Algorithmen auf Sequenzen

Vorlesung von Prof. Dr. Sven Rahmann
im Sommersemester 2008

Kapitel 2
Mustersuche mit verallgemeinerten Strings

Webseite zur Vorlesung
http://ls11-www.cs.tu-dortmund.de/people/rahmann/teaching/ss2008/AlgorithmenAufSequenzen

Sprechstunde
Mo 16-17 in OH14, R214

Übersicht

Verallgemeinerung von einfachen Strings durch

- Zeichenklassen (character classes)
- Lücken variabler beschränkter Länge (bounded gaps)
- Optionale Zeichen (optional characters)
- Wiederholbare Zeichen (repeatable characters)

In diesen Fällen arbeitet man am effizientesten mit bit-parallelen Algorithmen, nämlich Verallgemeinerungen von

- shift-and
- BNDM (backward non-deterministic DAWG matching)

1a. Zeichenklassen im Muster

Jede Stelle des Musters wird (statt durch ein einzelnes Zeichen aus *A*)
durch eine nichtleere Menge von Zeichen aus *A* beschrieben.
Das Muster ist damit ein String über 2^A (Potenzmenge von *A*).

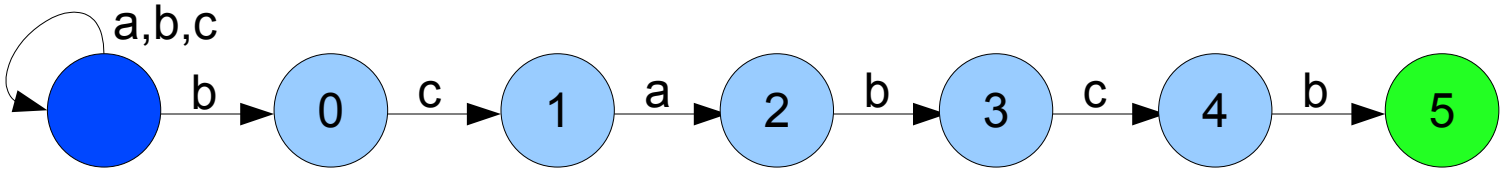
Beispiel

P = {a}{b}{a,c}{b}{a,b,c}{b,c},
vereinfacht geschrieben als
P = ab[ac]b*A*[bc],
passt z.B. auf ababab, ababac, ... (insgesamt auf 12 Strings).

Notation

Einelementige Mengen schreibt man wie bisher als Buchstaben.
Mehrelementige Mengen schreibt man in eckigen Klammern (ohne Komma)
Mengen, die das ganze Alphabet *A* bezeichnen, als *A*.

Shift-And Algorithmus



Erinnerung: Shift-And Algorithmus

- Aktualisiere eine bit-codierte aktive Zustandsmenge *z* im obigen Automaten
- Bit *q* von *z* ist 1 genau dann wenn Zustand *q* aktiv ist
- Initialisierung: *z*=0 (keiner der numerierten Zustände ist aktiv)
- Update beim Lesen von T_i: $z := ((z << 1) \mid 1) \,\&\, \text{mask}[T_i]$
- Match gefunden, wenn $(z \,\&\, 2^{m-1}) \mathrel{!}= 0$
- Definition von mask[*a*]: Bit *q* (*q*=0..*m*-1) ist genau dann gesetzt, wenn $P_{q+1}=a$.

Beobachtungen:

Das Pattern P ist nur in den |A| Bitmasken mask[a] codiert.
Jedes Bit q ist in genau einer der |A| Masken gesetzt, in den anderen nicht.

Shift-And mit Zeichenklassen

- Neue Definition von mask[*a*]: Bit *q* (*q*=0..*m*-1) ist genau dann gesetzt, wenn $a \in P_{q+1}$.

(Beachte: P_{q+1} ist jetzt eine Menge.)
Man muss nur bei der Initialisierung der Masken etwas ändern, nichts am Algorithmus!
Im übrigen kann man wieder den Shift-Or-Trick verwenden (Komplemente der Bits)!

Erweiterung von BNDM

Erinnerung

An Bit-parallelen Algorithmen, die mit Masken arbeiten,
kennen wir Shift-And (-Or) und BNDM.

- Shift-And (-Or) ist ein Präfix-basierter Algorithmus und muss jedes Textzeichen lesen.
- BNDM ist Teilstring-basiert und kann Textzeichen überspringen.

Man kann die Masken von BNDM in derselben Weise wie bei Shift-And modifizieren.
Man erhält eine Variante von BNDM, die mit Zeichenklassen arbeitet.

