Algorytmy i struktury danych do przetwarzania tekstów

czyli moje zainteresowania naukowe

Tomasz Kociumaka
kociumaka@mimuw.edu.pl

Wydział Matematyki, Informatyki i Mechaniki Uniwersytetu Warszawskiego

Wręczenie Nagrody im. Witolda Lipskiego
Warszawa, 4 października 2018

Teksty

Słowo — skończony ciąg symboli z ustalonego zbioru (alfabetu)

Teksty

Słowo — skończony ciąg symboli z ustalonego zbioru (alfabetu)

```
0001000001011011001111001111011001110111100001010111111010001001110110101
1110000011110110101101101000110100001110100000010110100101001111111011000
1011011010010000011001101110110111101011010001111010001001011000000000000

GTGGACCTCCCTGCAGGCCCTGGCTGAAGCAGCTTCCCCCTCCACTCTCTATCCTATTCCTCTTTTATTCAT
AGAATATATCACTGAATTTAACTCATGTATGTTTGTAAAAACATTTATTGTGTGGTACCCCGCGGGAATGCA
GGTGCCATGAGAGTGTCCTCATCTGTCTTTTCCATCACTGTACCACAGCACCTAAAAACACATAGGTATGTT

Fundacja Rozwoju Informatyki, przy współpracy z Polskim Stowarzyszeniem
dla Maszyn Liczących (Polish Chapter of ACM) i Polskim Towarzystwem
Informatycznym, z inicjatywy grupy polskich informatyków pracujących za
granicą ustanawia nagrodę dla młodych polskich naukowców za dorobek
w dziedzinie informatyki i jej zastosowań.

83 56 92 24 79 83 76 49 53 4 87 76 66 42 79 3 16 11 81 7 20 55 93 75 4 29
51 30 35 38 32 5 79 92 53 30 9 65 5 86 28 5 95 93 30 77 52 27 75 80 38 38
11 63 67 87 76 41 11 78 7 67 58 6 66 30 80 73 47 79 61 51 76 69 29 83 55
```

Wyszukiwanie wzorca: algorytmy i struktury danych

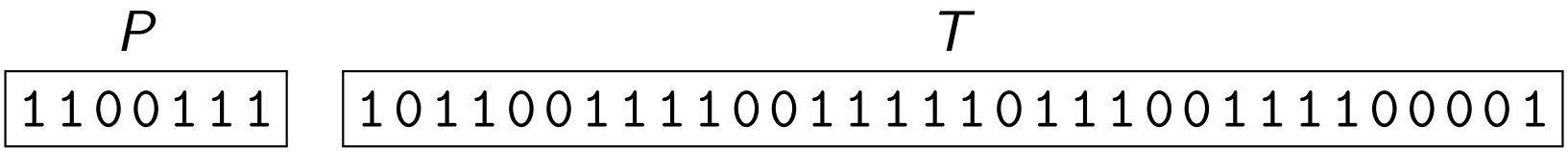
Problem

Znajdź fragmenty tekstu T (dł. n) pasujące do wzorca P (dł. m).

Wyszukiwanie wzorca: algorytmy i struktury danych

Problem

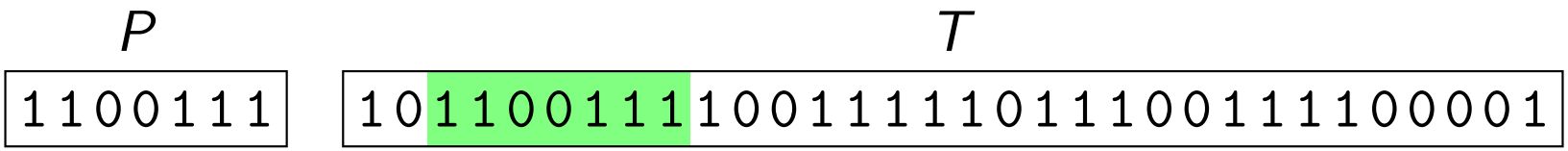
Znajdź fragmenty tekstu T (dł. n) pasujące do wzorca P (dł. m).



Wyszukiwanie wzorca: algorytmy i struktury danych

Problem

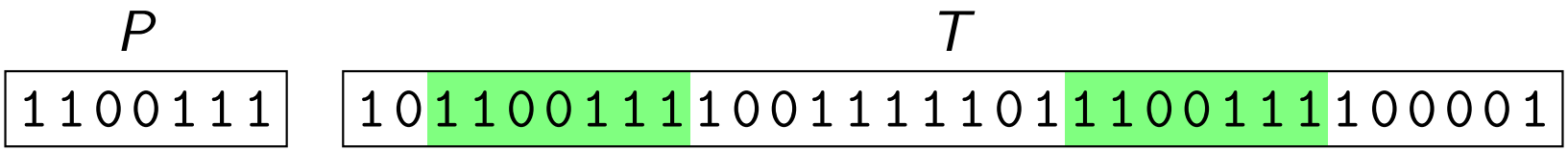
Znajdź fragmenty tekstu T (dł. n) pasujące do wzorca P (dł. m).



Wyszukiwanie wzorca: algorytmy i struktury danych

Problem

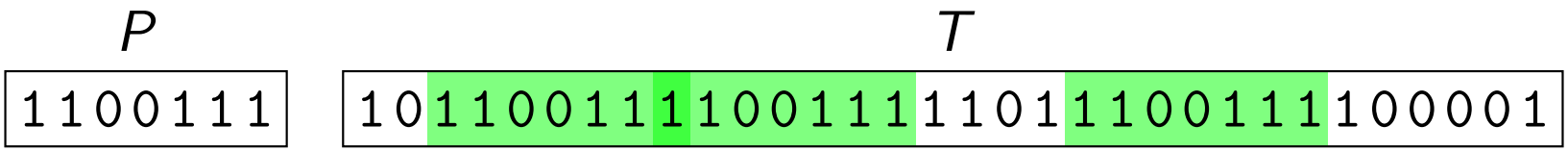
Znajdź fragmenty tekstu T (dł. n) pasujące do wzorca P (dł. m).



Wyszukiwanie wzorca: algorytmy i struktury danych

Problem

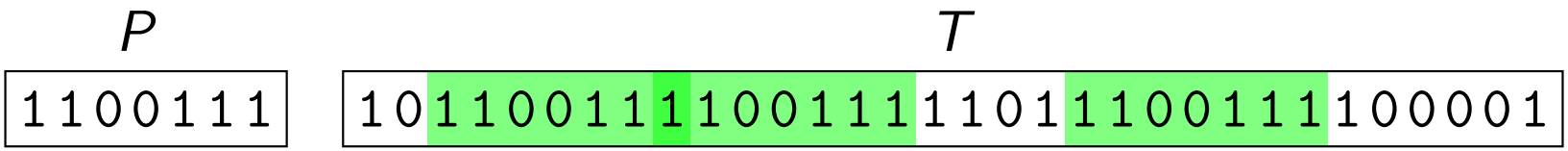
Znajdź fragmenty tekstu T (dł. n) pasujące do wzorca P (dł. m).



Wyszukiwanie wzorca: algorytmy i struktury danych

Problem

Znajdź fragmenty tekstu T (dł. n) pasujące do wzorca P (dł. m).

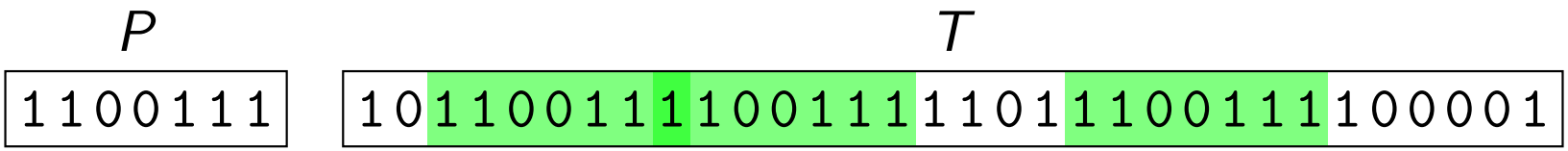


Dwie klasyczne wersje problemu:

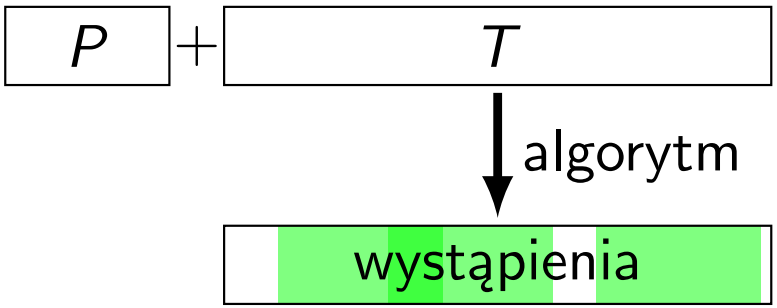
Wyszukiwanie wzorca: algorytmy i struktury danych

Problem

Znajdź fragmenty tekstu T (dł. n) pasujące do wzorca P (dł. m).



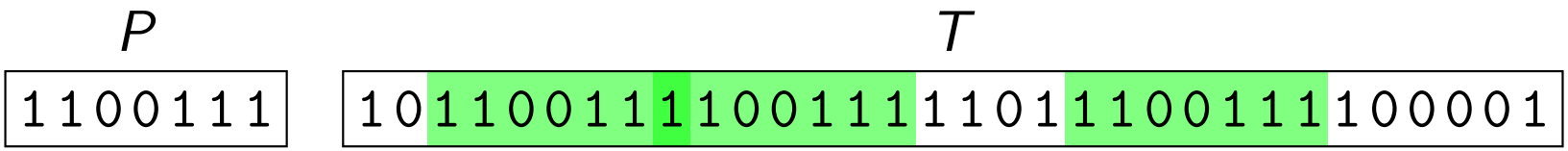
Dwie klasyczne wersje problemu:



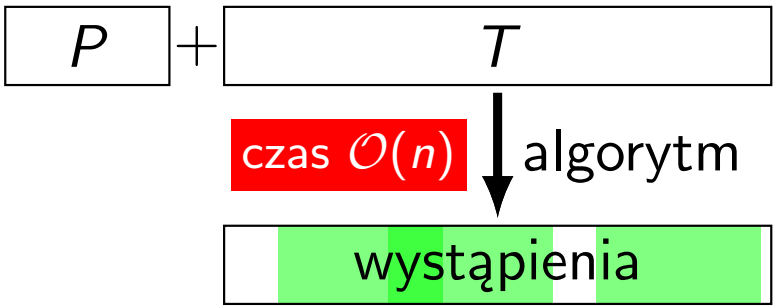
Wyszukiwanie wzorca: algorytmy i struktury danych

Problem

Znajdź fragmenty tekstu T (dł. n) pasujące do wzorca P (dł. m).



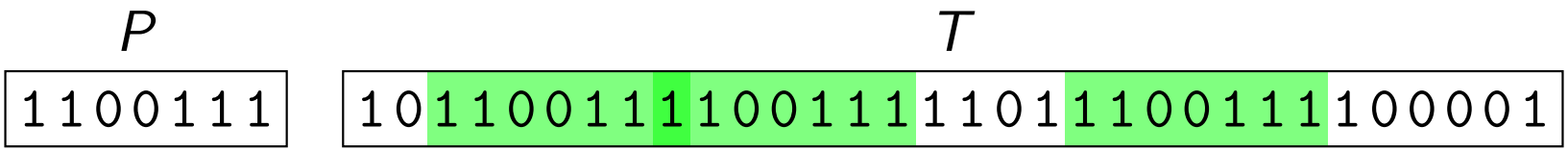
Dwie klasyczne wersje problemu:



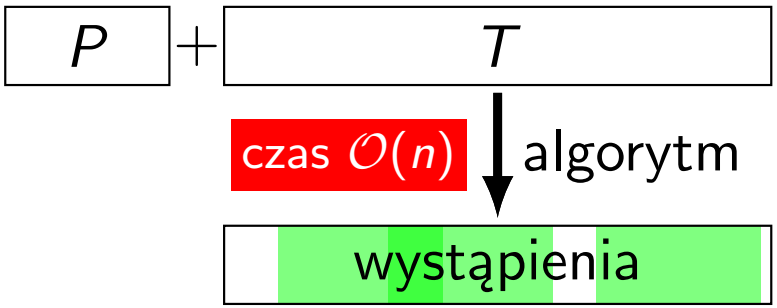
Wyszukiwanie wzorca: algorytmy i struktury danych

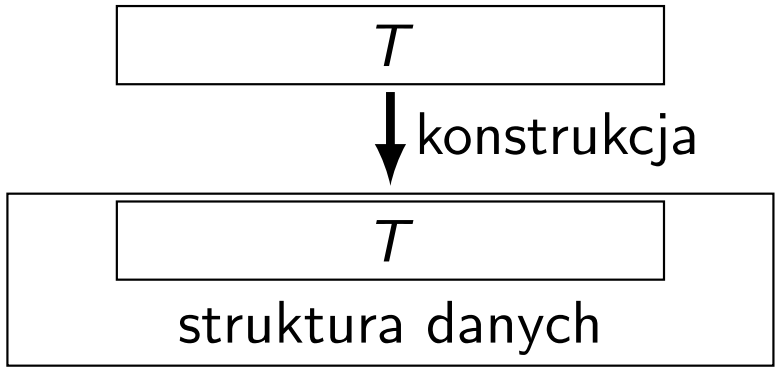
Problem

Znajdź fragmenty tekstu T (dł. n) pasujące do wzorca P (dł. m).



Dwie klasyczne wersje problemu:

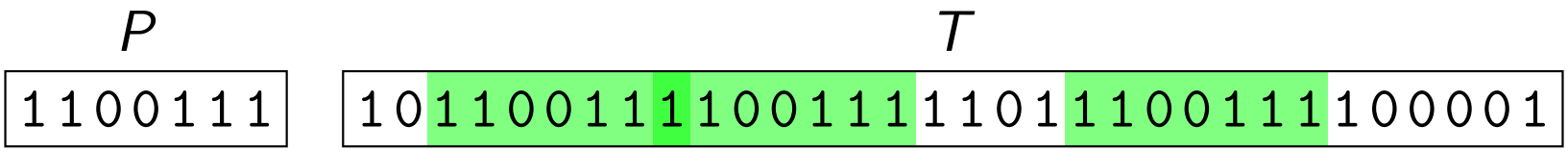




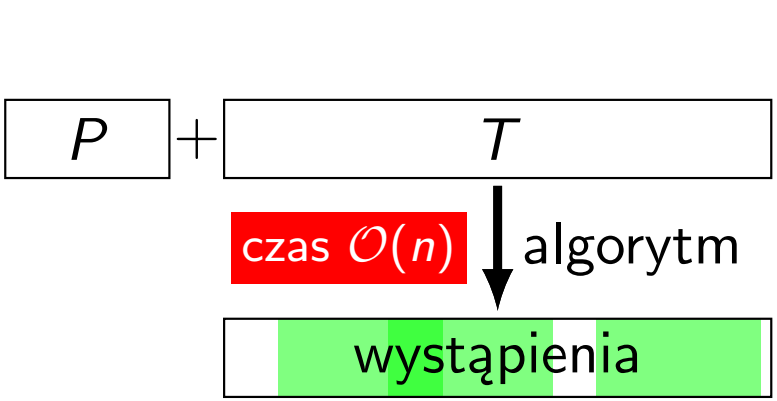
Wyszukiwanie wzorca: algorytmy i struktury danych

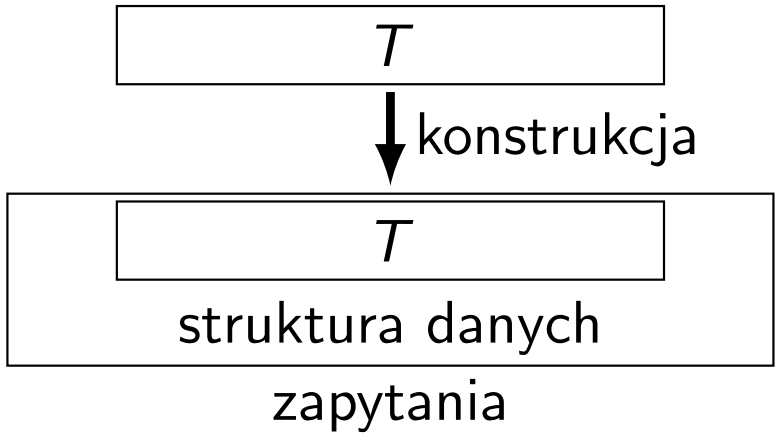
Problem

Znajdź fragmenty tekstu T (dł. n) pasujące do wzorca P (dł. m).



