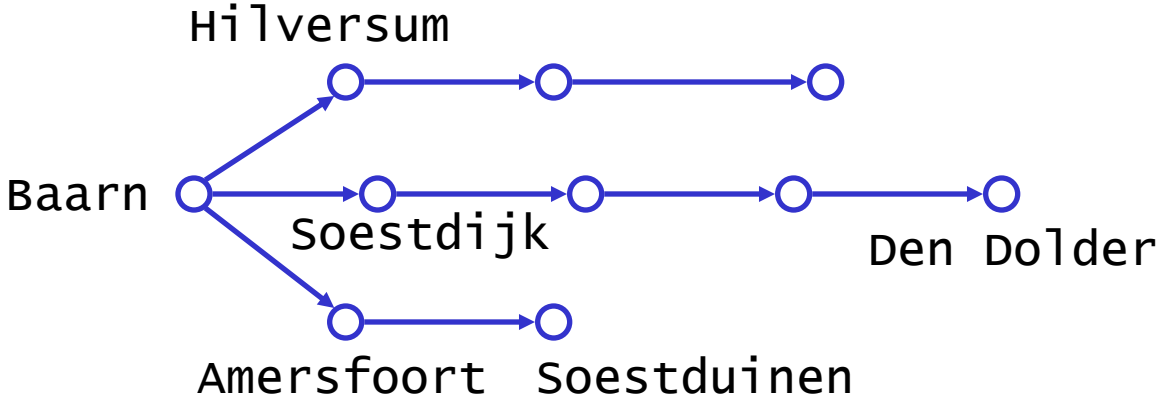
10 Bomen

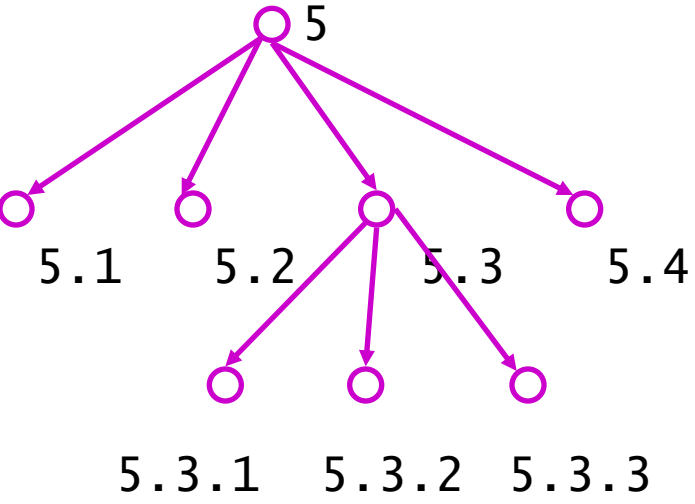
§8.8 ongerichte bomen
§9.4 gerichte bomen
ch 10. binaire bomen

voorbeelden



route boom

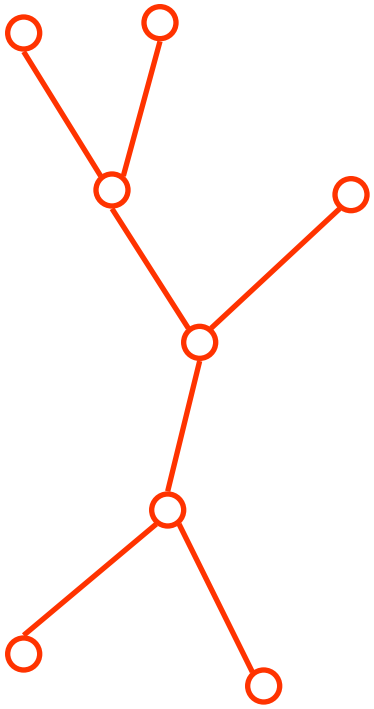
beslisboom



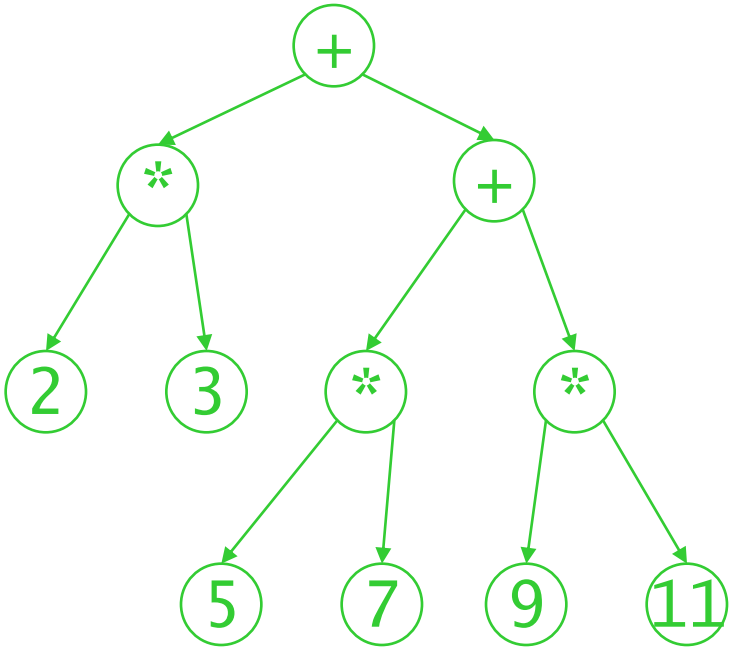
menuboom

decimal tree

gegevensboom



molecuul

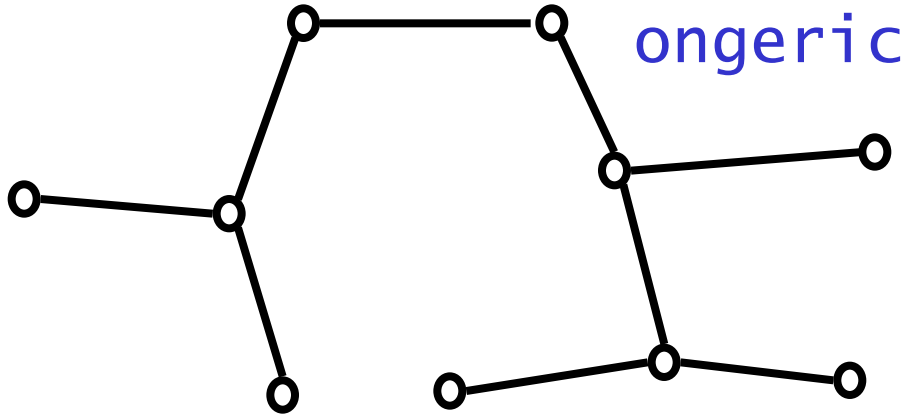


(2*3)+((5*7)+(9*11))

berekenboom

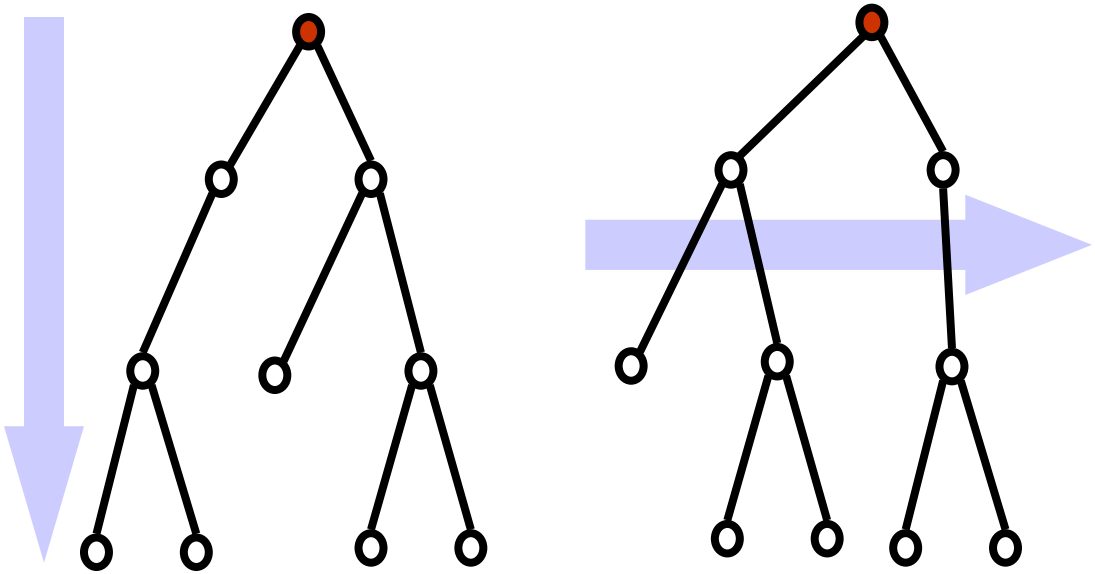
syntactische boom

arboretum



ongericht

§8.8 tree graphs
§9.4 rooted trees
ch.10 binary trees



gericht

geordend

links/rechts

definitie

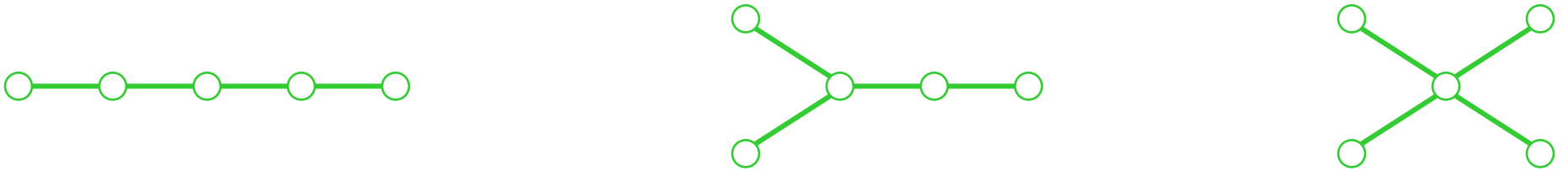
een *boom* is een samenhangende ongerichte graaf zonder cykels;

§8.8 tree graphs

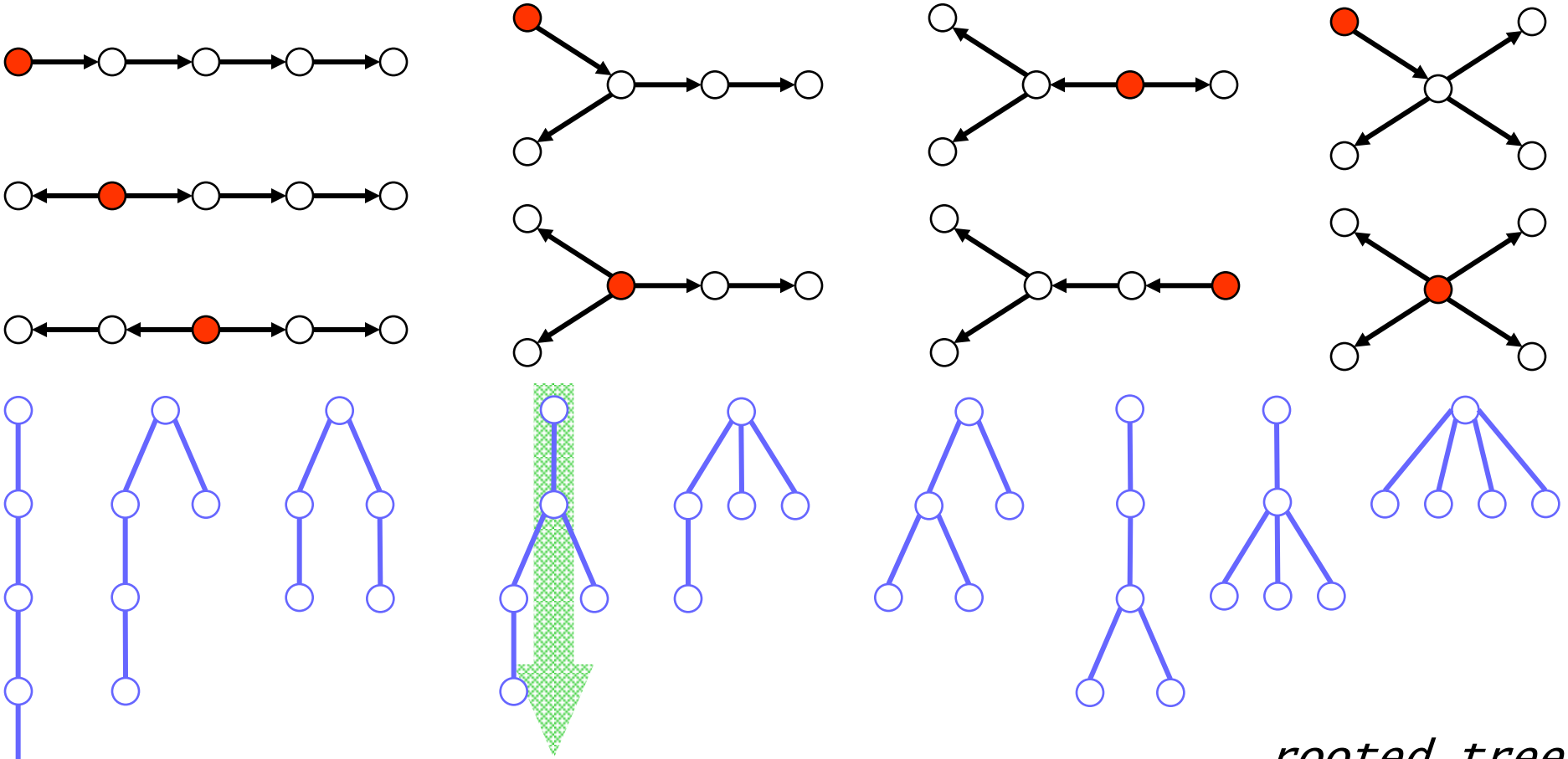
een *gewortelde boom* heeft een speciaal punt (de *wortel*), en daarmee een *richting* (vanuit de wortel)

