# Question 1

## Worst-Case Number of Symbols for Encoding in Roman Numerals

To start, the best-case number of symbols for encoding is a one-to-one mapping between the Roman numeral system and the decimal (base-10) system. For example, the decimal number is represented by the Roman numeral , meaning one symbol is required. This implies that, for a positive integer with symbols, the lower bound of the corresponding Roman numerals is also symbols.

Looking at the upper bound, one can prove that the encoding is bounded by . There exists some positive real for all in . Proof by construction will be used, where some concrete is proposed and shown that it works.

Trying some values of , such that the number of symbols needed increases:

|  |  |  |  |
| --- | --- | --- | --- |
| **Decimal Number** | **# of Symbols** | **Roman Numeral** | **# of Symbols** |
|  | 1 |  | 1 |
|  | 2 |  | 4 |
|  | 3 |  | 6 |
|  | 4 |  | 9 |
|  | 5 |  | 11 |

Seeing what would work:

It appears that works. Thus, this chosen will be adopted and checked with a proof by induction:

* Base case:
* Step: Seek to show given that for all
* Assume true for , show true for

Thus, the upper bound is . Furthermore, analyzing the examples used in the table above, it can be seen that the number of symbols needed for the decimal numbers, squared, are all the number of symbols needed for their Roman numeral equivalents. Therefore, the tight bound for this encoding is .

## Worst-Case Number of Symbols for Encoding in Decimal System

The worst-case number of symbols to encode a positive integer in decimal notation is . For every digit in the positive integer , the decimal system allows each to be represented by a symbol from . First prove that :

* By trial and error: It appears that for all
* To prove: For all positive integers
* By induction on . Base case:
* Step, assume: Indeed, true that for all
* To prove:
* Indeed:

Thus, the lower bound is . Now for the upper bound, a similar proof from part can be used to prove a bound of . Using and a proof by induction on :

* Base case:
* Step: Seek to show given that for all
* Assume true for , show true for

Thus, the upper bound is . Therefore, the tight bound for this encoding is . Comparing the Roman numeral system to the decimal system, the decimal encoding is better than the Roman numeral encoding from part . It is more efficient with respect to the number of symbols used, as it can represent these integers with the same or less symbols.

# Question 2

The goal for this problem is to prove that the Harmonic Series, namely , has an upper bound of . First, let to exponentiate and work with a proof involving . The Harmonic Series then becomes:

Consequently, this series can be expanded as follows:

To carry the proof through, group the terms by powers of two and compare, similar to how divergence is proven for the Harmonic Series [1]. Thus, the grouping is shown as:

There is also one extra term in the series. Then, a valid upper bound can be generated by using this strategy of increasing powers of two and applying it to the grouped terms as follows:

Clearly, , where represents the desired upper bound in question. Now, note that each grouped set of terms adds up to , therefore all the grouped terms added together equals . This, along with the extra term, leads to the following inequality:

Subbing back in leads to the following equation for :

Using the hint of adopting the exact form of , together with the power of two grouping, leads naturally to use and . Finally, this results in the following inequality:

With this result, the value of is an additive constant, meaning it can be omitted from the big O notation. This simplifies the inequality:

Therefore, the final upper bound of the Harmonic Series is . In mathematical form, this means that:

This concludes that the upper bound of the Harmonic Series is indeed .

# Question 3

* First adopt case-analysis (tackle each case one at a time)
* Use logical deduction for every case?
* More specifically, prove equality () in two parts: and
* For example, prove by proving and
* Example: Prove the case that both are even:
  + Let
  + Then, divides both and
  + Therefore, divides both and
  + Therefore,
  + Then proceed to prove
* To identify what is, first look at what input(s) the algorithm in question takes
  + And that is a pair of non-negative integers
* So, to encode (i.e., write down these two numbers), it would take symbols, which is the same as , so this is our

**Question 4**

* First enunciate a correctness property for that assumes only that and
* But doesn’t demand that nor that

**Question 5**

* We can infer things based on properties of mod
* E.g., since:
* Then we know that:

# References

|  |  |
| --- | --- |
| [1] | E. W. Weisstein, "Harmonic Series," Wolfram MathWorld, [Online]. Available: https://mathworld.wolfram.com/HarmonicSeries.html. [Accessed 23 January 2020]. |