No-2 Conjecture

July 2, 2022

1 Introduction

The *No-2 Conjecture* says that there is no consecutive sequence of ones with a size equal to two in the sequence T_3. In other words, no sequence of T translates to a sequence of T_3 that has exactly two consecutive ones.

Note that this pattern of exactly two consecutive ones does exist in T, but not T_3.

Lemma 1.1 *The smallest and only sequence that contains exactly two consecutive ones is 0110.*

Lemma 1.2 *There exists no consecutive sequences of ones or zeros greater than two in T.*

E.g. 0110 and 1001 exist in T, but 1110 and 0100 do not.

Theorem 1.3 *Any sequences of T that leads to a sequence of T_3 that contains two consecutive ones must have ones in locations 3, and 6.*

E.g. . . . 1 . . 1 . . .

Theorem 1.4 *Any sequences of T that leads to a sequence of T_3 that contains two consecutive ones must also have zeros in locations 0, and 9.*

E.g. 0 . . 1 . . 1 . . 0

Corollary 1.4.1 *The only sequences that translate to a sequence of T_3 that are equal to 0110 that follow Lemma 1.2, Theorem 1.2, and Theorem 1.3 are the following.*

Possible T sequences that translates to T_3 equal to 0110 = ['0011001010', '0011001100', '0011011010', '0101001010', '0101001100', '0101011010', '0101101010', '0101101100']

Lemma 1.5 *For any sequence of length at most 2^n, if the Thue-Morse sequence doesn't contain it in the first 2^{n+3} digits, it contains it nowhere.*

Corollary 1.5.1 *None of the sequences from the list of possible T sequences that translates to T_3 equal to 0110 are contained in first $2^{9+3} = 4096$ digits of T, and therefore exist nowhere in T.*

2 Code Verification

2.1 Possible T sequences that translates to T_3 equal to 0110 not found in first 4096 of T

The first 4096 of T do not contain any of the 8 possible T sequences that translates to T_3 equal to 0110.

```
first_4096 = ""

for x in range(0, 4096):
    first_4096 = "1" if bin(x).count('1') % 2 else "0"

possible = [
    '0011001010',
    '0011001100',
    '0011011010',
    '0101001010',
    '0101001100',
    '0101011010',
    '0101101010',
    '0101101100',
]

for x in possible:
    if x in first_4096:
        print("Found")
```

Listing 1: Possible T sequences that translates to T_3 equal to 0110 not found in first 4096 of T

2.2 Find possible T sequences that translates to T_3 equal to 0110

```
import itertools
from pprint import pprint

perms = [''.join(x) for x in itertools.product('01', repeat=10)]

def remove_less_than_two_in_a_row_t3(items: list[str]):
    exactly_two = []
    for x in items:
        if x[3] == '1' and x[6] == '1':
            exactly_two.append(x)

    return exactly_two

def remove_more_than_two_in_a_row_t3(items: list[str]):
    exactly_two = []
    for x in items:
        t3 = ""
        for a in range(0, 10, 3):
            t3 += x[a]

        if "111" not in t3:
            exactly_two.append(x)

    return exactly_two

def remove_more_than_two_in_a_row_t_new(items: list[str]):
    left = []
    for item in items:
        if "111" not in item and "000" not in item:
            left.append(item)
    return left

perms = remove_more_than_two_in_a_row_t_new(perms)
perms = remove_less_than_two_in_a_row_t3(perms)
perms = remove_more_than_two_in_a_row_t3(perms)
pprint(perms)
```

Listing 2: Find possible T sequences that translates to T_3 equal to 0110