

# Syntaks og semantik

## Lektion 1

5 februar 2008

- 1 Introduktion
- 2 Ord og sprog
- 3 Regulære udtryk

# Introduktion til kurset

- 1 Indhold
- 2 Form
- 3 Materiale
- 4 Eksamen
- 5 Folk

# Kursets emne

Grundlæggende aspekter ved programmeringssprog:

- Hvordan kan vi beskrive hvordan et sprog **ser ud**? (dets **form**)
- Hvordan kan vi beskrive hvordan et sprog skal **forstås**? (dets **adfærd**)

# Kursets emne

Grundlæggende aspekter ved programmeringssprog:

- Hvordan kan vi beskrive hvordan et sprog **ser ud**? (dets **form**)

## Syntaks:

- regulære sprog, endelige automater, regulære udtryk
  - kontekst-frie sprog, push-down-automater, kontekst-frie grammatikker
- Hvordan kan vi beskrive hvordan et sprog skal **forstås**? (dets **adfærd**)

## Semantik:

- operationel semantik
- denotationel semantik

# Kursets indhold

## Syntaks – regulære sprog:

- 1 Introduktion; sprog; regulære udtryk
- 2 Endelige automater
- 3 Sprog der *ikke* er regulære

## Syntaks – kontekstfrie sprog:

- 4 Kontekstfrie grammatikker
- 5 Pushdown-automater
- 6 Sprog der *ikke* er kontekstfrie

# Kursets indhold

## Semantik:

- 7 Operationelle semantikker for et simpelt imperativt sprog
- 8 Operationelle semantikker for diverse udvidelser af sproget
- 9 Blokke og procedurer
- 10 Paramettermekanismer
- 11 Denotationel semantik

## Teoretisk grundlag:

- 12 Domæneteori, rekursion og fikspunkter

# Hvad kan jeg bruge det til?

- Vil jeg lære et nyt programmeringssprog?

Nej.

- Skal vi se nogen smarte algoritmer?

Nej.

- Vil jeg blive bedre til at programmere?

Forhåbentlig.

- Vil jeg opnå større forståelse for hvordan programmeringssprog er opbygget?

Ja.

- Vil jeg opnå større forståelse for hvilke problemer computere kan løse?

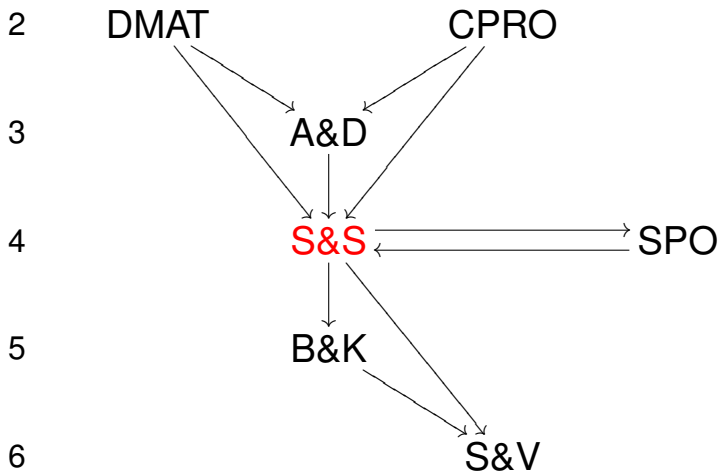
Til dels.

- Vil jeg blive bedre til at forstå teorien bag programmering?

Ja.



# Hvordan hænger det her sammen med andre kurser?



# Kursets form

- 8:15 – 10:00: Forelæsning
- 10:10 – 12:00: Opgaveregning

# Kursets form

- 8:15 – 10:00: Forelæsning
  - normalt i 0.1.95
  - Læs stoffet *hurtigt* inden forelæsningen, så I ved hvad det handler om, og læs det *grundigt* igen bagefter, så I er sikre på at have forstået det.
  - Kursets emner bygger ovenpå hinanden, så hvis der er noget man misser, er det svært at finde tilbage igen!
- 10:10 – 12:00: Opgaveregning
  - i grupperum
  - to større afleveringsopgaver
  - Forvent ikke at kunne forstå stoffet uden at regne opgaver.
  - Studerende der ikke regner opgaver, kan ikke opholde sig i grupperummet under opgaveregningen.

