Grammatica's en Ontleden – Deeltentamen 2 (van 2)

Woensdag 31 januari 2007 (8:30-10:30)

Dit examen bestaat uit vier meerkeuze vragen en drie open vragen. In totaal zijn er 100 punten te verdienen. Om te voorkomen dat je in tijdnood komt kan het verstandig zijn om eerst die onderdelen van het tentamen te maken die je het beste liggen. Als er om een Haskell functie wordt gevraagd dan mag je alles uit de `Prelude` *en uit de modules* `List`*,* `Char` *en* `ParseLib` *gebruiken. Het is niet toegestaan om tijdens het tentamen aantekeningen, het diktaat of ander lesmateriaal te raadplegen.*

Veel succes!

---

1) Geneste datatypen (7 punten)

De volgende twee datatypen zijn genest, oftewel, ze zijn gedefinieerd in termen van elkaar.

data $L\ a\ =\ Cons\ a\ (P\ a)\ |\ Nil$

data $P\ a = Pair\ (L\ a)\ (L\ a)$

Welke van de volgende type-definities is het meest geschikt als algebra voor L en P?

a) **type** $LPAlgebra\ a\ l\ p = ((a \to p \to p, l), l \to l \to l)$

b) **type** $LPAlgebra\ a\ l\ p = ((a \to p \to l, l), l \to l \to p)$

c) **type** $LPAlgebra\ a\ l\ p = ((a \to p\ a \to l\ a, l\ a), l\ a \to l\ a \to p\ a)$

d) **type** $LPAlgebra\ a\ l\ p = ((a \to p \to a, l), l \to a \to l \to a)$

2) LR ontleden (7 punten)

Hieronder staan een aantal uitspraken over LR ontleders. Welke uitspraak is correct?

a) Iedere grammatica kan worden getransformeerd naar een equivalente grammatica met de LR(1) eigenschap.

b) Een LR ontleder haalt telkens het eerste hulpsymbool van de stack en vervangt deze door een rechterkant (van dit hulpsymbool) uit de grammatica.

c) Een LR ontleder leest de invoer van rechts naar links (vandaar ook de R in de afkorting).

d) Bij de afleiding van een correcte zin eindigt een LR ontleder met een stack die alleen het startsymbool bevat.

3) Rep-min (7 PUNTEN)

Het "rep-min" probleem bestaat uit het vervangen van alle bladeren in een binaire boom door het minimum van alle bladeren. De vorm van de boom blijft hierbij gelijk. Stel dat we de volgende definities tot onze beschikking hebben:

data *Tree a* = *Leaf a* | *Node* (*Tree a*) (*Tree a*)

type *TreeAlgebra a b* = $(a \to b, b \to b \to b)$

foldTree :: *TreeAlgebra a b* → *Tree a* → *b*
repminAlg :: *Ord a* ⇒ *TreeAlgebra a* $(a \to (a,$ *Tree a*$))$

De implementaties van de twee functies zijn hier niet van belang. Het carrier type van de algebra *repminAlg* is een functie: gegeven het globale minimum wordt het lokale minimum en een nieuwe boom opgeleverd. Welke van de onderstaande definities voor *repmin* is correct?

a)
repmin :: *Ord a* ⇒ *Tree a* → *Tree a*
repmin = *foldTree repminAlg*

b)
repmin :: *Ord a* ⇒ *Tree a* → *Tree a*
repmin t = *foldTree m t*
where *m* = *foldTree repminAlg t*

c)
repmin :: *Ord a* ⇒ *Tree a* → *Tree a*
repmin t = *nt*
where (m, nt) = *foldTree repminAlg t m*

d)
repmin :: *Ord a* ⇒ *Tree a* → *Tree a*
repmin t = (*snd* . *f* . *fst*) *f*
where *f* = *foldTree repminAlg t*

4) Chomsky hierarchie (7 PUNTEN)

Hieronder staan drie beweringen:

- Bewering 1: *"Er bestaan reguliere talen die ook context-vrij zijn."*
- Bewering 2: *"Niet alle talen zijn context-vrij."*
- Bewering 3: *"Iedere taal die niet context-vrij is is ook niet regulier."*

Zijn deze beweringen correct?

a) Alleen bewering 1 is niet waar

b) Alleen bewering 2 is niet waar

c) Alleen bewering 3 is niet waar

d) Alle drie de beweringen kloppen

5) Folds en algebra's (25 PUNTEN)

