

# Z-Box Algorithmus

Silke Trißl

Wissensmanagement in der Bioinformatik



## Z-Algorithmus

- Zerlegung des Matching Problem in **zwei Phasen**
  - **Preprocessing**: Lerne möglichst viel über die Struktur von T und / oder P
  - **Search**: Nutze das Gelernte, um
    - P weiter nach rechts verschieben zu können
    - Bei einem Vergleich von P mit Substring von T nicht an Position 1 von P starten zu müssen
- **Aber**: Auch das Preprocessing muss schnell sein
  - Forderung kann aufgehoben werden, wenn bei vielen Suchen P oder T gleich ist
    - Beispiel: Suche von Promotorsequenzen im Genom

# Z-Algorithmus: Preprocessing

## Definition

- Sei  $i > 1$ . Dann ist  $Z_i(S)$  die Länge des **längsten Substrings**  $x$  von  $S$  mit
  - $x = S[i..i+|x|-1]$  (x startet an Position  $i$  in  $S$ )
  - $S[i..i+|x|-1] = S[1..|x|-1]$  (x ist auch Präfix von  $S$ )
- $x$  heißt **Z-Box** von  $S$  an Position  $i$  mit der Länge  $Z_i(S)$



# Beispiele

**S = aabcaabxaaz**

1(a)

0

0

3(aab)

1(a)

0

0

2(aa)

1(a)

0

**S = aaaaaa**

5

4

3

2

1

**S = baaaaa**

0

0

0

0

0

# Linearer Stringmatching Algorithmus

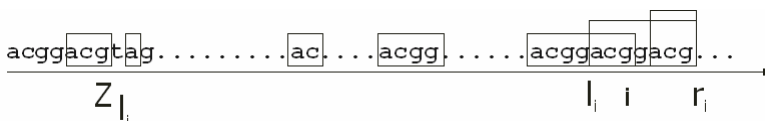
- Annahme: Z-Boxen lassen sich in  $O(|S|)$  berechnen
  - Das zeigen wir später
- Verwendung der Z-Boxen für String Matching

```
S := P|'\$'\|T; // ($ \notin \Sigma)
Berechne Z-Boxen von S;
for i = |P|+2 to |S|
  if (Zi(S)=|P|) then
    print Zi(S); // P in T at position i
  end if;
end if;
```

- Komplexität
  - Schleife wird  $m$ -Mal durchlaufen =>  $O(m)$

# Vorarbeiten

- Definition
  - Sei  $i > 1$ . Dann ist:
    - $r_i$  der **maximale Endpunkt** aller Z-Boxen, die bei oder vor  $i$  beginnen
    - $l_i$  ist die **Startposition** der Z-Box, die bei  $r_i$  endet
- Sprich:  $S[l_i..r_i]$  ist die Z-Box, die Position  $i$  von  $S$  enthält und am weitesten nach rechts reicht



# Lineare Berechnung der $Z_i$ Werte

- Trick
  - Verwenden von bereits berechneten  $Z_i$  zur Berechnung von  $Z_k$  ( $k > i$ )
- Grundaufbau
  - Lineares Durchlaufen des Strings
  - Kontinuierliches Vorhalten der aktuellen Werte  $l$  und  $r$
  - Größe der Z-Box an Position  $i$  ergibt sich mit konstantem Aufwand
- Induktive Erklärung
  - Induktionsanfang: Position  $k=2$ .
    - Berechne  $Z_2$
    - Wenn  $Z_2 > 0$ , setze  $r=r_2$  und  $l=l_2$ , sonst  $r=l=0$
  - Induktionsschritt: Position  $k > 2$ 
    - wir kennen  $r$ ,  $l$  und für alle  $Z_j$ ,  $j < k$

# Z-Box Algorithmus, Fall 1

- Möglichkeit 1:  $k > r$ 
  - D.h., dass es keine Z-Box gibt, die  $k$  enthält
  - Wir wissen nichts über den Bereich in  $S$  ab  $k$
  - Dann gehen wir primitiv vor
    - Berechne  $Z_k$  durch Zeichen-für-Zeichen matching
    - Wenn  $Z_k > 0$ :  $r=r_k$  und  $l=l_k$

Beispiel

$k$   
CTCAGATTGCAG  
0  
1  
0  
?

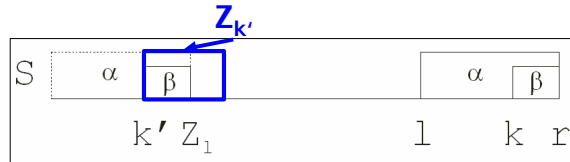
Gegenbeispiel

$l$   $k$   $r$   
CTACTACTTTGCAG  
0  
0  
5  
?

