Домашнее задание 3

Весь код архиватора лежит тут: https://github.com/Andrew-Zlobin/CMDC/tree/main/task3

Смотреть на результаты будем на примере теста из предыдущего задания. Поэтому, для начала откроем его:

In [1]:
```
text = None
with open('war_peace_ascii_Zlobin_AS.txt', 'r', encoding='utf-8') as file
    text = file.read()
```

In [2]:
```
text[:1000]
```

Out[2]:
```
'\nCHAPTER I\n\n"Well, Prince, so Genoa and Lucca are now just family es
tates of the\nBuonapartes. But I warn you, if you don\'t tell me that th
is means war,\nif you still try to defend the infamies and horrors perpe
trated by that\nAntichrist-I really believe he is Antichrist-I will have
nothing\nmore to do with you and you are no longer my friend, no longer
my\n\'faithful slave,\' as you call yourself! But how do you do? I see I
\nhave frightened you-sit down and tell me all the news."\n\nIt was in J
uly, 1805, and the speaker was the well-known Anna Pavlovna\nScherer, ma
id of honor and favorite of the Empress Marya Fedorovna.\nWith these wor
ds she greeted Prince Vasili Kuragin, a man of high\nrank and importance
, who was the first to arrive at her reception. Anna\nPavlovna had had a
cough for some days. She was, as she said, suffering\nfrom la grippe; gr
ippe being then a new word in St. Petersburg, used\nonly by the elite.\n
\nAll her invitations without exception, written in French, and delivere
d\nby a scarle'
```

В файлах https://github.com/Andrew-Zlobin/CMDC/blob/main/task3/BWT.py и https://github.com/Andrew-Zlobin/CMDC/blob/main/task3/DC.py реализованы алгориты BWT и DC соответсвенно.

In [3]:
```
from BWT import BWT
from DC import DC
from utils import BWT_DC_encode_pipeline, BWT_DC_decode_pipeline, alphabe
```

Напишем вспомогательные функции, чтобы убедиться, что они работают корректно:

In [4]:
```
def BWT_DC_encode(text):
    bwt_text = BWT.forward(text)
    alphabet = "".join(sorted(list(set(text))))
    int_alphabet = alphabet_to_number(alphabet)
    dc_text = DC.code(bwt_text, alphabet, BWT.get_char_spacing())
    text_len = len(bwt_text)
    array_to_encode = [text_len] + dc_text
    return array_to_encode, int_alphabet

def BWT_DC_decode(array_to_encode, int_alphabet):
    text_len = array_to_encode[0]
    dc_text = array_to_encode[1:]
```

```
    alphabet = number_to_alphabet(int_alphabet)
    dc_decoded = DC.decode(dc_text, alphabet, text_len)
    return dc_decoded
```

In [5]:
```
prepared_list, alph = BWT_DC_encode(text)
```

```
/home/dr_drew/Projects/CMDC_env/lib/python3.11/site-packages/pydivsufsort/
divsufsort.py:103: UserWarning: converting str argument uses more memory
  inp_p = _get_bytes_pointer(inp)
100%|
```

```
| 320165
0/3201650 [00:00<00:00, 5874511.81it/s]
```

In [6]:
```
decoded_text = BWT_DC_decode(prepared_list, alph)
```

закодируем, а потом раскодируем большой текст:

In [7]:
```
decoded_text[:100]
```

Out[7]:
```
'\nCHAPTER I\n\n"Well, Prince, so Genoa and Lucca are now just family es
tates of the\nBuonapartes. But I '
```

Строки совпадают, значит алгоритмы работают верно

In [8]:
```
[i for i, j in zip(text, decoded_text) if i != j]
```

Out[8]:
```
[]
```

Посмотрим на небольшую статистику массива, который получается после DC:

In [10]:
```
max(prepared_list[1:]), max(prepared_list)
```

Out[10]:
```
(2717697, 3201650)
```

In [11]:
```
len(text)
```

Out[11]:
```
3201649
```

In [12]:
```
len(prepared_list)
```

Out[12]:
```
3123963
```

In [13]:
```
sum([1 for el in prepared_list if el <= 254])
```

Out[13]:
```
3050242
```

In [14]:
```
sum([1 for el in prepared_list if el > 254 and el <= 65789])
```

Out[14]:
```
73522
```

In [15]:
```
sum([1 for el in prepared_list if el > 254 and el > 65789])
```

Out[15]:
```
199
```

Итого, получается, что текст длиной 3201649 символов, преобразуется в массив из 3123963 чисел, 3050004 меньше 254, 199 больше 65789 и 73522

