q-Gramm-Gruppen-Index

Sven Schrinner

14. Mai 2014

q-Gramme in Zahlen

q-Gramme fester Länge können als Zahlen aufgefasst werden:

- A $\rightarrow$ 00
- C $\rightarrow$ 01
- G $\rightarrow$ 10
- T $\rightarrow$ 11

q-Gramme in Zahlen

q-Gramme fester Länge können als Zahlen aufgefasst werden:

- A $\rightarrow$ 00
- C $\rightarrow$ 01
- G $\rightarrow$ 10
- T $\rightarrow$ 11

q-Gramm der Länge q benötigt $2q$ Bits an Speicher

- Jedes q-Gramm kodiert eine Zahl $g \in \{0, \ldots, 4^q - 1\}$

Naiver q-Gramm-Index

- Array A, liefert für jedes q-Gramm einen Startindex für ein Positionsarray P
- Array P, speichert Positionen der lexikographisch sortierten q-Gramme im Text

Naiver q-Gramm-Index

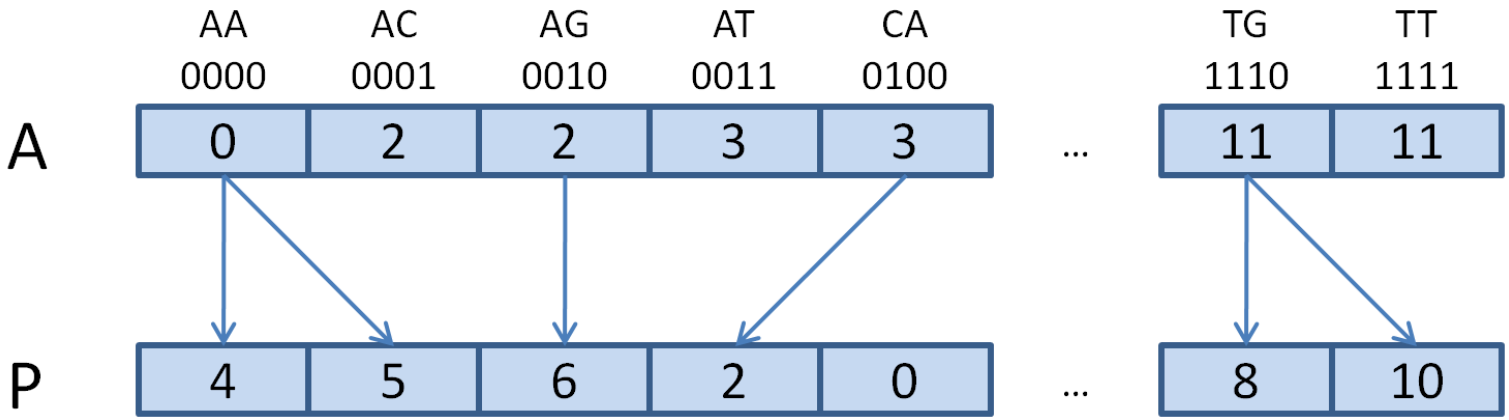
- Array A, liefert für jedes q-Gramm einen Startindex für ein Positionsarray P
- Array P, speichert Positionen der lexikographisch sortierten q-Gramme im Text

Vorkommen eines q-Gramms g:
$P[A[g]], P[A[g]+1], \ldots, P[A[g+1]-1]$

Naiver q-Gramm-Index

- Array A, liefert für jedes q-Gramm einen Startindex für ein Positionsarray P
- Array P, speichert Positionen der lexikographisch sortierten q-Gramme im Text

Vorkommen eines q-Gramms g:
$P[A[g]], P[A[g]+1], \ldots, P[A[g+1]-1]$



q-Gramm-Gruppen

Problem Naiver Ansatz benötigt $4^q + |T|$ Byte an Speicher.

Lösung Aufteilung der q-Gramme in Gruppen

q-Gramm-Gruppen

Problem Naiver Ansatz benötigt $4^q + |T|$ Byte an Speicher.

Lösung Aufteilung der q-Gramme in Gruppen

Sei w die gewählte Gruppengröße (bei Peanut: $w = 32$). Definiere dann die i-te Gruppe als:

$$G_i = \{g \in Q \mid \lfloor \frac{g}{w} \rfloor = i\}$$

wobei Q die Menge aller q-Gramme (in Zahlenform) ist.