Dwie klasyczne wersje problemu:

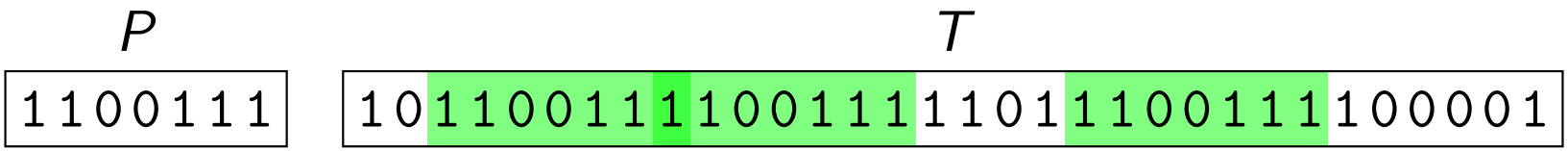




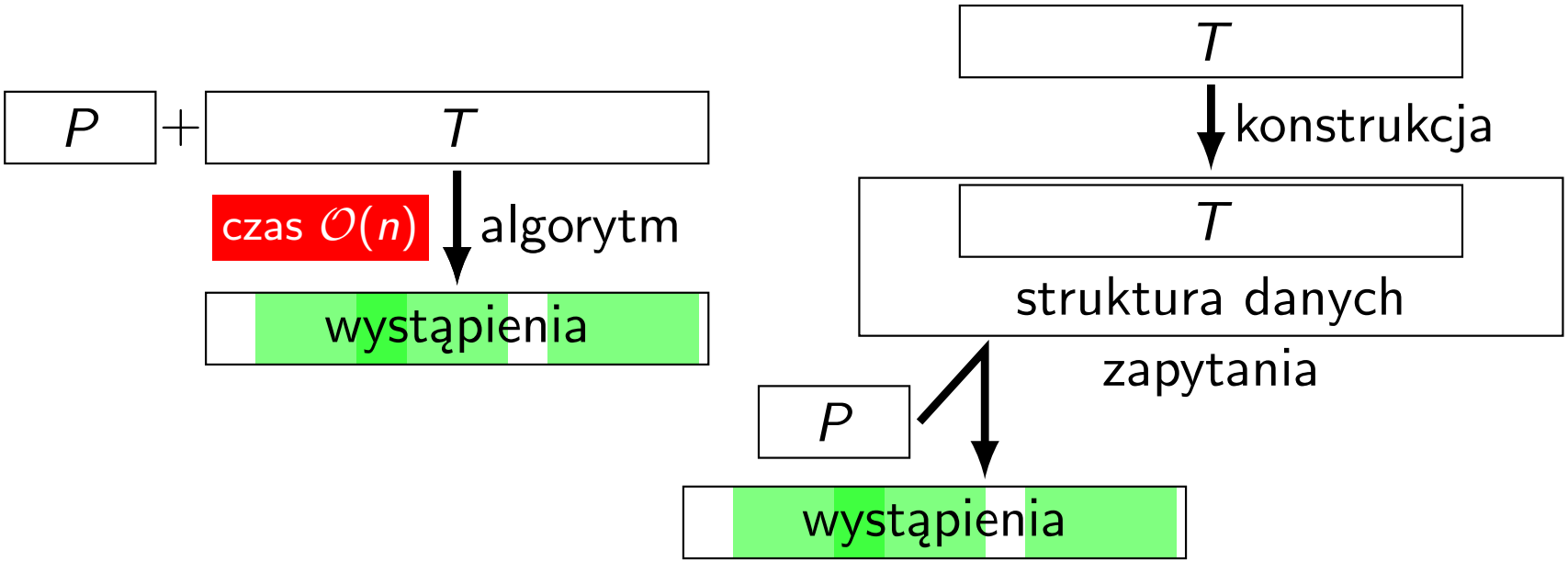
Wyszukiwanie wzorca: algorytmy i struktury danych

Problem

Znajdź fragmenty tekstu T (dł. n) pasujące do wzorca P (dł. m).



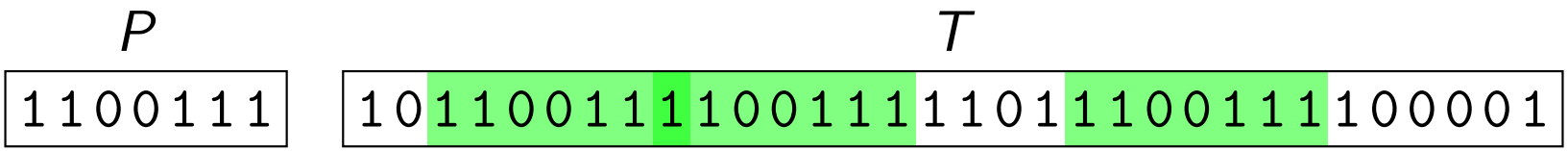
Dwie klasyczne wersje problemu:



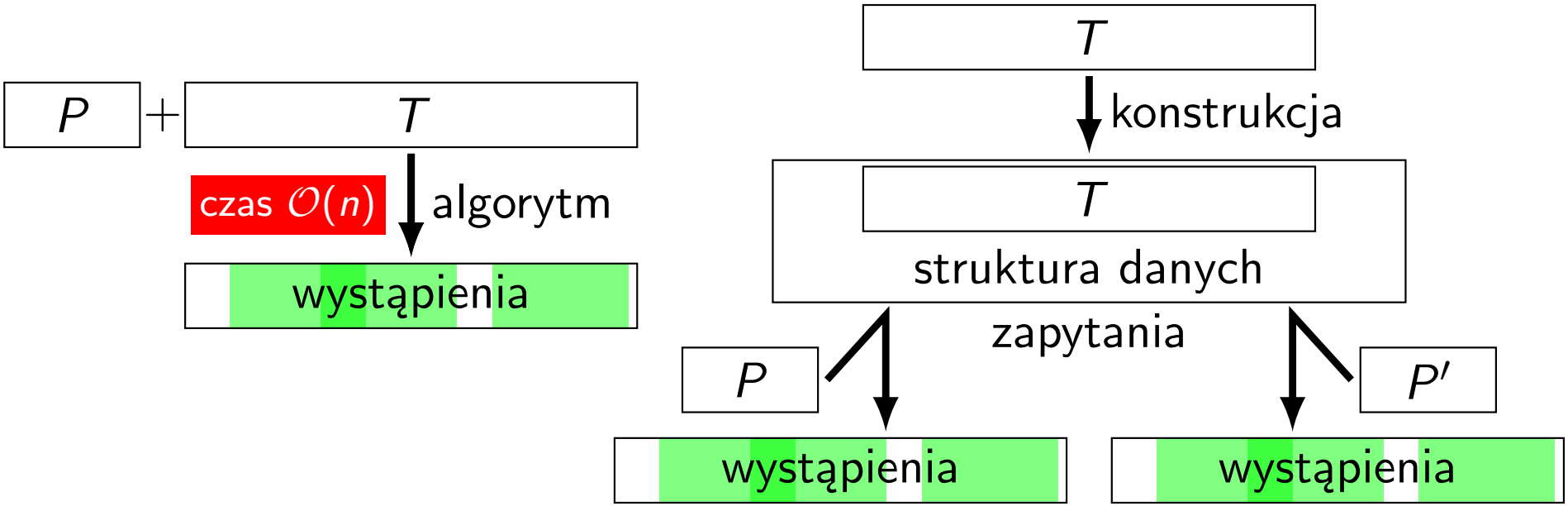
Wyszukiwanie wzorca: algorytmy i struktury danych

Problem

Znajdź fragmenty tekstu T (dł. n) pasujące do wzorca P (dł. m).



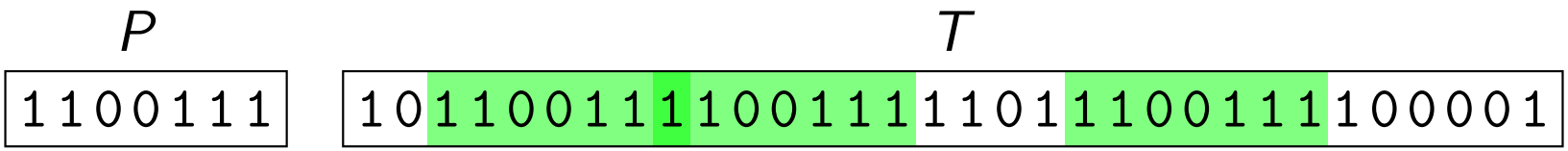
Dwie klasyczne wersje problemu:



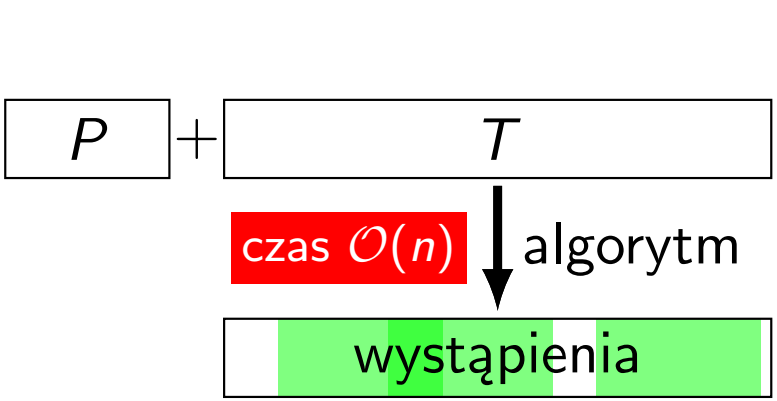
Wyszukiwanie wzorca: algorytmy i struktury danych

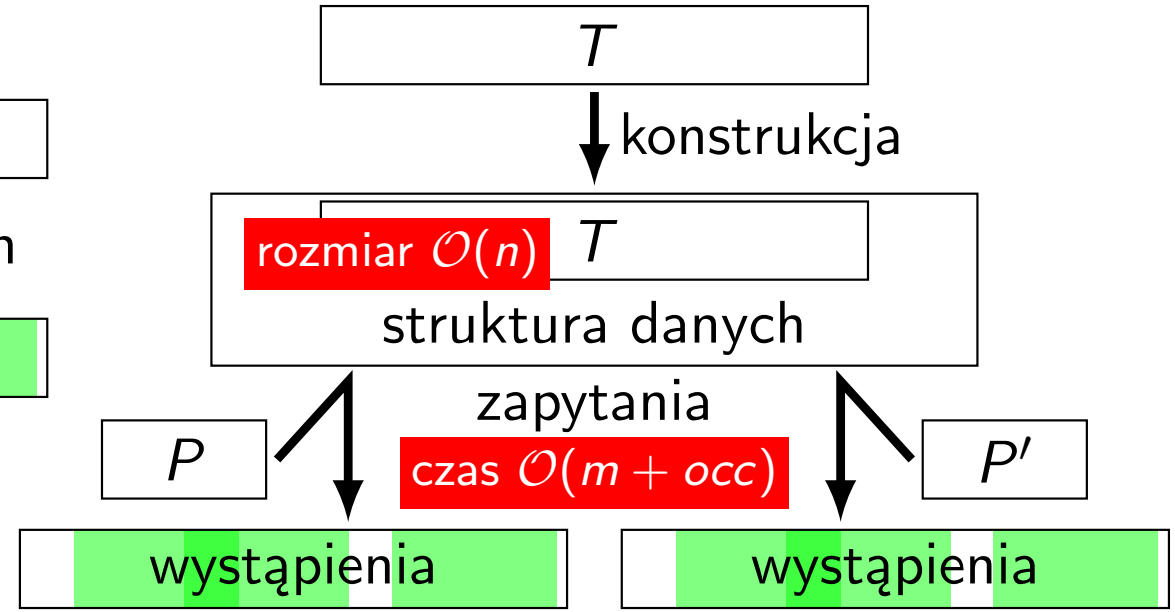
Problem

Znajdź fragmenty tekstu T (dł. n) pasujące do wzorca P (dł. m).



Dwie klasyczne wersje problemu:

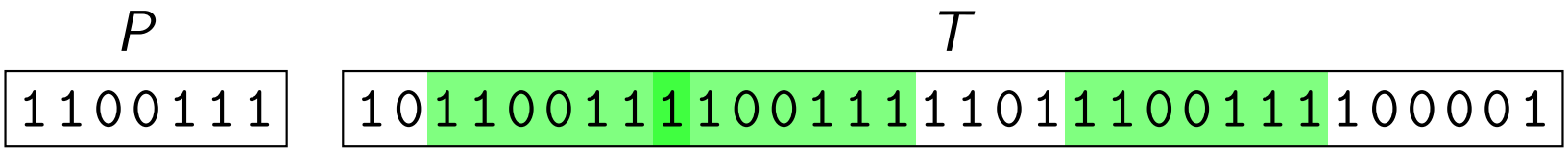




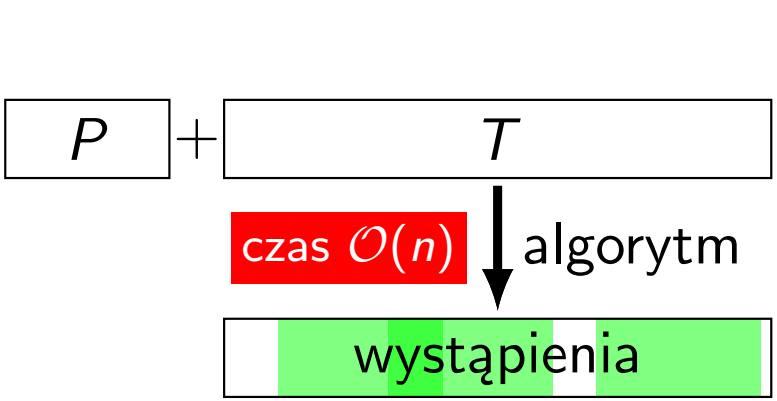
Wyszukiwanie wzorca: algorytmy i struktury danych

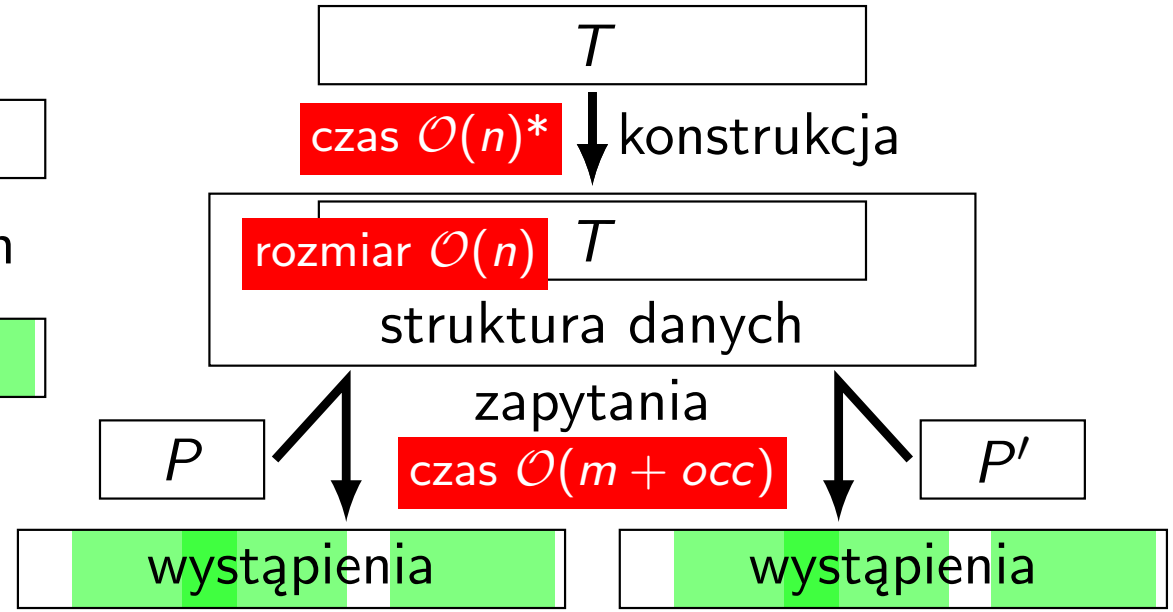
Problem

Znajdź fragmenty tekstu T (dł. n) pasujące do wzorca P (dł. m).



