En karakteristik af de regulære sprog

Et sprog *L* er regulært hvis og kun hvis

- *L* beskrives af et regulært udtryk
- *L* genkendes af en FA / NFA / NFA-Λ
- der *ikke* findes uendeligt mange strenge, der er parvist skelnelige mht. *L*

Skelnelighed (uge 15)

- x og y er ***skelnelige*** mht. L hvis
  $\exists z \in \Sigma^*: (xz \in L \wedge yz \notin L) \vee$
  $(xz \notin L \wedge yz \in L)$
- Hvis *skelnelige* strenge mht. L køres på en FA, der accepterer L, vil de ende i *forskellige* tilstande

- Intuition bag FA-minimering:

hvis to strenge er **u**skelnelige mht. FA'ens sprog, er der ingen grund til at den skelner mellem dem!

Uskelnelighedsrelationen I_L

Definition:

Givet et sprog $L \subseteq \Sigma^*$, definer relationen I_L over Σ^* ved:

$x \; I_L \; y \quad \Leftrightarrow \quad x$ og y er *uskelnelige* mht. L

Relationer

[Martin, kap. 1.4]

- En (binær) **relation** R over en mængde A er en delmængde af $A \times A$

- Eksempler:
  - $\leq$ er en relation over mængden af reelle tal
  - I_L er en relation over Σ^*

- Notation: $x \, R \, y \iff (x,y) \in R$

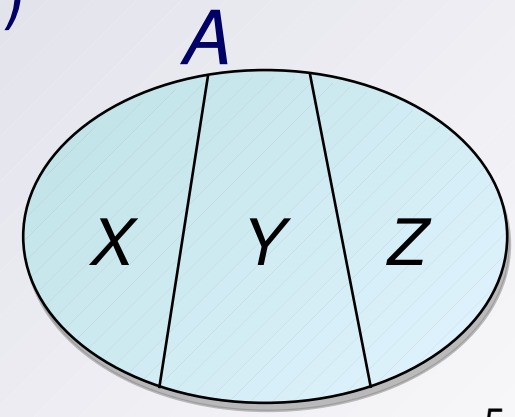
Ækvivalensrelationer

[Martin, kap. 1.4]

- R er en **ækvivalensrelation** hvis den er
  - *refleksiv* $(\forall x:\ x\ R\ x)$
  - *symmetrisk* $(\forall x,y:\ x\ R\ y \Rightarrow y\ R\ x)$
  - *transitiv* $(\forall x,y,z:\ x\ R\ y \wedge y\ R\ z \Rightarrow x\ R\ z)$

- En ækvivalensrelation over A definerer en **partitionering** af A (og omvendt)

- Notation: $[x] = \{y \mid x\ R\ y\}$ kaldes **ækvivalensklassen** for x mht. R



Egenskaber ved I_L

I_L er

- *refleksiv* $(\forall x\colon\ x\ I_L\ x)$
- *symmetrisk* $(\forall x,y\colon\ x\ I_L\ y \Rightarrow y\ I_L\ x)$
- *transitiv* $(\forall x,y,z\colon\ x\ I_L\ y \wedge y\ I_L\ z \Rightarrow x\ I_L\ z)$

dvs. I_L er en **ækvivalensrelation**

- I_L partitionerer Σ^*
- [x] er mængden af strenge, der er uskelnelige fra x mht. L

Quiz!

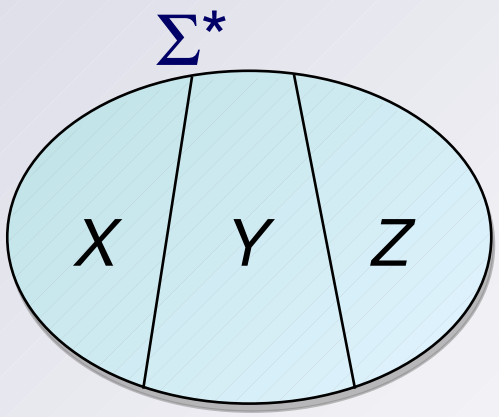
$L = \{0,1\}^*\{10\}$

Beskriv ækvivalensklasserne for I_L

Hint: der er 3 ækvivalensklasser...

Hint: find en streng, der er skelnelig fra Λ...

Hint: find en streng, der er skelnelig fra både Λ og 1...