§9.4 rooted trees

8.8 (ongerichte) boom:
samenhangende graaf zonder cykels

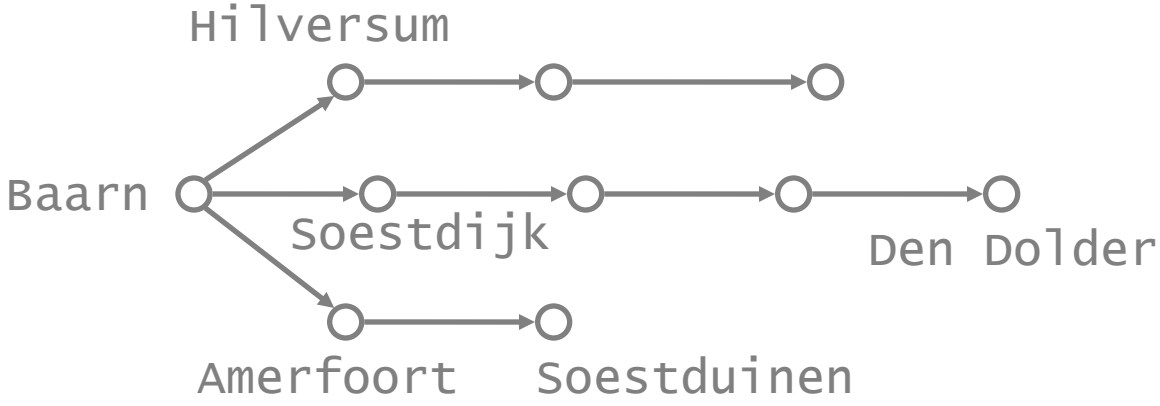


9.4 (gerichte) boom: kies wortel

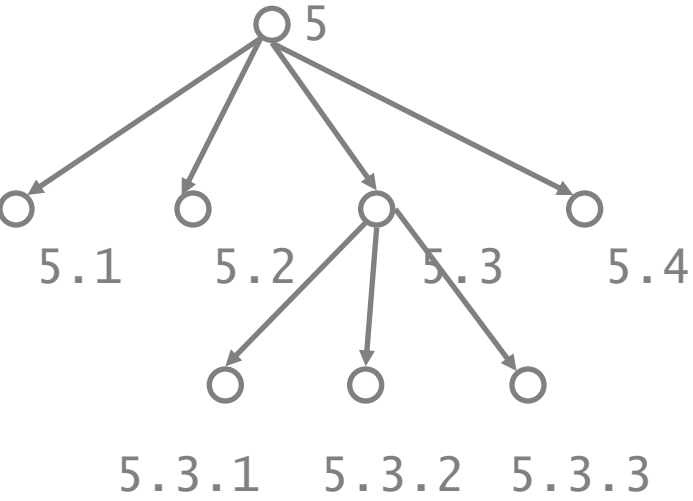


rooted tree

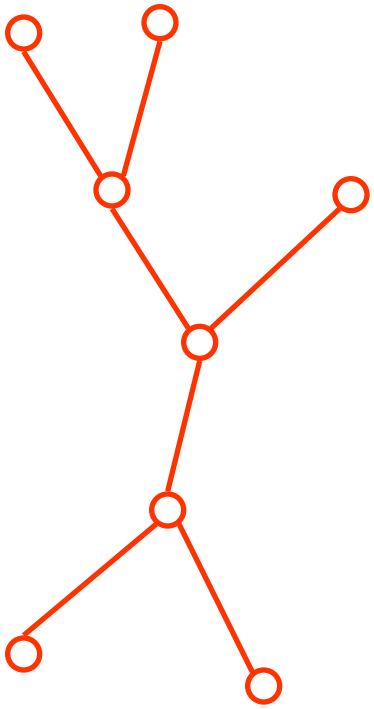
(ongericht) voorbeelden



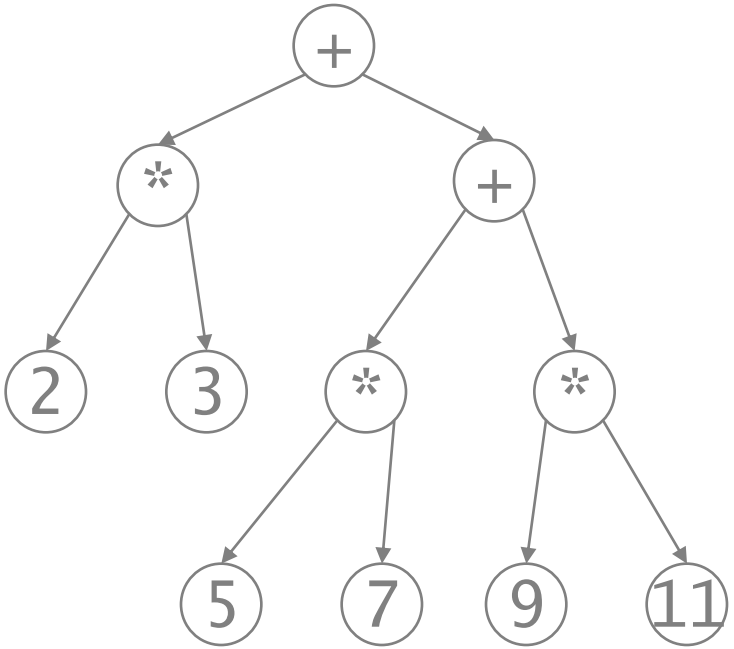
route boom
beslisboom



menuboom
decimal tree
gegevensboom



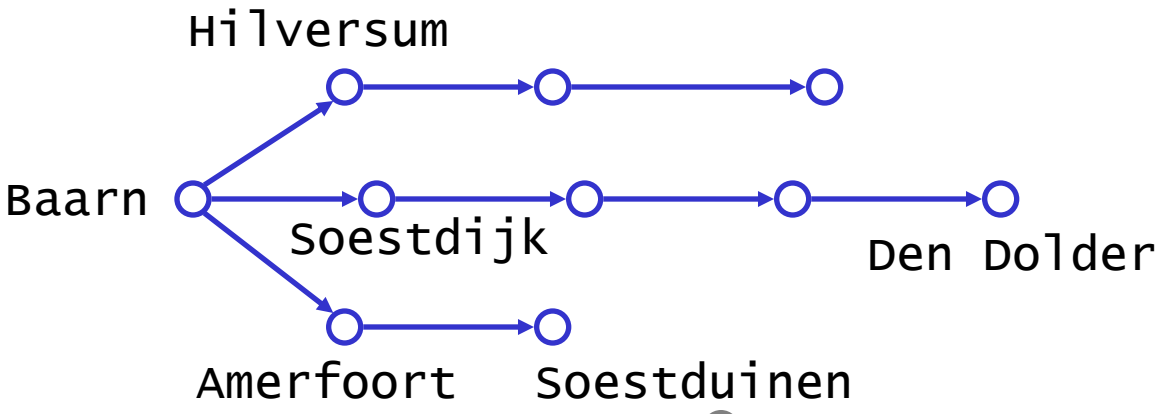
molecuul



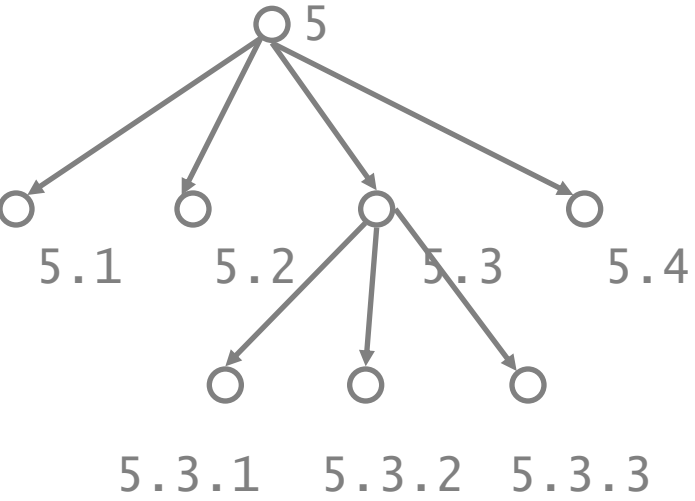
(2*3)+((5*7)+(9*11))

berekenboom
syntactische boom

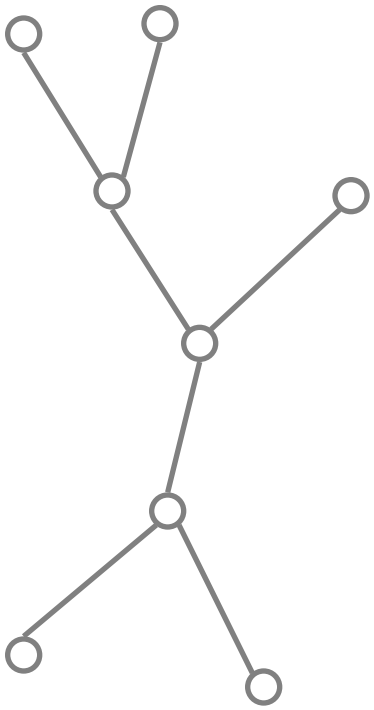
(gericht, ongeordend) voorbeelden



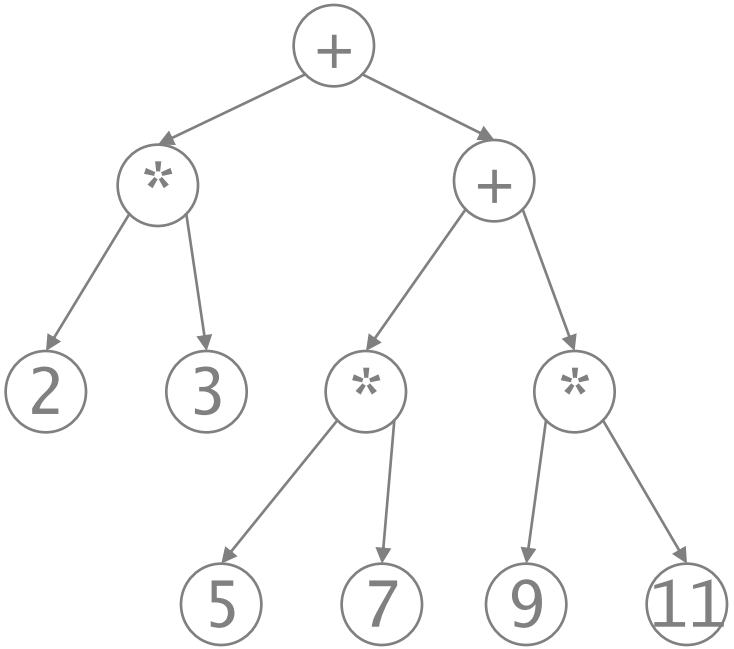
route boom
beslisboom



menuboom
decimal tree
gegevensboom



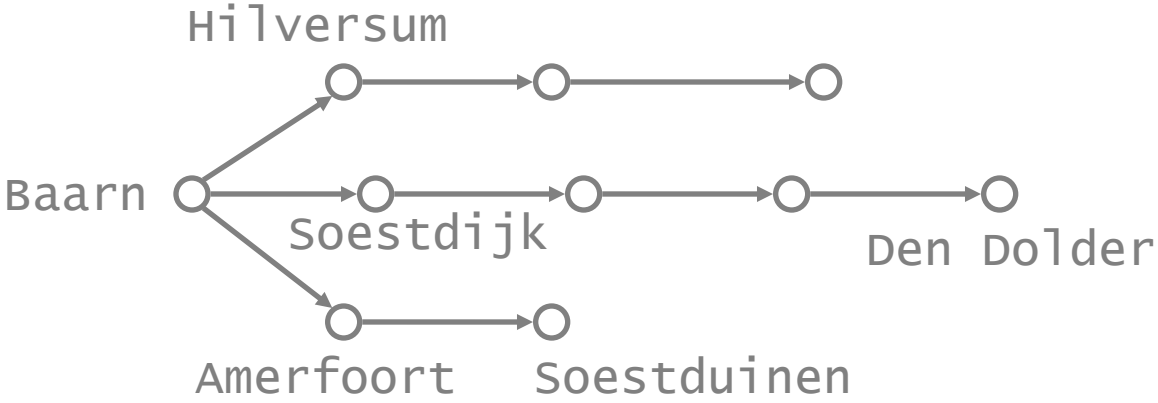
molecuul



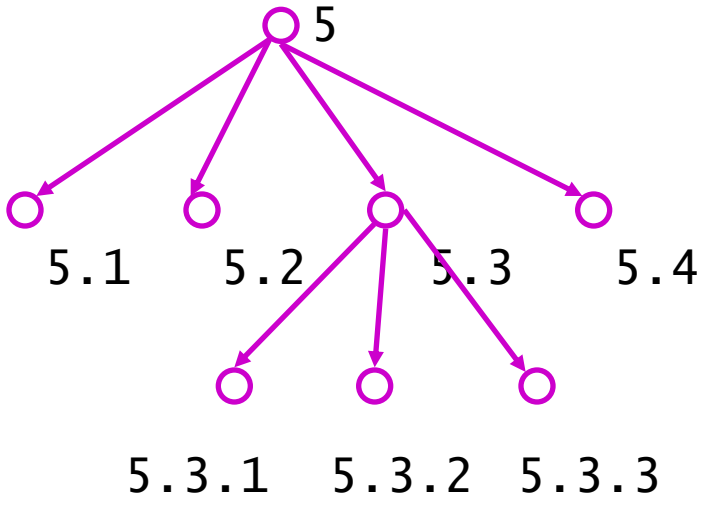
(2*3)+((5*7)+(9*11))

berekenboom
syntactische boom

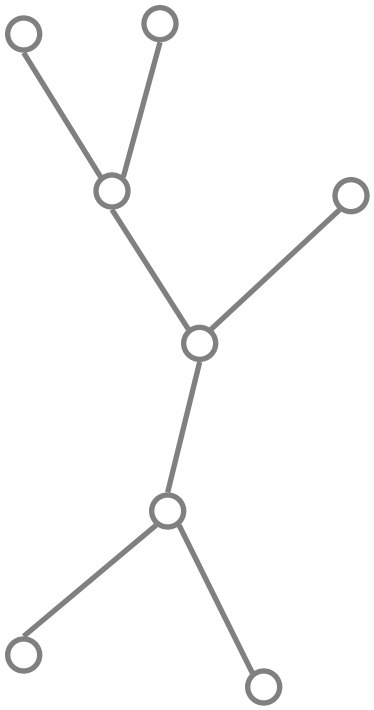
(gericht, geordend) voorbeelden



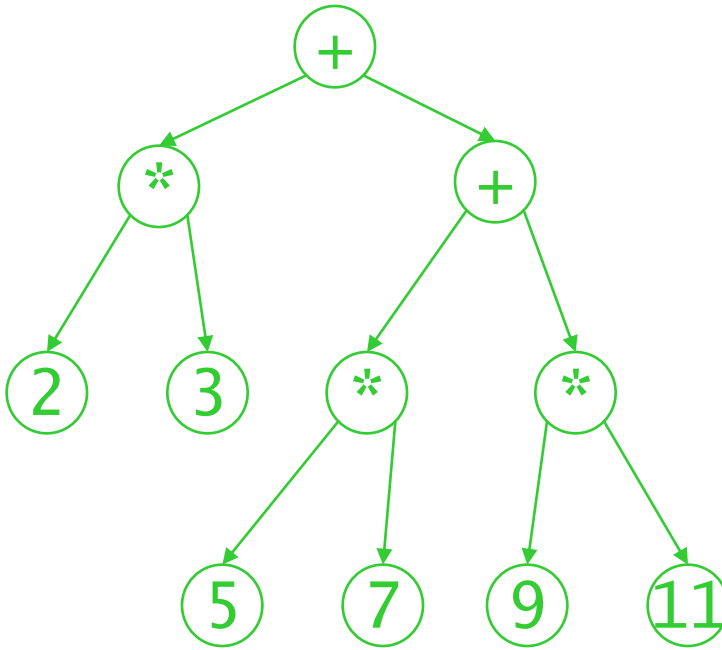
route boom
beslisboom



menuboom
decimal tree
gegevensboom



molecuul



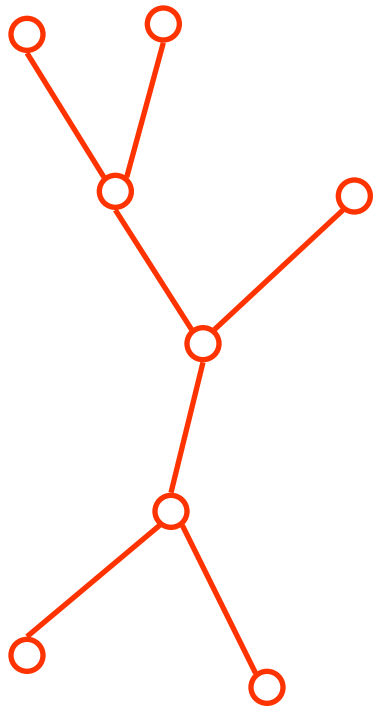
(2*3)+((5*7)+(9*11))

berekenboom
syntactische boom

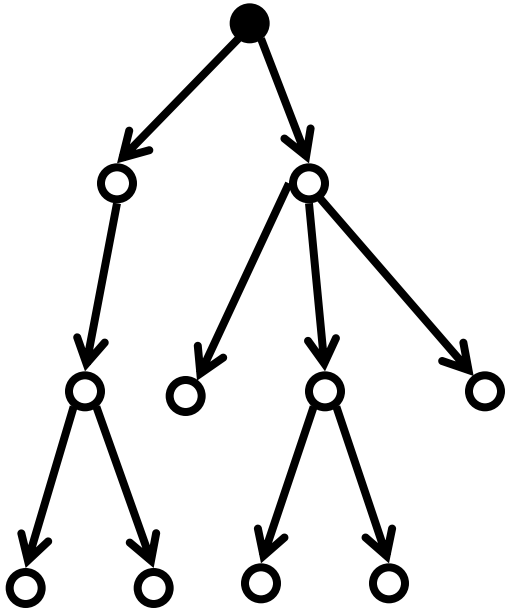
bomen: begrippen

terminologie

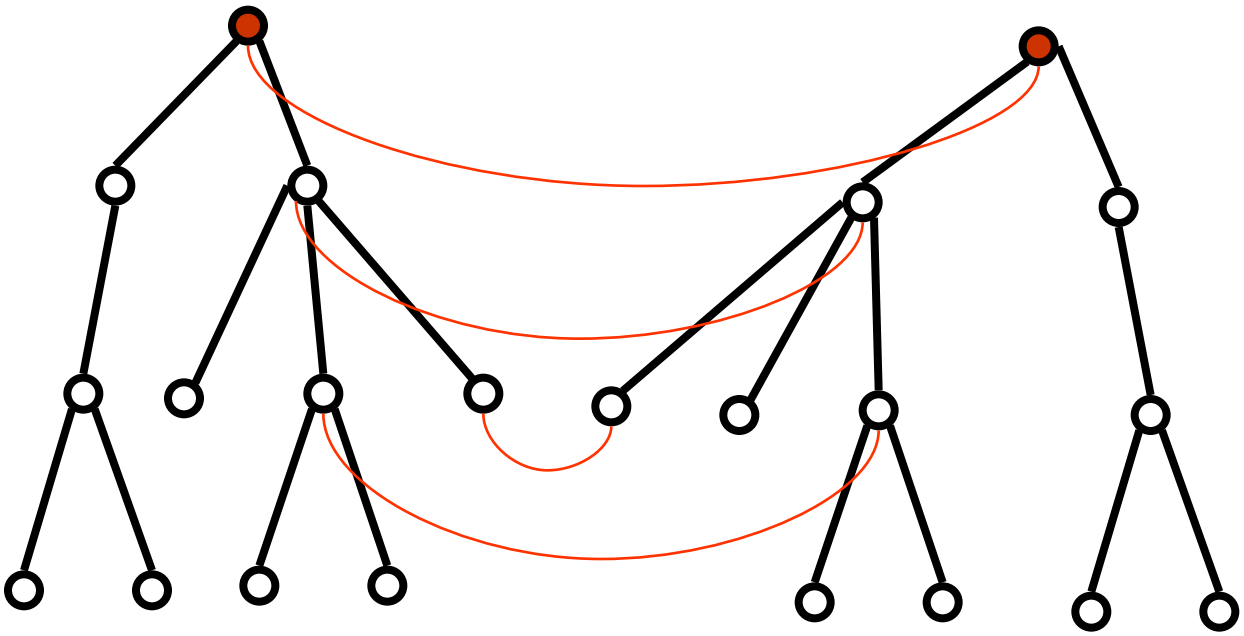
blad	eindpunt
intern	inwendig
tak	lijn



wortel	beginpunt
kind	benedenbuur
vader	ouder bovenbuur
broer	
nakomeling	opvolger
voorouder	voorganger



terminologie



(met wortel, ongeordend)