Dwie klasyczne wersje problemu:





Wybrane kierunki moich badań

Struktury danych do zapytań wewnętrznych

Utrzymywanie dynamicznych kolekcji słów

Przetwarzanie tekstów w małej pamięci roboczej

Wybrane kierunki moich badań

Struktury danych do zapytań wewnętrznych

Utrzymywanie dynamicznych kolekcji słów

Przetwarzanie tekstów w małej pamięci roboczej

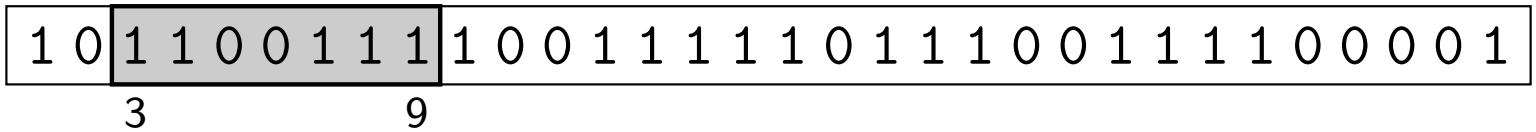
Zapytania wewnętrzne

1. Statyczny tekst T długości n.

1	0	1	1	0	0	1	1	1	1	0	0	1	1	1	1	0	1	1	1	0	0	1	1	1	1	0	0	0	1

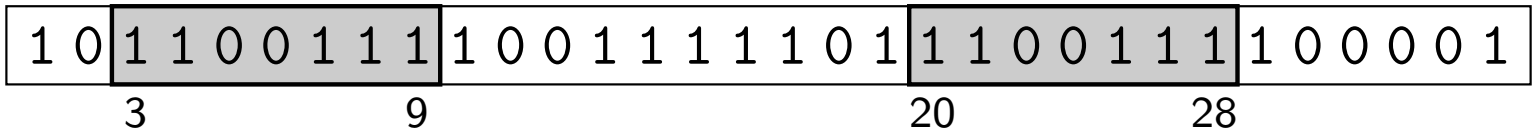
Zapytania wewnętrzne

1. Statyczny tekst T długości n.
2. Zapytania dotyczące fragmentów $T[i..j]$ tekstu T.
   - Każdy fragment $T[i..j]$ reprezentowany przez skrajne pozycje i oraz j.



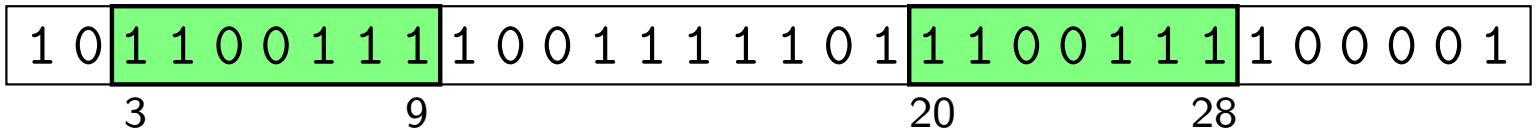
Zapytania wewnętrzne

1. Statyczny tekst T długości n.
2. Zapytania dotyczące fragmentów $T[i..j]$ tekstu T.
   - Każdy fragment $T[i..j]$ reprezentowany przez skrajne pozycje i oraz j.



Zapytania wewnętrzne

1. Statyczny tekst T długości n.
2. Zapytania dotyczące fragmentów $T[i..j]$ tekstu T.
   - Każdy fragment $T[i..j]$ reprezentowany przez skrajne pozycje i oraz j.

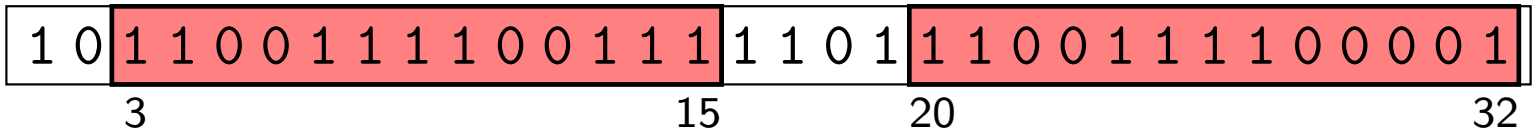


Podstawowe zapytania:

- Czy $T[i..j]$ pasuje do $T[i'..j']$?
  $T[3..9] = T[20..28]$

Zapytania wewnętrzne

1. Statyczny tekst T długości n.
2. Zapytania dotyczące fragmentów $T[i..j]$ tekstu T.
   - Każdy fragment $T[i..j]$ reprezentowany przez skrajne pozycje i oraz j.

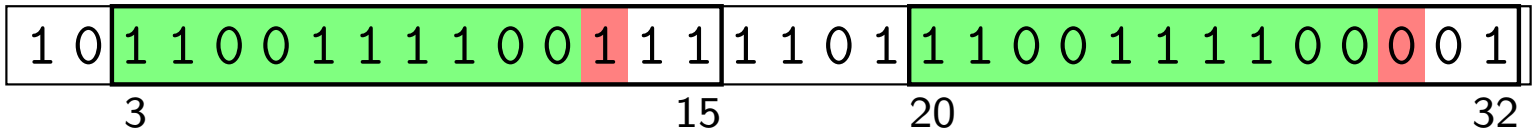


Podstawowe zapytania:

- Czy $T[i..j]$ pasuje do $T[i'..j']$?
  $T[3..9] = T[20..28]$, $T[3..15] \neq T[20..32]$.

Zapytania wewnętrzne

1. Statyczny tekst T długości n.
2. Zapytania dotyczące fragmentów $T[i..j]$ tekstu T.
   - Każdy fragment $T[i..j]$ reprezentowany przez skrajne pozycje i oraz j.

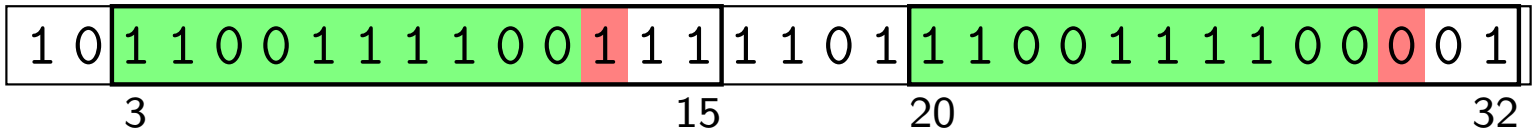


Podstawowe zapytania:

- Czy $T[i..j]$ pasuje do $T[i'..j']$?
  $T[3..9] = T[20..28]$, $T[3..15] \neq T[20..32]$.
- Jaki jest najdłuższy wspólny prefiks $T[i..j]$ oraz $T[i'..j']$?
  $\mathrm{lcp}(T[3..15], T[20..32]) = 10$.

Zapytania wewnętrzne

1. Statyczny tekst T długości n.
2. Zapytania dotyczące fragmentów $T[i..j]$ tekstu T.
   - Każdy fragment $T[i..j]$ reprezentowany przez skrajne pozycje i oraz j.

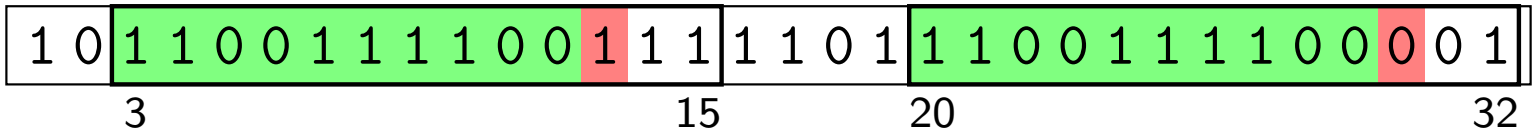


Podstawowe zapytania:

- Czy $T[i..j]$ pasuje do $T[i'..j']$?
  $T[3..9] = T[20..28]$, $T[3..15] \neq T[20..32]$.
- Jaki jest najdłuższy wspólny prefiks $T[i..j]$ oraz $T[i'..j']$?
  $\mathrm{lcp}(T[3..15], T[20..32]) = 10$.
- Czy $T[i..j]$ jest leksykograficznie mniejszy niż $T[i'..j']$?
  $T[3..15] > T[20..32]$.

Zapytania wewnętrzne

1. Statyczny tekst T długości n.
2. Zapytania dotyczące fragmentów $T[i..j]$ tekstu T.
   - Każdy fragment $T[i..j]$ reprezentowany przez skrajne pozycje i oraz j.



Podstawowe zapytania:

- Czy $T[i..j]$ pasuje do $T[i'..j']$?
  $T[3..9] = T[20..28]$, $T[3..15] \neq T[20..32]$.
- Jaki jest najdłuższy wspólny prefiks $T[i..j]$ oraz $T[i'..j']$?
  $\mathrm{lcp}(T[3..15], T[20..32]) = 10$.
- Czy $T[i..j]$ jest leksykograficznie mniejszy niż $T[i'..j']$?
  $T[3..15] > T[20..32]$.

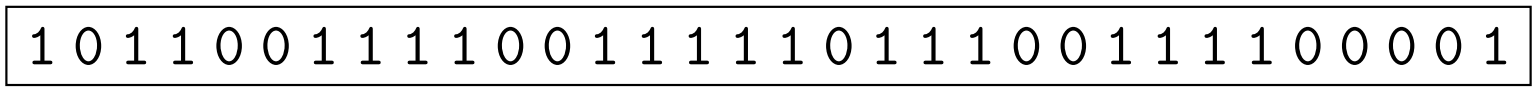
Motywacja: Podproblemy wielu algorytmów i struktur danych.

Wewnętrzne wyszukiwanie wzorca

K, Radoszewski, Rytter, Waleń (SODA 2015)

Zapytania

Znajdź wystąpienia fragmentu x zawarte we fragmencie y.

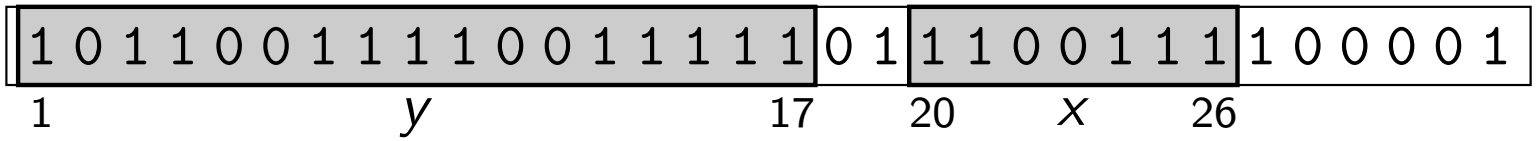


Wewnętrzne wyszukiwanie wzorca

K, Radoszewski, Rytter, Waleń (SODA 2015)

Zapytania

Znajdź wystąpienia fragmentu x zawarte we fragmencie y.



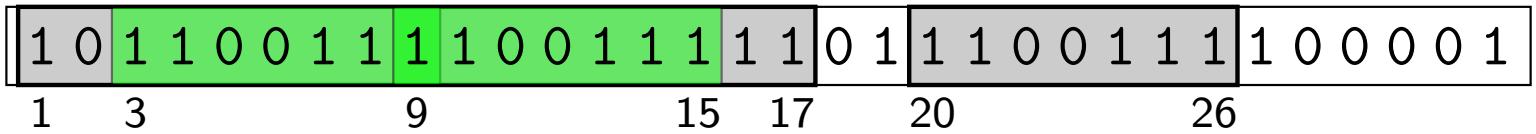
Wystąpienia $x = T[20..26]$ w $y = T[1..17]$ to: .

Wewnętrzne wyszukiwanie wzorca

K, Radoszewski, Rytter, Waleń (SODA 2015)

Zapytania

Znajdź wystąpienia fragmentu x zawarte we fragmencie y.



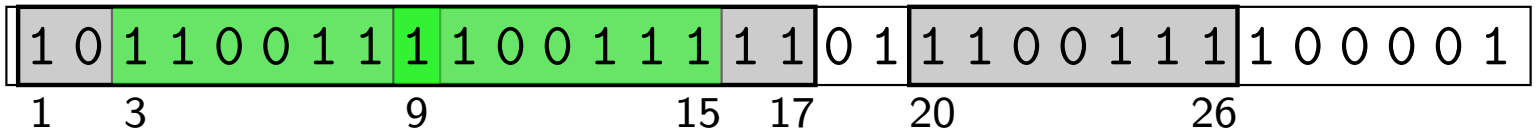
Wystąpienia $x = T[20..26]$ w $y = T[1..17]$ to: $T[3..9]$ oraz $T[9..15]$.

Wewnętrzne wyszukiwanie wzorca

K, Radoszewski, Rytter, Waleń (SODA 2015)

Zapytania

Znajdź wystąpienia fragmentu x zawarte we fragmencie y.



Wystąpienia $x = T[20..26]$ w $y = T[1..17]$ to: $T[3..9]$ oraz $T[9..15]$.

- Czas zapytania $\mathcal{O}(|y|/|x|)$
- Rozmiar $\mathcal{O}(n)$
- Czas konstrukcji $\mathcal{O}(n)$

Najmniejszy i największy leksykograficznie sufiks podsłów

Babenko, Gawrychowski, K, Starikovskaya (CPM 2014), K (CPM 2016)

Zapytania

Wyznacz najmniejszy oraz największy leksykograficznie sufiks fragmentu x.

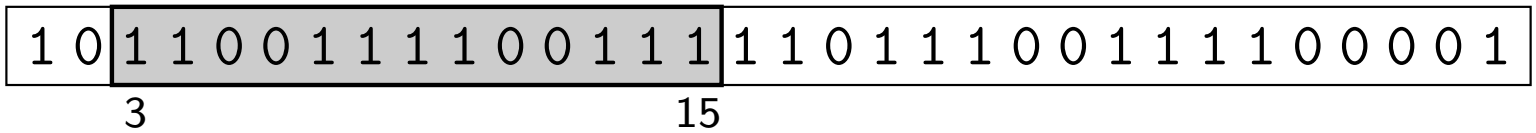
1	0	1	1	0	0	1	1	1	1	0	0	1	1	1	1	1	0	1	1	1	0	0	1	1	1	1	0	0	0	1

Najmniejszy i największy leksykograficznie sufiks podsłów

Babenko, Gawrychowski, K, Starikovskaya (CPM 2014), K (CPM 2016)

Zapytania

Wyznacz najmniejszy oraz największy leksykograficznie sufiks fragmentu x.