# Afleveringsopgaver

- to gennemgående opgaver som I skal bruge en del af opgaveregningen på, *hver gang*
- afleveres til mig, kommenteres bagefter først af jeres kolleger og til sidst af mig
- vil være del af eksamenspensum
- kan for PE-studerende erstattes af tilsvarende opgaver der har *relation til projektet*
- Syntaksopgave
- Semantikopgave

# Afleveringsopgaver

- to gennemgående opgaver som I skal bruge en del af opgaveregningen på, *hver gang*
- afleveres til mig, kommenteres bagefter først af jeres kolleger og til sidst af mig
- vil være del af eksamenspensum
- kan for PE-studerende erstattes af tilsvarende opgaver der har *relation til projektet*
- Syntaksopgave
  - tilgængelig nu
  - afleveres 10 marts
  - evt. erstatningsopgave skal indleveres senest 15 februar
- Semantikopgave
  - vil blive offentliggjort i starten af marts

# Bøger

- Michael Sipser: *Introduction to the Theory of Computation*, Second Edition, PWS Publishing Co. 2005.  
Brug ikke ældre udgaver, der er lavet for meget om!
- Hans Hüttel: *Pilen ved træets rod*, Aalborg Universitet 2007.
- Sipser skal vi bruge *nu*
- Hüttel først i marts

# Hjemmeside

<http://sands07.twoday.net>

- slides
- opgaver
- andet materiale
- interessante links
- RSS-feed
- kommentarfunktion!

# Eksamen

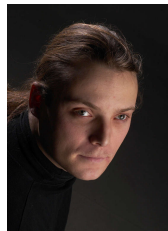
- mundtlig, 20min
- et antal spørgsmål kendt på forhånd – træk ét af dem
- 20min forberedelsestid
- ekstern censor, karakter
- pensum og spørgsmål fastlægges ved tredjesidste kursusgang
- afleveringsopgaver indgår som hver deres spørgsmål





Uli Fahrenberg  
underviser

`uli@cs.aau.dk`



Jens Alsted  
hjælpelærer

`alsted@cs.aau.dk`

# Sprog og regulære udtryk

- 6 Motivation: Regulære udtryk
- 7 Bogstaver, ord
- 8 Sprog
- 9 At sammensætte ord
- 10 Operationer på sprog
- 11 Regulære udtryk igen
- 12 Regulære sprog

Regulære udtryk bruges til tekstbehandling:

- at **søge** efter mønstre
- at **erstatte**

Eksempler:

- `grep 'Hans' manual.tex`
- `grep 'vi[ ,.]' manual.tex`
- `grep 'o[a-zA-Z]*o[a-zA-Z]*o' manual.tex`
- `sed 's:\\[a-zA-Z]*: :g' manual.tex`
- `sed 's:\\usepackage{[a-zA-Z]*}: :g' manual.tex`

Et regulært udtryk definerer et **sprog**:

- $[[\text{Hans}]] = \{\text{Hans}\}$
- $[[v_i[ , . ]]] = \{v_i , v_i , , v_i .\}$
- $[[o[a-zA-Z]^*o]] = \{\text{otto}, \text{othello}, \text{ohMyGodNo}, \text{oo}, \dots\}$
- $[[\\[a-zA-Z]^*]] = \{\backslash\text{section}, \backslash\text{emph}, \backslash\text{prut}, \backslash, \dots\}$
- $[[\\usepackage\\(\\[a-zA-Z]^*\\)\\?{[a-zA-Z]^*}]]$   
 $= \{\backslash\text{usepackage}\{\text{url}\}, \backslash\text{usepackage}[\text{danish}]\{\text{babel}\},$   
 $\backslash\text{usepackage}\{\}, \backslash\text{usepackage}[\ ]\{\}, \dots\}$

**Mål** for i dag: At gøre det her *præcist*

- $\Sigma$  – en endelig mængde af **bogstaver** eller **symboler**  
– et **alfabet**
- et **ord**: en endelig følge af bogstaver  
– normalt skrevet uden parenteser eller kommaer
- eksempel:  $\Sigma = \{0, 1\}$   
ord over  $\Sigma$ : f.x. 0, 1, 00, 01, 1001010110101
- eksempel:  $\Sigma = \{a, b, c, d, r\}$   
ord over  $\Sigma$ : f.x. *a, b, c, d, r, abba, abracadabra*
- eksempel:  $\Sigma = \{\text{gik, jeg, land, mig, og, over, sø, to, vi}\}$   
ord over  $\Sigma$ : f.x. “jeg og mig og vi to”  
eller “jeg gik mig over sø og land”
- eksempel:  $\Sigma = \{\text{else, if, then, } Exp, Stm\}$   
ord over  $\Sigma$ : f.x. “if *Exp* then *Stm* else *Stm*”

- et **sprog**: en mængde af ord (endelig eller uendelig)
- mængden af *alle* ord over et alfabet  $\Sigma$  skrives  $\Sigma^*$   
(den er altid uendelig (medmindre  $\Sigma$  er tom ...))
- eksempel:  $\Sigma = \{0, 1\}$   
 $\Sigma^* = \{\varepsilon, 0, 1, 00, 01, 10, 11, 000, 001, 010, \dots\}$
- $\varepsilon$  – det *tomme* ord; ordet af længde 0
- **længden** af et ord:  $|w|$  = antallet af bogstaver i ordet
- $\emptyset$  – det *tomme sprog*; mængden uden indhold
- **Bemærk**:  $\varepsilon$  er et *ord*,  $\emptyset$  er et *sprog*. Og  $\{\varepsilon\} \neq \emptyset$