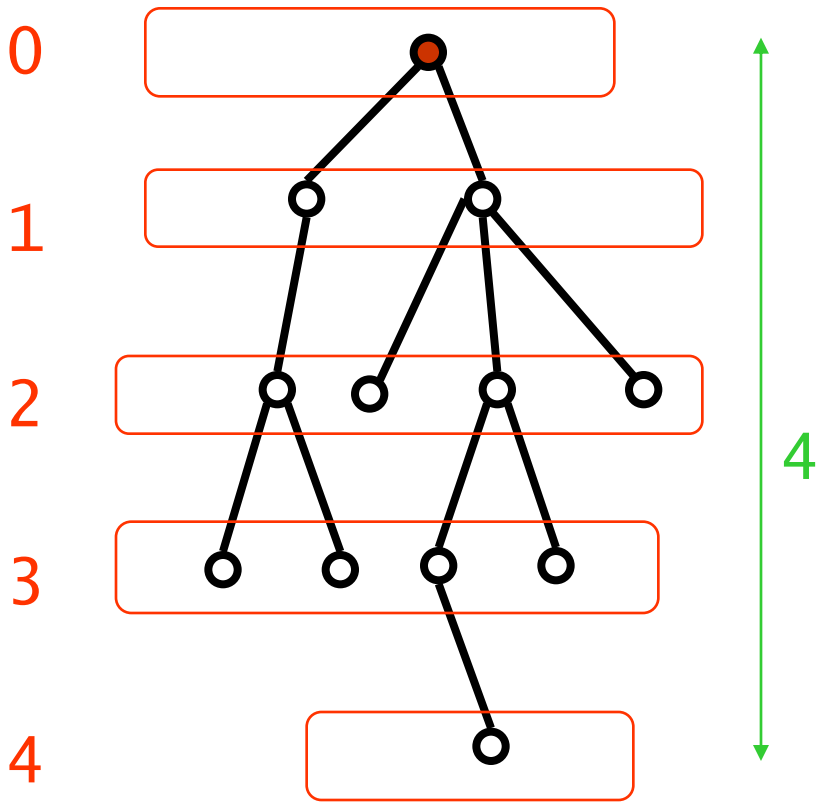
gelijkheid
(isomorfie)

relatie behoudt
- takken
- wortel
- ordening

waar van toepassing

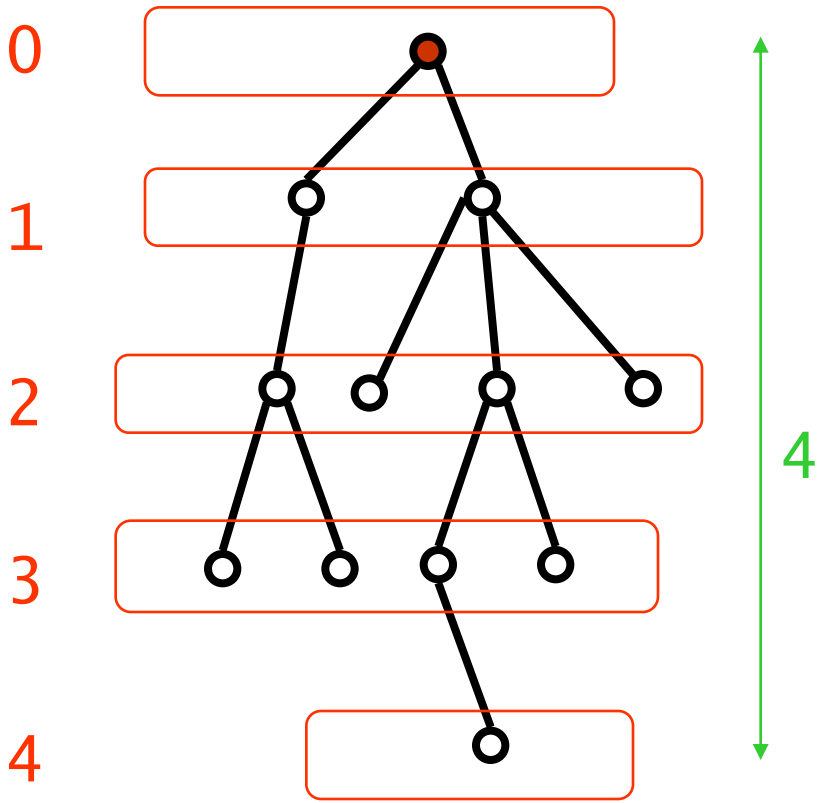
nivo's

diepte / hoogte
nivo *level*



nivo's

diepte / hoogte
nivo *level*

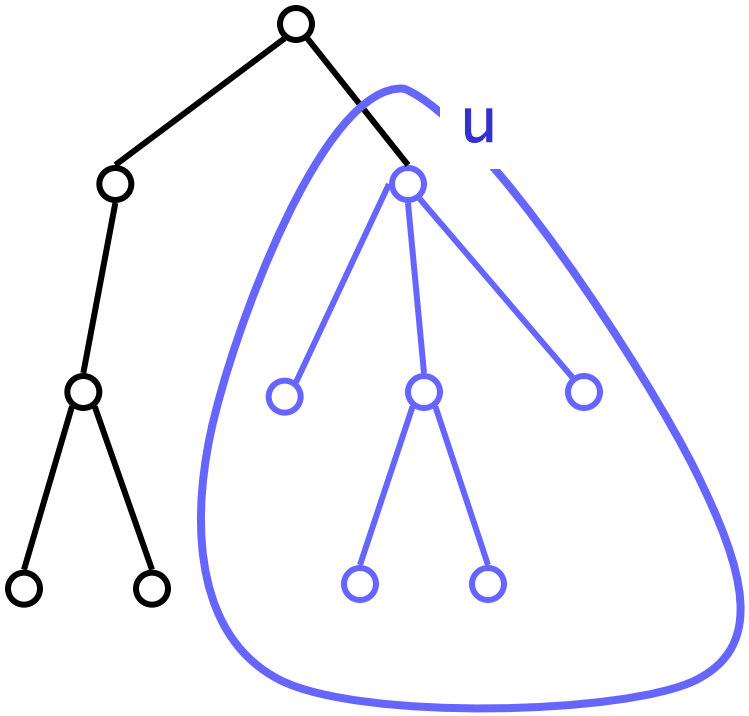


technisch is er verschil tussen diepte en hoogte: je kunt van elke knoop kijken hoe ver die van de wortel af ligt, maar ook hoe ver van het verste blad.

hier reken ik vanaf de wortel, en anders zeg ik het erbij.

deelboom

Zij $T = (V,A)$ een boom en u een punt in T. Dan is T_u de *deelboom* van T bestaande uit u, al zijn nakomelingen en alle pijlen tussen deze punten.



bomen: eigenschappen

definitie

een *boom* is een samenhangende ongerichte graaf zonder cykels;

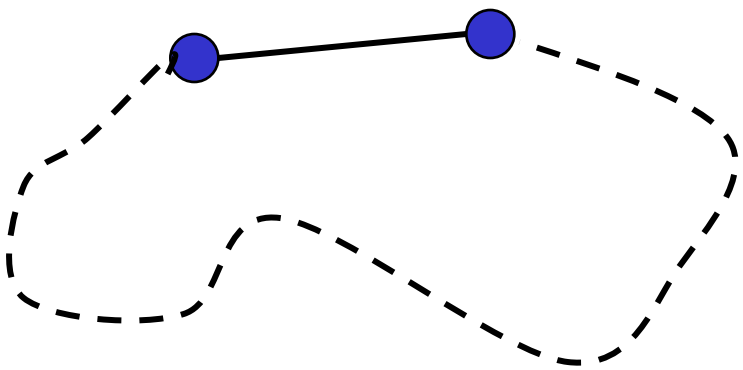
§8.8 tree graphs

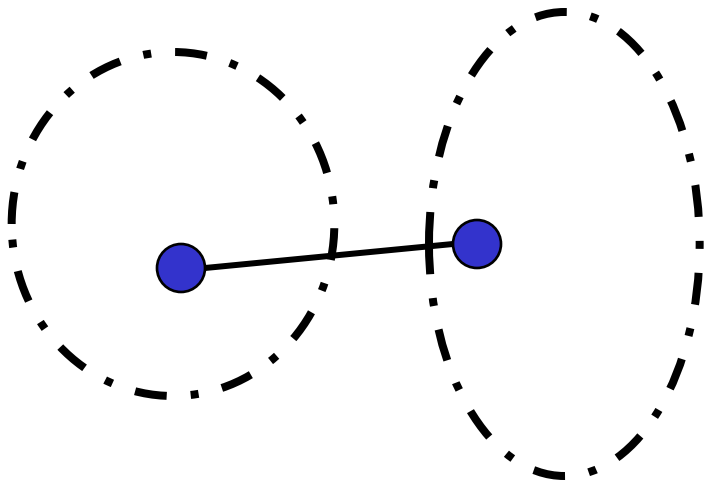
een *gewortelde boom* heeft een speciaal punt (de *wortel*), en daarmee een *richting* (vanuit de wortel)

§9.4 rooted trees

karakterisatie

1. (ongerichte) boom
   samenhangend, acyclisch
2. maximaal acyclisch
   lijn erbij ⇒ cykel
3. minimaal samenhangend
   lijn weg ⇒ onsamenhangend

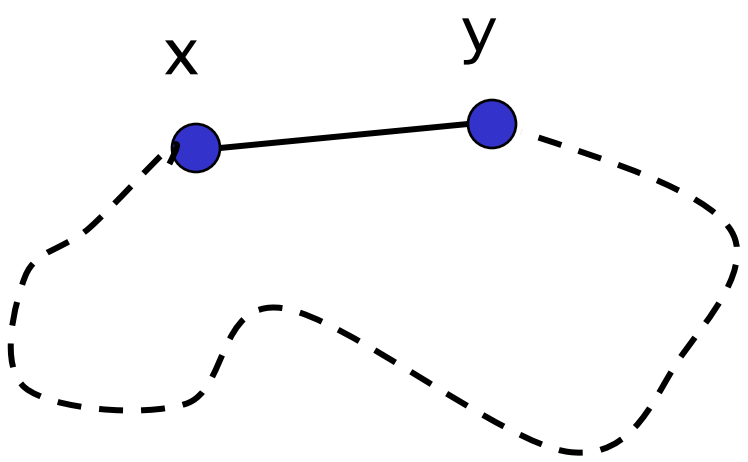




(Exercise 8.13)

bewijs 1⇔2

1. boom
   samenhangend, acyclisch
2. maximaal acyclisch
   lijn erbij ⇒ cykel
3. minimaal samenhangend
   lijn weg ⇒ onsamenhangend



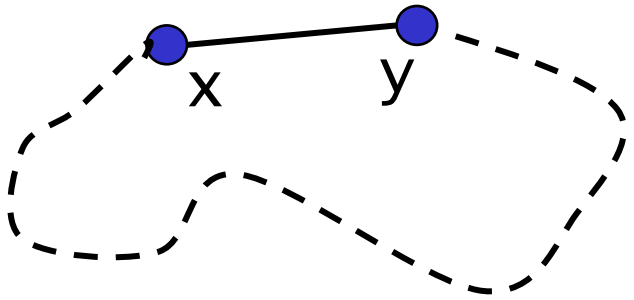
1⇒2
maximaal
2⇒1
samenhangend

bewijs 1⇔2

1. boom — samenhangend, acyclisch
2. maximaal acyclisch — tak erbij ⇒ cykel

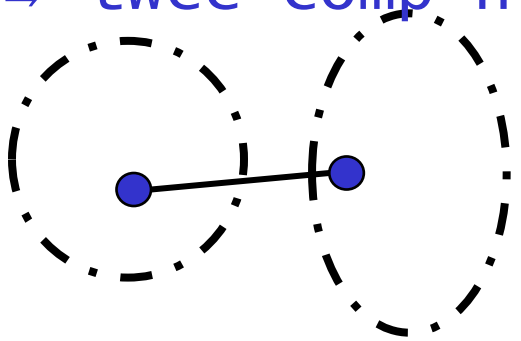
1⇒2 voeg lijn {x,y} toe aan G. omdat G samenhangend is bestond er al een pad tussen x en y. samen met de lijn ontstaat een cykel.

2⇒1 we moeten nog laten zien dat G samenhangend is. kies willekeurige x en y, en we beredeneren dat x en y verbonden zijn. als er een lijn tussen x en y ligt, dan zijn ze verbonden. als er geen lijn is voeg die dan toe. volgens gegeven ontstaat een cykel. dat betekent dat er al een pad tussen x en y moest bestaan.



bewijs 1⇔3

1. boom — samenhangend, acyclisch
3. minimaal samenh. — lijn eraf ⇒ twee comp'n



1⇒3 verwijder lijn {x,y} uit G. nu is G niet langer samenhangend, want x en y zijn niet meer verbonden via pad: anders zou er oorspronkelijk een cykel geweest zijn.

3⇒1 we moeten nog laten zien dat G acyclisch is. stel G bevat wél een cykel. verwijderen van een lijn in de cykel levert nog steeds een samenhangende graaf, dus de graaf is dan niet minimaal samenhangend. (tegenspraak). G heeft dus geen cykels.

stelling

Het aantal takken van [ongerichte] boom $T = (V,E)$ is één minder dan het aantal knopen van T: $|E| = |V|-1$.

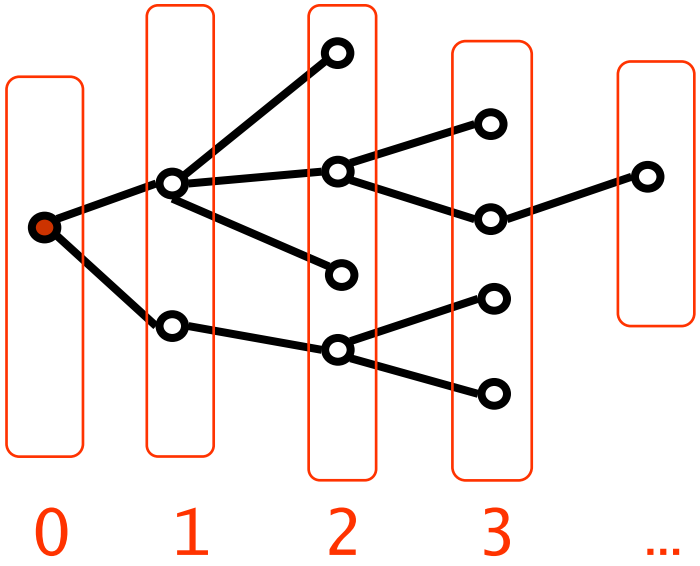
V(ertices)
E(dges)

stelling

Het aantal takken van [ongerichte] boom T = (V,E) is één minder dan het aantal knopen van T: |E| = |V|-1.

V(ertices)
E(dges)

bewijs (één)



kies willekeurig beginknoop, dan alle buren, etc.
acyclisch, we vinden nooit eerdere bezochte knopen
elke knoop heeft één inkomende tak, behalve …

stelling

Het aantal takken van boom $T = (V,E)$ is één minder dan het aantal knopen van T:
$|E| = |V|-1$.

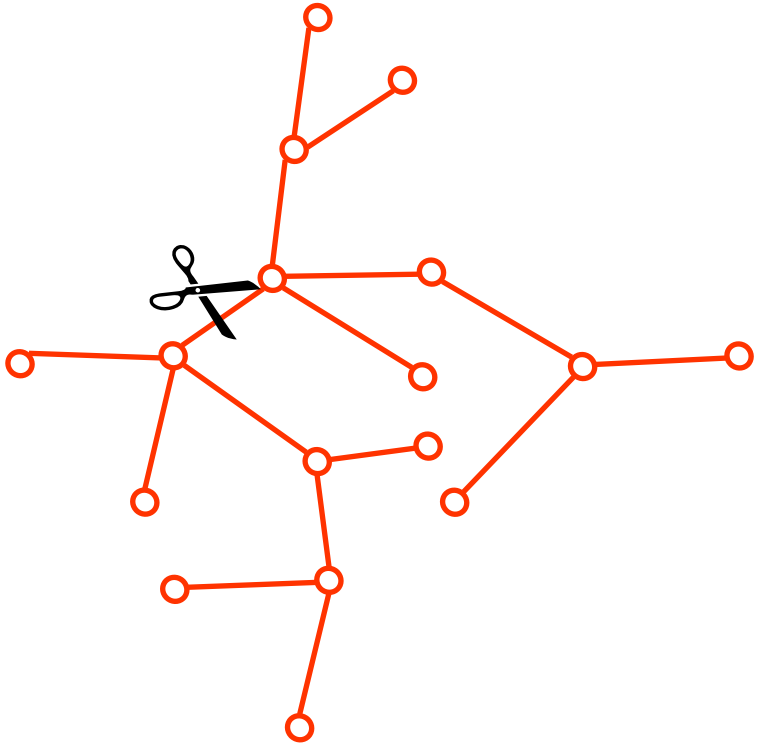
bewijs (*alternatief*)

- basis: enkele knoop

- inductiestap
  $|E_1| = |V_1|-1$
  $|E_2| = |V_2|-1$

$E=E_1 \cup E_2 \cup \{t\}$ $V=V_1 \cup V_2$

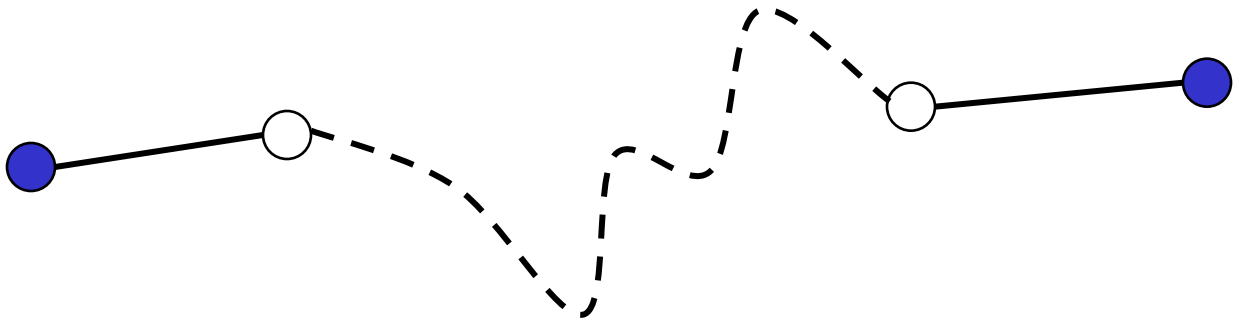
optellen:
$|E|-1 = |V|-2$



(Exercise 8.38)

een (samenhangende) graaf zonder cykels met ten minste één tak heeft ten minste twee knopen van graad 1

bewijs

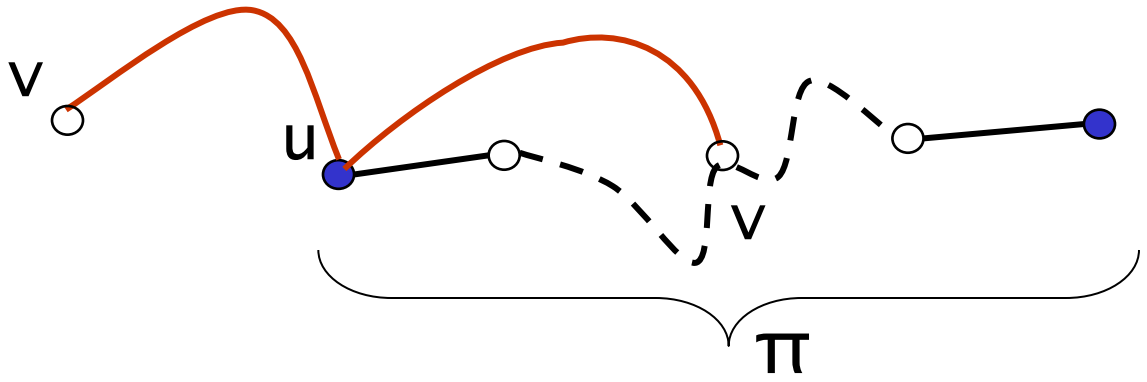


maximaal simpel pad

alternatief: graden tellen (som van de graden is …)

bewijs Ex.8.38

een samenhangende graaf zonder cykels met ten minste één tak heeft ten minste twee knopen van graad 1



bewijs.
neem een maximaal simpel pad π in graaf G en laat u een eindpunt van π zijn. heeft knoop u nog andere buren? nee: want dan zou òf het pad langer worden, òf zou er een cykel ontstaan. dus: u heeft graad 1