- X: $\{\Lambda, 0\} \cup \{0,1\}^*\{00\}$ $= [\Lambda]$
- Y: $\{0,1\}^*\{1\}$ $= [1]$
- Z: $\{0,1\}^*\{10\}$ $= [10]$

en *repræsentant* for hver ækvivalensklasse

MyHill-Nerode-sætningen

L er regulært $\Leftrightarrow$ I_L har endeligt mange ækvivalensklasser

- "$\Rightarrow$": (uge 15) hvis I_L har **u**endeligt mange ækvivalensklasser, så er L **ikke** regulært
- "$\Leftarrow$": Bevis følger...

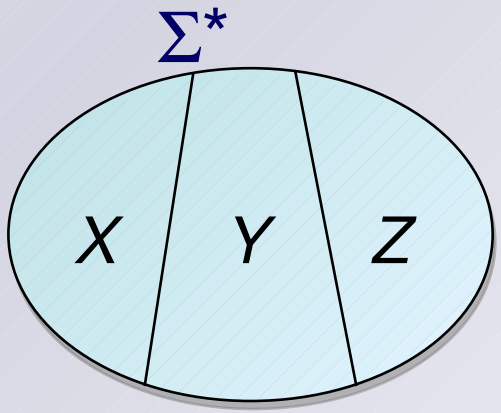
Konstruktion af en FA fra I_L

- Givet et sprog $L \subseteq \Sigma^*$, antag I_L har endeligt mange ækvivalensklasser

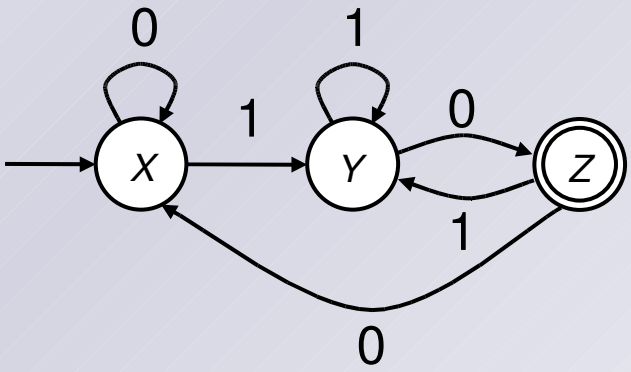
- Vi kan definere en FA, hvor **tilstandene er ækvivalensklasserne** af I_L

Eksempel

- Ækvivalensklasserne for I_L når $L = \{0,1\}^*\{10\}$:
  - X: $\{\Lambda, 0\} \cup \{0,1\}^*\{00\}$
  - Y: $\{0,1\}^*\{1\}$
  - Z: $\{0,1\}^*\{10\}$



- M_L :



Konstruktion af en FA fra I_L

Definer en FA:

$M_L=(Q, \Sigma, q_0, A, \delta)$ hvor

- $Q = Q_L$ hvor Q_L er ækvivalensklasserne af I_L
- $q_0 = [\Lambda]$
- $A = \{ q \in Q \mid q \cap L \neq \emptyset \}$

$\forall\, \delta(q, a) = p$ hvis $q=[x]$ og $p=[xa]$ for en streng x

(δ er veldefineret idet $x\, I_L\, y \Rightarrow xa\, I_L\, ya$)

Påstand: $L(M_L) = L$

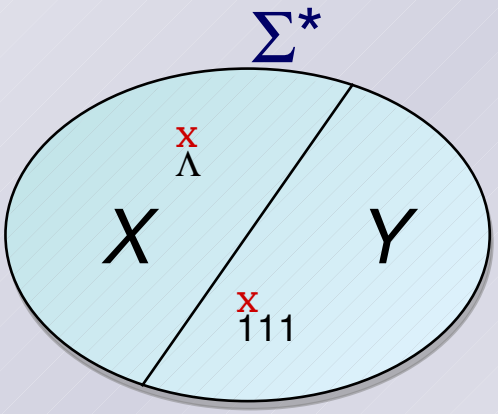
Quiz!