лежат между 254 и 65789

Для сжатия этого массива будем использовать дельта-код Элиаса, арифметическое, и кодирование с переполнением Они реалзованы соответственно в:
https://github.com/Andrew-Zlobin/CMDC/blob/main/task3/compression/elias.py
https://github.com/Andrew-Zlobin/CMDC/blob/main/task3/compression/arithmetic.py
https://github.com/Andrew-Zlobin/CMDC/blob/main/task3/compression/overflow.py

И результаты работы на тексте из предыдущего задания

Дельта код:

Кодирование

In [28]:
```
%%time
!python3 compressor.py war_peace_ascii_Zlobin_AS.txt -d -f result.compres
```

```
/home/dr_drew/Projects/CMDC_env/lib/python3.11/site-packages/pydivsufsort/
divsufsort.py:103: UserWarning: converting str argument uses more memory
  inp_p = _get_bytes_pointer(inp)
100%|████████████████████████████| 3201650/3201650 [00:00<00:00, 5635599.8
9it/s]
CPU times: user 28.6 ms, sys: 8.34 ms, total: 36.9 ms
Wall time: 3.96 s
```

Размер закодированного файла:

In [29]:
```
!ls -l result.compressed
```

```
-rw-r--r-- 1 dr_drew dr_drew 2286674 июн 22 19:19 result.compressed
```

Декодирование:

In [ ]:
```
%%time
!python3 compressor.py result.compressed -d -f decoded.txt
```

CPU times: user 15.3 s, sys: 3.11 s, total: 18.4 s Wall time: 27min 5s

Проверяем, чтобы файл совпал с исходным:

In [31]:
```
!cmp war_peace_ascii_Zlobin_AS.txt decoded.txt
```

Арифметическое кодирование

Кодирование

In [32]:
```
%%time
!python3 compressor.py war_peace_ascii_Zlobin_AS.txt -a -f result.compres
```

```
/home/dr_drew/Projects/CMDC_env/lib/python3.11/site-packages/pydivsufsort/
divsufsort.py:103: UserWarning: converting str argument uses more memory
  inp_p = _get_bytes_pointer(inp)
100%|████████████████████████████| 3201650/3201650 [00:00<00:00, 5833532.8
3it/s]
CPU times: user 2.38 s, sys: 563 ms, total: 2.94 s
Wall time: 9min 32s
```

Размер закодированного файла:

In [33]:
```
!ls -l result.compressed
```

```
-rw-r--r-- 1 dr_drew dr_drew 2056202 июн 22 19:55 result.compressed
```

Декодирование:

In [34]:
```
%%time
!python3 compressor.py result.compressed -a -f decoded.txt
```

```
CPU times: user 2.64 s, sys: 580 ms, total: 3.22 s
Wall time: 10min 41s
```

Проверяем, чтобы файл совпал с исходным:

In [35]:
```
!cmp war_peace_ascii_Zlobin_AS.txt decoded.txt
```

Кодирование с переполнением:

Кодирование

In [36]:
```
%%time
!python3 compressor.py war_peace_ascii_Zlobin_AS.txt -o -f result.compres
```

```
/home/dr_drew/Projects/CMDC_env/lib/python3.11/site-packages/pydivsufsort/
divsufsort.py:103: UserWarning: converting str argument uses more memory
  inp_p = _get_bytes_pointer(inp)
100%|████████████████████████████| 3201650/3201650 [00:00<00:00, 5806683.6
6it/s]
CPU times: user 15.5 ms, sys: 8.66 ms, total: 24.1 ms
Wall time: 3.22 s
```

Размер закодированного файла:

In [37]:
```
!ls -l result.compressed
```

```
-rw-r--r-- 1 dr_drew dr_drew 3272233 июн 22 20:06 result.compressed
```

Декодирование:

In [38]:
```
%%time
!python3 compressor.py result.compressed -o -f decoded.txt
```

```
CPU times: user 38.6 ms, sys: 8.37 ms, total: 47 ms
Wall time: 8.65 s
```

Проверяем, чтобы файл совпал с исходным:

In [39]:
```
!cmp war_peace_ascii_Zlobin_AS.txt decoded.txt
```

In [40]:
```
!ls -l war_peace_ascii_Zlobin_AS.txt
```

```
-rw-r--r-- 1 dr_drew dr_drew 3201649 июн 20 15:02 war_peace_ascii_Zlobin_A
S.txt
```

Итого получилось, что исходный текст объёмом 3.05 мб удалось сжать до 1.95 мб арифметическим кодированием и до 2.18 мб дельта кодом. При кодировании с переполнением объём остался почти таким же, это может быть обусловлено тем, что после DC количество чисел было всего на 77686 меньше количества символов в исходном тексте, при этом если в исходном тексте один символ занимал один байт, то при сжимании массива 73522 символов кодировалось уже большим числом байт, засчёт чего объём закодированного массива увеличился.

In [ ]: