Bird's aanpak van nonterminatie - een haastige notitie

Maarten Fokkinga, 26 mei 1987

Het gebruik van $\perp$ in Bird's boek kan tot misverstanden en onenigheid leiden, omdat Bird niet erg precies is in de rol die $\perp$ speelt. Ik geef hieronder aan wat die rol is, en wat alternatieven voor de behandeling van nonterminatie zijn.

De noodzaak van nonterminatie

Men kan aantonen dat voldoend krachtige berekeningssystemen (computers met programmeertalen), noodgedwongen ook nonterminatie (oneindig voortdurende berekeningen) toelaten. Daarenboven zal soms de berekening van een expressie niet termineren terwijl de wiskundige betekenis van die expressie wel degelijk bepaald is. Bijvoorbeeld vergelijk de gelijkheidstest $\sin = \cos$ met de gelijkheid $\sin = \cos$, en de lidmaatschapstest $17 \in [2,4,\ldots]$ met het lidmaatschap $17 \in [2,4,\ldots]$. Voorts is er nog het verschijnsel dat de mechanische evaluatie van een expressie

niet termineert, maar naarmate de berekening vordert wél steeds meer van de uitkomst oplevert (en in-de-limiet de hele uitkomst). Denk bijvoorbeeld aan een priemgetallen-generator.

Redeneren over nonterminatie

Er zijn verscheidene methoden om met nonterminatie om te gaan. Ieder heeft zo zijn voor- en nadelen. Ik denk dat het een kwestie van smaak is, welke je de voorkeur geeft.

1. Nonterminatie doodzwijgen.

Volgens deze methode kan je niet formeel over nonterminatie uitspraken doen: je beperkt je in de uitspraken over expressies tot gelijkheden die je op grond van de eigenschappen van "=" (nml. reflexiviteit, symmetrie, transitiviteit, congruentie en substitutiviteit) en uitgaande van de definities in het programma (de extra =-axioma's over de identifiers) kunt afleiden.

Het voordeel van deze methode is de elegante en

eenvoudige formalisering; een voordeel op meta-niveau. (Of dit ook in alle opzichten een voordeel op gebruiksniveau is, valt nog te bezien!) Een nadeel van de methode is dat uitspraken van de vorm "de evaluatie van deze expressie termineert niet" niet of nauwelijks te doen (i.e. af te leiden) zijn. In de context van:

```
f :: num → num
x :: num
f y = f y
x = x + 1
```

valt er wel wat voor te zeggen om (f 0) en x als "gelijkwaardig" te beschouwen: f 0 = x. Dat is volgens deze methode niet mogelijk (i.e. niet te bewijzen).

2. Nonterminatie als "nieuwe waarde"

We voeren $\perp$ in als een "nieuwe waarde"; d.w.z. we breiden de bestaande verzamelingen zoals $\mathbb{N} = \{0, 1, 2, \ldots\}$, $\mathbb{B} = \{\text{True}, \text{False}\}$, $\mathbb{Z} = \{\ldots, -1, 0, +1, \ldots\}$ uit met $\perp$ (een aparte $\perp$ voor elke verzameling; $\perp$ is een polymorfe waarde): $\mathbb{N}^\perp = \mathbb{N} \cup \{\perp\}$, $\mathbb{B}^\perp = \mathbb{B} \cup \{\perp\}$ $\mathbb{Z} = \mathbb{Z} \cup \{\perp\}$ enzovoorts. Tevens moeten we nu definiëren wat de operaties voor resultaat geven indien

een der operanden $\perp$ is. Dat doen we zó dat $\perp$ terecht als "nonterminatie" gezien kan worden. Bijvoorbeeld:

$$\perp +^{\perp} x = x +^{\perp} \perp = \perp \qquad (\forall x \in \mathbb{N}^{\perp})$$
$$x +^{\perp} y = x + y \qquad (\forall x, y \in \mathbb{N})$$

Operaties die $\perp$ opleveren indien een der operanden $\perp$ is, noemen we <u>strikt</u>. De meeste operaties zullen tot strikte uitgebreid worden. Maar dat hoeft niet. Je kunt voor vermenigvuldiging kiezen uit de volgende 2 mogelijkheden:

- $x *^{\perp} 0 = 0$ zelfs als $x = \perp$ $(\forall x \in \mathbb{N}^{\perp})$
- $\begin{cases} x *^{\perp} 0 = 0 & \text{alleen voor } x \in \mathbb{N} \\ \perp *^{\perp} 0 = \perp \end{cases}$

Het hangt er maar van af welke evaluatie je voorschrijft/modelleert/definiëert voor $*^{\perp}$.

Het voordeel van deze methode is dat "nonterminatie" nu geformaliseerd is, en vatbaar is voor formele bewijsvoering. Bedenk dat de "negatieve" informatie "dit programma termineert niet" soms best wel nuttig is!
Het nadeel van deze methode moet niet onderschat worden: vanwege de uitbreidingen van dataverzamelingen en operaties, gelden de gebruikelijke axioma's niet zonder meer: het is nu niet meer

waar dat $x *' 0 = 0$ voor alle x, maar er geldt wel $x \neq \bot \Rightarrow x *' 0 = 0$ voor alle x ($\in \mathbb{N}^\bot$). Merk ook op dat we hierboven <u>niet</u> $=^\bot$ schreven maar $=$, dus de échte gelijkheid op $\mathbb{N}^\bot$ volgens welke $\bot = \bot$ waar is. (Eventueel vallen $=^\bot$ en $=$ samen, wanneer we $=^\bot$ als de non-strikte uitbreiding van $=$ definiëren.)

