} \in \mathbb{R}^{r}$. Given the above described algorithms to compute matrix-vector and matrix-matrix multiplications, find the number of floating-point operations required to compute the following:

   i. $\mathbf{ABx}$

   ii. $\mathbf{ABC}$

   Write each expression in terms of m, n, r and p. Then find the minimum number of operations required if $m = 10000$, $n = 5000$, $r = 500$ and $p = 150$. (4)