密码学第一章作业

2112514　辛浩然

信息安全、法学

9/17/2023

1　习题 1.18

若 $(z_0, z_1, z_2, z_3) = (0, 0, 0, 0)$，$z_4 = 0 \pmod 2$，得到 $z_i = 0 \pmod 2, i \geq 0$，所以密钥流周期为 1；

若 $(z_0, z_1, z_2, z_3) = (0, 0, 0, 1)$，$z_4 = 1 \pmod 2$，可以得到密钥流：$(0, 0, 0, 1) \longrightarrow (1, 0, 0, 0) \longrightarrow (1, 1, 0, 0) \longrightarrow (0, 1, 1, 0) \longrightarrow (0, 0, 1, 1) \longrightarrow (0, 0, 0, 1)$。对该密钥流中的所有状态而言，周期都是 5；

若 $(z_0, z_1, z_2, z_3) = (0, 0, 1, 0)$，$z_4 = 1 \pmod 2$，可以得到密钥流：$(0, 0, 1, 0) \longrightarrow (1, 0, 0, 1) \longrightarrow (0, 1, 0, 0) \longrightarrow (1, 0, 1, 0) \longrightarrow (0, 1, 0, 1) \longrightarrow (0, 0, 1, 0)$。对该密钥流中的所有状态而言，周期都是 5；

若 $(z_0, z_1, z_2, z_3) = (0, 1, 1, 1)$，$z_4 = 1 \pmod 2$，可以得到密钥流：$(0, 1, 1, 1) \longrightarrow (1, 0, 1, 1) \longrightarrow (1, 1, 0, 1) \longrightarrow (1, 1, 1, 0) \longrightarrow (1, 1, 1, 1) \longrightarrow (0, 1, 1, 1)$。对该密钥流中的所有状态而言，周期都是 5；

综上，当初始向量 $(z_0, z_1, z_2, z_3) = (0, 0, 0, 0)$，周期为 1；其他情况下，周期为 5.

2　习题 1.21-b

密文 HJV 共出现在五个位置，起始位置分别为 107、125、263、317 和 329，其距离分别为 18、138、54 和 12.

猜测 $m = 6$;

当 $m = 5$ 时，重合指数分别为 0.057、0.057、0.047、0.050、0.057；

当 $m = 6$ 时，重合指数分别为 0.079、0.100、0.066、0.082、0.060、0.090，基本都约为 0.065，故密钥长度为 6；

下面对任意的 $1 \leq i \leq 6, 0 \leq g \leq 25$，计算其对应的 M_g 的值. 其中，$M_g = \sum_{i=0}^{25} \frac{p_i f_{i+g}}{n'}$；对每个 i，寻找其 M_g 值接近于 0.065 的那一个.

计算结果如表所示：

i	$M_g(y_i)$ 的值
1	0.031 0.036 0.065 0.039 0.034 0.042 0.037 0.031 0.042 0.046 0.025 0.034 0.038 0.042 0.038 0.046 0.036 0.040 0.042 0.033 0.030 0.039 0.043 0.034 0.042 0.033
2	0.038 0.039 0.048 0.041 0.039 0.036 0.045 0.030 0.026 0.035 0.044 0.030 0.035 0.047 0.040 0.032 0.035 0.070 0.036 0.029 0.029 0.035 0.029 0.037 0.045 0.036
3	0.035 0.036 0.034 0.037 0.035 0.041 0.028 0.038 0.034 0.042 0.041 0.046 0.040 0.043 0.036 0.032 0.035 0.039 0.042 0.031 0.039 0.033 0.035 0.043 0.059 0.045
4	0.045 0.038 0.043 0.037 0.036 0.037 0.031 0.033 0.038 0.037 0.036 0.051 0.040 0.031 0.034 0.066 0.037 0.029 0.039 0.040 0.025 0.035 0.041 0.033 0.034 0.044
5	0.040 0.033 0.034 0.039 0.044 0.033 0.043 0.046 0.046 0.034 0.034 0.035 0.034 0.035 0.033 0.044 0.034 0.035 0.035 0.055 0.040 0.035 0.043 0.044 0.030 0.032
6	0.042 0.038 0.037 0.042 0.038 0.027 0.033 0.039 0.037 0.036 0.035 0.048 0.035 0.025 0.037 0.070 0.042 0.032 0.038 0.033 0.040 0.041 0.035 0.037 0.039 0.048

由表中数据可知，密钥是 $K = (2, 17, 24, 15, 19, 14)$

i	k	$M_g(y_i)$ 的值
1	2	0.065
2	17	0.070
3	24	0.059
4	15	0.066
5	19	0.055
6	14	0.070

解密后的明文如下（加入了空格和标点，以便于阅读）：

I LEARNED HOW TO CALCULATE THE AMOUNT OF PAPER NEEDED FOR A ROOM WHEN I WAS AT SCHOOL. YOU MULTIPLY THE SQUARE FOOTAGE OF THE WALLS BY THE CUBIC CONTENTS OF THE FLOOR AND CEILING COMBINED, AND DOUBLE IT. YOU THEN ALLOW HALF THE TOTAL FOR OPENINGS SUCH AS WINDOWS AND DOORS. THEN YOU ALLOW THE OTHER HALF FOR MATCHING THE PATTERN. THEN YOU DOUBLE THE WHOLE THING AGAIN TO GIVE A MARGIN OF ERROR, AND THEN YOU ORDER THE PAPER.