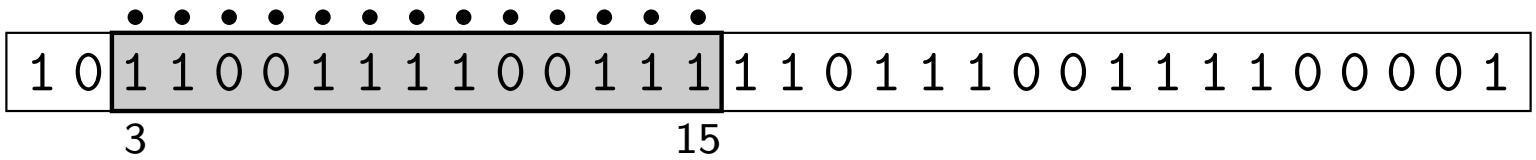
Najmniejszy leksykograficznie sufiks $T[3..15]$ to

Najmniejszy i największy leksykograficznie sufiks podsłów

Babenko, Gawrychowski, K, Starikovskaya (CPM 2014), K (CPM 2016)

Zapytania

Wyznacz najmniejszy oraz największy leksykograficznie sufiks fragmentu x.



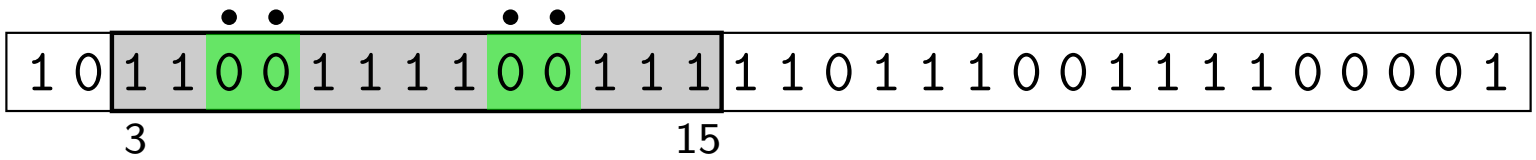
Najmniejszy leksykograficznie sufiks $T[3..15]$ to

Najmniejszy i największy leksykograficznie sufiks podsłów

Babenko, Gawrychowski, K, Starikovskaya (CPM 2014), K (CPM 2016)

Zapytania

Wyznacz najmniejszy oraz największy leksykograficznie sufiks fragmentu x.



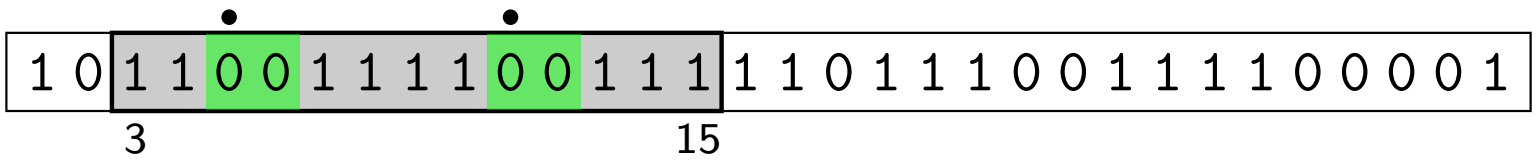
Najmniejszy leksykograficznie sufiks $T[3..15]$ to 0

Najmniejszy i największy leksykograficznie sufiks podsłów

Babenko, Gawrychowski, K, Starikovskaya (CPM 2014), K (CPM 2016)

Zapytania

Wyznacz najmniejszy oraz największy leksykograficznie sufiks fragmentu x.



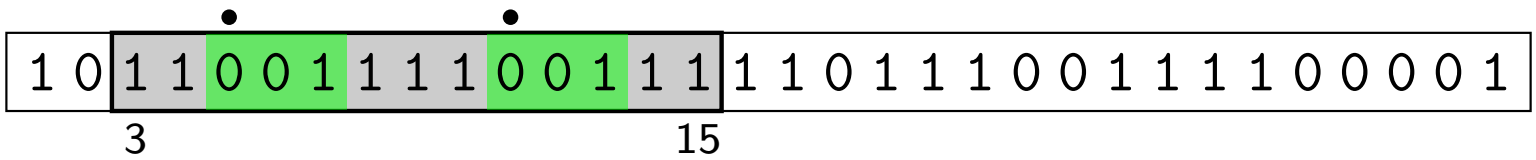
Najmniejszy leksykograficznie sufiks $T[3..15]$ to 00

Najmniejszy i największy leksykograficznie sufiks podsłów

Babenko, Gawrychowski, K, Starikovskaya (CPM 2014), K (CPM 2016)

Zapytania

Wyznacz najmniejszy oraz największy leksykograficznie sufiks fragmentu x.



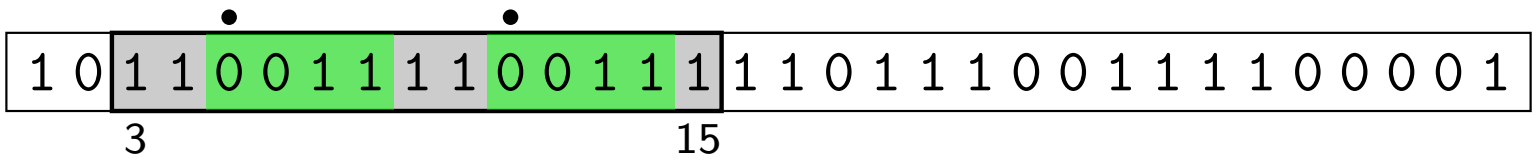
Najmniejszy leksykograficznie sufiks $T[3..15]$ to 001

Najmniejszy i największy leksykograficznie sufiks podsłów

Babenko, Gawrychowski, K, Starikovskaya (CPM 2014), K (CPM 2016)

Zapytania

Wyznacz najmniejszy oraz największy leksykograficznie sufiks fragmentu x.



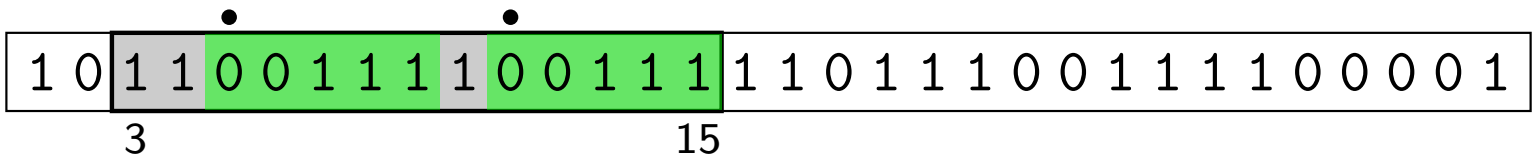
Najmniejszy leksykograficznie sufiks $T[3..15]$ to 0011

Najmniejszy i największy leksykograficznie sufiks podsłów

Babenko, Gawrychowski, K, Starikovskaya (CPM 2014), K (CPM 2016)

Zapytania

Wyznacz najmniejszy oraz największy leksykograficznie sufiks fragmentu x.



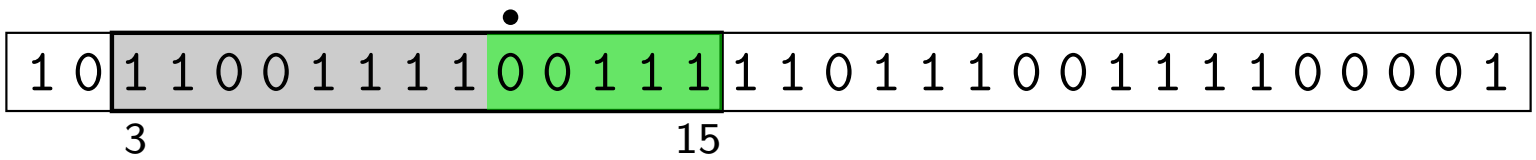
Najmniejszy leksykograficznie sufiks $T[3..15]$ to 00111

Najmniejszy i największy leksykograficznie sufiks podsłów

Babenko, Gawrychowski, K, Starikovskaya (CPM 2014), K (CPM 2016)

Zapytania

Wyznacz najmniejszy oraz największy leksykograficznie sufiks fragmentu x.



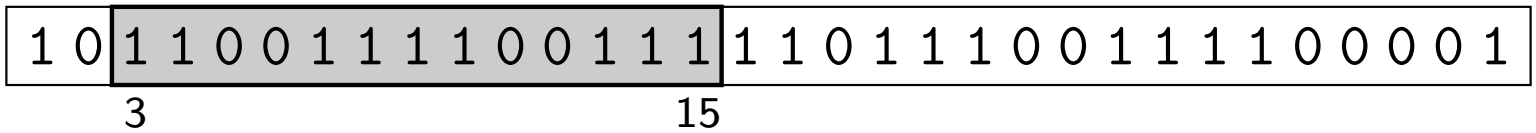
Najmniejszy leksykograficznie sufiks $T[3..15]$ to $00111 = T[11..15]$.

Najmniejszy i największy leksykograficznie sufiks podsłów

Babenko, Gawrychowski, K, Starikovskaya (CPM 2014), K (CPM 2016)

Zapytania

Wyznacz najmniejszy oraz największy leksykograficznie sufiks fragmentu x.



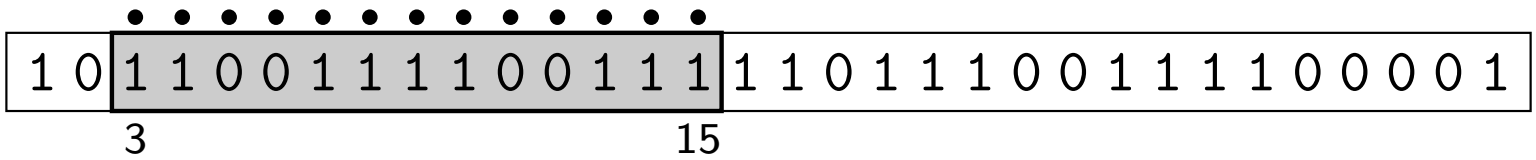
Najmniejszy leksykograficznie sufiks $T[3..15]$ to $00111 = T[11..15]$.
Największy leksykograficznie sufiks $T[3..15]$ to

Najmniejszy i największy leksykograficznie sufiks podsłów

Babenko, Gawrychowski, K, Starikovskaya (CPM 2014), K (CPM 2016)

Zapytania

Wyznacz najmniejszy oraz największy leksykograficznie sufiks fragmentu x.



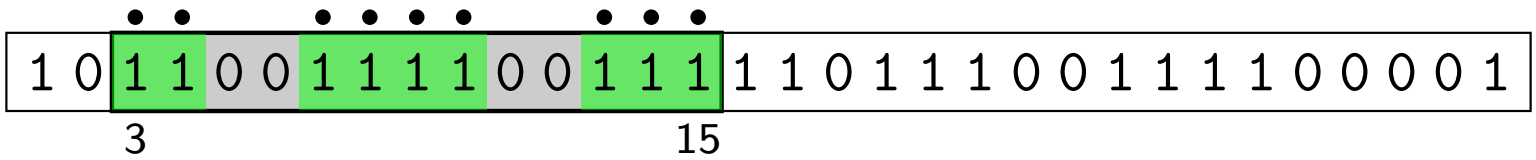
Najmniejszy leksykograficznie sufiks $T[3..15]$ to $00111 = T[11..15]$.
Największy leksykograficznie sufiks $T[3..15]$ to

Najmniejszy i największy leksykograficznie sufiks podsłów

Babenko, Gawrychowski, K, Starikovskaya (CPM 2014), K (CPM 2016)

Zapytania

Wyznacz najmniejszy oraz największy leksykograficznie sufiks fragmentu x.



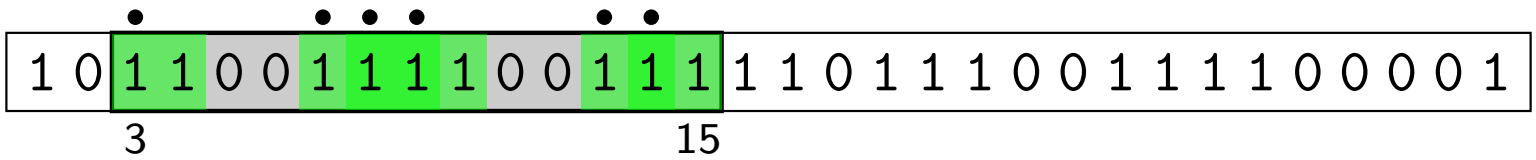
Najmniejszy leksykograficznie sufiks $T[3..15]$ to $00111 = T[11..15]$.
Największy leksykograficznie sufiks $T[3..15]$ to 1

Najmniejszy i największy leksykograficznie sufiks podsłów

Babenko, Gawrychowski, K, Starikovskaya (CPM 2014), K (CPM 2016)

Zapytania

Wyznacz najmniejszy oraz największy leksykograficznie sufiks fragmentu x.



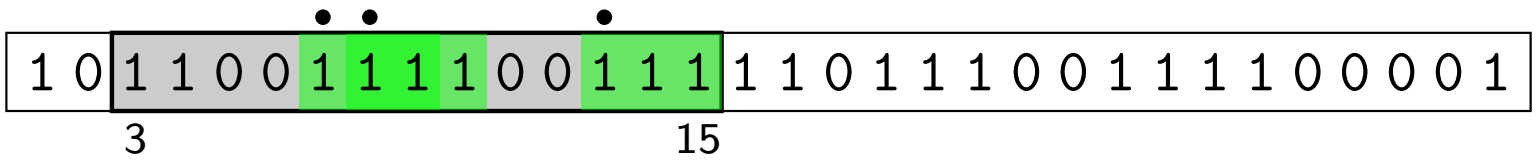
Najmniejszy leksykograficznie sufiks $T[3..15]$ to $00111 = T[11..15]$.
Największy leksykograficznie sufiks $T[3..15]$ to 11

Najmniejszy i największy leksykograficznie sufiks podsłów

Babenko, Gawrychowski, K, Starikovskaya (CPM 2014), K (CPM 2016)

Zapytania

Wyznacz najmniejszy oraz największy leksykograficznie sufiks fragmentu x.



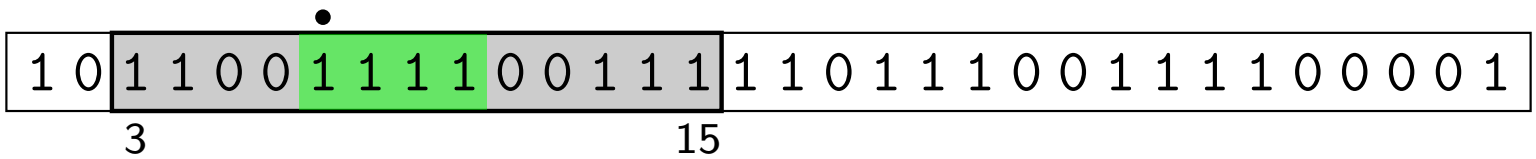
Najmniejszy leksykograficznie sufiks $T[3..15]$ to $00111 = T[11..15]$.
Największy leksykograficznie sufiks $T[3..15]$ to 111

Najmniejszy i największy leksykograficznie sufiks podsłów

Babenko, Gawrychowski, K, Starikovskaya (CPM 2014), K (CPM 2016)

Zapytania

Wyznacz najmniejszy oraz największy leksykograficznie sufiks fragmentu x.



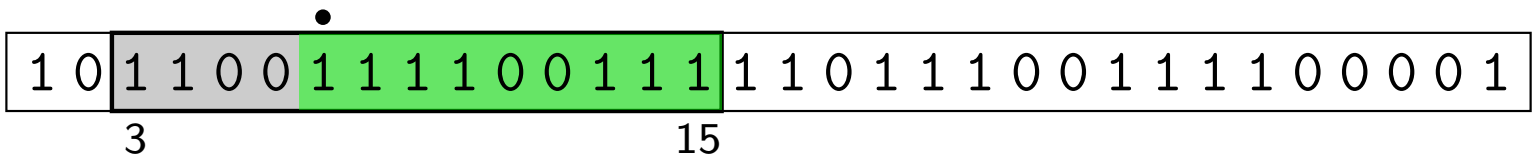
Najmniejszy leksykograficznie sufiks $T[3..15]$ to $00111 = T[11..15]$.
Największy leksykograficznie sufiks $T[3..15]$ to 1111

Najmniejszy i największy leksykograficznie sufiks podsłów

Babenko, Gawrychowski, K, Starikovskaya (CPM 2014), K (CPM 2016)

Zapytania

Wyznacz najmniejszy oraz największy leksykograficznie sufiks fragmentu x.