Bovenstaand nadeel is misschien meer een nadeel op meta-nivo (weinig elegante, ingewikkelde theorie) dan op gebruikers-nivo (in de praktijk ben ik nog niet tegen moeilijkheden aangelopen).

Een voorbeeld van het nut van deze methode is als volgt. In een eerder verhaaltje, [Bökinga, 29-4-87], heb ik de optelling van cijfer-lijsten (met de eenheden aan de staart) als volgt gekarakteriseerd:

```
---
ss = map sum xycs
xycs = zip (zip (xs', ys'), cs)
cs = tail (map (div 10) ss ++ [0])
```

Dit zijn weliswaar geldige gelijkheden voor de bedoelde grootheden, maar als definitie beschouwd zijn ze circulair, i.e. recursief. Soms is recursie niet bezwaarlijk (zoals bij de faculteitsfunktie), soms

is het wel bezwaarlijk (zoals bij fac n = fac n). Volgens methode 1 kan ik niet veel over ss, zeycs en cs afleiden. Het bewijs dat niemand er ook maar iets over kan afleiden, heb ik nog niet gevonden. Maar volgens methode 2 kan ik héél eenvoudig aantonen dat ss = zeycs = cs = ⊥. Dus weet ik dat de recursie in deze vorm niet goed is: het programma termineert niet!

Bird volgt, mijns inziens, deze methode in zijn boek. De misverstanden komen vooral voort vanwege het gebrek aan een duidelijke uitleg over ⊥ én ook vanwege het weglaten van de superscripten ⊥ bij de namen der verzamelingen en operaties. Meestal wordt met een operatie $\otimes$ de operatie $\otimes^{\perp}$ bedoeld! Maar met name wanneer hij refereert aan de "wiskundige betekenis" van $\otimes$, is dat niet het geval. Bovendien heeft hij voor bijvoorbeeld de vermenigvuldiging niet gezegd of

$$0 *^{\perp} \perp = 0 \qquad \text{dan wel} \qquad 0 *^{\perp} \perp = \perp$$

gekozen wordt. (Gezien de implementatie van * in Miranda, en gezien de wens om ⊥ als nonterminatie-volgens-Miranda op te vatten, moet hij voor de rechter mogelijkheid kiezen.)

Opmerking. Een voordeel van $\bot$ (in methode 2) dat hierboven nog niet genoemd is, is het volgende. We kunnen de data-verzamelingen nu ordenen, zeg met ordeningssymbool $\sqsubseteq$. Met name:

- voor $\mathbb{N}^\bot$: $\bot \sqsubseteq x$
- voor $\mathbb{B}^\bot$: $\bot \sqsubseteq x$
- voor lijsten: $x:xs \sqsubseteq y:ys$ wanneer $x \sqsubseteq y$ én $xs \sqsubseteq ys$
  $\bot \sqsubseteq xs$

etcetera. Deze ordeningsrelatie (partiële ordening) laat nu nog "fijnere" uitspraken over programma's toe, en laat nu ook recursie en manipulaties op oneindige lijsten. alternatieve verklaringen toe van Let wel, ik zeg niet dat recursie en oneindige lijsten alléén maar met behulp van $\bot$ en $\sqsubseteq$ verklaard kunnen worden. Dit is slechts een alternatief voor de verklaring volgens methode 1, en heeft dus zowel vóór- alsook nadelen.
(Einde opmerking.)

3. Een andere interpretatie van $\bot$.

We zouden $\bot$ ook kunnen interpreteren als "een of andere expressie (mogelijk met nonterminerende evaluatie)".

Anders dan bij methode 2 is $\bot$ nu zelf geen waarde; expressies met $\bot$ erin staan nu voor verzamelingen van expressies, en een gelijkheid zoals expr1 = expr2 staat nu voor de bewering "alle mogelijke uitkomsten ~~kunnen~~ van (instanties van) expr1 kunnen ook door (instanties van) expr2 verkregen worden, en omgekeerd". Dus $\bot < 3 = \bot$ (nml. = True, of = False, of heeft geen uitkomst). En na:

def. $f\,x = 3$

geldt dat: $f\bot = 3$. We kunnen nu ook weer een partiële ordening $\sqsubseteq$ definiëren op expressies-met-mogelijk-$\bot$-erin, met de betekenis dat in expr1 $\sqsubseteq$ expr2 "de uitkomsten van expr1 zeker ook door expr2 verkregen kunnen worden".

Ook al is $\bot$ nu géén waarde, en kun je niet over de waarde van $\bot < 3$ spreken, toch komt deze methode in uitwerking aardig overeen met methode 2. [Ik zou op dit moment geen verschil aan kunnen geven!].

Dankbetuiging Een ("felle") discussie met Gerrit van der Hoeven over opgave 7 van Ch2 van Bird's boek heeft mij tot het schrijven van dit verhaal getriggerd.