习题 1.21-b 代码附录

分组并求解重合指数:

```
    from collections import Counter

def split_string_into_groups(input_string):
    groups = [[] for _ in range(6)] # 创建包含6个空列表的列表，每个列表代表一个组

    for i, char in enumerate(input_string):
        group_number = i % 6 # 计算字符在哪个组
        groups[group_number].append(char) # 将字符添加到相应的组中

    return [''.join(group) for group in groups] # 将每个组中的字符列表连接成字符串

def calculate_probability_squared(string):
    # 统计每个字母出现的次数
    letter_count = Counter(string)

    # 计算字符串的总长度
    total_length = len(string)

    # 计算每个字母的概率，并将概率平方并累加
    probability_squared_sum = 0.0
    for letter, count in letter_count.items():
        probability = count / total_length
        probability_squared = probability ** 2
        probability_squared_sum += probability_squared

    return probability_squared_sum

input_string =
    "KCCPKBGUFDPHQTYAVINRRTMVGRKDNBVFDETDGILTXRGUDDKOTFMBPVGEGLTGCKQRACQCWDNAWCRXIZAKFTLEWRPTYCQKYVXCHKFTPONCQ
result = split_string_into_groups(input_string)

for i, group in enumerate(result):
    print(f'Group {i+1}: {group}')

for i, string in enumerate(result, start=1):
    result = calculate_probability_squared(string)
    print(f"字符串{i}的概率平方累加结果:", result)
```

求解 $M_g(y_i)$:

```
    from collections import Counter
```

```python

# 频率表
freq = {
    'A': 0.082, 'B': 0.015, 'C': 0.028, 'D': 0.043, 'E': 0.127,
    'F': 0.022, 'G': 0.020, 'H': 0.061, 'I': 0.070, 'J': 0.002,
    'K': 0.008, 'L': 0.040, 'M': 0.024, 'N': 0.067, 'O': 0.075,
    'P': 0.019, 'Q': 0.001, 'R': 0.060, 'S': 0.063, 'T': 0.091,
    'U': 0.028, 'V': 0.010, 'W': 0.023, 'X': 0.001, 'Y': 0.020,
    'Z': 0.001
}

# 输入字符串列表
input_strings = [
    "KGQNGVGGTGCQWAWQHNJEPJTKQFWAPJGHPWKCTAQVNCIVJFVNIVCPQJQJT",
    "CUTRRFIUFEKCCKRKKCVTKVRCDRSFRRKFZTEEJFNYWKKKVFYVRFDFIVIV",
    "CFYRKDLDMGQWRFPYFQAMQDLGZLJSJJMPLFBBRSRCDAFCLSCREEYDYLBN",
    "PDATDETDBLRDXTTVTQJCDASCXSTIAUIDVPDSWPWGDWTGNQLWPXGTCNTP",
    "KPVMNTXKPTANILYXPRUMYHVZGWBAHMTILLPHXEXAKBIGHEABBOZKWHKI",
    "BHIVBDROVGCAZECCOHWSHCSQSCHSKVZSGKGCBZCOABOHISCBBSWFHIHS"
]

# 初始化总的累加值
total_accumulated_value = 0.0

# 遍历每个字符串并计算字母频率并累加
for i, string in enumerate(input_strings, start=1):
    letter_count = Counter(string)
    total_length = len(string)

    for m in range(26):
        accumulated_value = 0.0
        for letter, count in letter_count.items():
            frequency = count / total_length

            # 计算字符串中(i+2) mod 26位置的字母
            index = (ord(letter) - ord('A') - m) % 26
            next_letter = chr(ord('A') + index)

            # 计算res，即频率相乘
            res = frequency * freq[next_letter]
            accumulated_value += res

            total_accumulated_value += accumulated_value
```

```
45        print(f"字符串{i}; m值为{m}的累加结果:", accumulated_value)
```

解密:

```
1   def decrypt_and_combine(text):
2   lines = text.split('\n')
3   decrypted_lines = []
4
5   for i, line in enumerate(lines):
6       shift = 0
7       if i == 0:
8           shift = 2
9       elif i == 1:
10          shift = 17
11      elif i == 2:
12          shift = 24
13      elif i == 3:
14          shift = 15
15      elif i == 4:
16          shift = 19
17      elif i == 5:
18          shift = 14
19
20      decrypted_line = ""
21      for char in line:
22          if char.isalpha():
23              char_code = ord(char)
24              char_code -= ord('A')
25              char_code = (char_code - shift) % 26
26              decrypted_char = chr(char_code + ord('A'))
27              decrypted_line += decrypted_char
28          else:
29              decrypted_line += char
30
31      decrypted_lines.append(decrypted_line)
32
33  combined_text = ""
34  current_index = 0
35
36  while True:
37      for line in decrypted_lines:
38          if current_index < len(line):
39              combined_text += line[current_index]
```

```
        current_index += 1
        if current_index >= max(len(line) for line in decrypted_lines):
            break

    return combined_text

encrypted_text = [
    "KGQNGVGGTGCQWAWQHNJEPJTKQFWAPJGHPWKCTAQVNCIVJFVNIVCPQJQJT",
    "CUTRRFIUFEKCCKRKKCVTKVRCDRSFRRKFZTEEJFNYWKKKVFYVRFDFIVIV",
    "CFYRKDLDMGQWRFPYFQAMQDLGZLJSJJMPLFBBRSRCDAFCLSCREEYDYLBN",
    "PDATDETDBLRDXTTVTQJCDASCXSTIAUIDVPDSWPWGDWTGNQLWPXGTCNTP",
    "KPVMNTXKPTANILYXPRUMYHVZGWBAHMTILLPHXEXAKBIGHEABBOZKWHKI",
    "BHIVBDROVGCAZECCOHWSHCSQSCHSKVZSGKGCBZCOABOHISCBBSWFHIHS"
]

combined_decrypted_text = decrypt_and_combine('\n'.join(encrypted_text))
print(combined_decrypted_text)
```