□

alternatief bewijs Ex.8.38

een samenhangende graaf zonder cykels met ten minste één tak heeft ten minste twee knopen van graad 1

graden tellen: *bewijs dmv. tegenspraak* 'contradictie'

neem aan: G=(V,E) heeft max één knoop van graad 1 (en geen knopen van graad 0)

Laat σ som van de graden van G, dan geldt

(1) ten minste $\sigma \geq 1 + 2(|V|-1)$

(2) stelling $\sigma = 2|E|$

(3) voor bomen $|E| = |V|-1$

$2|E| = \sigma \geq 1 + 2(|V|-1) = 1+2|E|$ tegenspraak

dus moeten er (ten minste) twee knopen van graad 1 zijn

karakterisatie

*definitie: een *boom* is een samenhangende ongerichte graaf zonder cykels

*eigenschap: T = (V,E) dan |E| = |V|-1

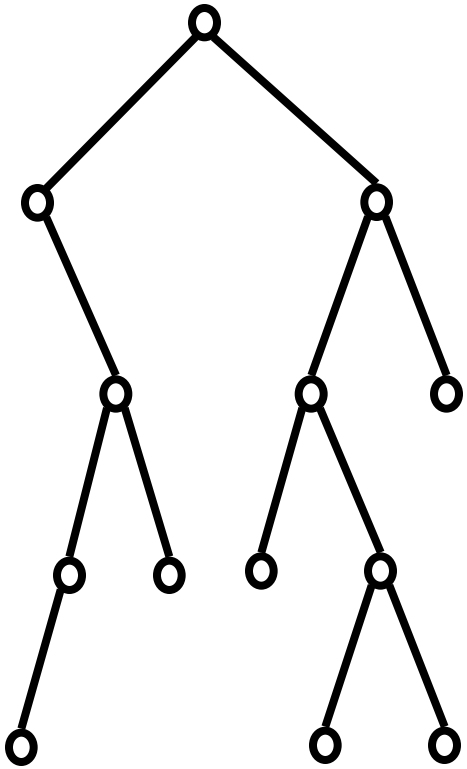
Theorem 8.6 (Exercise 8.14)

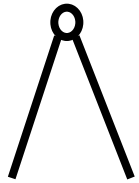
G is een graaf met n knopen.
De volgende beweringen zijn equivalent:

- G is een boom (samenhangend, zonder cykels)
- G is zonder cykels, heeft n-1 takken
- G is samenhangend, heeft n-1 takken

boom, kies twee uit:

- G is zonder cykels
- G is samenhangend
- G heeft n-1 takken





links/rechts

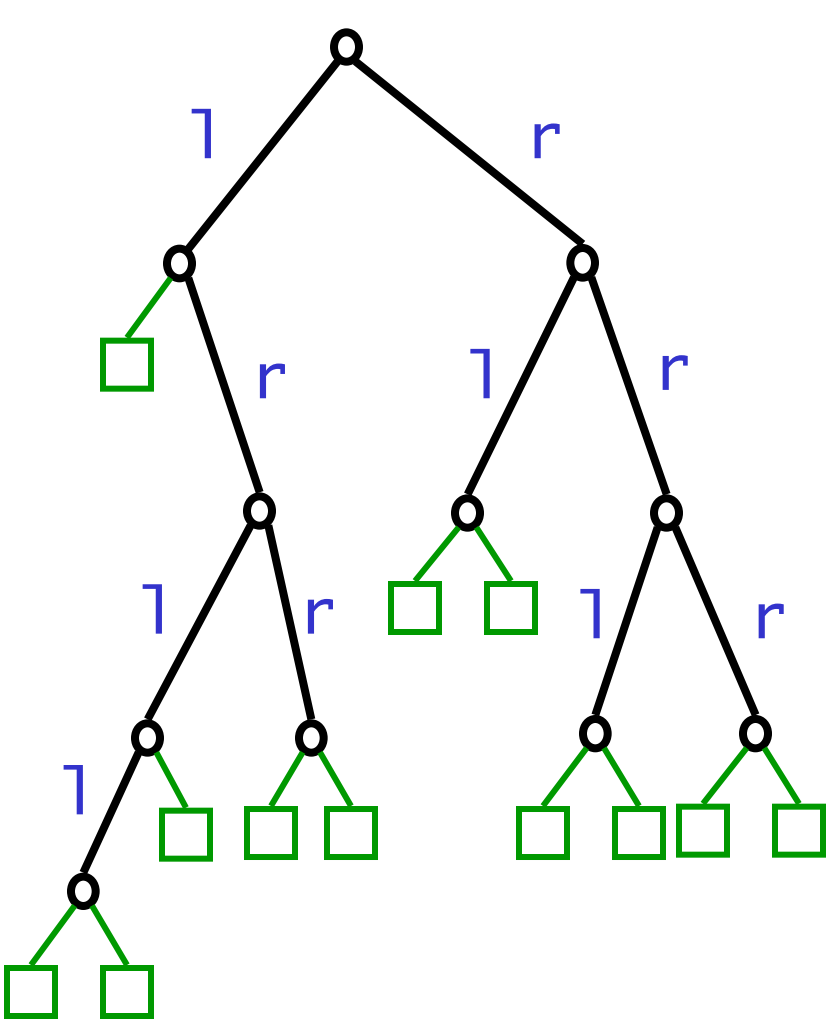
binair

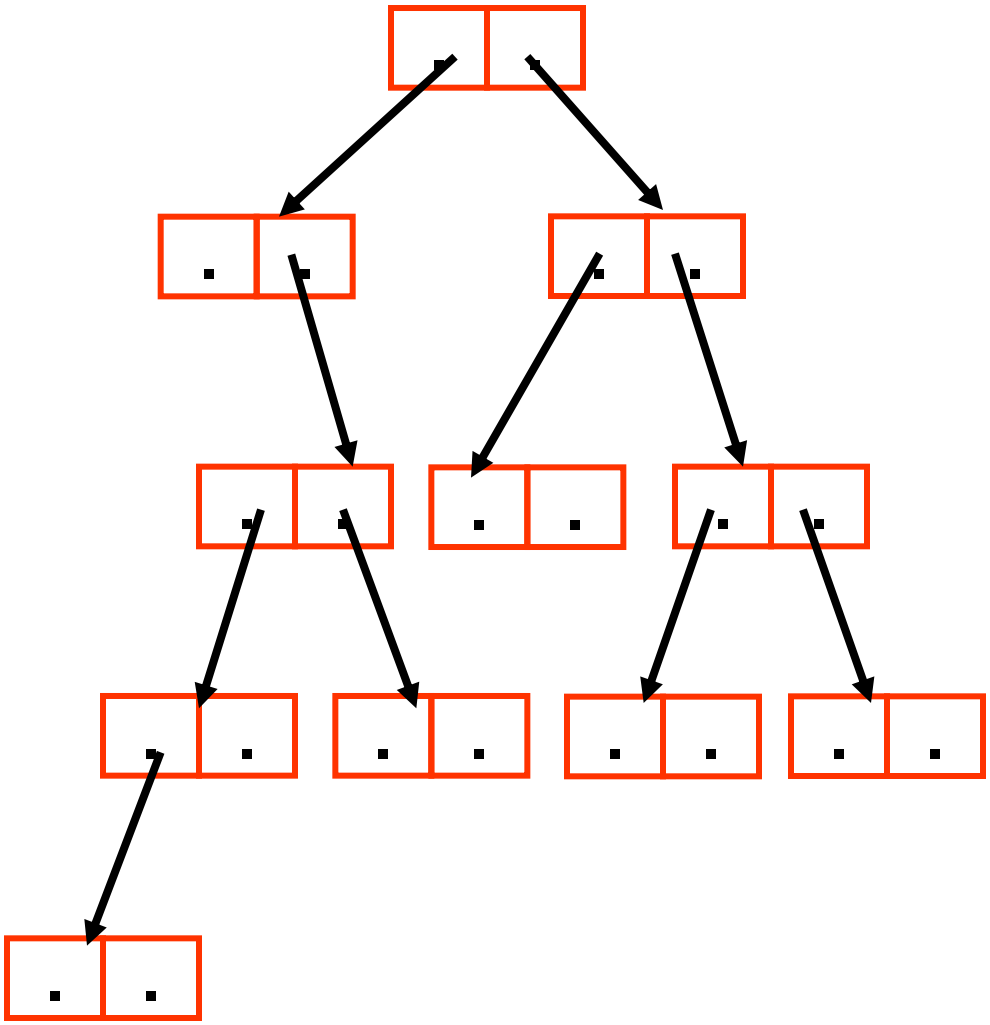
links en rechts is belangrijk in binaire bomen: zelfs als er maar één kind is maakt het uit of dat links dan wel rechts zit.

het begrip is natuurlijk binnen de informatica want past bij een standaard implementatie van bomen met pointers.

veelgebruikt, bv. binaire zoekbomen.

§10.4 binaire boom: pointer structuur

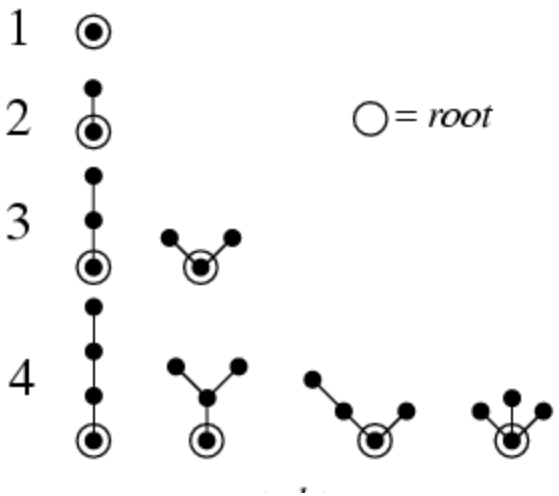




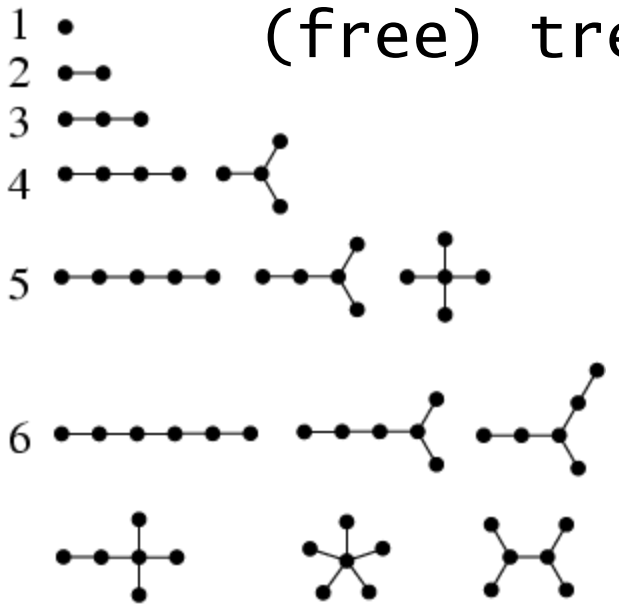
classificatie

tree (graph)	boom (als graaf) ongerichte boom
rooted tree	gewortelde boom gerichte boom (wortel naar bladeren)
ordered rooted tree	geordend en gericht (standaard boom ?)
binary tree	binaire boom links en rechts (informatica!)