q-Gramm-Gruppen

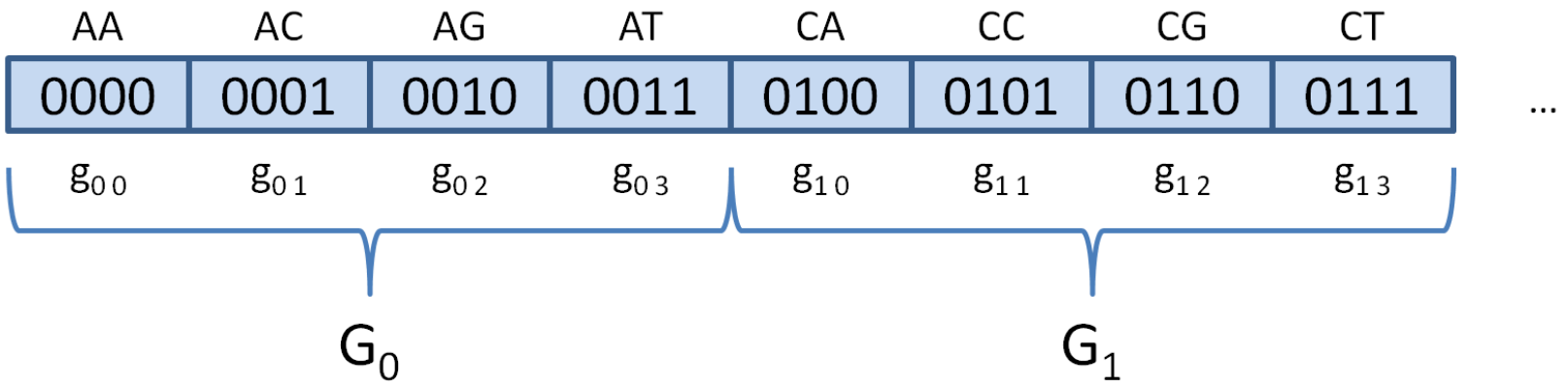
Problem Naiver Ansatz benötigt $4^q + |T|$ Byte an Speicher.

Lösung Aufteilung der q-Gramme in Gruppen

Sei w die gewählte Gruppengröße (bei Peanut: $w = 32$). Definiere dann die i-te Gruppe als:

$$G_i = \{g \in Q \mid \lfloor \frac{g}{w} \rfloor = i\}$$

wobei Q die Menge aller q-Gramme (in Zahlenform) ist.



Zusammensetzung des Gruppenindex

Index besteht aus vier Arrays:

- I besitzt pro Gruppe einen Eintrag
- S besitzt pro Gruppe einen Eintrag
- S' besitzt pro vorkommenden q-Gramm einen Eintrag
- O besitzt pro Textposition einen Eintrag

Zusammensetzung des Gruppenindex

Index besteht aus vier Arrays:

- I besitzt pro Gruppe einen Eintrag
- S besitzt pro Gruppe einen Eintrag
- S' besitzt pro vorkommenden q-Gramm einen Eintrag
- O besitzt pro Textposition einen Eintrag

$I[i]$ besitzt pro q-Gramm ein Bit, welches angibt, ob dieses q-Gramm im Text vorkommt

$\rightarrow$ Daher $w = 32$

Erlaubt schnellen Test, ob q-Gramm im Text vorkommt

Zusammensetzung des Gruppenindex

Lookup für g_{ij}:

1. Falls $I[i]_j = 1$, schaue in $S[i]$

Zusammensetzung des Gruppenindex

Lookup für g_{ij}:

1. Falls $I[i]_j = 1$, schaue in $S[i]$
2. Sei $j' = \sum_{a=0}^{j-1} I[i]_a$

Zusammensetzung des Gruppenindex

Lookup für g_{ij}:

1. Falls $I[i]_j = 1$, schaue in $S[i]$
2. Sei $j' = \sum_{a=0}^{j-1} I[i]_a$
3. Schauen in $S'[S[i] + j']$

Zusammensetzung des Gruppenindex

Lookup für g_{ij}:

1. Falls $I[i]_j = 1$, schaue in $S[i]$
2. Sei $j' = \sum_{a=0}^{j-1} I[i]_a$
3. Schauen in $S'[S[i] + j']$
4. Ergebnis ist Startindex für Array O