Gegeven is het volgende datatype voor proposities:

```
data Prop a = Var a                          -- variabele
            | Con Bool                       -- constante
            | Niet    (Prop a)               -- ontkenning
            | En      (Prop a) (Prop a)      -- conjunctie
            | Of      (Prop a) (Prop a)      -- disjunctie
            | AlsDan (Prop a) (Prop a)       -- implicatie
```

Dit datatype abstraheert over de representatie van variabelen (de type-variabele a). Aan een propositie met daarin *Strings* als variabelen wordt bijvoorbeeld het type *Prop String* toegekend.

a) Geef een type-definitie voor algebra's van het *Prop* datatype.

b) Definieer de *fold* functie voor het *Prop* datatype en noem deze *foldProp*. Geef ook met meest algemene type van *foldProp* en gebruik daarbij de type-definitie die je bij onderdeel **a** hebt gegeven.

De laatste drie onderdelen van deze vraag beantwoord je steeds door een algebra te geven voor het beschreven probleem. Je hoeft *foldProp* dus niet aan te roepen. Schrijf voor iedere algebra ook het meest algemene type op.

c) Verzamel in een lijst alle variabelen die voorkomen in een propositie. Dubbele voorkomens hoeven niet te worden verwijderd.

d) Om een propositie te evalueren moeten we de waarheidswaarde van alle variabelen weten. Hiervoor komen de volgende definities van pas:

```
type Env a b = [(a, b)]

(?) :: Eq a => Env a b -> a -> b
((a, b) : rest) ? x
  | a == x    = b
  | otherwise = rest ? x
```

Evalueer een propositie met een gegeven waarheidstabel van het type *Env a Bool*. Het carrier type van deze algebra moet dus een functie zijn.

e) De binaire operatoren *En* en *Of* zijn beide symmetrisch: de propositie *En p q* is semantisch gelijk aan *En q p* (hetzelfde geldt voor *Of*). Transformeer een propositie zodanig dat de grootte van de linker operand van een conjunctie of disjunctie altijd kleiner of gelijk is dan de grootte van de rechter operand. Doe dit door operanden om te draaien. Om de grootte van een propositie te bepalen tellen we het aantal constructoren.

Hint: de algebra moet twee resultaten opleveren. Na het aanroepen van foldProp wordt dan één van de twee resultaten teruggegeven.

6) Pompstellingen (22 PUNTEN)

Bestudeer de taal van goed-geneste haakjesparen. Deze taal heeft het openings- en het sluithaakje als enige twee eindsymbolen. Voorbeelden van zinnen uit de taal zijn:

((((( ))))) (( ) ( )) ( )

Ook de lege string rekenen we tot deze taal.

a) Toon aan dat de taal van goed-geneste haakjesparen context-vrij is.

b) Geef de omgekeerde pompstelling voor reguliere talen waarmee kan worden bewezen dat een taal niet regulier is.

c) De taal die we bestuderen kan op de volgende manier zeer precies worden gespecificeerd: de string s behoort tot de taal van goed-geneste haakjesparen *dan en slechts dan als*

- geen enkele prefix van s minder openingshaakjes heeft dan sluithaakjes; *en*
- het aantal openings- en sluithaakjes in s gelijk is.

Bewijs dat de taal van goed-geneste haakjesparen niet regulier is.

7) LL(1) ontleden (25 PUNTEN)

Gegeven is de volgende grammatica met S als startsymbool en de verzameling $\{a, b, c\}$ als eindsymbolen:

$$\begin{array}{lcl} S & \rightarrow & XaY \\ X & \rightarrow & bSa \\ X & \rightarrow & \epsilon \\ Y & \rightarrow & X \\ Y & \rightarrow & c \end{array}$$

a) Bepaal de *empty* eigenschap en de verzamelingen *firsts* en *follow* voor ieder hulpsymbool van de bovenstaande grammatica.

b) Bepaal aan de hand van *empty*, *firsts* en *follow* de *lookAhead* verzameling van iedere productieregel in de bovenstaande grammatica.

c) Is de bovenstaande grammatica LL(1)? Geef een korte toelichting hoe je dit kunt bepalen met behulp van de berekende *lookAhead* verzamelingen.

d) De string *baaac* behoort tot de taal van de grammatica. Laat zien hoe een LL(1) ontleder deze zin herkent door gebruik te maken van een stack. Geef stapsgewijs aan wat het nog niet geconsumeerde gedeelte van de invoer is samen met de toestand van de stack.