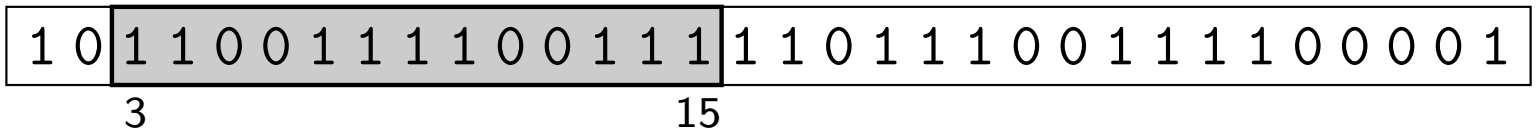
Najmniejszy leksykograficznie sufiks $T[3..15]$ to $00111 = T[11..15]$.
Największy leksykograficznie sufiks $T[3..15]$ to $111100111 = T[7..15]$.

Najmniejszy i największy leksykograficznie sufiks podsłów

Babenko, Gawrychowski, K, Starikovskaya (CPM 2014), K (CPM 2016)

Zapytania

Wyznacz najmniejszy oraz największy leksykograficznie sufiks fragmentu x.



Najmniejszy leksykograficznie sufiks $T[3..15]$ to $00111 = T[11..15]$.
Największy leksykograficznie sufiks $T[3..15]$ to $111100111 = T[7..15]$.

- Czas zapytania $\mathcal{O}(1)$
- Rozmiar $\mathcal{O}(n)$
- Czas konstrukcji $\mathcal{O}(n)$

Selekcja sufiksów podsłów

Babenko, Gawrychowski, K, Starikovskaya (SODA 2015)

Zapytania

Wyznacz k-ty najmniejszy leksykograficznie sufiks fragmentu x.

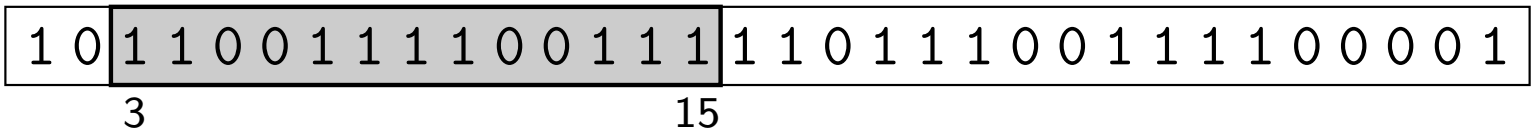
1 0 1 1 0 0 1 1 1 1 0 0 1 1 1 1 0 1 1 1 0 0 1 1 1 1 0 0 0 0 1

Selekcja sufiksów podsłów

Babenko, Gawrychowski, K, Starikovskaya (SODA 2015)

Zapytania

Wyznacz k-ty najmniejszy leksykograficznie sufiks fragmentu x.

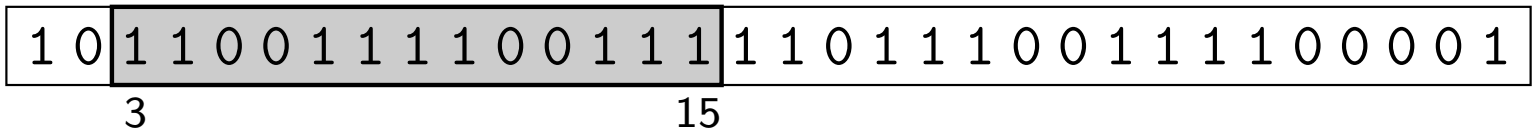


Selekcja sufiksów podsłów

Babenko, Gawrychowski, K, Starikovskaya (SODA 2015)

Zapytania

Wyznacz k-ty najmniejszy leksykograficznie sufiks fragmentu x.



$T[11..15] = 00111$
$T[5..15] = 00111100111$
$T[12..15] = 0111$
$T[6..15] = 0111100111$

$T[15..15] = 1$
$T[10..15] = 100111$
$T[4..15] = 100111100111$
$T[14..15] = 11$
$T[9..15] = 1100111$

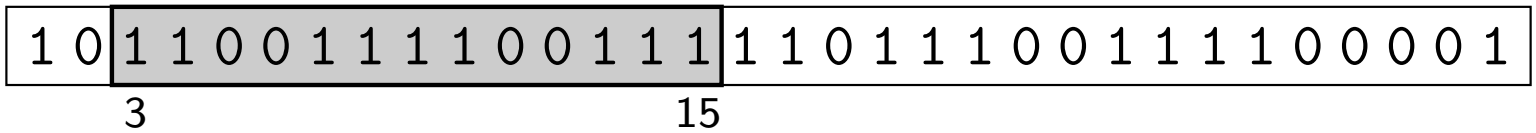
$T[3..15] = 1100111100111$
$T[13..15] = 111$
$T[8..15] = 11100111$
$T[7..15] = 111100111$

Selekcja sufiksów podsłów

Babenko, Gawrychowski, K, Starikovskaya (SODA 2015)

Zapytania

Wyznacz k-ty najmniejszy leksykograficznie sufiks fragmentu x.



$T[11..15] = 00111$
$T[5..15] = 00111100111$
$T[12..15] = 0111$
$T[6..15] = 0111100111$

$T[15..15] = 1$
$T[10..15] = 100111$
$T[4..15] = 100111100111$
$T[14..15] = 11$
$T[9..15] = 1100111$

$T[3..15] = 1100111100111$
$T[13..15] = 111$
$T[8..15] = 11100111$
$T[7..15] = 111100111$

- Czas zapytania $\mathcal{O}(\log |x|)$
- Rozmiar $\mathcal{O}(n)$
- Czas konstrukcji $\mathcal{O}(n\sqrt{\log n})$

Kompresja podsłów

Babenko, Gawrychowski, K, Starikovskaya; K, Radoszewski, Rytter, Waleń (SODA 2015)

Zapytania (Cormode i Muthukrishnan, SODA 2005)

Wyznacz skompresowaną postać podsłowa występującego jako fragment x.

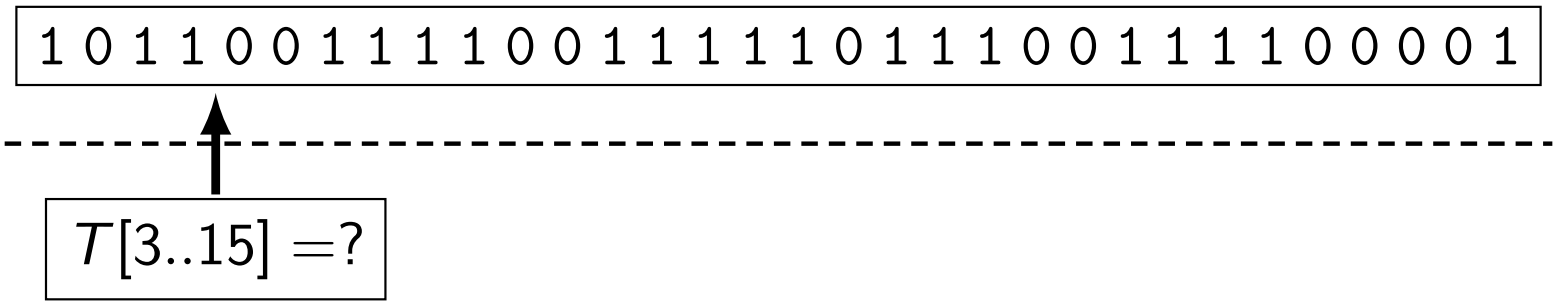
1	0	1	1	0	0	1	1	1	1	0	0	1	1	1	1	1	0	1	1	1	0	0	1	1	1	1	0	0	0	1

Kompresja podsłów

Babenko, Gawrychowski, K, Starikovskaya; K, Radoszewski, Rytter, Waleń (SODA 2015)

Zapytania (Cormode i Muthukrishnan, SODA 2005)

Wyznacz skompresowaną postać podsłowa występującego jako fragment x.

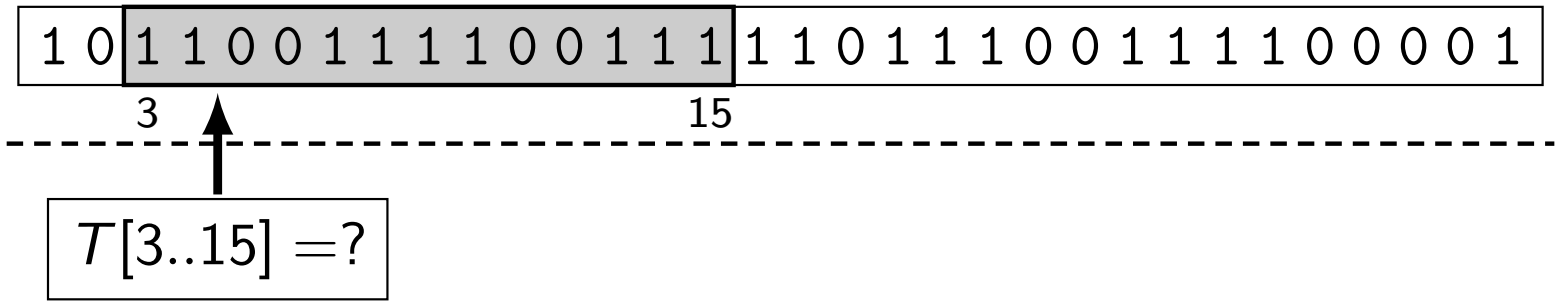


Kompresja podsłów

Babenko, Gawrychowski, K, Starikovskaya; K, Radoszewski, Rytter, Waleń (SODA 2015)

Zapytania (Cormode i Muthukrishnan, SODA 2005)

Wyznacz skompresowaną postać podsłowa występującego jako fragment x.

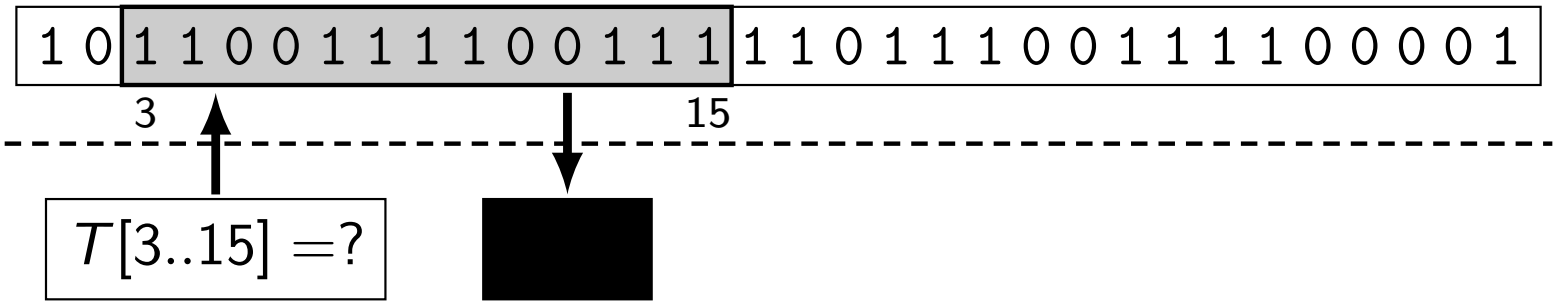


Kompresja podsłów

Babenko, Gawrychowski, K, Starikovskaya; K, Radoszewski, Rytter, Waleń (SODA 2015)

Zapytania (Cormode i Muthukrishnan, SODA 2005)

Wyznacz skompresowaną postać podsłowa występującego jako fragment x.

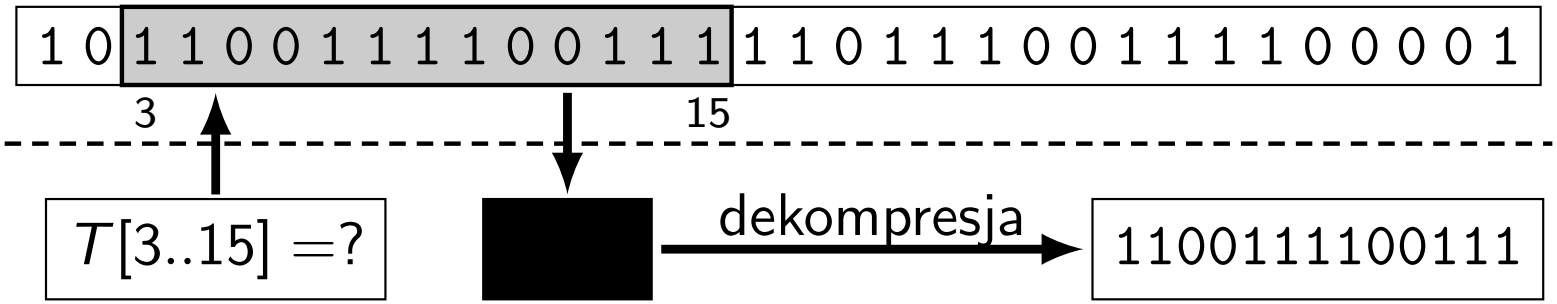


Kompresja podsłów

Babenko, Gawrychowski, K, Starikovskaya; K, Radoszewski, Rytter, Waleń (SODA 2015)

Zapytania (Cormode i Muthukrishnan, SODA 2005)

Wyznacz skompresowaną postać podsłowa występującego jako fragment x.

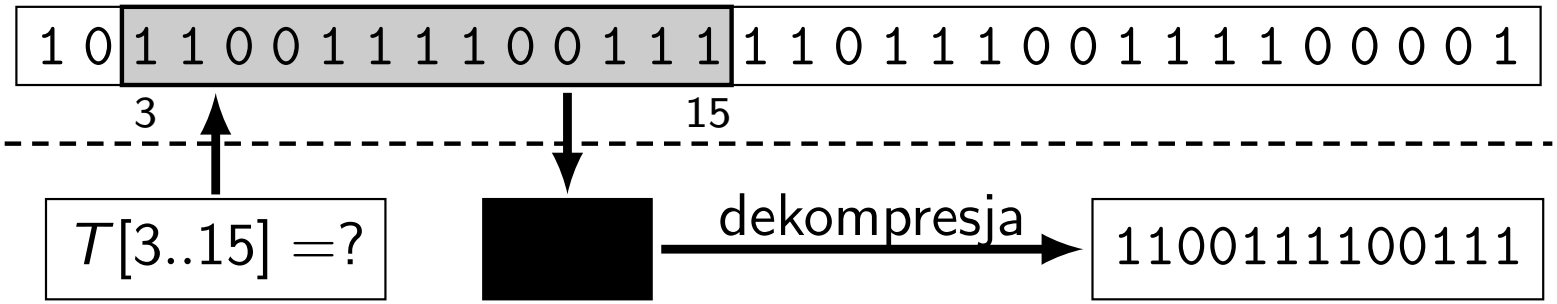


Kompresja podsłów

Babenko, Gawrychowski, K, Starikovskaya; K, Radoszewski, Rytter, Waleń (SODA 2015)

Zapytania (Cormode i Muthukrishnan, SODA 2005)

Wyznacz skompresowaną postać podsłowa występującego jako fragment x.



Zastosowania technik stworzonych do prostszych zapytań wewnętrznych

- Wewnętrzne wyszukiwanie wzorca:
  - kompresja przy użyciu algorytmu Lempela-Ziva LZ77 (np. zip, gzip).
- Selekcja sufiksów:
  - kompresja przy użyciu transformaty Burrowsa-Wheelera (np. bzip2).

Najdłuższy wspólny prefiks podsłów

Klasyczna struktura danych:

- Czas zapytania $\mathcal{O}(1)$
- Rozmiar $\mathcal{O}(n)$
- Czas konstrukcji $\mathcal{O}(n)$

Czy można lepiej?