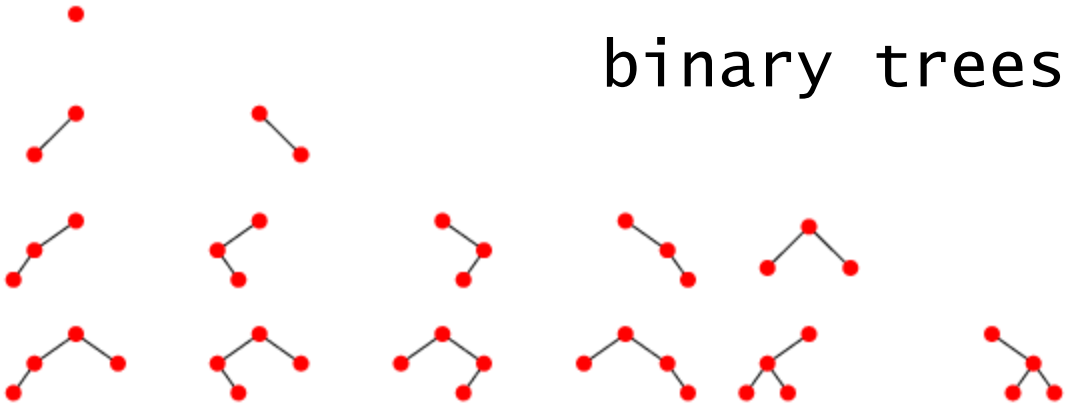
trees



rooted trees



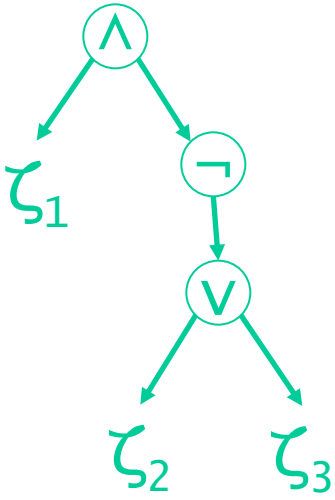
(free) trees



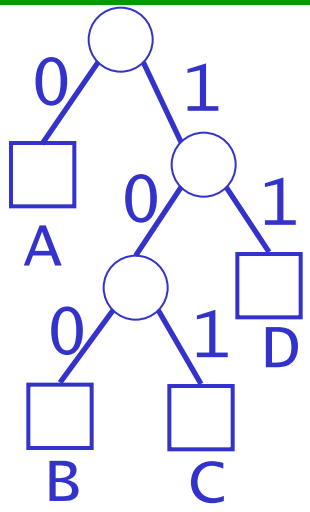
binary trees

http://mathworld.wolfram.com/

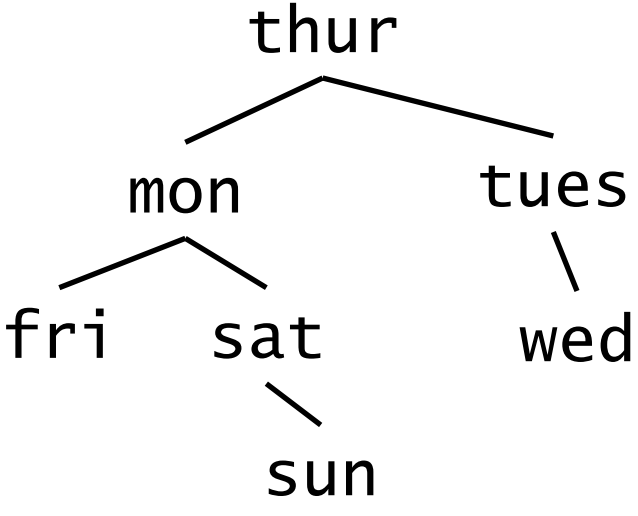
bomen, bomen, bomen …



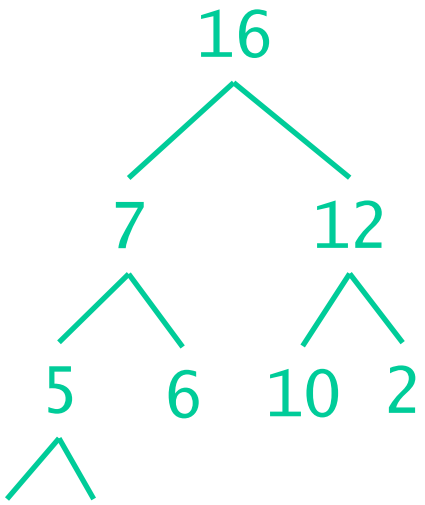
expressie



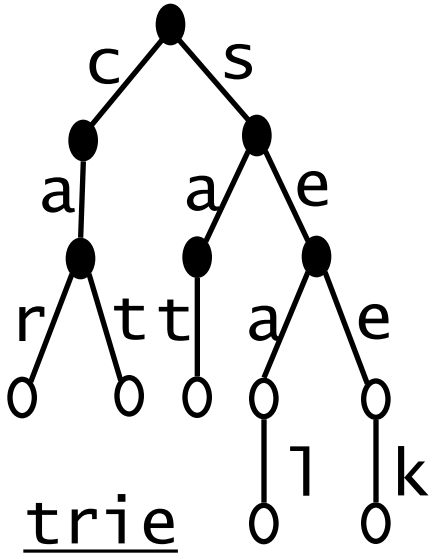
code



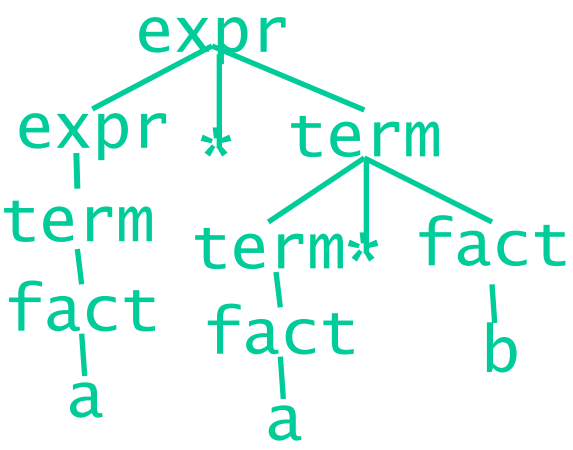
binaire zoek



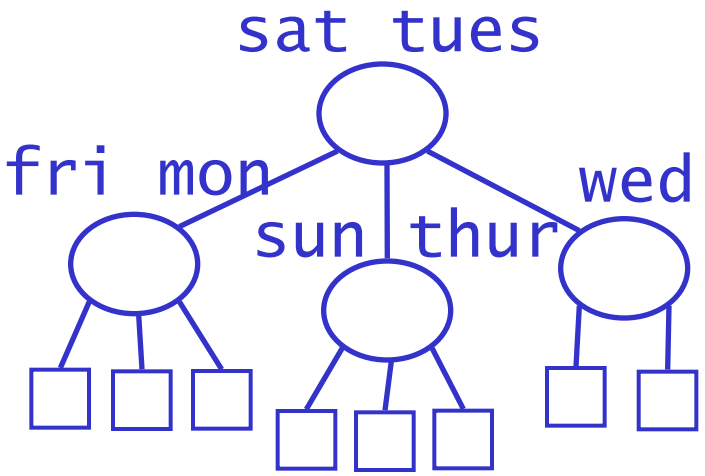
heap



trie

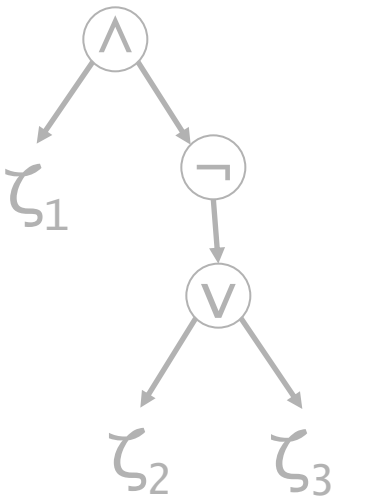


syntax

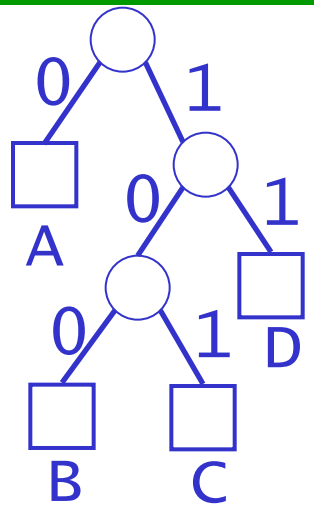


2,3 boom

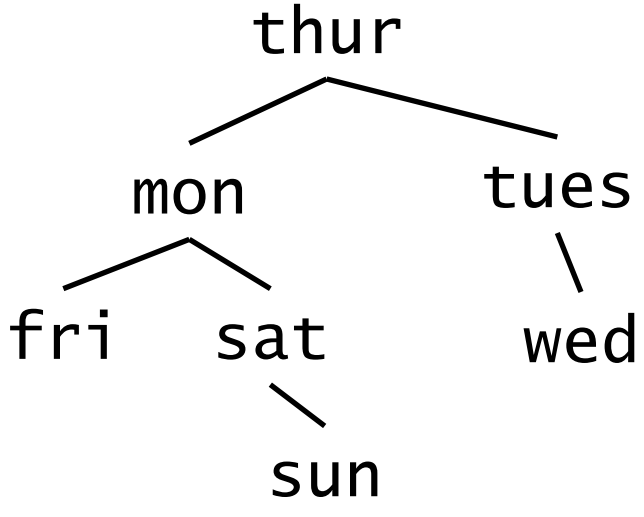
bomen, bomen, bomen …



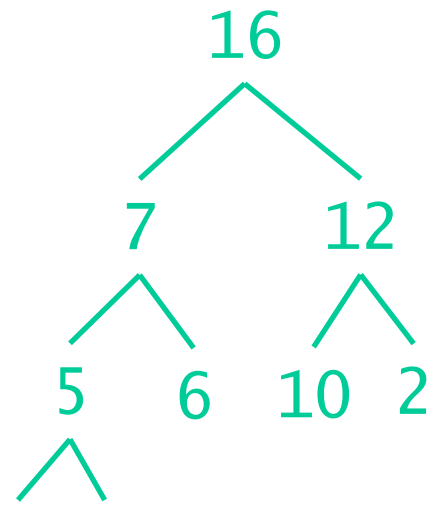
expressie



code

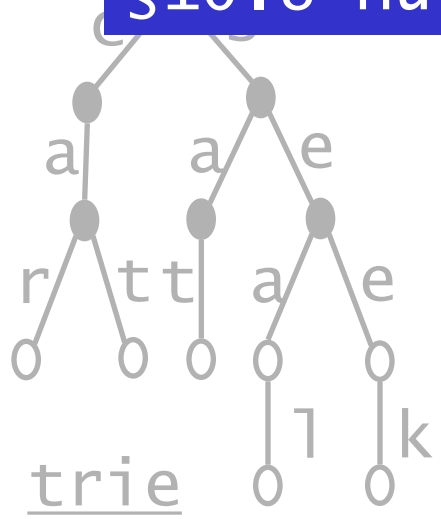


binaire zoek

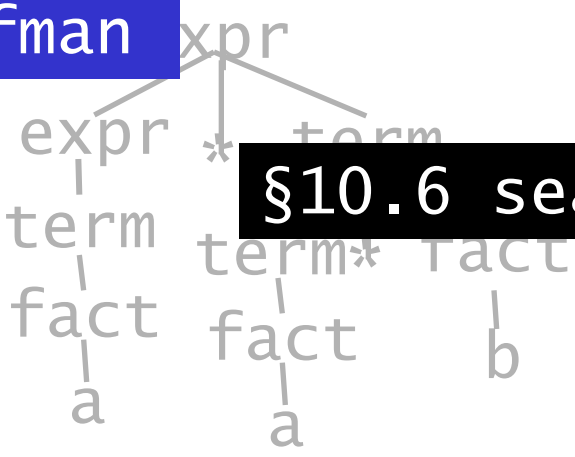


heap

§10.8 Huffman



trie



syntax

§10.7 heap

§10.6 search trees

2,3 boom

bomen

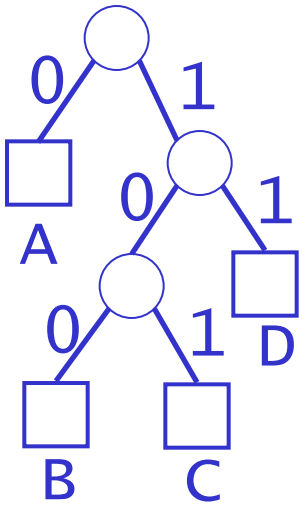
komen aan de orde bij de colleges Algoritmiek en Datastructuren.

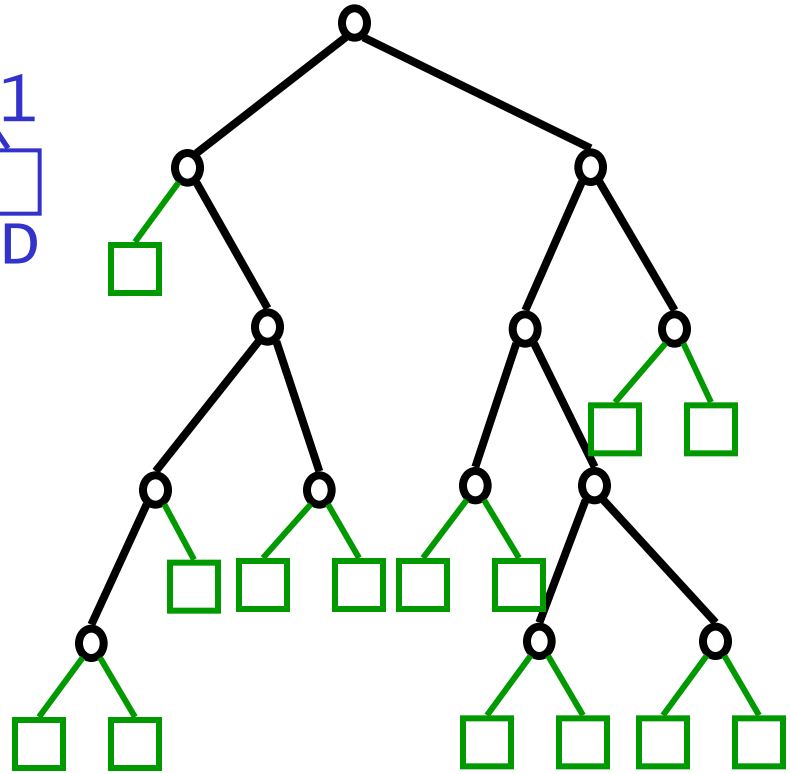
Huffman code: binaire representatie van letters zodat de boodschap minimale lengte heeft bij gegeven frequentie van de letters

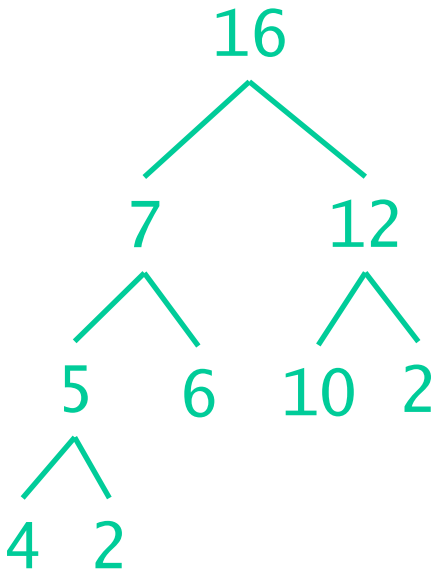
Zoekbomen: vind elk element, kleiner dan links, groter dan rechts

Heap: ordening op 'prioriteit' van boven naar beneden, om snel grootste element te vinden (en verwijderen). slimme algoritmen om elementen toe te voegen en te verwijderen

§10.3 mooie vormen







complete boom

2-boom (volle boom)

uitgebreide boom

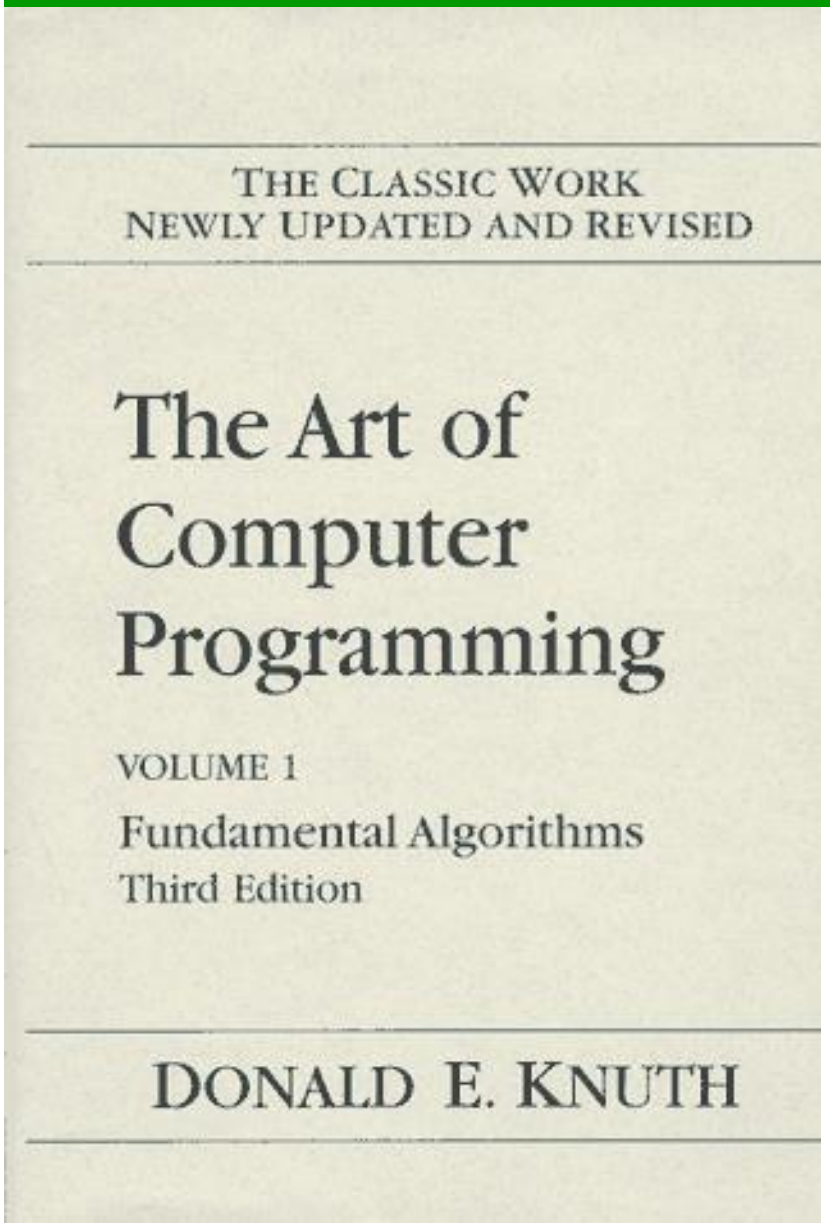
intern / extern

vormen

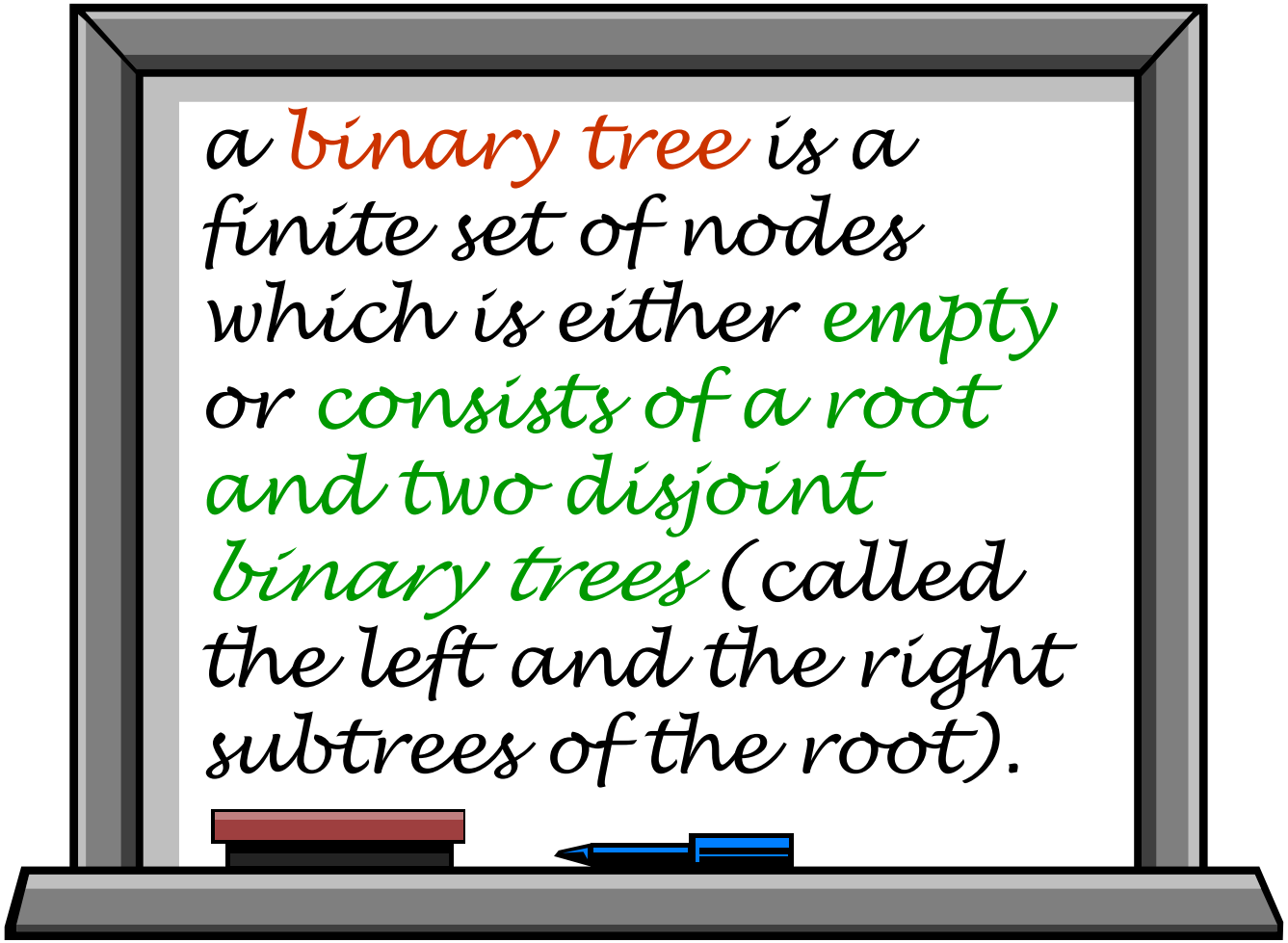
vol: élke knoop is óf een blad of heeft twee kinderen. Er zijn dus geen knopen met één kind. Elke boom kan vol gemaakt worden door bij elk ontbrekend kind een ‘speciaal’ blad aan te leggen. Ook de bladeren krijgen bladeren onder zich.
(dit lijkt op het aangeven van de NIL-pointers in een pointer implementatie)

compleet: voeg de kinderen nivo-voor-nivo toe, van links naar rechts.
(past bij een array implementatie, maar dat leert u later …)

Donald E. Knuth TAoCP



Knuth says:

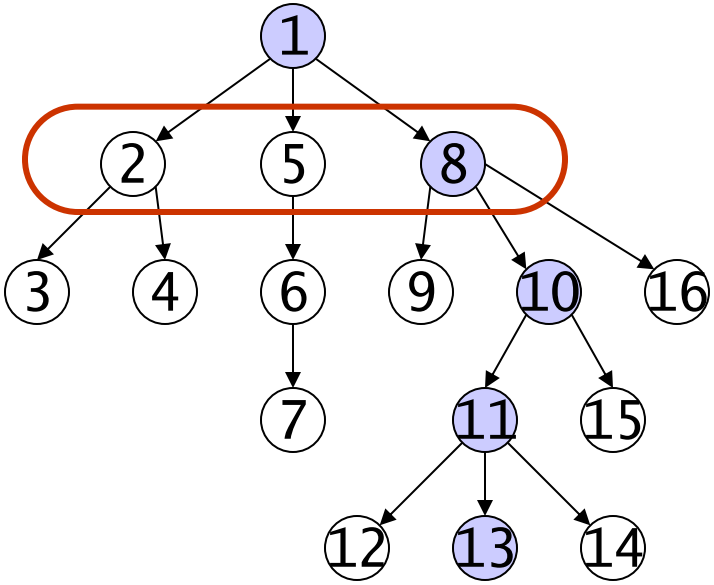


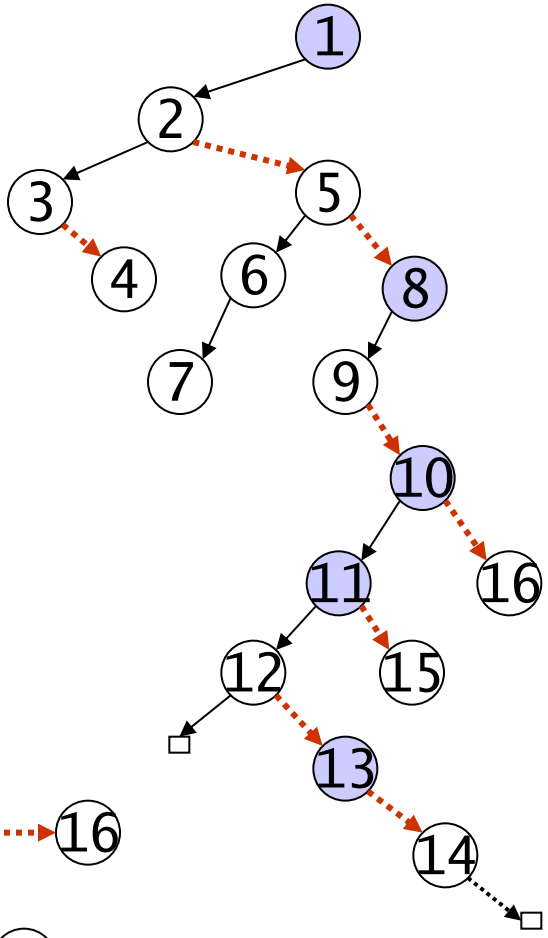
recursion!