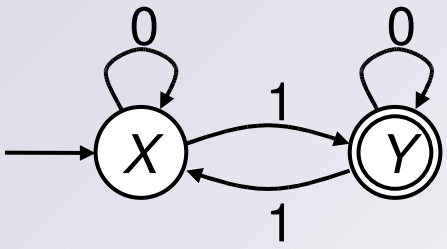
Antag ækvivalensklasserne for I_L er

- $X = \{x \in \{0,1\}^* \mid$ antal 1'er i x er lige$\}$
- $Y = \{x \in \{0,1\}^* \mid$ antal 1'er i x er ulige$\}$

og $111 \in L$

Lav en FA, der accepterer L





Bevis for korrekthed af konstruktionen

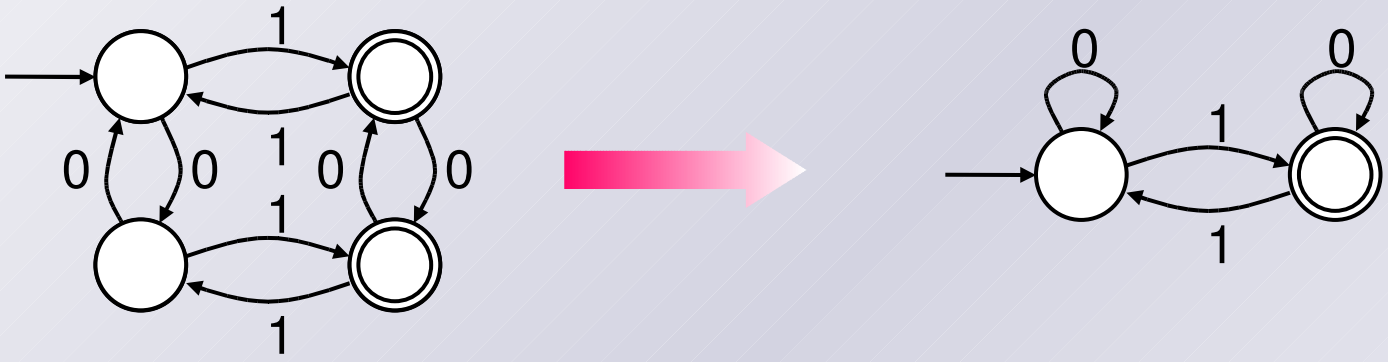
- Påstand: $L(M_L) = L$
- Lemma: $\forall x,y \in \Sigma^*: \quad \delta^*([x], y) = [xy]$

  Bevis: induktion i strukturen af *y*...
- $\delta^*(q_0, x) = \delta^*([\Lambda], x) = [x]$ (følger af lemmaet og def. af q_0)
- $x \in L(M_L) \Leftrightarrow \delta^*(q_0, x) \in A \Leftrightarrow [x] \in A \Leftrightarrow [x] \cap L \neq \emptyset$
- $x \in L \Rightarrow [x] \cap L \neq \emptyset$ (da $x \in [x]$)
- $[x] \cap L \neq \emptyset \Rightarrow x \in L$ (bruger def. af I_L)
- dvs. $x \in L(M_L) \Leftrightarrow x \in L$

M_L er minimal!

- Lad n være antallet af ækvivalensklasser af $\mathcal{I}_L$
- M_L har 1 tilstand for hver ækvivalensklasse af $\mathcal{I}_L$
- Vælg en streng x_i fra hver ækvivalensklasse
- For ethvert par x_i, x_j, $i \neq j$:
  x_i og x_j er skelnelige mht. L
- dvs. enhver FA der genkender L har mindst n tilstande (jfr. Første seminar)
- og M_L har netop n tilstande, så M_L er **minimal**!

Minimering af automater



- Man kan i visse tilfælde opnå en mindre FA ved at “slå tilstande sammen”...
- *Kan vi gøre det systematisk?*
- *Vil den resulterende FA blive **minimal**?*

En algoritme til FA-minimering

Fra MyHill-Nerode-sætningen kan vi udlede en algoritme, der

givet en vilkårlig FA $M=(Q, \Sigma, q_0, A, \delta)$,
finder en **minimal** FA M_1 hvor $L(M_1)=L(M)$

To partitioner af Σ^*

#1 Ækvivalensklasserne af I_L

(svarer til tilstandene i den minimale FA M_L)

#2 En opdeling af alle $x \in \Sigma^*$ efter værdien af $\delta^*(q_0, x)$

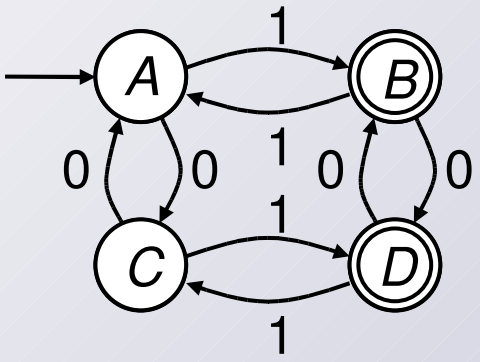
(svarer til tilstandene i den givne FA M)