Najdłuższy wspólny prefiks podsłów

Klasyczna struktura danych:

- Czas zapytania $\mathcal{O}(1)$
- Rozmiar $\mathcal{O}(n)$
- Czas konstrukcji $\mathcal{O}(n)$

Czy można lepiej?

- NIE! (dla dużych alfabetów)

Najdłuższy wspólny prefiks podsłów

Klasyczna struktura danych:

- Czas zapytania $\mathcal{O}(1)$
- Rozmiar $\mathcal{O}(n)$
- Czas konstrukcji $\mathcal{O}(n)$

Czy można lepiej?

- NIE! (dla dużych alfabetów)
- TAK! (dla małych alfabetów)

Najdłuższy wspólny prefiks podsłów

Klasyczna struktura danych:

- Czas zapytania $\mathcal{O}(1)$
- Rozmiar $\mathcal{O}(n)$
- Czas konstrukcji $\mathcal{O}(n)$

Czy można lepiej?

- NIE! (dla dużych alfabetów)
- TAK! (dla małych alfabetów)

Standardowy model obliczeń (word RAM):

Najdłuższy wspólny prefiks podsłów

Klasyczna struktura danych:

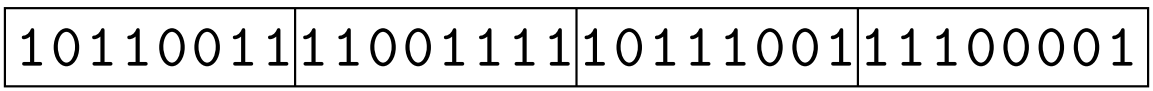
- Czas zapytania $\mathcal{O}(1)$
- Rozmiar $\mathcal{O}(n)$
- Czas konstrukcji $\mathcal{O}(n)$

Czy można lepiej?

- NIE! (dla dużych alfabetów)
- TAK! (dla małych alfabetów)

Standardowy model obliczeń (word RAM):

- Słowa maszynowe po $w = \Omega(\log n)$ bitów (w praktyce $w = 64$)
- Operacje bitowe i arytmetyczne w czasie stałym



$w = 8$

Najdłuższy wspólny prefiks podsłów

Klasyczna struktura danych:

- Czas zapytania $\mathcal{O}(1)$
- Rozmiar $\mathcal{O}(n)$
- Czas konstrukcji $\mathcal{O}(n)$

Nowy wynik (alfabet rozmiaru σ):

- Czas zapytania $\mathcal{O}(1)$
- Rozmiar $\mathcal{O}(n/\log_\sigma n)$
- Czas konstrukcji $\mathcal{O}(n/\log_\sigma n)$

Czy można lepiej?

- NIE! (dla dużych alfabetów)
- TAK! (dla małych alfabetów)

Standardowy model obliczeń (word RAM):

- Słowa maszynowe po $w = \Omega(\log n)$ bitów (w praktyce $w = 64$)
- Operacje bitowe i arytmetyczne w czasie stałym

10110011	11001111	10111001	11100001

$w = 8$

Wybrane kierunki moich badań

Struktury danych do zapytań wewnętrznych

Utrzymywanie dynamicznych kolekcji słów

Przetwarzanie tekstów w małej pamięci roboczej

Model dynamicznych kolekcji słów

Mehlhorn, Sundar, Uhrig (SODA 1994); Alstrup, Brodal, Rauhe (SODA 2000)

Utrzymujemy kolekcję $\mathcal{W}$ niepustych słów, wspierając operacje:

Model dynamicznych kolekcji słów

Mehlhorn, Sundar, Uhrig (SODA 1994); Alstrup, Brodal, Rauhe (SODA 2000)

Utrzymujemy kolekcję $\mathcal{W}$ niepustych słów, wspierając operacje:

- `make_string`(w): Wstaw w do $\mathcal{W}$.

$\mathcal{W}$: 01

1

$$\texttt{make\_string}(01) = 1$$

Model dynamicznych kolekcji słów

Mehlhorn, Sundar, Uhrig (SODA 1994); Alstrup, Brodal, Rauhe (SODA 2000)

Utrzymujemy kolekcję $\mathcal{W}$ niepustych słów, wspierając operacje:

- `make_string`(w): Wstaw w do $\mathcal{W}$.
- `concat`(w_1, w_2): Dla $w_1, w_2 \in \mathcal{W}$, wstaw $w_1 w_2$ do $\mathcal{W}$.

$\mathcal{W}$: 01 (1) 0101 (2)

$$\texttt{concat}(1, 1) = 2$$

Model dynamicznych kolekcji słów

Mehlhorn, Sundar, Uhrig (SODA 1994); Alstrup, Brodal, Rauhe (SODA 2000)

Utrzymujemy kolekcję $\mathcal{W}$ niepustych słów, wspierając operacje:

- `make_string`(w): Wstaw w do $\mathcal{W}$.
- `concat`(w_1, w_2): Dla $w_1, w_2 \in \mathcal{W}$, wstaw $w_1 w_2$ do $\mathcal{W}$.
- `split`(w, k): Dla $w \in \mathcal{W}$, wstaw $w[1..k]$ oraz $w[k+1..|w|]$ do $\mathcal{W}$.

$\mathcal{W}$:	01	0101	0	101
	1	2	3	4

$$\texttt{split}(2, 1) = (3, 4)$$

Model dynamicznych kolekcji słów

Mehlhorn, Sundar, Uhrig (SODA 1994); Alstrup, Brodal, Rauhe (SODA 2000)

Utrzymujemy kolekcję $\mathcal{W}$ niepustych słów, wspierając operacje:

- `make_string`(w): Wstaw w do $\mathcal{W}$.
- `concat`(w_1, w_2): Dla $w_1, w_2 \in \mathcal{W}$, wstaw $w_1 w_2$ do $\mathcal{W}$.
- `split`(w, k): Dla $w \in \mathcal{W}$, wstaw $w[1..k]$ oraz $w[k+1..|w|]$ do $\mathcal{W}$.
- `equal`(w_1, w_2): Czy słowa $w_1, w_2 \in \mathcal{W}$ są równe?

$\mathcal{W}$:	01	0101	0	101
	1	2	3	4

`equal`$(1, 3) =$ **false**

Model dynamicznych kolekcji słów

Mehlhorn, Sundar, Uhrig (SODA 1994); Alstrup, Brodal, Rauhe (SODA 2000)

Utrzymujemy kolekcję $\mathcal{W}$ niepustych słów, wspierając operacje:

- `make_string`(w): Wstaw w do $\mathcal{W}$.
- `concat`(w_1, w_2): Dla $w_1, w_2 \in \mathcal{W}$, wstaw $w_1 w_2$ do $\mathcal{W}$.
- `split`(w, k): Dla $w \in \mathcal{W}$, wstaw $w[1..k]$ oraz $w[k+1..|w|]$ do $\mathcal{W}$.
- `equal`(w_1, w_2): Czy słowa $w_1, w_2 \in \mathcal{W}$ są równe?

$\mathcal{W}$:	01	0101	0	101	10	1
	1	2	3	4	5	6

$$\texttt{split}(4, 2) = (5, 6)$$

Model dynamicznych kolekcji słów

Mehlhorn, Sundar, Uhrig (SODA 1994); Alstrup, Brodal, Rauhe (SODA 2000)

Utrzymujemy kolekcję $\mathcal{W}$ niepustych słów, wspierając operacje:

- `make_string`(w): Wstaw w do $\mathcal{W}$.
- `concat`(w_1, w_2): Dla $w_1, w_2 \in \mathcal{W}$, wstaw $w_1 w_2$ do $\mathcal{W}$.
- `split`(w, k): Dla $w \in \mathcal{W}$, wstaw $w[1..k]$ oraz $w[k+1..|w|]$ do $\mathcal{W}$.
- `equal`(w_1, w_2): Czy słowa $w_1, w_2 \in \mathcal{W}$ są równe?

$\mathcal{W}$:	01	0101	0	101	10	1	10
	1	2	3	4	5	6	7

$$\texttt{concat}(6, 3) = 7$$

Model dynamicznych kolekcji słów

Mehlhorn, Sundar, Uhrig (SODA 1994); Alstrup, Brodal, Rauhe (SODA 2000)

Utrzymujemy kolekcję $\mathcal{W}$ niepustych słów, wspierając operacje:

- `make_string`(w): Wstaw w do $\mathcal{W}$.
- `concat`(w_1, w_2): Dla $w_1, w_2 \in \mathcal{W}$, wstaw $w_1 w_2$ do $\mathcal{W}$.
- `split`(w, k): Dla $w \in \mathcal{W}$, wstaw $w[1..k]$ oraz $w[k+1..|w|]$ do $\mathcal{W}$.
- `equal`(w_1, w_2): Czy słowa $w_1, w_2 \in \mathcal{W}$ są równe?

$\mathcal{W}$:	01	0101	0	101	10	1	10
	1	2	3	4	5	6	7

$$\texttt{equal}(5, 7) = \textbf{true}$$

Model dynamicznych kolekcji słów

Mehlhorn, Sundar, Uhrig (SODA 1994); Alstrup, Brodal, Rauhe (SODA 2000)

Utrzymujemy kolekcję $\mathcal{W}$ niepustych słów, wspierając operacje:

- `make_string`(w): Wstaw w do $\mathcal{W}$.
- `concat`(w_1, w_2): Dla $w_1, w_2 \in \mathcal{W}$, wstaw $w_1 w_2$ do $\mathcal{W}$.
- `split`(w, k): Dla $w \in \mathcal{W}$, wstaw $w[1..k]$ oraz $w[k+1..|w|]$ do $\mathcal{W}$.
- `equal`(w_1, w_2): Czy słowa $w_1, w_2 \in \mathcal{W}$ są równe?

$\mathcal{W}$:	01	0101	0	101	10	1	10	1001
	1	2	3	4	5	6	7	8

$$\texttt{concat}(7, 1) = 8$$

Model dynamicznych kolekcji słów

Mehlhorn, Sundar, Uhrig (SODA 1994); Alstrup, Brodal, Rauhe (SODA 2000)

Utrzymujemy kolekcję $\mathcal{W}$ niepustych słów, wspierając operacje:

- `make_string`(w): Wstaw w do $\mathcal{W}$.
- `concat`(w_1, w_2): Dla $w_1, w_2 \in \mathcal{W}$, wstaw $w_1 w_2$ do $\mathcal{W}$.
- `split`(w, k): Dla $w \in \mathcal{W}$, wstaw $w[1..k]$ oraz $w[k+1..|w|]$ do $\mathcal{W}$.
- `equal`(w_1, w_2): Czy słowa $w_1, w_2 \in \mathcal{W}$ są równe?
- `LCP`(w_1, w_2): Jaki jest najdłuższy wspólny prefiks $w_1, w_2 \in \mathcal{W}$?

$\mathcal{W}$:	01	0101	0	101	10	1	10	1001
	1	2	3	4	5	6	7	8

$$\texttt{LCP}(4, 8) = 2$$

Model dynamicznych kolekcji słów

Mehlhorn, Sundar, Uhrig (SODA 1994); Alstrup, Brodal, Rauhe (SODA 2000)

Utrzymujemy kolekcję $\mathcal{W}$ niepustych słów, wspierając operacje:

- `make_string`(w): Wstaw w do $\mathcal{W}$.
- `concat`(w_1, w_2): Dla $w_1, w_2 \in \mathcal{W}$, wstaw $w_1 w_2$ do $\mathcal{W}$.
- `split`(w, k): Dla $w \in \mathcal{W}$, wstaw $w[1..k]$ oraz $w[k+1..|w|]$ do $\mathcal{W}$.
- `equal`(w_1, w_2): Czy słowa $w_1, w_2 \in \mathcal{W}$ są równe?
- `LCP`(w_1, w_2): Jaki jest najdłuższy wspólny prefiks $w_1, w_2 \in \mathcal{W}$?
- `compare`(w_1, w_2): Porównaj leksykograficznie w_1 oraz $w_2 \in \mathcal{W}$.

$\mathcal{W}$:	01	0101	0	101	10	1	10	1001
	1	2	3	4	5	6	7	8

`compare`$(4, 8) =$ '>'

Wyniki

Gawrychowski, Karczmarz, K, Łącki, Sankowski (SODA 2018)

	Alstrup i in.	Nowe wyniki	
`make_string`	$\mathcal{O}(\|w\| \log n)$	$\mathcal{O}(\|w\| + \log n)$	~~$o(\|w\| \log n)$~~
`concat`	$\mathcal{O}(\log n \log^* n)$	$\mathcal{O}(\log n)$	~~$o(\log n)$~~
`split`	$\mathcal{O}(\log n \log^* n)$	$\mathcal{O}(\log n)$	~~$o(\log n)$~~
`equal`	$\mathcal{O}(1)$	$\mathcal{O}(1)$	~~$o(\log n)$~~
`compare`	$\mathcal{O}(1)$	$\mathcal{O}(1)$	-
`LCP`	$\mathcal{O}(\log n)$	$\mathcal{O}(1)$	-
Randomizacja	Las Vegas	Las Vegas	Monte Carlo
Czas	pesymistyczny	pesymistyczny	amortyzowany

Oznaczenia:

n — łączna długość słów w kolekcji $\mathcal{W}$

$\log^* n$ — logarytm iterowany: $\min\{k : \underbrace{\log \log \cdots \log}_{k \text{ razy}} n < 1\}$.

Wybrane kierunki moich badań

Struktury danych do zapytań wewnętrznych

Utrzymywanie dynamicznych kolekcji słów

Przetwarzanie tekstów w małej pamięci roboczej

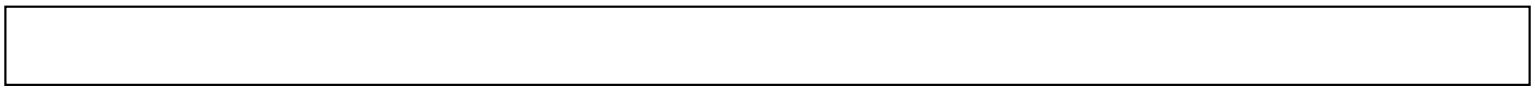
Dostęp tylko do odczytu

Model:

Dostęp tylko do odczytu

Model:

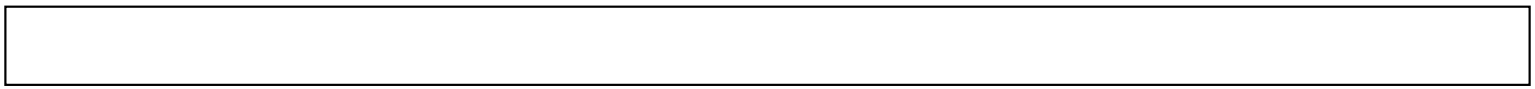
- Dane wejściowe dostępne przez wyrocznię.
- Złożoność pamięciowa obejmuje tylko pamięć roboczą.



Dostęp tylko do odczytu

Model:

- Dane wejściowe dostępne przez wyrocznię.
- Złożoność pamięciowa obejmuje tylko pamięć roboczą.

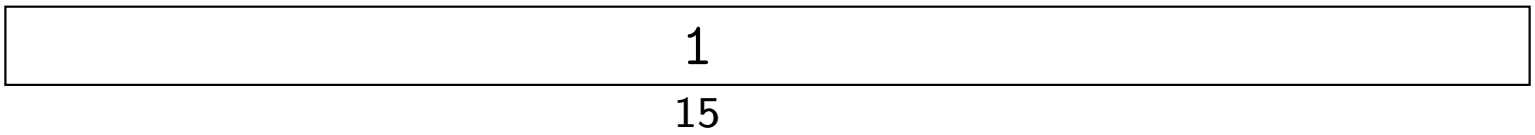


15

Dostęp tylko do odczytu

Model:

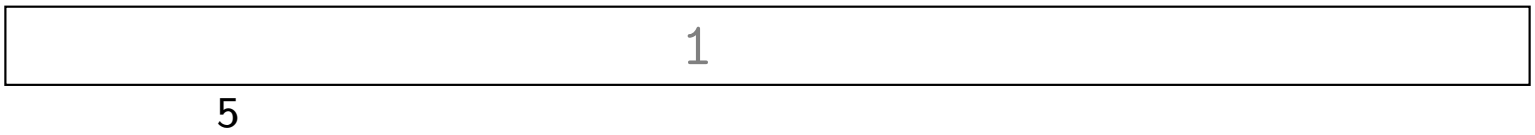
- Dane wejściowe dostępne przez wyrocznię.
- Złożoność pamięciowa obejmuje tylko pamięć roboczą.