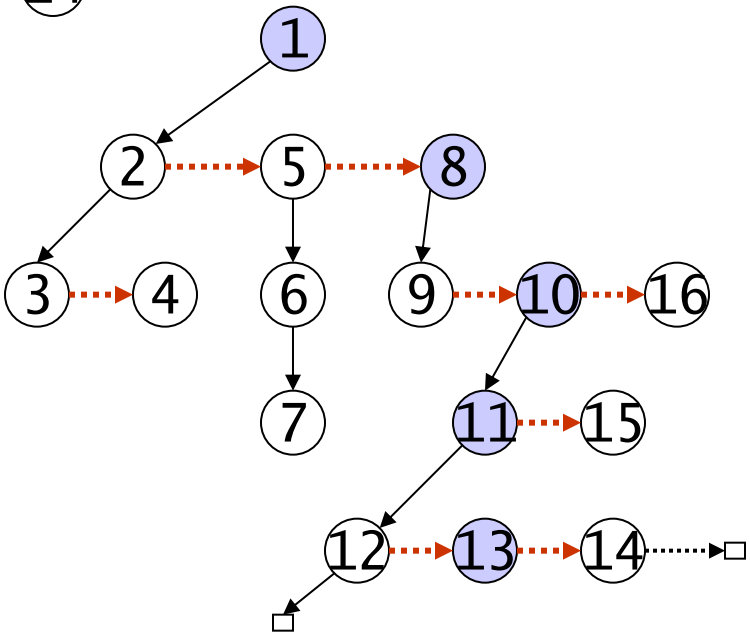
donald knuth. held.

(vraag me om te vertellen van de cheques …)

§10.9 eerste kind, rechter broer





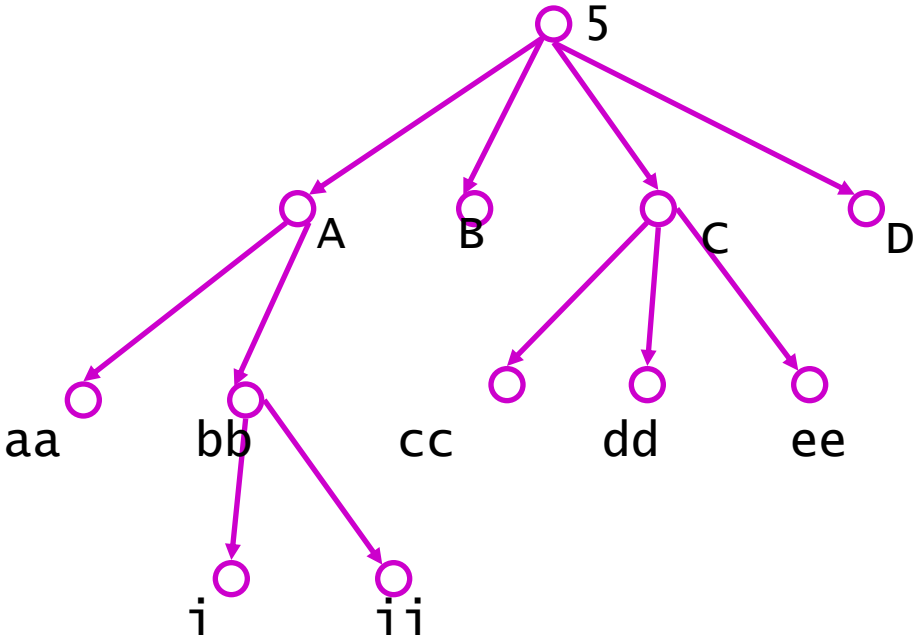


de eerste-kind rechter-broer (sorry, wij kennen geen onzijdig woord voor ‘sibling’, ‘brusje’ wordt soms geprobeerd…)
representatie vormt elke boom om in een binaire boom, links en rechts krijgen dan die nieuwe betekenis van kind en broer (resp.)

bomen: knopen ordenen

p.238 Polish notation
p.241 traversing binary trees

knopen ordenen

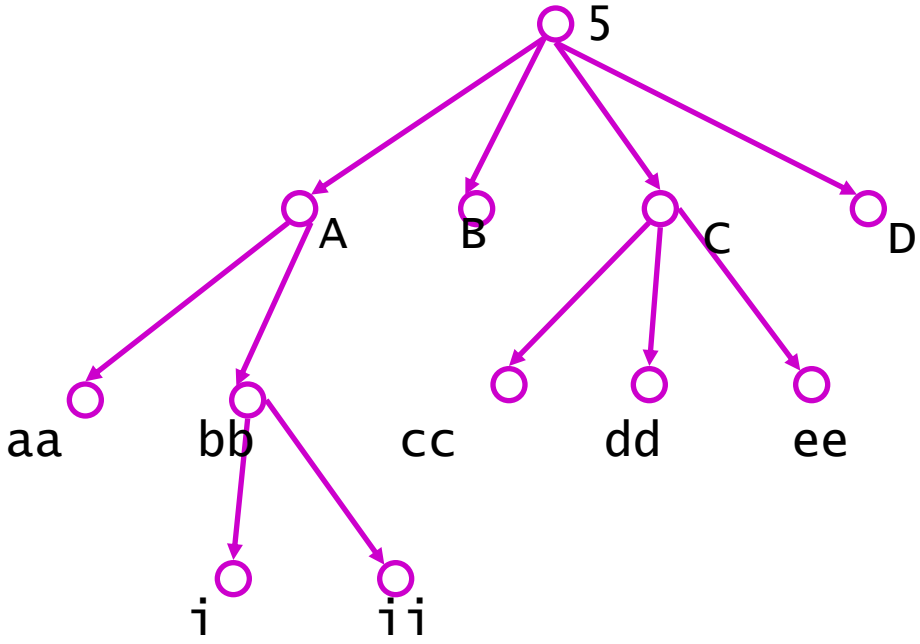


- 5
- A
  - aa
  - bb
    - i
    - ii
- B
- C
  - cc
  - dd
  - ee
- D

  - aa
    - i
    - ii
  - bb
- A
- B
  - cc
  - dd
  - ee
- C
- D

5

knopen ordenen



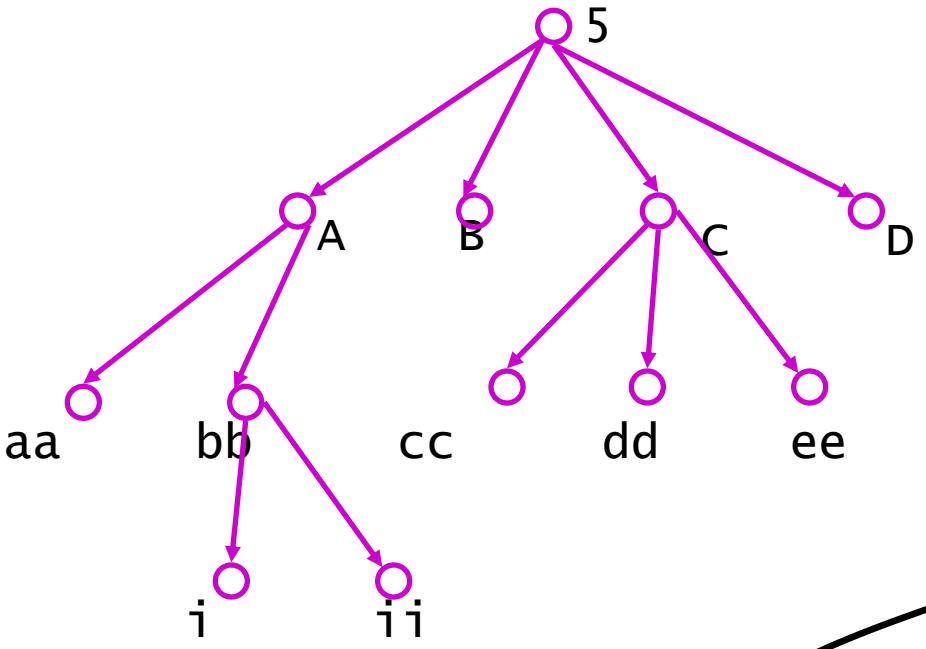
- 5
  - A
    - aa
    - bb
      - i
      - ii
  - B
  - C
    - cc
    - dd
    - ee
  - D

inhoudsopgave

- aa
  - i
  - ii
- bb
- A
- B
  - cc
  - dd
  - ee
- C
- D
- 5

inhoud tellen
(eerst de onderdelen)

preorde

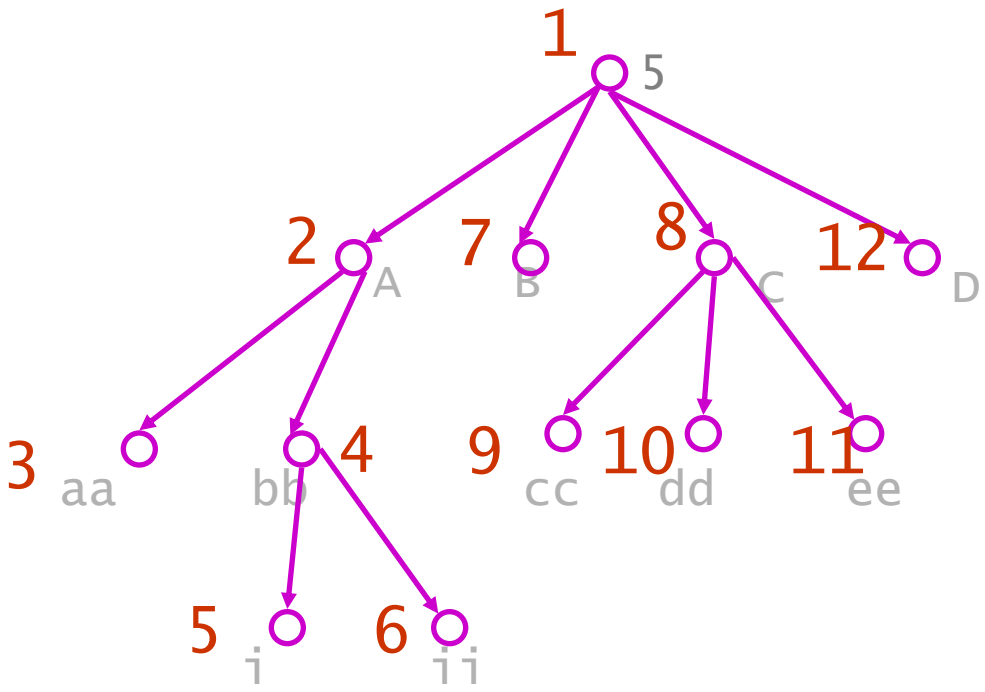


volgorde

- 5
  - A
    - aa
    - bb
      - i
      - ii
  - B
  - C
    - cc
    - dd
    - ee
  - D

‘eerst knoop, dan (subbomen van) kinderen’

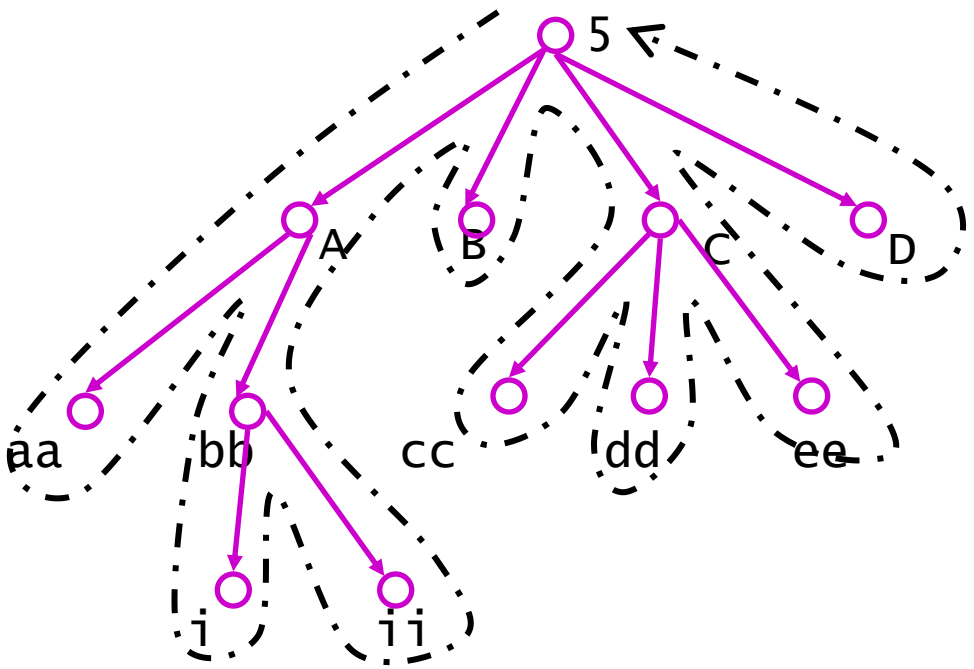
preorde



nummering

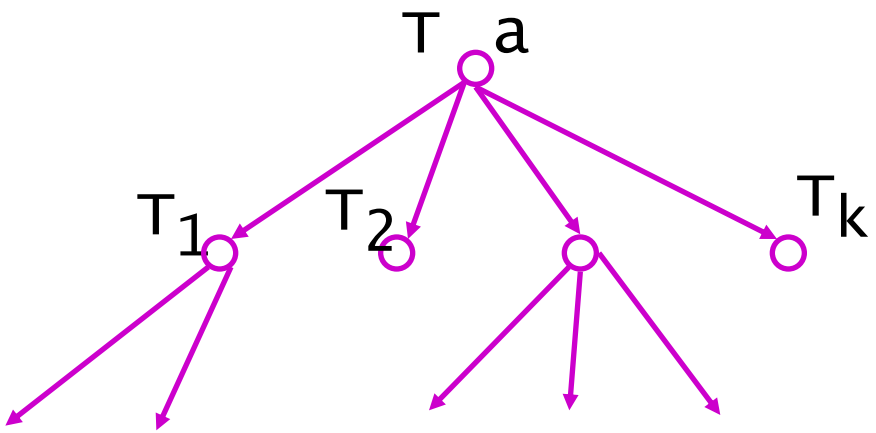
1	5
2	• A
3	• aa
4	• bb
5	•i
6	•ii
7	• B
8	• C
9	• cc
10	• dd
11	• ee
12	• D

omloopmethode: preorde



- 5
  - A
    - aa
    - bb
      - i
      - ii
  - B
  - C
    - cc
    - dd
    - ee
  - D

preorde



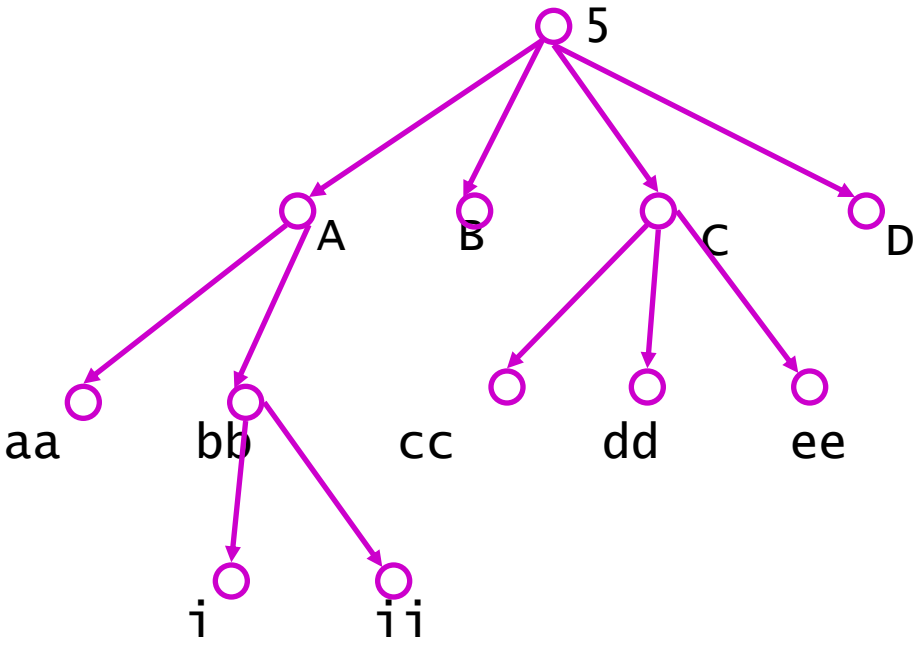
recursieve definitie

boom T
deelbomen T_1, T_2, …, T_k van wortel

$pre(T)$ = wortel(T),
$pre(T_1)$, $pre(T_2)$, …, $pre(T_k)$

‘eerst knoop, dan (subbomen van) kinderen’