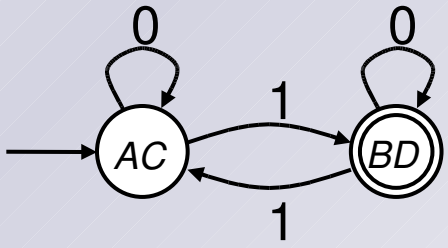
Definer for alle $q \in Q$:

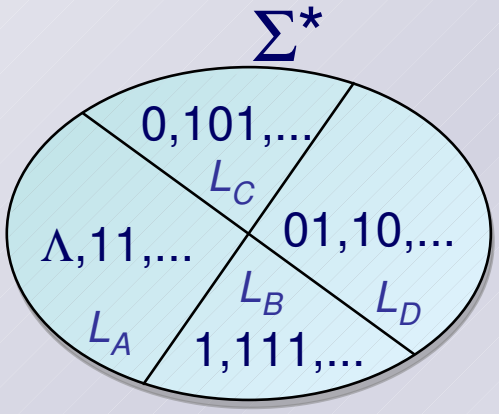
$$L_q = \{ x \in \Sigma^* \mid \delta^*(q_0, x) = q \}$$

Kan vi konstruere #1 ud fra #2?

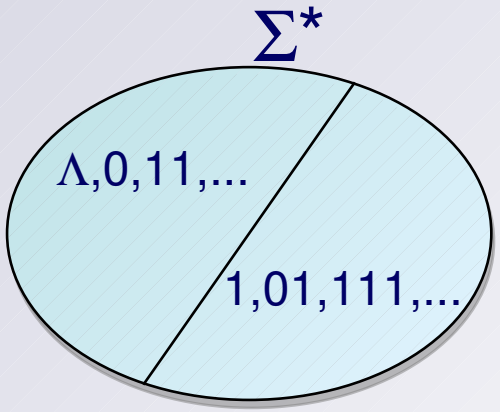
Eksempel







#2



#1

Fjern uopnåelige tilstande

- Ækvivalensklasserne af I_L indeholder alle mindst 1 streng
- Det er muligt at $L_q = \emptyset$ for en eller flere $q \in Q$ (hvis q er **uopnåelig** fra q_0)
- Vi har en algoritme, der kan fjerne uopnåelige tilstande fra en FA uden at ændre sproget
- Vi kan derfor antage at $L_q \neq \emptyset$ for alle $q \in Q$

Opnåelige tilstande

- Givet en FA $M=(Q, \Sigma, q_0, A, \delta)$

- Lad R være den mindste mængde, der opfylder
  - $q_0 \in R$

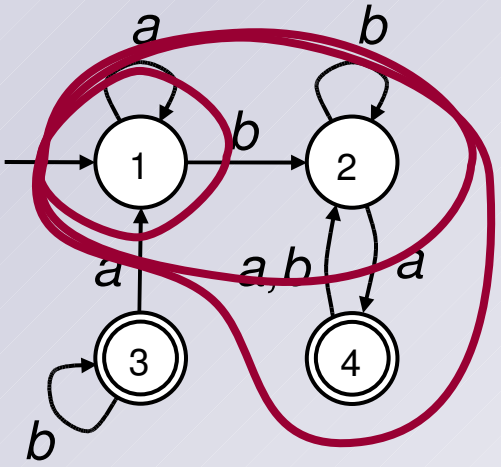
  $\forall\forall q \in R, a \in \Sigma: \ \delta(q, a) \in R$

(ligner definitionen af Λ-lukning...)

- R er mængden af **opnåelige tilstande** i M

Eksempel

- R kan findes med en fixpunktsalgoritme:



- $1 \in R$
- $\delta(1, b)=2 \in R$
- $\delta(2, a)=4 \in R$
- fixpunkt er nu nået

dvs. de opnåelige tilstande er 1,2,4

Forholdet mellem partition #1 og #2

- Fra 1. seminar: $\delta^*(q_0, x)=\delta^*(q_0, y) \Rightarrow x \, I_L \, y$
- Dvs. enhver L_q–mængde er helt indeholdt i én I_L-ækvivalensklasse
- Enhver ækvivalensklasse af I_L er derfor foreningen af en eller flere af L_q–mængderne
- Da $L_q \neq \emptyset$ er hver af disse foreninger **unik**
- Definition: $p \equiv q \Leftrightarrow L_p$ og L_q er delmængder af samme I_L-ækvivalensklasse
- Dvs. hvis $p \equiv q$, så svarer p og q til samme tilstand i den minimale automat!