Dostęp tylko do odczytu

Model:

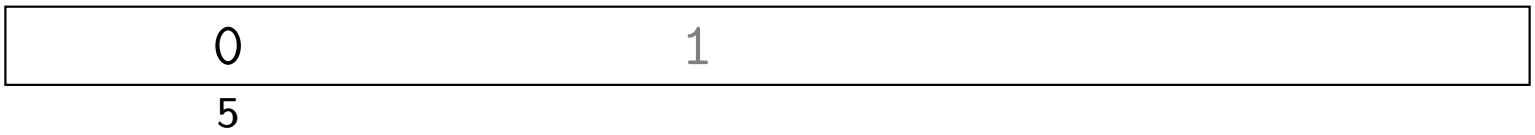
- Dane wejściowe dostępne przez wyrocznię.
- Złożoność pamięciowa obejmuje tylko pamięć roboczą.



Dostęp tylko do odczytu

Model:

- Dane wejściowe dostępne przez wyrocznię.
- Złożoność pamięciowa obejmuje tylko pamięć roboczą.



Dostęp tylko do odczytu

Model:

- Dane wejściowe dostępne przez wyrocznię.
- Złożoność pamięciowa obejmuje tylko pamięć roboczą.

Dostęp tylko do odczytu

Model:

- Dane wejściowe dostępne przez wyrocznię.
- Złożoność pamięciowa obejmuje tylko pamięć roboczą.

Wyszukiwanie wzorca (Galil i Seiferas, STOC 1981):

- Czas $\mathcal{O}(n)$
- Pamięć robocza $\mathcal{O}(1)$

Sortowanie wskazanych sufiksów

Gawrychowski, K (SODA 2017)

Rzadka tablica sufiksowa

Posortuj leksykograficznie sufiksy $T[i_j..n]$ dla danych pozycji $i_1, \ldots, i_b$.

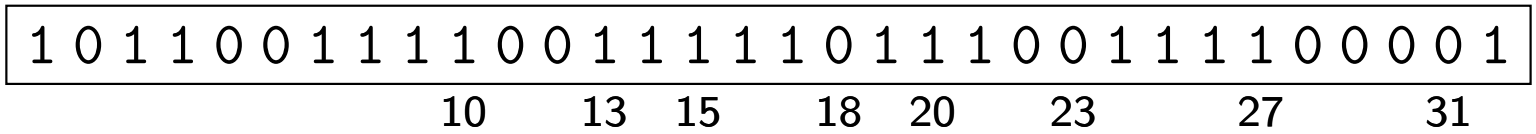
1	0	1	1	0	0	1	1	1	1	0	0	1	1	1	1	1	0	1	1	1	0	0	1	1	1	1	0	0	0	1

Sortowanie wskazanych sufiksów

Gawrychowski, K (SODA 2017)

Rzadka tablica sufiksowa

Posortuj leksykograficznie sufiksy $T[i_j..n]$ dla danych pozycji $i_1, \ldots, i_b$.

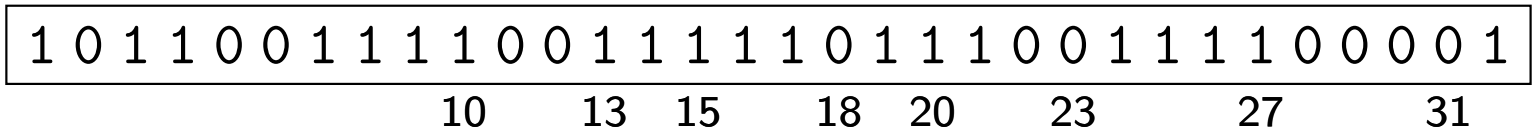


Sortowanie wskazanych sufiksów

Gawrychowski, K (SODA 2017)

Rzadka tablica sufiksowa

Posortuj leksykograficznie sufiksy $T[i_j..n]$ dla danych pozycji $i_1, \ldots, i_b$.



31	01
18	011100111100001
23	0111100001
27	100001
10	10011111011100111100001
20	1100111100001
15	1110111100111100001
13	11111011100111100001

Sortowanie wskazanych sufiksów

Gawrychowski, K (SODA 2017)

	Czas	Pamięć	Randomizacja
Bille i in. (ICALP 2013)	$\mathcal{O}(n \log^2 b)$	$\mathcal{O}(b)$	Monte Carlo
Bille i in. (ICALP 2013)	$\mathcal{O}((n + b^2) \log^2 b)$	$\mathcal{O}(b)$	Las Vegas
I i in. (STACS 2014)	$\mathcal{O}(n)$	$\mathcal{O}(b \log b)$	Monte Carlo
I i in. (STACS 2014)	$\mathcal{O}(n \log b)$	$\mathcal{O}(b)$	Las Vegas
Nowy wynik	$\mathcal{O}(n)$	$\mathcal{O}(b)$	Monte Carlo
Nowy wynik	$\mathcal{O}(n\sqrt{\log b})$	$\mathcal{O}(b)$	Las Vegas

Algorytmy strumieniowe

Model:

Algorytmy strumieniowe

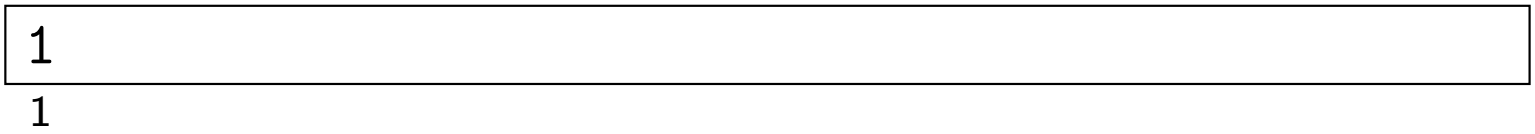
Model:

- Jednokrotny, sekwencyjny odczyt danych wejściowych.
- Złożoność pamięciowa obejmuje tylko pamięć roboczą.

Algorytmy strumieniowe

Model:

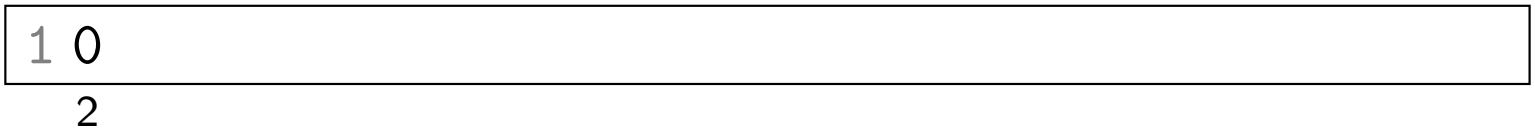
- Jednokrotny, sekwencyjny odczyt danych wejściowych.
- Złożoność pamięciowa obejmuje tylko pamięć roboczą.



Algorytmy strumieniowe

Model:

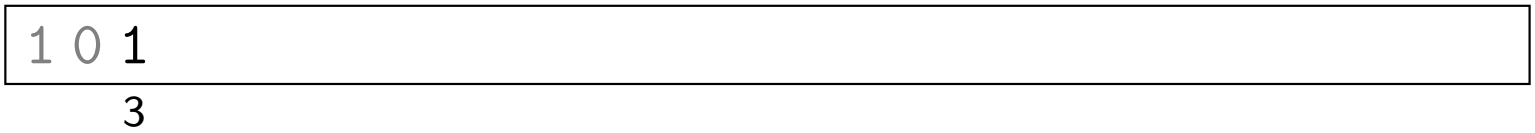
- Jednokrotny, sekwencyjny odczyt danych wejściowych.
- Złożoność pamięciowa obejmuje tylko pamięć roboczą.



Algorytmy strumieniowe

Model:

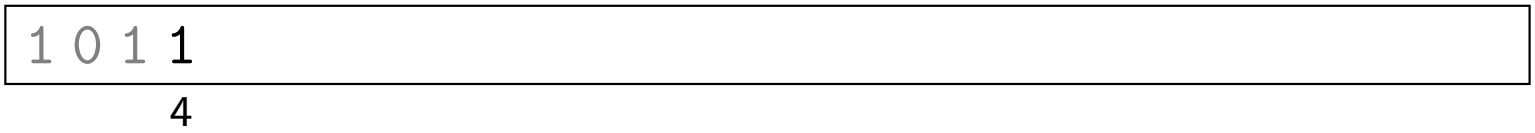
- Jednokrotny, sekwencyjny odczyt danych wejściowych.
- Złożoność pamięciowa obejmuje tylko pamięć roboczą.



Algorytmy strumieniowe

Model:

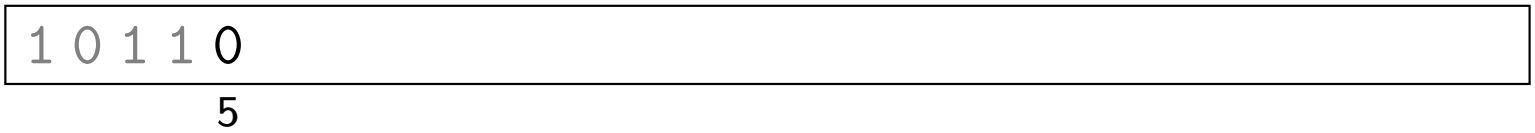
- Jednokrotny, sekwencyjny odczyt danych wejściowych.
- Złożoność pamięciowa obejmuje tylko pamięć roboczą.



Algorytmy strumieniowe

Model:

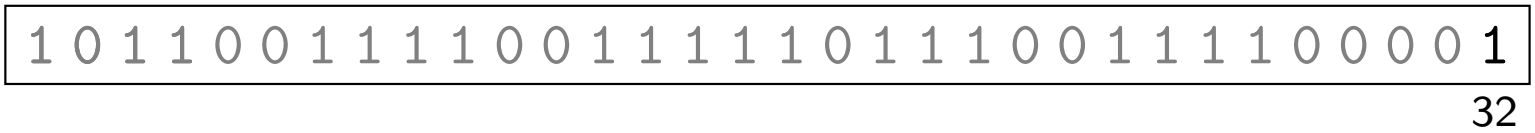
- Jednokrotny, sekwencyjny odczyt danych wejściowych.
- Złożoność pamięciowa obejmuje tylko pamięć roboczą.



Algorytmy strumieniowe

Model:

- Jednokrotny, sekwencyjny odczyt danych wejściowych.
- Złożoność pamięciowa obejmuje tylko pamięć roboczą.



Algorytmy strumieniowe

Model:

- Jednokrotny, sekwencyjny odczyt danych wejściowych.
- Złożoność pamięciowa obejmuje tylko pamięć roboczą.

Algorytmy strumieniowe

Model:

- Jednokrotny, sekwencyjny odczyt danych wejściowych.
- Złożoność pamięciowa obejmuje tylko pamięć roboczą.
- Przetwarzanie danych w czasie rzeczywistym:
  - Odpowiedź dla znanej części przed wczytaniem kolejnego symbolu.

Algorytmy strumieniowe

Model:

- Jednokrotny, sekwencyjny odczyt danych wejściowych.
- Złożoność pamięciowa obejmuje tylko pamięć roboczą.
- Przetwarzanie danych w czasie rzeczywistym:
  - Odpowiedź dla znanej części przed wczytaniem kolejnego symbolu.

Wyszukiwanie wzorca:

- W strumieniu wzorzec poprzedza tekst.
- Wystąpienie $T[i..j]$ wzorca P raportowane przed wczytaniem $T[j+1]$.

Algorytmy strumieniowe

Model:

- Jednokrotny, sekwencyjny odczyt danych wejściowych.
- Złożoność pamięciowa obejmuje tylko pamięć roboczą.
- Przetwarzanie danych w czasie rzeczywistym:
  - Odpowiedź dla znanej części przed wczytaniem kolejnego symbolu.

Wyszukiwanie wzorca:

- W strumieniu wzorzec poprzedza tekst.
- Wystąpienie $T[i..j]$ wzorca P raportowane przed wczytaniem $T[j+1]$.

Porat i Porat, FOCS 2009; Breslauer i Galil, CPM 2011:

- Czas na symbol $\mathcal{O}(1)$
- Pamięć robocza $\mathcal{O}(\log m)$
- Randomizacja typu Monte Carlo

Wyszukiwanie wzorca z niezgodnościami

Clifford, K, Porat (SODA 2019)

Problem

Znajdź fragmenty tekstu T różniące się od wzorca P na $\leqslant k$ pozycjach.

Wyszukiwanie wzorca z niezgodnościami

Clifford, K, Porat (SODA 2019)

Problem

Znajdź fragmenty tekstu T różniące się od wzorca P na $\leqslant k$ pozycjach.

$k = 2$

P: 0100111

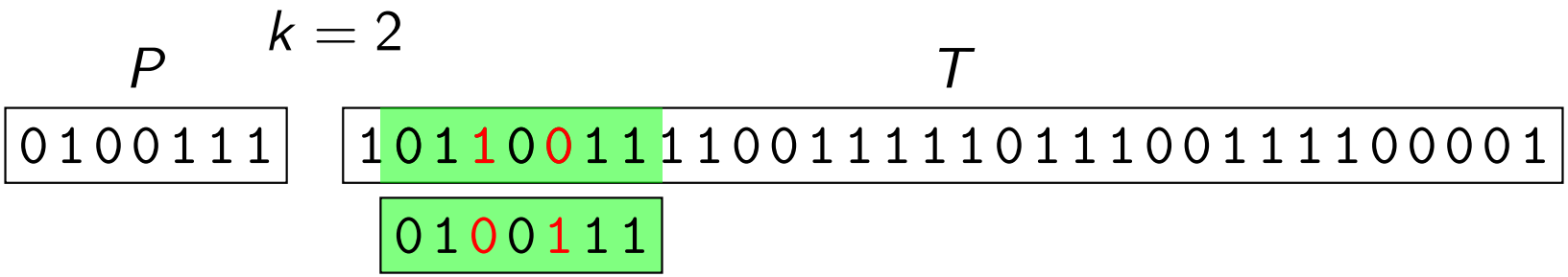
T: 10110011110011111011100111100001

Wyszukiwanie wzorca z niezgodnościami

Clifford, K, Porat (SODA 2019)

Problem

Znajdź fragmenty tekstu T różniące się od wzorca P na $\leqslant k$ pozycjach.

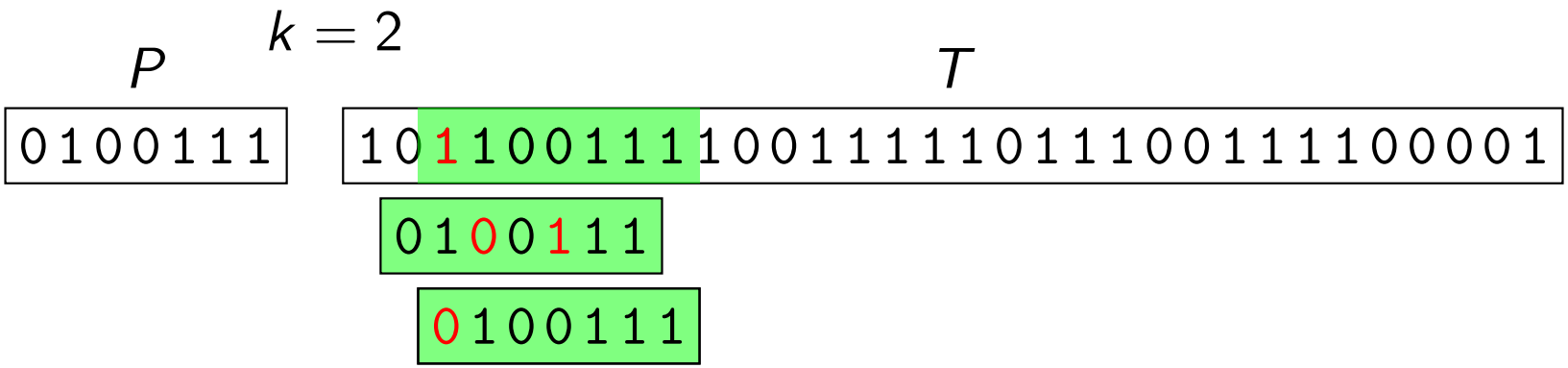


Wyszukiwanie wzorca z niezgodnościami

Clifford, K, Porat (SODA 2019)

Problem

Znajdź fragmenty tekstu T różniące się od wzorca P na $\leqslant k$ pozycjach.