preorde

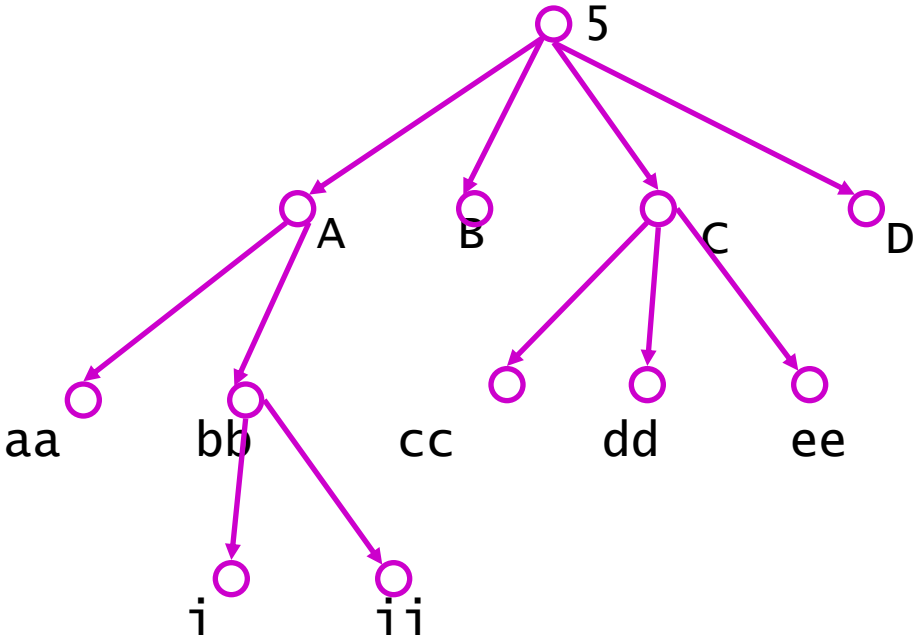


- 5
  - A
    - aa
    - bb
      - i
      - ii
  - B
  - C
    - cc
    - dd
    - ee
  - D

pre-ordening

$pre(T) = wortel(T), pre(T_1), pre(T_2), \ldots, pre(T_k)$

postorde

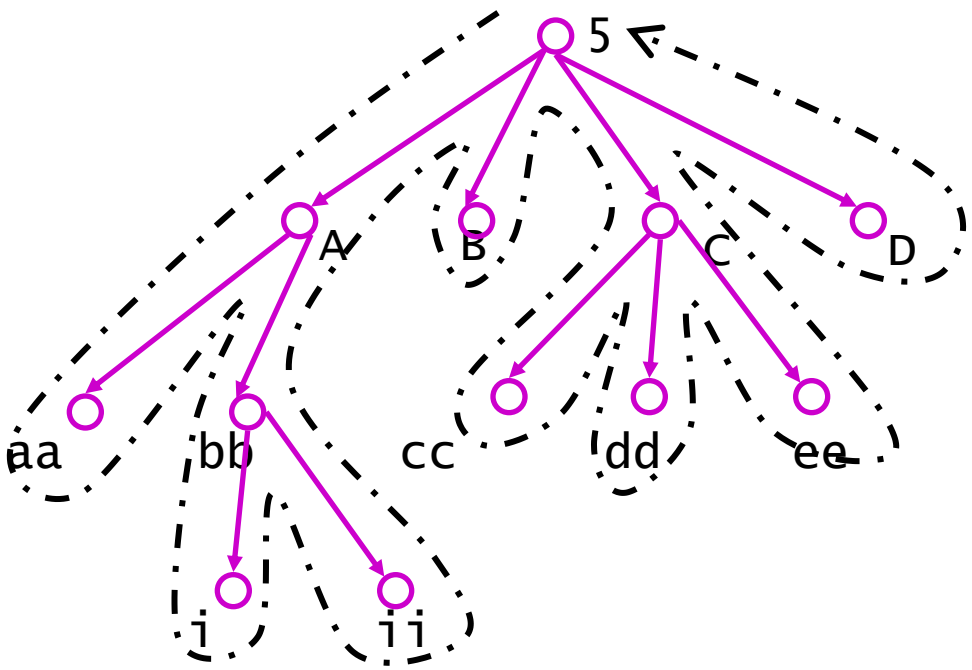


- aa
  - i
  - ii
- bb
- A
- B
  - cc
  - dd
  - ee
- C
- D

5

‘eerst (subbomen van) kinderen, dan knoop’

postorde



Omloopmethode

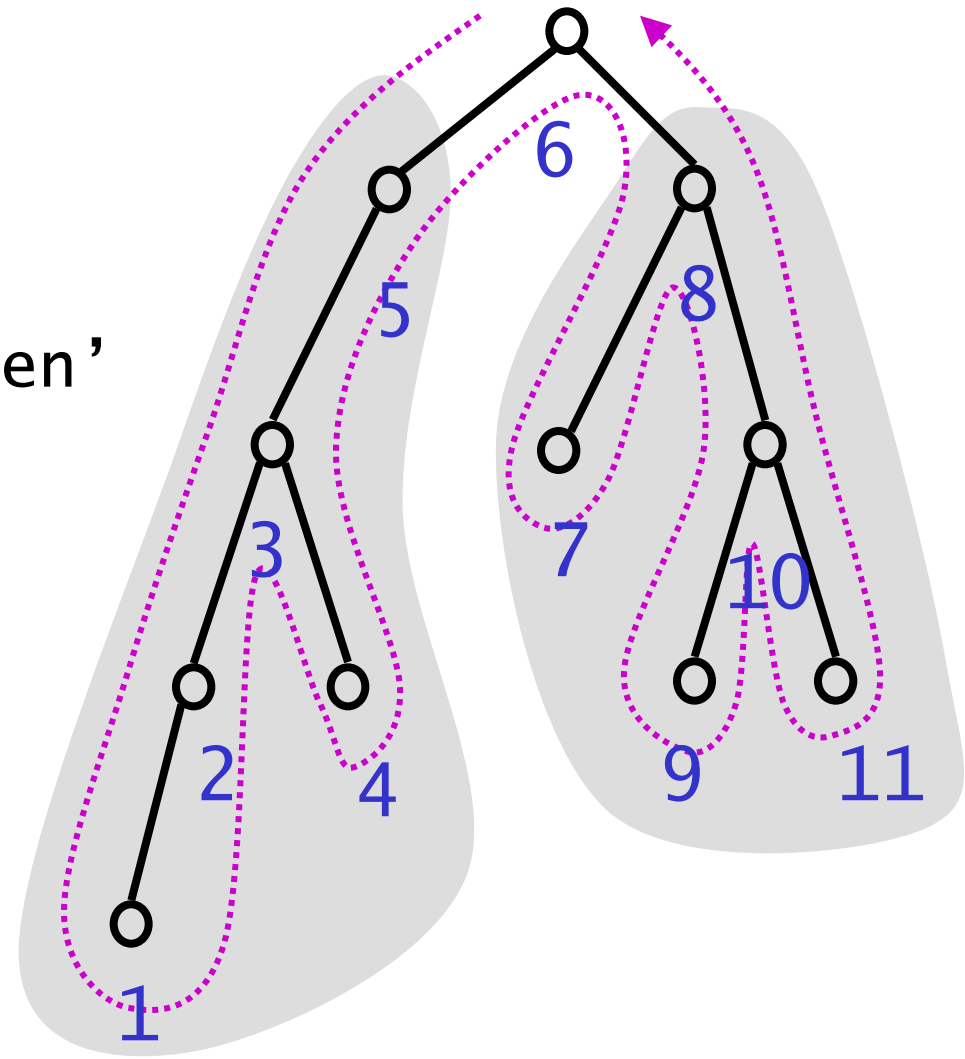
- aa
  - i
  - ii
- bb
- A
- B
  - cc
  - dd
  - ee
- C
- D

5

symmetrische ordening

bij binaire bomen

‘knoop tussen
(subbomen van) kinderen’

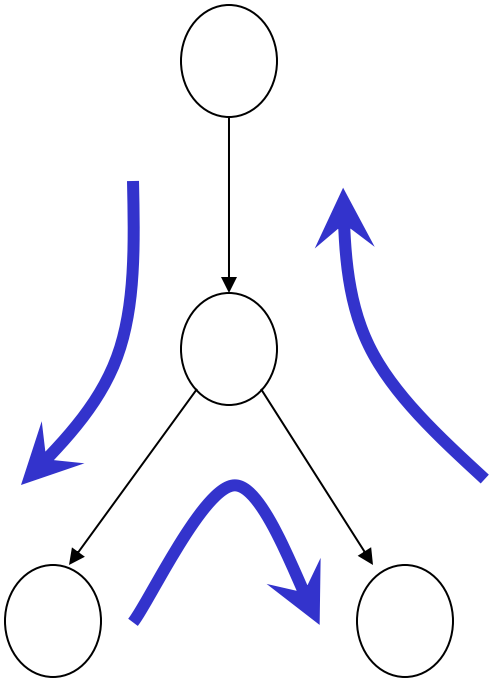


Omloopmethode

$symm(T) = symm(T_\ell), wortel(T), symm(T_r)$

wandelingen

(1)
preorde
WLR



(3)
postorde
LRW

(2)
symmetrisch
LWR

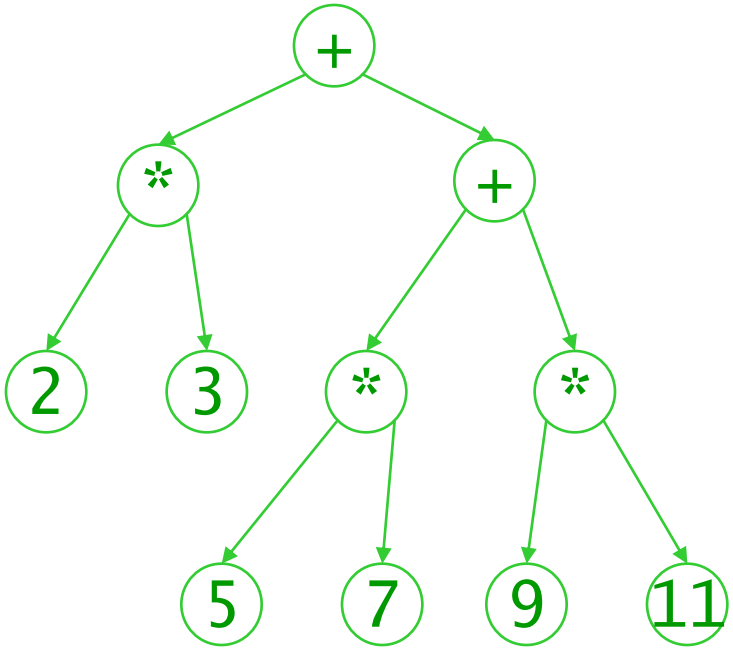
(bij binaire bomen)

wandelingen

bij de omloopmethode zijn er voor binaire bomen drie natuurlijke momenten om (systematisch) een knoop te ‘bezoeken’, bij de eerste, tweede, of derde keer dat we een knoop tegenkomen.
dat leidt dan tot pre-orde, symmetrische, en post-orde wandelingen.

expressies

intern: operatoren
bladeren: waardes



omloopmethode

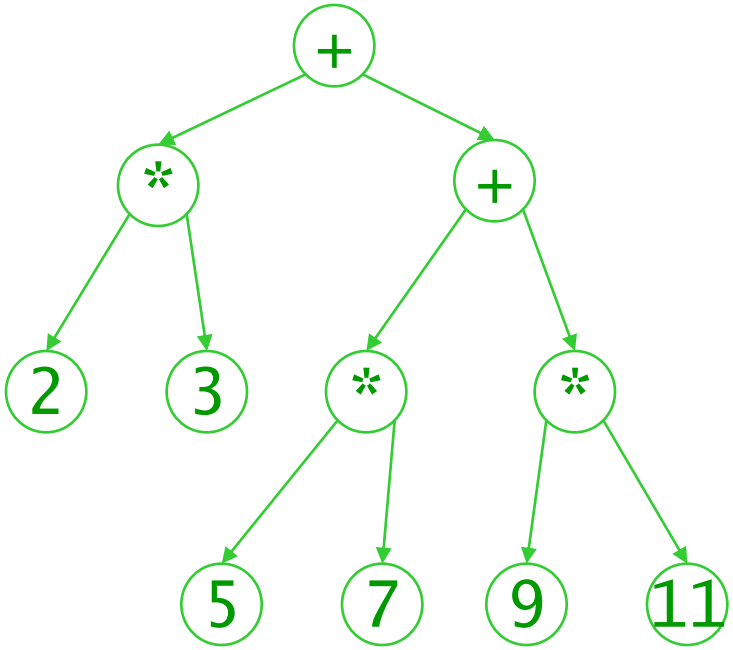
infix, volledig gehaakt
((2*3)+((5*7)+(9*11)))

prefix *Polish notation*
+ * 2 3 + * 5 7 * 9 11

postfix *reverse Polish*
2 3 * 5 7 * 9 11 * + +

zonder haakjes!
'ariteit' bekend

expressies



infix, volledig gehaakt
((2*3)+((5*7)+(9*11)))

haakjes ivm volgorde
evt. voorrangsregels

prefix *Polish notation*
+ * 2 3 + * 5 7 * 9 11
postfix
2 3 * 5 7 * 9 11 * + +

géén haakjes (echt niet!)

recept

prefix *Polish notation*

```
+ * 2 3 + * 5 7 * 9 11
```

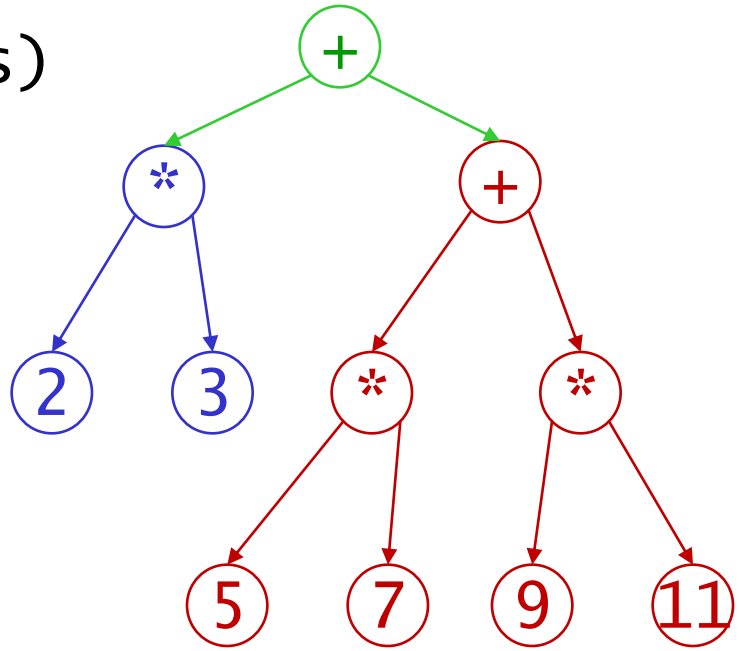
van prefix naar infix:

herhaal

vervang @ x y

(dwz operator + twee waardes)

door (x @ y)




```
+ * 2 3 + * 5 7 * 9 11

+(2*3)+(5*7)(9*11)

+(2*3)((5*7)+(9*11))

((2*3)+((5*7)+(9*11)))
```

recept

postfix *Polish notation*

2 3 * 5 7 * 9 11 * + +

van postfix naar infix:

herhaal

vervang x y @

(dwz operator + twee waardes)

door één nieuwe waarde

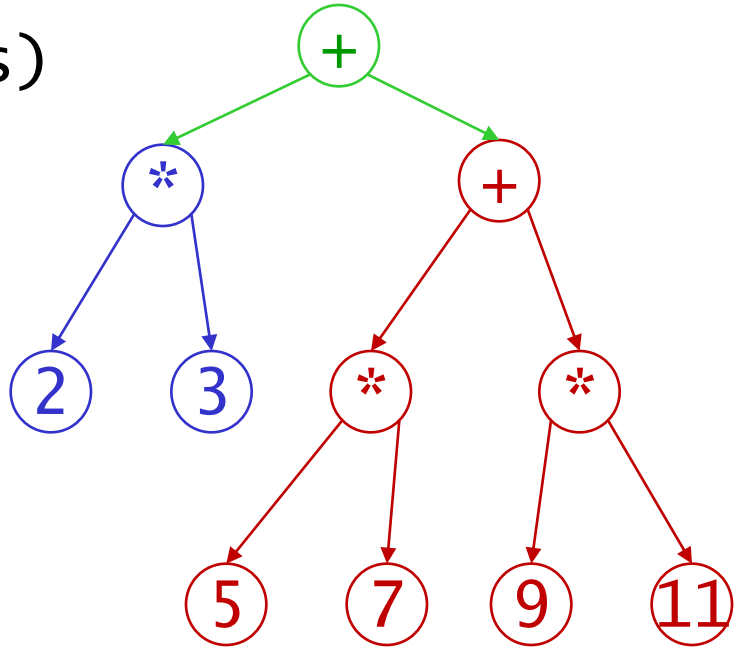
aangegeven door haakjes

2 3 * 5 7 * 9 11 * + +

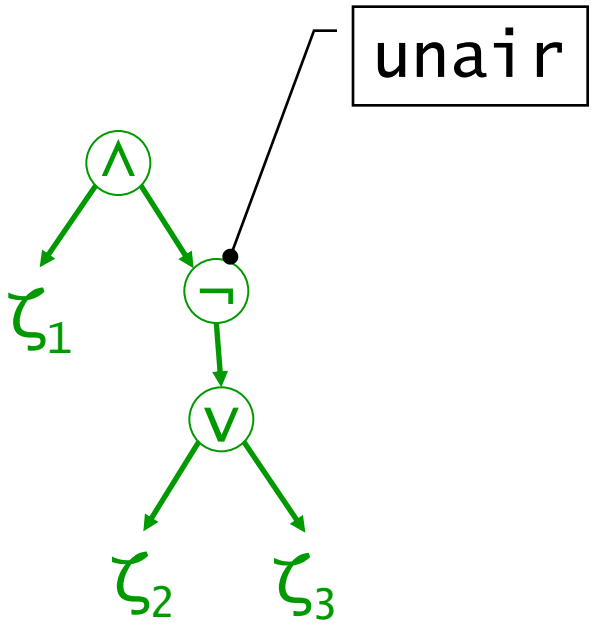
(2*3) (5*7) (9*11) + +

(2*3) ((5*7)+(9*11)) +

((2*3)+((5*7)+(9*11)))



