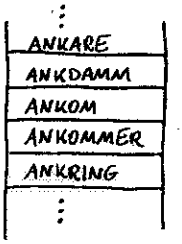
FÖRELÄSNING

TILLÄMPNING: STAVNINGSKONTROLL

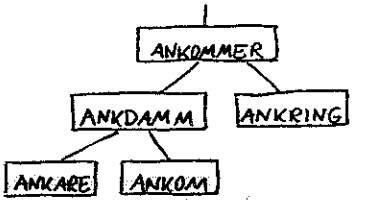
- BLOOMFILTER FÖR ORDUPPSLAGNING
- HANTERING AV SAMMANSÄTTNINGAR
- HANTERING AV BÖJNINGSFORMER
- GENERERING AV RÄTTELSEFÖRSLAG
- OPTIMERING

DATASTRUKTURER FÖR ORDLISTAN

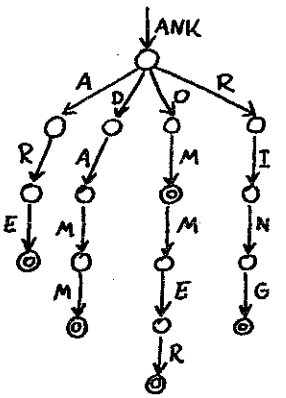
1. SORTERAD VEKTOR



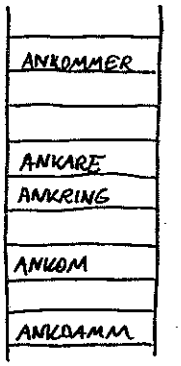
2. BINÄRT TRÄD



3. AUTOMAT



4. HASHTABELL



BLOOMFILTER

DATASTRUKTUR FÖR SNABB KOLL AV MÄNGDTILLHÖRIGHET I STORA MÄNGDER.

BARA TVÅ OPERATIONER:

INSERT(x) STOPPAR IN X I MÄNGDEN

ISIN(x) KOLLAR OM X ÄR I MÄNGDEN

EGENSKAPER:

- BÅDA OPERATIONERNA HAR TIDSKOMPLEXITET $O(1)$
- LITET MINNESUTRYMME KRÄVS
- ELEMENTEN LAGRAS INTE I KLARTEXT
- LITEN SANNOLIKHET FÖR ATT ISIN(x) ÄR SANT TROTS ATT X INTE INGÅR I MÄNGDEN.

TILLÄMPNINGSEXEMPEL:

LAGRING AV ORDLISTAN I STAVNINGSKONTROLL.

LAGRING AV BESÖKTA LÄNKAR I WEBBLÄSARE.

STAVA: BLOOMFILTER

ORDLISTAN LAGRAS SOM ETT BLOOMFILTER, DVS MED HASHNING I EN TABELL MED ENDAST NOLLOR OCH ETTOR.

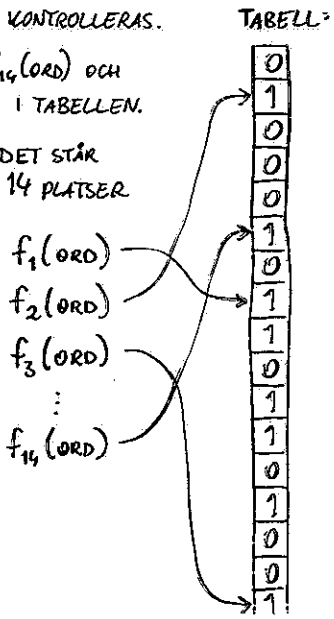
ANVÄND 14 HASHFUNKTIONER

f_i: ORD → PLATS I HASHTABELL

$1 \le i \le 14$

GIVET ORD SOM SKA KONTROLLERAS.

- BERÄKNA $f_1(\text{ORD}), \ldots, f_{14}(\text{ORD})$ OCH UNDERSÖK DESSA PLATSER I TABELLEN.
- ACCEPTERA ORDET OM DET STÅR ETTOR PÅ ALLA DESSA 14 PLATSER



SANNOLIKHET FÖR FEL

LÅT n VARA ANTALET ORD I ORDLISTAN
m —"— BITAR I VEKTORN
k —"— HASHFUNKTIONER

$$P\left[\begin{matrix}\text{EN VISS BIT SÄTTS TILL SANT}\\ \text{VID EN VISS HASHNING}\end{matrix}\right] = \frac{1}{m}$$

$$P\left[\begin{matrix}\text{EN VISS BIT SÄTTS INTE TILL SANT}\\ \text{VID EN VISS HASHNING}\end{matrix}\right] = 1 - \frac{1}{m}$$

$$P\left[\begin{matrix}\text{EN VISS BIT ÄR FORTFARANDE}\\ \text{FALSK EFTER } kn \text{ HASHNINGAR}\end{matrix}\right] = \left(1 - \frac{1}{m}\right)^{kn}$$

$$P\left[\begin{matrix}\text{EN VISS BIT ÄR SANN}\\ \text{EFTER } kn \text{ HASHNINGAR}\end{matrix}\right] = 1 - \left(1 - \frac{1}{m}\right)^{kn}$$

$$P\left[\begin{matrix}k \text{ SLUMPVISA UPPSLAGNINGAR I}\\ \text{VEKTORN GER ALLA SVARET SANT}\end{matrix}\right] = \underbrace{\left[1 - \left(1 - \frac{1}{m}\right)^{kn}\right]^k}_{f(k)}$$