- at **sammensætte** ord:  $abe \circ kat = abekat$   
(svarer til at gange tal sammen, men *ikke kommutativt*!)  
( $\circ$ -tegnet udelades de fleste gange)
- $\varepsilon$  er **identiteten**:  $w \circ \varepsilon = w$  og  $\varepsilon \circ w = w$  for alle ord  $w$ .  
(ligesom tallet 1 er identiteten for multiplikation)
- gentagen sammensættelse skrives som **potenser**:  $a^2 = aa$ ,  
 $a^3 = aaa$ ,  $a^9 = aaaaaaaaaa$  etc.

Hvis  $L_1$  og  $L_2$  er sprog over et alfabet  $\Sigma$ , kan vi danne

- foreningsmængden  $L_1 \cup L_2 = \{w \mid w \in L_1 \text{ eller } w \in L_2\}$ 
  - sproget med alle de ord der er i  $L_1$  **eller**  $L_2$
- fællesmængden  $L_1 \cap L_2 = \{w \mid w \in L_1 \text{ og } w \in L_2\}$ 
  - sproget med alle de ord der er i  $L_1$  **og**  $L_2$
- sammensætningen  $L_1 \circ L_2 = \{w_1 \circ w_2 \mid w_1 \in L_1 \text{ og } w_2 \in L_2\}$ 
  - sproget med alle de ord der er **sammensætninger** af et ord fra  $L_1$  efterfulgt af et ord fra  $L_2$
- stjernen  $L_1^* = \{w_1 \circ w_2 \circ \dots \circ w_k \mid \text{alle } w_i \in L_1\}$ 
  - sproget med alle de ord der er **sammensætninger** af **vilkårligt mange** ord fra  $L_1$
  - indeholder  $\varepsilon$ : det tomme ord = sammensætningen af **0** ord fra  $L_1$  ...



Vi kan beskrive sprog som mængder: (her lader vi  $\Sigma = \{a, b\}$  )

- $L_1 = \{a, b, ab\}$  (et *endeligt* sprog)
- $L_2 = \{a^n \mid n \in \mathbb{N}\}$  – alle ord der indeholder kun  $a$ , af vilkårlig længde
- $L_3 = \{a^n b a^m \mid n, m \in \mathbb{N}\}$  – alle ord der indeholder præcist ét  $b$
- $L_4 = \{a^n b^n\}$  – alle ord der indeholder et antal  $a$  og så *samme* antal  $b$

eller ved hjælp af **regulære udtryk**:

- $L_1 = a \cup b \cup ab$
- $L_2 = a^*$
- $L_3 = a^* \circ b \circ a^*$
- $L_4 = ???$

(vi skal senere se at  $L_4$  *ikke* kan beskrives ved regulære udtryk!)

**Definition 1.52:** Et **regulært udtryk** over et alfabet  $\Sigma$  er et udtryk af formen

- ❶  $a$  for et  $a \in \Sigma$ ,
  - ❷  $\varepsilon$ ,
  - ❸  $\emptyset$ ,
  - ❹  $(R_1 \cup R_2)$ , hvor  $R_1$  og  $R_2$  er regulære udtryk,
  - ❺  $(R_1 \circ R_2)$ , hvor  $R_1$  og  $R_2$  er regulære udtryk, eller
  - ❻  $(R_1^*)$ , hvor  $R_1$  er et regulært udtryk.
- en **induktiv** (eller **rekursiv**) definition: 1. til 3. giver de basale *byggesten*, og 4. til 6. giver *byggeregler* til hvordan man kan sætte ting sammen.
- parenteserne udelades ofte

**Definition 1.52:** Et **regulært udtryk** over et alfabet  $\Sigma$  er et udtryk af formen

- 1  $a$  for et  $a \in \Sigma$ ,
- 2  $\varepsilon$ ,
- 3  $\emptyset$ ,
- 4  $(R_1 \cup R_2)$ , hvor  $R_1$  og  $R_2$  er regulære udtryk,
- 5  $(R_1 \circ R_2)$ , hvor  $R_1$  og  $R_2$  er regulære udtryk, eller
- 6  $(R_1^*)$ , hvor  $R_1$  er et regulært udtryk.