Konstruktion af ≡ (minimeringsalgoritmen)

Lad S være den mindste mængde, der opfylder:

b) $(p \in A \wedge q \notin A) \vee (p \notin A \wedge q \in A) \Rightarrow (p, q) \in S$

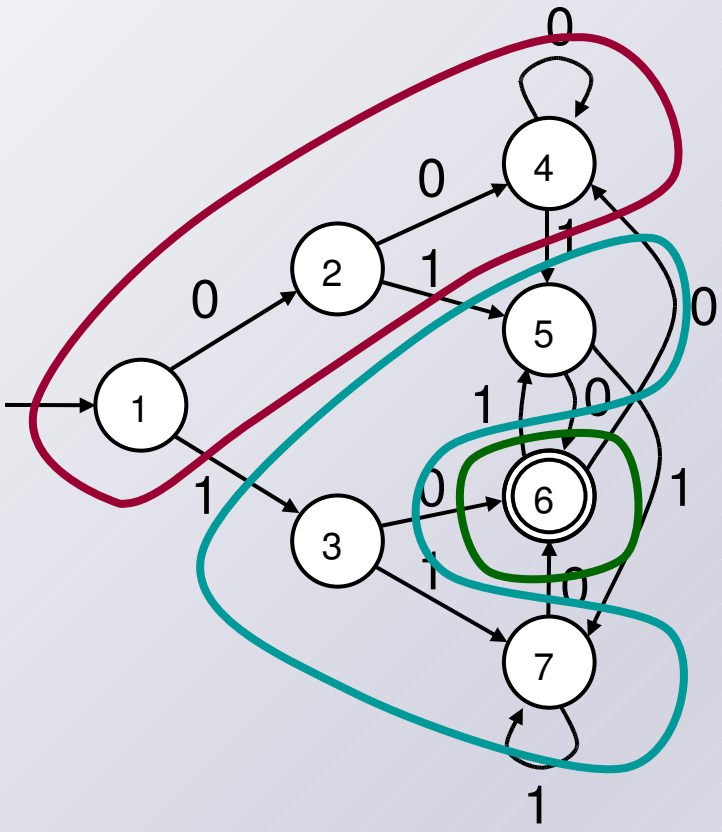
c) $(\exists a \in \Sigma: (\delta(p, a), \delta(q, a)) \in S) \Rightarrow (p, q) \in S$

Påstand: $p \not\equiv q$ hvis og kun hvis $(p, q) \in S$

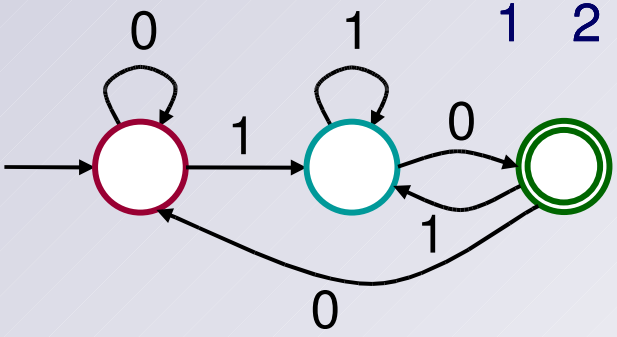
S kan beregnes med en fixpunktsalgoritme

(i stil med opnåelige tilstande og Λ-lukning tidligere...)

