Problem 55 - Lychrel Numbers

Gautam Manohar

16 July 2018

This document originally appeared as a blog post on my website. Find it at gautammanohar.com/euler/55.

1 Problem Statement

If we take 47, reverse it, and add, we get $47 + 74 = 121$, which is a palindrome.

Not all numbers produce palindromes so quickly. For example,

$$\begin{aligned} 349 + 943 &= 1292, \\ 1292 + 2921 &= 4213, \\ 4213 + 3124 &= 7337. \end{aligned}$$

That is, 349 takes 3 iterations to arrive at a palindrome.

Although no one has proved it yet, it is thought that some numbers, like 196, never produce a palindrome. A number that never forms a palindrome through the reverse and add process is called a Lychrel number. Due to the theoretical nature of these numbers, and for the purpose of this problem, we shall assume that a number is Lychrel until proven otherwise. In addition, you are given that for every number below 10000, it will either become a palindrome in less than fifty iterations, or, no one, with all the computing power that exists, has managed so far to map it to a palindrome.

Many numbers converge to the same palindrome; for example,

$$19, 28, 29, 37, 38, 46, 47, 56, 64, 65, 73, 74, 82, 83, 91, 92, 110, 121$$

all converge to 121, a total of 18 numbers.

Given N, find the palindrome to which the most numbers from 1 to N, inclusive, converge.

2 My Algorithm

To check whether a number is a palindrome, we use string comprehension. We maintain a frequency dictionary that maps a palindrome to the count of numbers that converge to it. For each $n \leq N$, we apply up to 60 iterations of the reverse-add process. If at any point we get a palindrome, we break and add it to the frequency table. Otherwise, we increment the count corresponding to 0. To remove the numbers which did not yield a palindrome in 60 iterations, we set `freq[0] = 0`.

Finally, we sort the frequency table and get the palindrome with the highest associated count. For each n, we perform at most 60 iterations, and so this solution has time complexity $O(N)$.