expressies



omloopmethode

infix, waar nodig gehaakt

$\zeta_1 \wedge \neg(\zeta_2 \vee \zeta_3)$

prefix *Polish notation*

$\wedge\ \zeta_1\ \neg\ \vee\ \zeta_2\ \zeta_3$

postfix

$\zeta_1\ \zeta_2\ \zeta_3\ \vee\ \neg\ \wedge$

uniek te ontcijferen míts
ariteit bekend van de operatoren

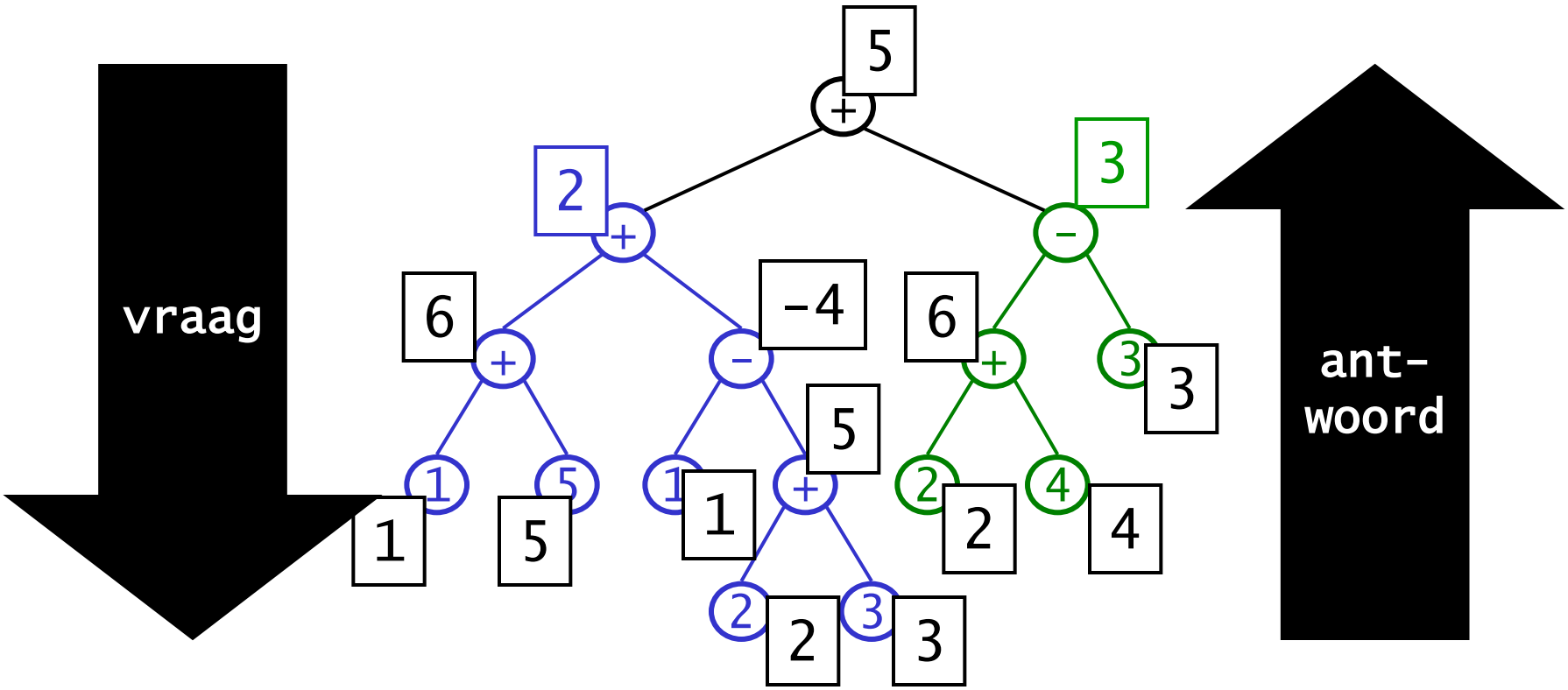
recursieve functies bij bomen

basis

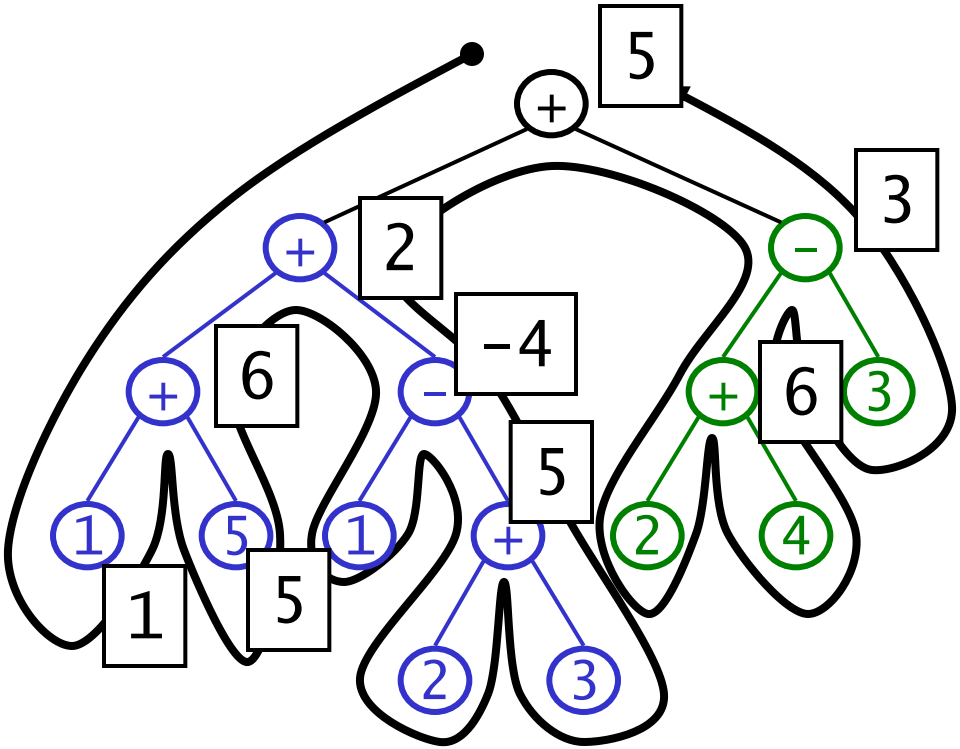
blad ‘x’ f(blad) = getalswaarde x

recursie

knoop ‘@’ f(knoop) = f(links) @ f(rechts)



evaluatie



postorde evaluatie

6 5 -4 2

1 5 + 1 2 3 + - + 2 4 + 3 - +

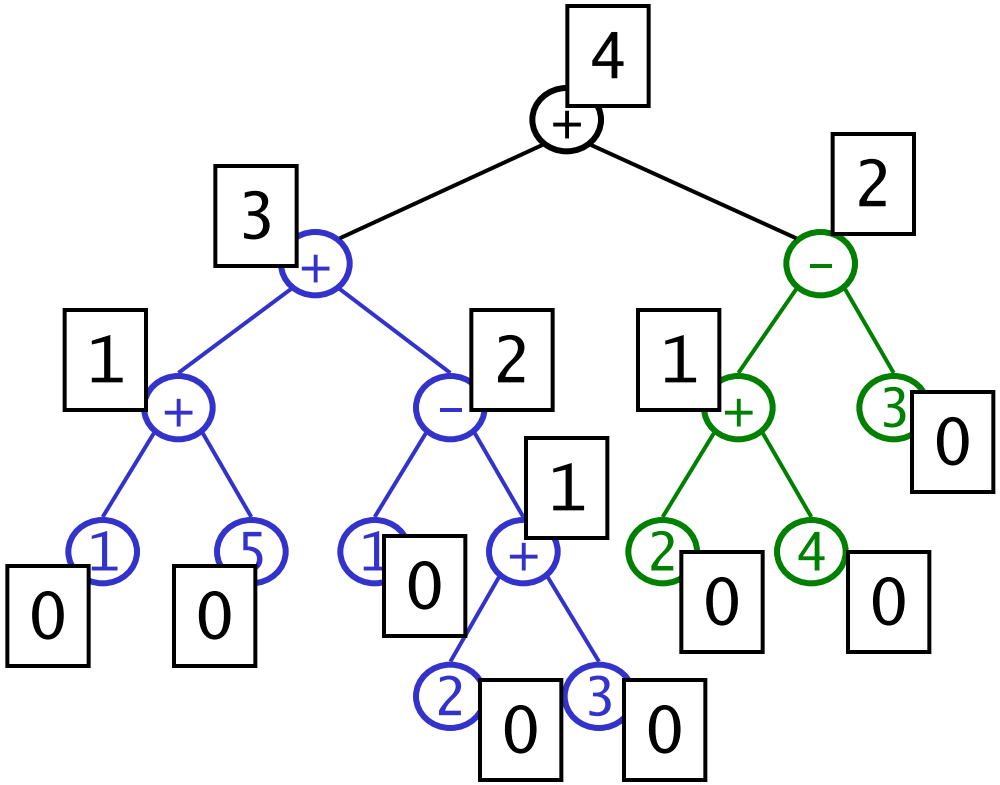
recursieve functies bij bomen

basis

blad f(blad) = 0

recursie

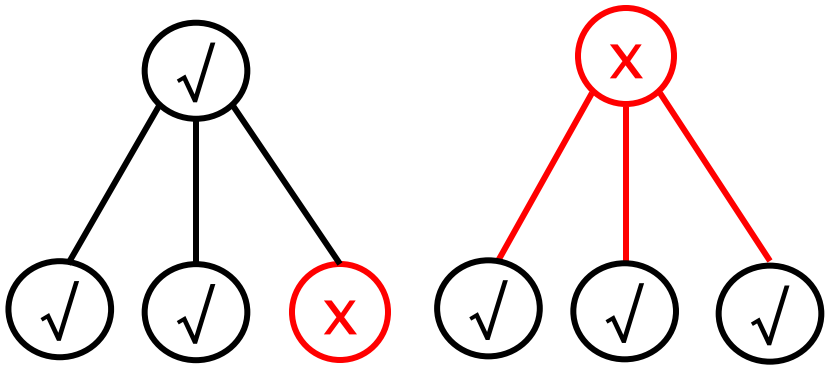
knoop f(knoop) = 1+max{f(links),f(rechts)}

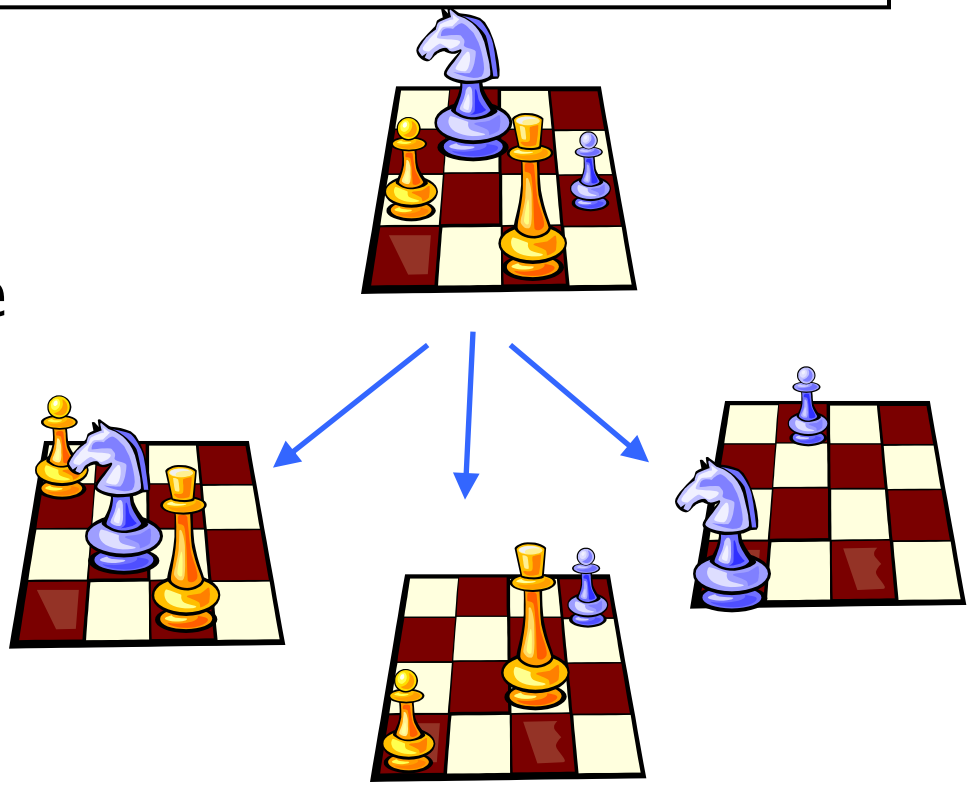


spel-evaluaties

basis
 blad al of niet gewonnen (bij beurt)
recursie
 knoop winnend : ∃ niet-winnend kind
  (verloren: ∀ kind winnend)

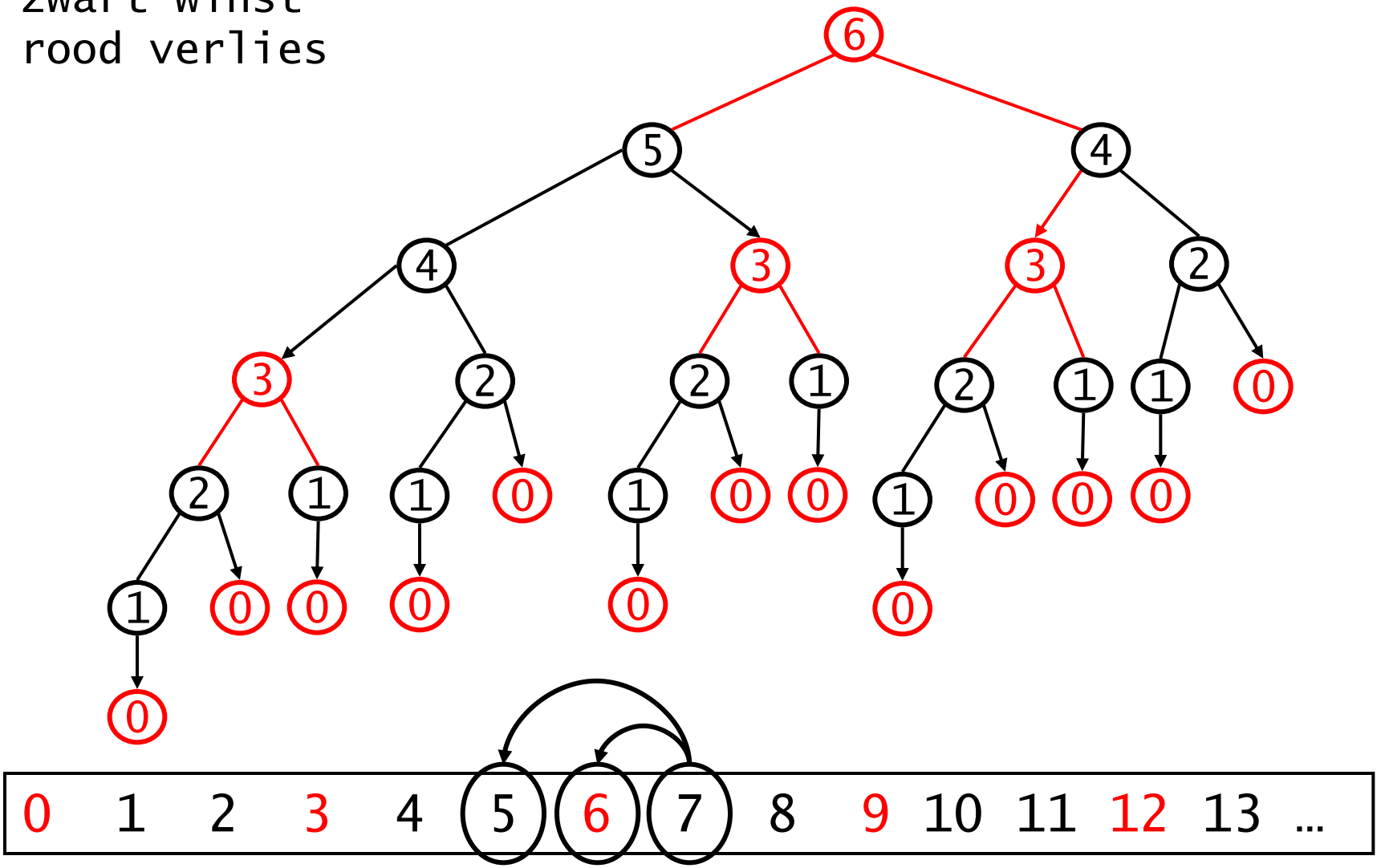
boom niet expliciet
te veel knopen:
 boom knotten +
 evaluatie-functie





Nim

zwart winst
rood verlies



end...