Zadanie 1 z listy 5 - “Kompresja Danych”

Łukasz Klasiński

26 kwietnia 2020

Zadanie 2

Dla alfabetu wejściowego o rozmiarze a i liczby naturalnej k skonstruuj jak najdłuższy tekst wejściowy, dla którego algorytm LZ78/LZW z nieograniczonym rozmiarem słownika nie użyje ani razu dopasowania dłuższego niż k-literowe.

Rozwiązanie

Pomysł: Chcemy budować tak słowo, aby na koniec w słowniku były wszystkie kombinacje liter ze słownika o wszystkich długościach $\leq k$.

Przykład:

· $k = 1$

· alfabet = $[a, b]$

Na początku mamy pusty słownik (LZ78), zatem układamy słowo tak jak robiłby to słownik:

· słowo = a

· słownik $\{a\}$

· słowo = $a \oplus b$

· słownik $\{a, b\}$

Teraz dla $k = 2$ robimy to samo co wcześniej tylko dodatkowo rozważamy kombinacje długości 2:

· słowo = $ab \oplus aa$

· słownik $\{a, b, aa\}$

· słowo = $abaa \oplus ab$

· słownik = $\{a, b, aa, ab\}$

I tak dalej. Powyższy algorytm możemy bardzo łatwo zapisać w $MUJP$:

```
s = ""
n = len(alphabet)

for k in range(1, n + 1):
    for combination in combinations_with_replacement(alphabet, k):
        s += ''.join(combination)
```

Taka konstrukcja wymusi zapisanie każdej możliwej wartości w słowniu, zatem nie da się zrobić dłuższego, bo po jego zapełnieniu kolejne k elementów słowa na pewno już by w nim było, więc zostało by zmachowane z nim $+1$ elementem za nim występującym. Ostatecznie otrzymamy słowo długości $|a|^k$.