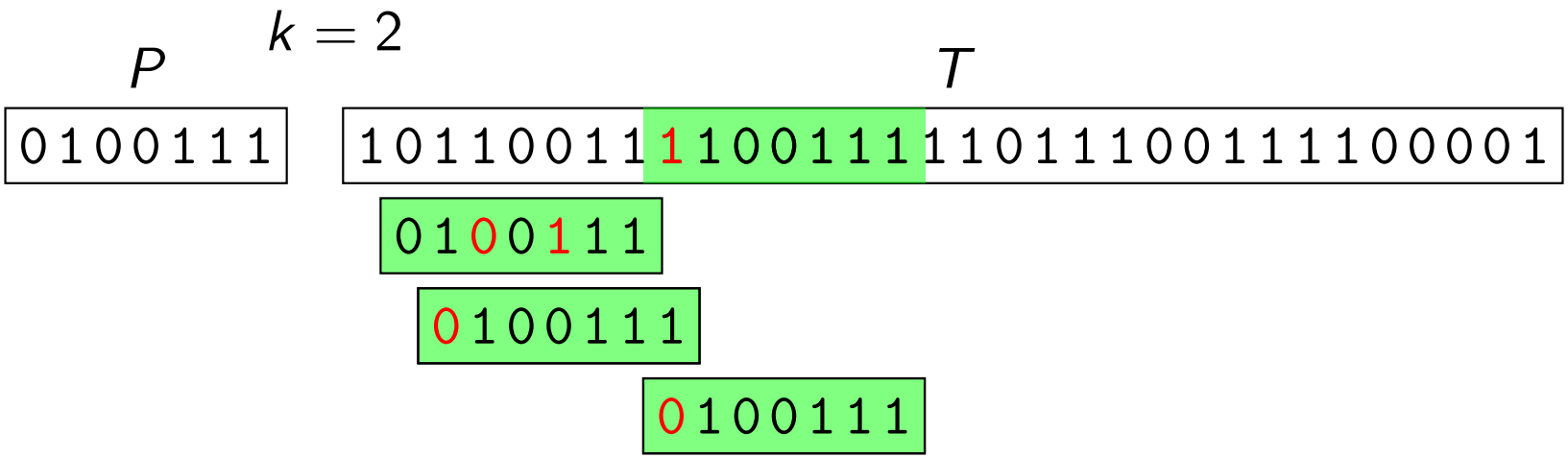


Wyszukiwanie wzorca z niezgodnościami

Clifford, K, Porat (SODA 2019)

Problem

Znajdź fragmenty tekstu T różniące się od wzorca P na $\leqslant k$ pozycjach.

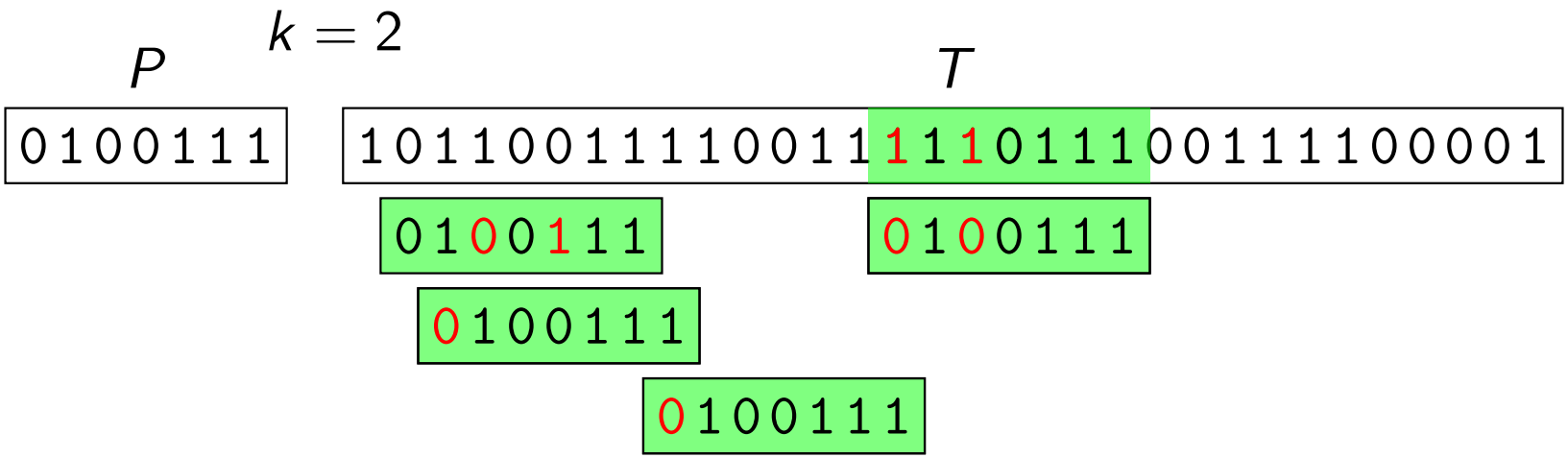


Wyszukiwanie wzorca z niezgodnościami

Clifford, K, Porat (SODA 2019)

Problem

Znajdź fragmenty tekstu T różniące się od wzorca P na $\leqslant k$ pozycjach.

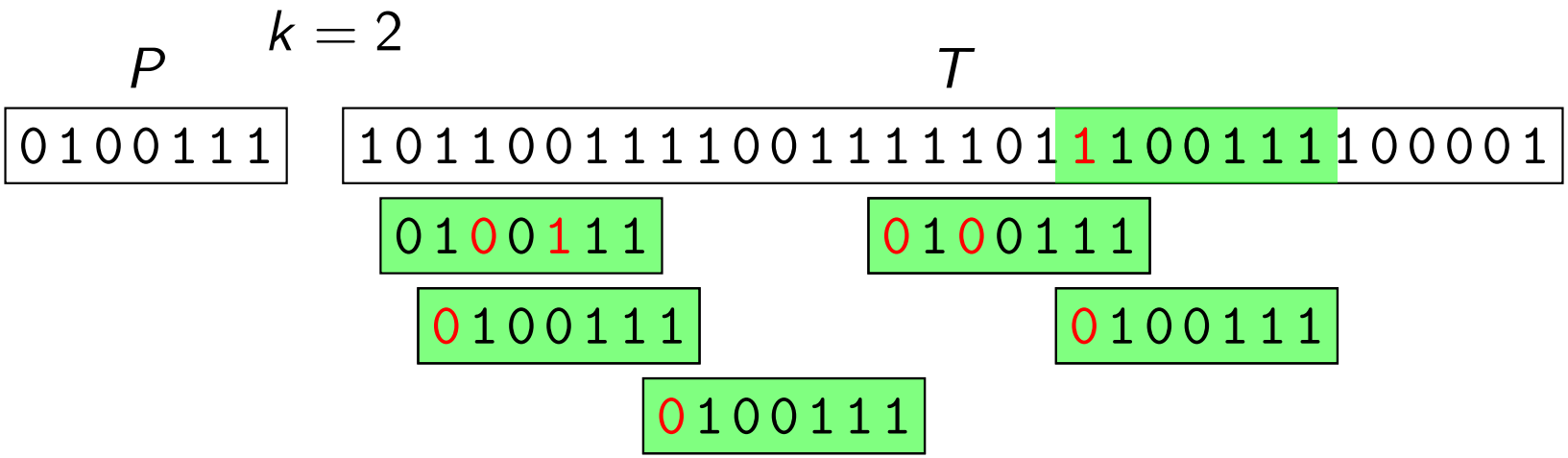


Wyszukiwanie wzorca z niezgodnościami

Clifford, K, Porat (SODA 2019)

Problem

Znajdź fragmenty tekstu T różniące się od wzorca P na $\leqslant k$ pozycjach.

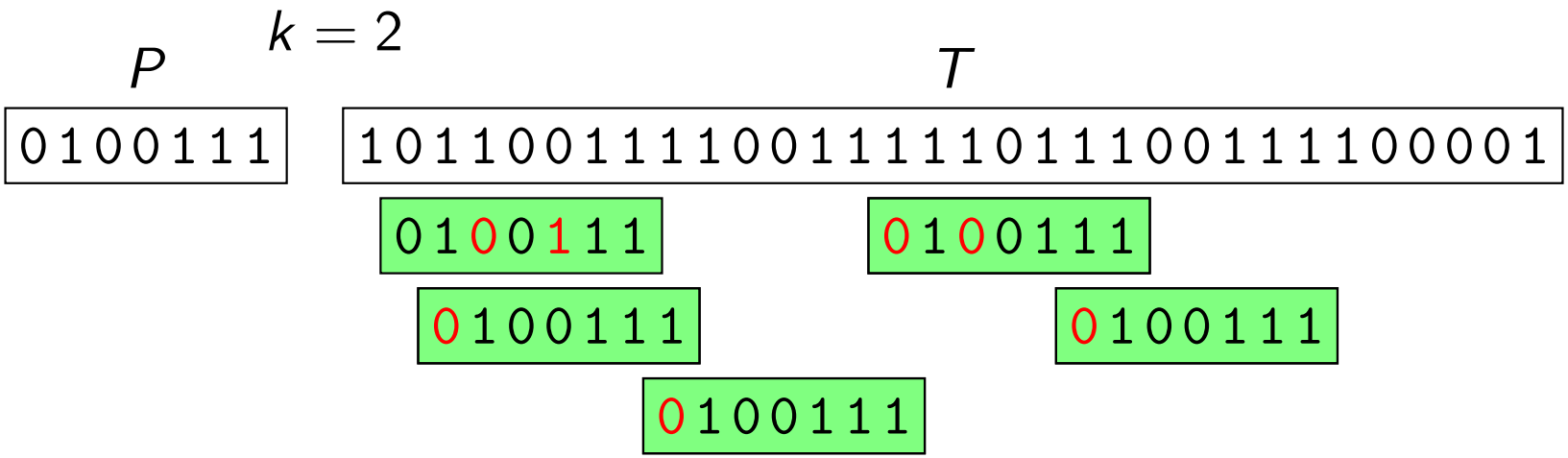


Wyszukiwanie wzorca z niezgodnościami

Clifford, K, Porat (SODA 2019)

Problem

Znajdź fragmenty tekstu T różniące się od wzorca P na $\leqslant k$ pozycjach.



Wyszukiwanie wzorca z niezgodnościami

Clifford, K, Porat (SODA 2019)

	Czas na symbol	Pamięć
Porat i Porat (FOCS 2009)	$\tilde{\mathcal{O}}(k^2)$	$\tilde{\mathcal{O}}(k^3)$
Clifford i in. (SODA 2016)	$\tilde{\mathcal{O}}(\sqrt{k})$	$\tilde{\mathcal{O}}(k^2)$
Golan i in. (ICALP 2018)	$\tilde{\mathcal{O}}(k)$	$\tilde{\mathcal{O}}(k)$
Nowy wynik	$\tilde{\mathcal{O}}(\sqrt{k})$	$\tilde{\mathcal{O}}(k)$

Wyszukiwanie wzorca z niezgodnościami

Clifford, K, Porat (SODA 2019)

	Czas na symbol	Pamięć
Porat i Porat (FOCS 2009)	$\tilde{\mathcal{O}}(k^2)$	$\tilde{\mathcal{O}}(k^3)$
Clifford i in. (SODA 2016)	$\tilde{\mathcal{O}}(\sqrt{k})$	$\tilde{\mathcal{O}}(k^2)$
Golan i in. (ICALP 2018)	$\tilde{\mathcal{O}}(k)$	$\tilde{\mathcal{O}}(k)$
Nowy wynik	$\tilde{\mathcal{O}}(\sqrt{k})$	$\mathcal{O}(k \log m)$

Podziękowania dla współautorów

Anna Adamaszek
Mai Alzamel
Maxim Babenko
Golnaz Badkobeh
Evangelos Bampas
Hideo Bannai
Carl Barton
Anudhyan Boral
Panagiotis Charalampopoulos
Raphaël Clifford
Maxime Crochemore
Marek Cygan
Jerzy Czyżowicz
Gabriele Fici
Johannes Fischer
Tomáš Flouri
Travis Gagie
Michał Gańczorz
Paweł Gawrychowski
Leszek Gąsieniec
Garance Gourdel
Szymon Grabowski
Fabrizio Grandoni
David Ilcinkas
Costas S. Iliopoulos
Shunsuke Inenaga
Artur Jeż
Adam Karczmarz
Ralf Klasing
Ignat Kolesnichenko
Dmitry Kosolobov
Marcin Kubica
Ritu Kundu
Jakub Łącki
Alessio Langiu
Thierry Lecroq
Arnaud Lefebvre
Chang Liu
Manal Mohamed
Jakub Pachocki
Dominik Pająk
Marcin Pilipczuk
Michał Pilipczuk
Solon P. Pissis
Ely Porat
Élise Prieur-Gaston
Simon J. Puglisi
Jakub Radoszewski
Wojciech Rytter
Piotr Sankowski
Arseny M. Shur
William F. Smyth
Tatiana Starikovskaya
Juliusz Straszyński
Shiho Sugimoto
Bartosz Szreder
Wojciech Tyczyński
Hjalte Wedel Vildhøj
Tomasz Waleń
Bartłomiej Wiśniewski
Michał Włodarczyk
Wiktor Zuba

Podziękowania dla współautorów

Anna Adamaszek
Mai Alzamel
Maxim Babenko
Golnaz Badkobeh
Evangelos Bampas
Hideo Bannai
Carl Barton
Anudhyan Boral
Panagiotis Charalampopoulos
Raphaël Clifford
Maxime Crochemore
Marek Cygan
Jerzy Czyżowicz
Gabriele Fici
Johannes Fischer
Tomáš Flouri
Travis Gagie
Michał Gańczorz
Paweł Gawrychowski
Leszek Gąsieniec
Garance Gourdel
Szymon Grabowski
Fabrizio Grandoni
David Ilcinkas
Costas S. Iliopoulos
Shunsuke Inenaga
Artur Jeż
Adam Karczmarz
Ralf Klasing
Ignat Kolesnichenko
Dmitry Kosolobov
Marcin Kubica
Ritu Kundu
Jakub Łącki
Alessio Langiu
Thierry Lecroq
Arnaud Lefebvre
Chang Liu
Manal Mohamed
Jakub Pachocki
Dominik Pająk
Marcin Pilipczuk
Michał Pilipczuk
Solon P. Pissis
Ely Porat
Élise Prieur-Gaston
Simon J. Puglisi
Jakub Radoszewski
Wojciech Rytter
Piotr Sankowski
Arseny M. Shur
William F. Smyth
Tatiana Starikovskaya
Juliusz Straszyński
Shiho Sugimoto
Bartosz Szreder
Wojciech Tyczyński
Hjalte Wedel Vildhøj
Tomasz Waleń
Bartłomiej Wiśniewski
Michał Włodarczyk
Wiktor Zuba

Pytania?

Dziękuję za uwagę!