Eksempler (med  $\Sigma = \{a, b\}$ ):

$a, b, a \cup b, (a \cup b)^*, (a \cup b)^* \circ b, ((a \cup b)^* \circ b)^*$

**Definition 1.52, fortsat: Sproget**, som et regulært udtryk  $R$  beskriver, betegnes  $\llbracket R \rrbracket$  og er defineret som følger:

- 1  $\llbracket a \rrbracket = \{a\}$
- 2  $\llbracket \varepsilon \rrbracket = \{\varepsilon\}$
- 3  $\llbracket \emptyset \rrbracket = \emptyset$
- 4  $\llbracket R_1 \cup R_2 \rrbracket = \llbracket R_1 \rrbracket \cup \llbracket R_2 \rrbracket$
- 5  $\llbracket R_1 \circ R_2 \rrbracket = \llbracket R_1 \rrbracket \circ \llbracket R_2 \rrbracket$
- 6  $\llbracket R_1^* \rrbracket = \llbracket R_1 \rrbracket^*$

**Definition 1.52, fortsat: Sproget**, som et regulært udtryk  $R$  beskriver, betegnes  $\llbracket R \rrbracket$  og er defineret som følger:

- 1  $\llbracket a \rrbracket = \{a\}$
- 2  $\llbracket \varepsilon \rrbracket = \{\varepsilon\}$
- 3  $\llbracket \emptyset \rrbracket = \emptyset$
- 4  $\llbracket R_1 \cup R_2 \rrbracket = \llbracket R_1 \rrbracket \cup \llbracket R_2 \rrbracket$
- 5  $\llbracket R_1 \circ R_2 \rrbracket = \llbracket R_1 \rrbracket \circ \llbracket R_2 \rrbracket$
- 6  $\llbracket R_1^* \rrbracket = \llbracket R_1 \rrbracket^*$

– Sipser skriver  $L(R)$  i stedet for  $\llbracket R \rrbracket$ . Jeg vil bruge begge notationer

### Udvidelser:

- $\Sigma = a_1 \cup a_2 \cup \dots \cup a_n$  (hvis sproget er  $\Sigma = \{a_1, a_2, \dots, a_n\}$  )
- $R^+ = R \circ R^*$

## Eksempler (1.53): (for $\Sigma = \{0, 1\}$ )

- 1  $\llbracket 0^*10^* \rrbracket$  = sproget med alle ord der indeholder symbolet 1  
*præcist én gang*
- 2  $\llbracket \Sigma^*1\Sigma^* \rrbracket$  = sproget med alle ord der indeholder symbolet 1  
*mindst én gang*
- 4  $\llbracket (01^+)^* \rrbracket$  = sproget af alle ord hvori ethvert 0 efterfølges af  
mindst ét 1
- 5  $\llbracket (\Sigma\Sigma)^* \rrbracket = \{w \mid |w| \text{ er et lige tal}\}$
- 8  $\llbracket 0\Sigma^*0 \cup 1\Sigma^*1 \cup 0 \cup 1 \rrbracket = \{w \mid \text{start- og slutsymbolet i } w \text{ er ens}\}$
- 9  $(0 \cup \varepsilon)1^* = 01^* \cup 1^*$
- X  $(0 \cup 1)^* = (0^*1^*)^*$

Sipser	grep, sed etc.	kommentarer
$a$	$a$	
$\Sigma$	$.$	
$\varepsilon, \emptyset$		findes ikke; se nedenfor
$R_1 \cup R_2$	$R_1 \mid R_2$	
$R_1 \circ R_2$	$R_1 R_2$	
$R^*$	$R^+$	
$R^+$	$R^+$	
$(, )$	$\backslash (, \backslash )$	
$R \cup \varepsilon$	$R?$	højst én forekomst af $R$
	$[abcd]$	svarer til $a \cup b \cup c \cup d$
	$[:alpha:]$	matcher alle bogstaver
	$[:digit:]$	matcher alle cifre
	$^$	negation. <b>Rigtige regexps</b> <b>indeholder ikke negation!</b>

se også `info sed` eller `man 7 regex`

**Definition:** Et sprog kaldes **regulært** hvis det kan frembringes af et regulært udtryk.

- eller måske: Et sprog  $L \subseteq \Sigma^*$ , for  $\Sigma$  et alfabet, siges at være **regulært** hvis der findes et regulært udtryk  $R$  over  $\Sigma$  for hvilket  $L = \llbracket R \rrbracket$ .

**Spørgsmål** der trænger sig på:

- Er alle sprog regulære? Nej. Se afsnit 1.4 (lektion 4)
- Findes der andre måder at frembringe regulære sprog på?  
Ja. F.x. endelige automater; se afsnit 1.1 (lektion 2)
- Findes der også måder at frembringe ikke-regulære sprog på?  
Ja. F.x. kontekstfrie grammatikker; se afsnit 2.1 (lektion 5)
- Kan alle sprog så beskrives vha. disse metoder?  
Nej, langt fra. Se afsnit 2.3 (lektion 7)
- Hvad gør vi så? Venter på B&K-kurset næste semester