$f(k)$ MINIMERAS AV

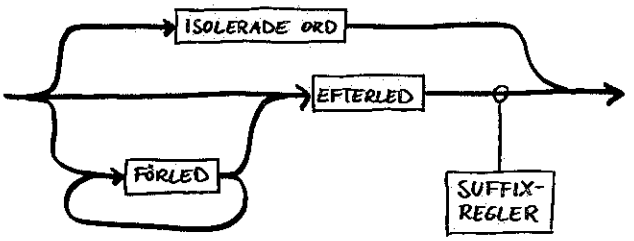
$$k = -\frac{\ln 2}{n \cdot \ln\left(1 - \frac{1}{m}\right)} \approx \ln 2 \cdot \frac{m}{n} \approx 0{,}69\,\frac{m}{n}$$

OCH GER DÅ $f(k) = 2^{-k}$

STAVA: HANTERING AV SAMMANSÄTTNINGAR

TRE ORDLISTOR:

- ISOLERADE ORD (TEX ÖMSOM, ELLER)
- FÖRLED (TEX FOTBOLLS, MEDIE, STAVNINGS)
- EFTERLED (TEX KUNSKAP, STORT, LÅTA, MEDIUM)



STAVA: SUFFIXREGLER

VI LAGRAR VISSA BESTÄMDA FORMER AV ORDEN I ORDLISTAN OCH INFÖR SUFFIXREGLER AV TYPEN

$$-\text{ORNA} \leftarrow -\text{A}, -\text{AN}, -\text{OR}$$

VILKET BETYDER "OM ORDEN XA, XAN OCH XOR FINNS I ORDLISTAN SÅ SKA OCKSÅ XORNA GODKÄNNAS"

EXEMPEL: DOCKA, DOCKAN, DOCKOR ⇒ DOCKORNA

BARA TVÅ AV 1585 ORD SOM PASSAR IN PÅ DENNA REGEL ÄR FEL.

EXEMPEL (FEL): DEKA, DEKAN, DEKOR ⇒ DEKORNA

REGLERNA UTFORMAS SÅ ATT DENNA TYP AV FEL MINIMERAS.

Exempel på rättstavning

Felstavat ord: STRUTN

- Kasta om intilliggande bokstäver:
  TSRUTN, SRITUN, STURTN, STRTUN, STRUNT
- Sätt in en extra bokstav:
  ...ESTRUTN, SETRUTN, STERUTN, STREUTN, STRUETN, STRUTEN, STRUTNE, ...
- Ta bort en bokstav:
  TRUTN, SRUTN, STUTN, STRTN, STRUN, STRUT
- Ersätt en bokstav:
  ... YTRUTN, ... SXRUTN, ... STWUTN,
  ... STRVTN, ... STRUUN, ... STRUTS, ...

Problem: Väldigt många ord måste genereras, varav många är helt okiga.
Exempel: felstavat 10-bokstavsord ger 618 ord

Grafotaktisk tabell

Tabell som innehåller alla 4-bokstavsföljder (fyrgram) som förekommer i språket.

Exempel: STRUET, RUTNET, SXRU¢T

Ta också hänsyn till ordbörjan och ordslut.

I svenska språket finns 6% av alla fyrgram

Två användningsområden:

1. Alternativ till Bloomfilter för att avgöra om ett ord finns i språket
2. Hitta var i ett ord en felstavning förekommer

Exempel: STRUTN
Alla fyrgram utom UTN■ finns i T.
Alltså måste felstavningen vara bland dom tre sista bokstäverna i ordet.

Optimering av hashning

Observation:
Nästan all tid av rättstavningen tillbringas i hashningsfunktionen.

Försök optimera hashningen!

Ursprungliga hashfunktioner:

$$f_i(w) = \sum_j k_{i,j} \cdot w[j] \mod p_i$$

För ett ord med n bokstäver görs
- n multiplikationer
- n moduloberäkningar
- n-1 additioner
- 3n indexeringar

vid varje hashfunktionsberäkning.

Moduloberäkning kräver division och tar längst tid av dessa operationer.

Moduloberäkning med flyttal

$x \bmod p$ Resten då x divideras med p.

$= x - \left\lfloor \frac{x}{p} \right\rfloor \cdot p$ Låt $q = \frac{1}{p}$.

$= x - \lfloor x \cdot q \rfloor \cdot p$

Förbehandla genom att beräkna $\frac{1}{p_i}$ för alla hashfunktioner.

Därefter kan en moduloberäkning ersättas av
- En omvandling heltal → flyttal
- En flyttalsmultiplikation
- En omvandling flyttal → heltal
- En heltalsmultiplikation
- En heltalssubtraktion