Eksempel på FA-minimering



	1	2	3	4	5	6
2						
3	X	X				
4			X			
5	X	X		X		
6	X	X	X	X	X	
7	X	X		X		X



1. Fjern uopnåelige tilstande (ingen i denne FA)
2. Find ≡ ved at udfylde en tabel for S (fixpunktsberegning)
3. Kombiner tilstande, der svarer til umærkede par

Bevis for korrekthed

Antag $p,q \in Q$, $x \in L_p$, $y \in L_q$

(dvs. $\delta^*(q_0, x)=p$ og $\delta^*(q_0, y)=q$)

Lemma: Følgende udsagn er ækvivalente:

5. $p \equiv q$
6. $x \; I_L \; y$
7. $\forall z \in \Sigma^*: \; \delta^*(p, z) \in A \; \Leftrightarrow \; \delta^*(q, z) \in A$

Bevis for korrekthed

- Påstand: $p \not\equiv q$ hvis og kun hvis $(p, q) \in S$

- Iflg. lemmaet:

$$p \not\equiv q \;\Leftrightarrow\; \big(\exists z \in \Sigma^*:\; (\delta^*(p, z) \in A \wedge \delta^*(q, z) \notin A) \vee (\delta^*(p, z) \notin A \wedge \delta^*(q, z) \in A)\big)$$

- $p \not\equiv q \;\Rightarrow\; (p, q) \in S$ (brug lemmaet, lav induktion i z)
- $(p, q) \in S \;\Rightarrow\; p \not\equiv q$ (brug lemmaet, lav induktion i S)

FA minimering i dRegAut Java-pakken

”pseudo-kode”:

uformel mellemting mellem
de matematiske definitioner
og Java-koden

FA.findReachableStates(), version 1

```
Set findReachableStates() {
    R = { q0 }
    done = false
    while ¬done do
        done = true
        for each q∈ R do
            for each a∈ Σ do
                p = δ(q, a)
                if p∉ R then
                    add p to R
                    done = false
    return R
}
```

FA.findReachableStates(), version 2

```
Set findReachableStates() {
    R = ∅
    pending = { q0 }
    while pending≠∅ do
        pick and remove an element q from pending
        add q to R
        for each c∈Σ do
            p = δ(q, c)
            if p∉R then add p to pending
    return R
}
```

Ved hjælp af *pending* undgår vi at besøge hver tilstand flere gange

FA.minimize()

```
FA minimize() {
    phase 1:    remove unreachable states
    phase 2a:   divide into accept/reject states
    phase 2b:   iteration
    phase 3:    build resulting minimal automaton n
}
```

FA.minimize(), phase 2a

define some ordering on the states Q

declare *marks*: a set of pairs (p,q) where $p,q \in Q$ and $p<q$

$marks = \emptyset$

for each pair $p,q \in Q$ where $p<q$ do

    if $\neg(p \in A \Leftrightarrow q \in A)$ then

        add (p,q) to *marks*

Mange muligheder for Java-representation af *marks*...

FA.minimize(), phase 2b

done = false
while ¬*done* do
    done = true
    for each pair $p,q \in Q$ where $p<q$ do
        if $(p,q) \notin$ *marks* then
            for each $a \in \Sigma$ do
                $r = \delta(p, a)$
                $s = \delta(q, a)$
                if $r > s$ then swap r and s
                if $(r,s) \in$ *marks* then
                    add (p,q) to *marks*
                    done = false

Kunne også laves smartere med

en "*pending*" worklist...

FA.minimize(), phase 3

FA n = new FA with same alphabet as f
but with no states or transitions yet

initialize empty maps *old2new*: $f.Q \to n.Q$ and *new2old*: $n.Q \to f.Q$

for each $r \in f.Q$ in order do
  if $(s,r) \in$ *marks* for every $s<r$ then
    add a new state p to $n.Q$
    add *old2new*(r) = p and *new2old*(p) = r
    if $r \in f.A$ then add p to $n.A$
  else
    add *old2new*(r) = *old2new*(s)
  if $r = f.q_0$ then set $n.q_0$ = *old2new*(r)

for each state p in n do
  add $n.\delta(p,c)$ = *old2new*($f.\delta$(*new2old*(p),c)) for each $c \in \Sigma$

Eksempel

```
Alphabet a = new Alphabet('0', '1');
RegExp r = new RegExp("0+(1*+01*+10*+001*01)*0*", a);

NFALambda n1 = r.toNFALambda();
NFA n2 = n1.removeLambdas();
FA n3 = n2.determinize();

System.out.println("Før: "+n3.getNumberOfStates());
FA n4 = n3.minimize();
System.out.println("Efter: "+n4.getNumberOfStates());
```

```
Før: 13
Efter: 1
```

Resume

- MyHill-Nerode-sætningen:
  endnu en karakteristik af de regulære sprog
- en algoritme til FA minimering
- en algoritme til at fjerne uopnåelige tilstande i en FA