T&A Case study - Rekenmachine

Charlie Gerhardus, s3050009

February 24, 2012

1 Inleiding

Rekenmachine is een programma geschreven in C++ dat wiskundige berekeningen kan uitvoeren. In deze berekeningen kunnen we gebruik maken van de volgende operatoren $+, -, *, /$ en haakjes (). De rekenmachine houd rekening met de binding van de operatoren zodat $3 + 2 * 3 = 9 \neq (3 + 2) * 3 = 15$.

Om berekeningen uit te kunnen voeren heeft de rekenmachine twee bestanden nodig. Het eerste bestand 'rekenmachine.woorden' word gebruikt om te beschrijven welke afzonderlijke woorden zich in de taal bevinden. De notatie in dit bestand komt niet overeen met die van de cursus. Daarom hieronder een overzicht van alle woorden en hun reguliere expressie.

woord_getal = $(0U1U2U..U9)^+((,(0U1U2U..U9)^+)U\lambda)$
woord_plus = +
woord_min = -
woord_keer = *
woord_deel = /
woord_open = (
woord_sluit =)

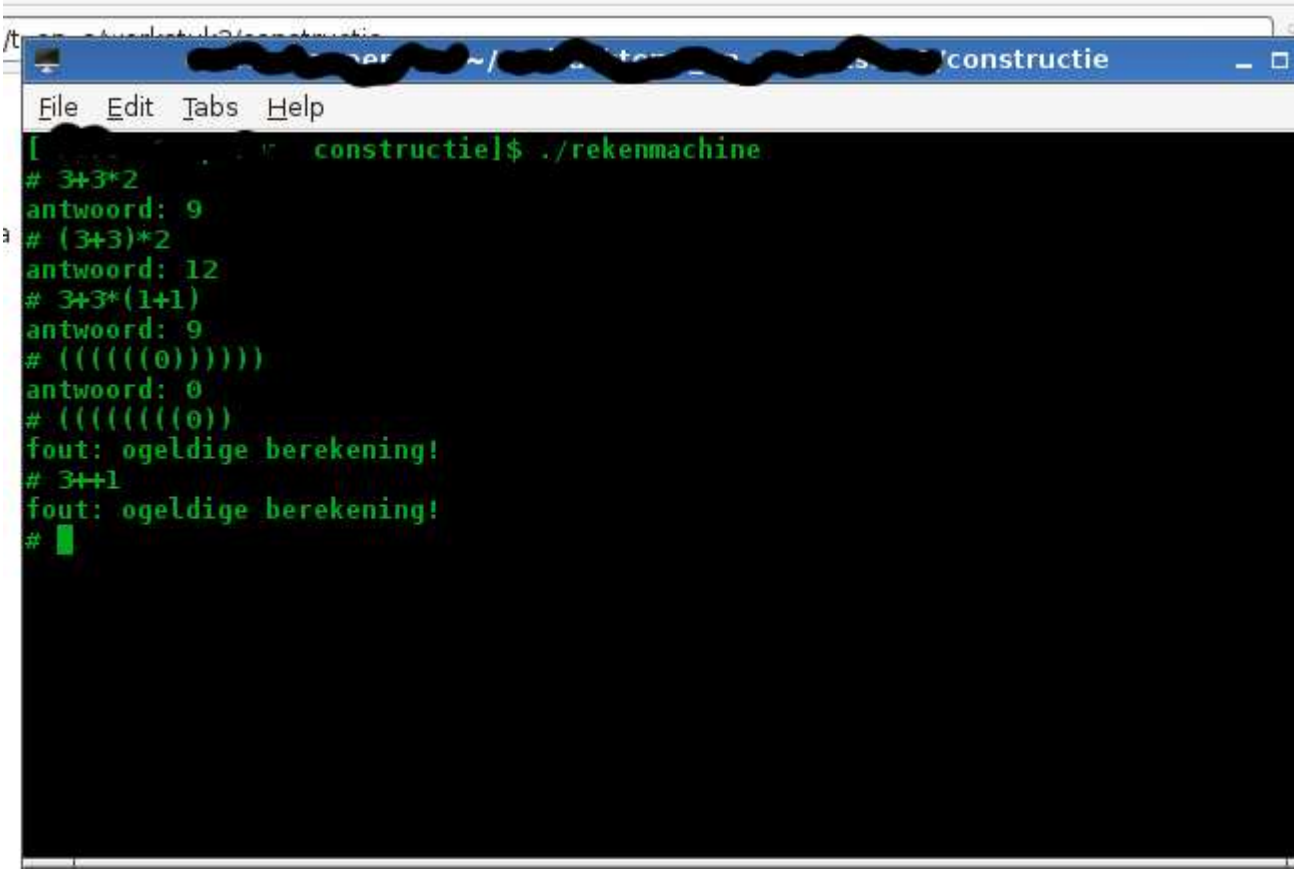
Nadat de rekenmachine de invoer in een lijst met woorden heeft getransformeerd zullen deze door een grammatica verwerkt worden tot een boom. Hieronder de grammatica die "compatible" is met de rekenmachine.

```
S –> L | L P

L –> G | L K
R –> L | L P

P –> OP R

K  –> OK RK
RK –> LK | LK K
LK –> G

OP –> woord_plus |woord_min
```