Laufzeitvergleich

- Bei einfachen Strings ist BNDM meist besser als Shift-Or (außer bei kurzen Mustern).
- Die Laufzeit von Shift-And ist unabhängig davon, ob Zeichenklassen vorkommen.
- BNDM wird beim Auftreten von Zeichenklassen langsamer,
  da es mehr Strings gibt, die als Faktoren von P vorkommen.

1b. Zeichenklassen im Text

Wir betrachten jetzt den Fall, dass sowohl *P* als auch *T* Strings über 2^A sind.

Möglichkeit 1

Wenn wenige verschiedene Teilmengen von *A* als Zeichen im Text vorkommen, führt man dafür neue Symbole ein.
Beispiel: Aminosäurealphabet bei Proteinen:

A	Alanine
C	Cysteine
D	Aspartic acid
E	Glutamic acid
F	Phenylalanine
G	Glycine
H	Histidine
I	Isoleucine
K	Lysine
L	Leucine
M	Methionine
N	Asparagine
P	Proline
Q	Glutamine
R	Arginine
S	Serine
T	Threonine
V	Valine
W	Tryptophan
Y	Tyrosine

Zusätzlich
B = {D,N},
Z = {E,Q},
X = {A,C,D,E,F,G,H,I,K,L,M,N,P,Q,R,S,T,V,Q,X,Y}

Berechnung der Masken

Zeichenklasse $K \in 2^A$ im Muster passt zu Textsymbol $S \in 2^A$,
genau dann wenn $K \cap S \neq \{\}$.

Die Masken für die neuen Textsymbole *S* werden also durch
ver-oder-ung der Masken der in *S* enthaltenen Symbole berechnet,
nachdem deren Masken berechnet wurden.

Beispiel:
A = {D,E,N,Q} mit zusätzlichen Symbolen B = {D,N}, Z = {E,Q}.
Muster *P* = QB[DNQ][DE] führt zunächst auf Bit-Masken
mask[D] = 1110,
mask[E] = 1000,
mask[N] = 0110,
mask[Q] = 0101.

Dann werden noch
mask[B] = 1110,
mask[Z] = 1101
durch ver-oder-ung berechnet.

Übung: Wie arbeitet Shift-And auf dem Text ZDQZBQE?

Berechnung der Masken

Möglichkeit 2

Bei kleinen Alphabeten A,
wenn fast alle Teilmengen von A als Zeichen im Text auftreten (können),
kann die Potenzmenge von A wiederum Bit-parallel codieren.

Beispiel DNA:

Wir setzen A=1, C=2, G=4, T=8 und codieren alle Teilmengen von {A,C,G,T}
als Zahlen zwischen 0 und 15. {A,C,T} wird z.B. durch 11 = $(1011)_2$ repräsentiert.
Sowohl T als auch P werden jetzt als String über {0,1,...,15} angegeben.

Gegeben die „alten“ Masken mask[a] für alle $a \in A = \{0, ..., |A|-1\}$,
kann man mask+[S] für alle $S \in 2^A$ wie effizient folgt berechnen.

```
mask+[0] := 0
for c := 0 .. |A|-1:
    for j := 0 .. 2^c -1:
        mask+[2^c + j] := mask+[j] | mask[c]
```

Danach ist mask+[2^a] = mask[a], und auch die Kombinationen sind richtig gesetzt.

2. Gaps beschränkter Länge

Motivation

Die Prosite-Datenbank (http://www.expasy.ch/prosite/) enthält Beschreibungen von funktionalen Bereichen in Proteinsequenzen.
Prosite-Patterns sind verallgemeinerte Strings mit Zeichenklassen und zusätzlich mit sogenannten Gaps (variabler, aber) beschränkter Länge.

Diese drücken eine Folge von beliebigen Zeichen aus,
deren Länge zwischen zwei gegebenen Konstanten *m* und *n* liegt.

Prosite-Notation

- Die Elemente eines Prosite-Patterns werden durch einen Strich (-) getrennt.
- Einzelne Buchstaben repräsentieren sich selbst.
- Zeichenklassen werden wie gehabt mit eckigen Klammern geschrieben ([ ]).
- Das Komplement einer Zeichenklasse wird mit { } ausgedrückt.
- Ein Gap der Länge m bis n wird als x(*m*,*n*) geschrieben.
- Statt x(*m*,*m*) schreibt man nur x(*m*); statt x(1) nur x.

Beispiel: Prosite-Pattern ZF_RING_1 (PS00518; Zinc finger RING-type signature)
C - x - H - x - [LIVMFY] - C - x(2) - C - [LIVMYA]

Beispiel: Prosite-Pattern COLD_SHOCK (PS00352; 'Cold-shock' domain signature)
[FYKH] - G - [FL] - [IL] - x(6,7) - [DER] - [LIVM] - [FQ] - x - H - x - [STKR] - x - [LIVMFYC]

NFAs für Gaps beschränkter Länge

lala