## Z-Box Algorithmus, Fall 2

- Möglichkeit 2:  $k \leq r$

- Die Situation:



- Also

- Z-Box  $Z_l$  ist Präfix von  $S$
    - Substring  $\beta = S[k..r]$  kommt auch an  $k' = k - l + 1$  von  $S$  vor
    - Was wissen wir über diesen Substring? Natürlich:  $Z_{k'}$
    - D.h., dass  $S[k.. ]$  ist Präfix von  $S$  mit mindestens Länge  $\min(Z_{k'}, |\beta|)$

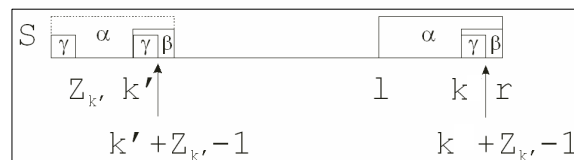
## Z-Box Algorithmus, Fall 2.1

- Fallunterscheidung

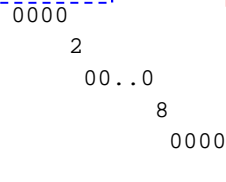
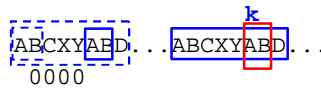
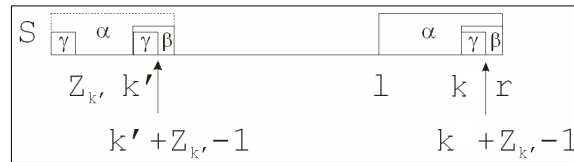
- $Z_{k'} < |\beta|$

Dann ist das Zeichen an  $k' + Z_{k'}$ , ein Mismatch bei der Präfixverlängerung. Dann ist das Zeichen  $S(k + Z_{k'})$  der gleiche Mismatch. Also

$Z_k = Z_{k'}$ ;  $r$  und  $l$  unverändert



# Beispiel



$\beta = |ABD|$ ;  $k' = 6$ ;  $Z_6 = 2 < |\beta|$

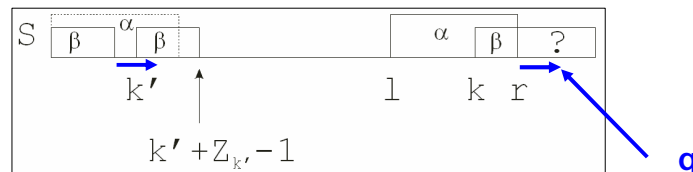
# Z-Box Algorithmus, Fall 2.2

- $Z_k \geq \beta$ : Dann ist  $\beta$  ein Präfix von  $S$ 
  - ... dass sich vielleicht sogar verlängern lässt
  - Wir wissen, dass  $S(|\beta| + 1) = S(k' + |\beta|)$
  - Wir wissen aber nichts über  $S(r + 1)$  – dieses Zeichen wurde noch nie betrachtet

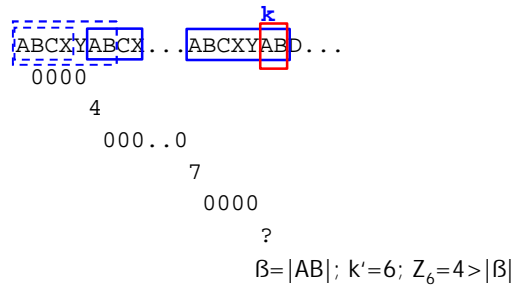
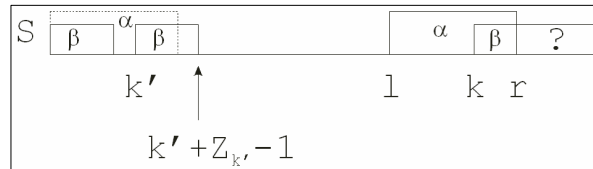
Matche Substring  $S[r + 1..]$  mit  $S[|\beta| + 1..]$

Sei der erste Mismatch an Position  $q$

Dann:  $Z_k = q - k$ ;  $r = q - 1$ ; wenn  $q \neq r + 1$ :  $l = k$



## Beispiel



## Algorithmus

```

match  $Z_2$ 
set  $l, r$ ;
for  $k = 3$  to  $|S|$ 
  if  $k > r$  then
    match  $Z_k$ ;
    set  $r, l$ ;
  else
     $k' := k - 1 + 1$ ;
     $b := r - k + 1$ ; // This is  $\beta$ 
    if  $Z_{k'} < b$  then
       $Z_k := Z_{k'}$ ;
    else
      match  $S[r+1..]$  with  $S[b+1..]$  until  $q$ ;
       $Z_k := q - k$ ;  $r := q - 1$ ;
      if  $q = r + 1$  then  $l := k$ ;
    end if;
  end if;
end for;

```

# Aufgabe

---



- Exaktes Stringmatching mit dem Z-Box Algorithmus
  - Implementiert den Z-Box Algorithmus
  - gibt die Position der exakten Matches an
  
- Aufgabe
  - Bestimmt die Zahl an Vergleichen, die nötig war