OK –> woord_keer |woord_deel

G –> woord_getal | woord_open S woord_sluit

Intern gebruikt de rekenmachine een andere notatie om de grammatica te definiren, deze maakt het mogelijk om non-terminals een kind te laten genereren dat niet de zelfde naam heeft als de non-terminal en niet uniek hoeft te zijn. In de daadwerkelijke grammatica word dus geen onderscheid gemaakt tussen OP OK en LK L en RK R en heten deze 'op', 'links', 'rechts' respectievelijk.



Screenshot van rekenmachine

2 Regulier?

De rekenmachine accepteert de volgende invoer, "((((0))))". Het antwoord op deze berekening is 0. De taal vereist dat elk openend haakje een corresponderend sluitend haakje heeft. De volgende expressie genegeerd woorden die onderdeel zijn van de rekenmachine taal.
$y = ({}^{i}0)^{i}$

Stel: De rekenmachine taal is regulier.
Bewijs: Voor elk woord w met $lengte(w) >= k$ waarbij k het aantal states in de automaat is geld, dat $w = uvw$ waar $lengte(uv) <= k$ en v herhaald mag worden zodat $uv^{i}w$ ook onderdeel is van de rekenmachine taal.

$w = y$ Met $i = k$
Omdat $lengte(uv) <= k$ betekend dit dat uv alleen maar opende haakjes kan bevatten. Als deze gepompt worden ontstaat er een ongeldig woord met een ongelijk aantal opende en sluitende haakjes.

Dus de rekenmachine taal is niet regulier.

3 Rekenmachine in werking

De manier waarop de rekenmachine het antwoord berekend is hieronder in pseudo code gegeven.

```
functie double bereken(blad, [links_gegeven=false, links_in])

als blad bevat woord_getal:
        return woord_getal

double links = 0.0
als links_gegeven:
        links = links_in
anders:
        links = bereken( blad.kind["links"] )

als blad heeft kind "op":
        rechts = bereken( blad.kind["rechts"]["links"] )
        links  = links "op" rechts
```

```
        als blad heeft kind "rechts""op":
                links = bereken (blad.kind["rechts"], true, links)

return links
```

Wat de grammatica dus in wezen voor de rekenmachine doet is gedeelten die voorrang krijgen in een berekening in ["links"] of in ["rechts"]["links"] zetten. Als de ["rechts"] van de huidige operatie een ["op"] bevat betekend dit dat ["rechts"] de volgende operatie is, ["links"] hoeft nu niet meer berekend te worden en is gegeven.

Invoer en hun boom:

```
# 1+2
regel_wiskunde:
  links:
    woord_getal = 1
  op:
    woord_plus = +
  rechts:
    links:
      woord_getal = 2

antwoord: 3


# 3+2*3
regel_wiskunde:
  links:
    woord_getal = 3
  op:
    woord_plus = +
  rechts:
    links: << DEZE KOMT VOOR 3+
      links:
        woord_getal = 2
      op:
        woord_keer = *
```

```
      rechts:
        links:
          woord_getal = 3

antwoord: 9

# (3+2)*3
regel_wiskunde:
  links:
    links: << NU IS DEZE EERST
      links:
        woord_getal = 3
      op:
        woord_plus = +
      rechts:
        links:
          woord_getal = 2
    op:
      woord_keer = *
    rechts:
      links:
        woord_getal = 3

antwoord: 15
```