Zusammensetzung des Gruppenindex

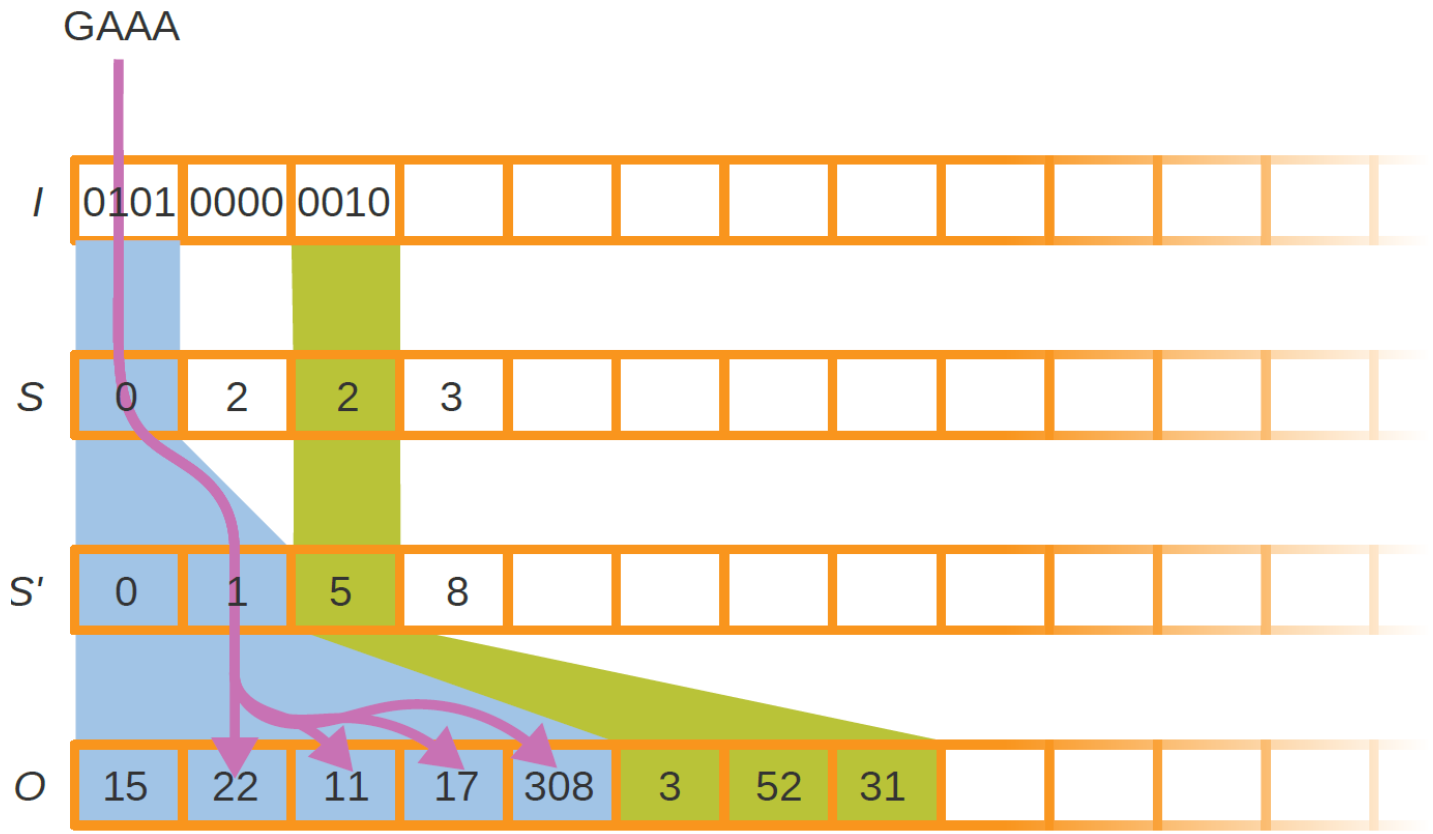
Lookup für g_{ij}:

1. Falls $I[i]_j = 1$, schaue in $S[i]$
2. Sei $j' = \sum_{a=0}^{j-1} I[i]_a$
3. Schauen in $S'[S[i] + j']$
4. Ergebnis ist Startindex für Array O

Vorkommen von g_{ij} an:

$$O[S'[S[i] + j']], O[S'[S[i] + j'] + 1], \ldots, O[S'[S[i] + j' + 1] - 1]$$

Kompletter Arrayzugriff



Analyse der q-Gramm-Gruppen

Für kleine q:

- Kein Vorteil, da O und P-Array den Speicherbedarf dominieren

Analyse der q-Gramm-Gruppen

Für kleine q:

- Kein Vorteil, da O und P-Array den Speicherbedarf dominieren

Für große q:

- I und S für $w = 32$ nur $\frac{1}{16}$ so groß wie A-Array
- S' verbraucht nur Speicher für tatsächlich vorkommende q-Gramme
- Je größer q, desto wahrscheinlicher ist Ersparnis bei S'

Analyse der q-Gramm-Gruppen

Für kleine q:

- Kein Vorteil, da O und P-Array den Speicherbedarf dominieren

Für große q:

- I und S für $w = 32$ nur $\frac{1}{16}$ so groß wie A-Array
- S' verbraucht nur Speicher für tatsächlich vorkommende q-Gramme
- Je größer q, desto wahrscheinlicher ist Ersparnis bei S'

$\rightarrow$ Speicherverbrauch für Index konnte bei Peanut um etwa 90% reduziert werden [Köster and Rahmann, 2014].

Literatur

Köster, J. and Rahmann, S. (2014).
Massively parallel read mapping on gpus with peanut.
CoRR, abs/1403.1706.