Naam:
Adres:
Postcode en
Woonplaats:

Studentnummer:
Studierichting:
Jaar van eerste inschrijving:

Bladnr.: 1 / 3
Tentamen: Vertaalboom
Datum: 09-02-2009
Naam docent: Jongejan

a)

FIRST

S → ABC	ab
A → aB	a
A → b	b

B → Bb	ab
B → S	ab
C → cA	c

	FIRST	FOLLOW
S	ab	$ abc
A	ab	abc$
B	ab	abc$
C	c	$abc

$ staat voor end of file

b

niet LL(1), er is een first/first conflict op beide B → regels.

poging om het conflict op te lossen: niet gelukt

B → S B' follow(B') ∩ follow(b) = b

B' → ε

B' → b B' first = {b}

first/follow conflict

Dus niet makkelijk op te lossen, dus herschrijven.

c)

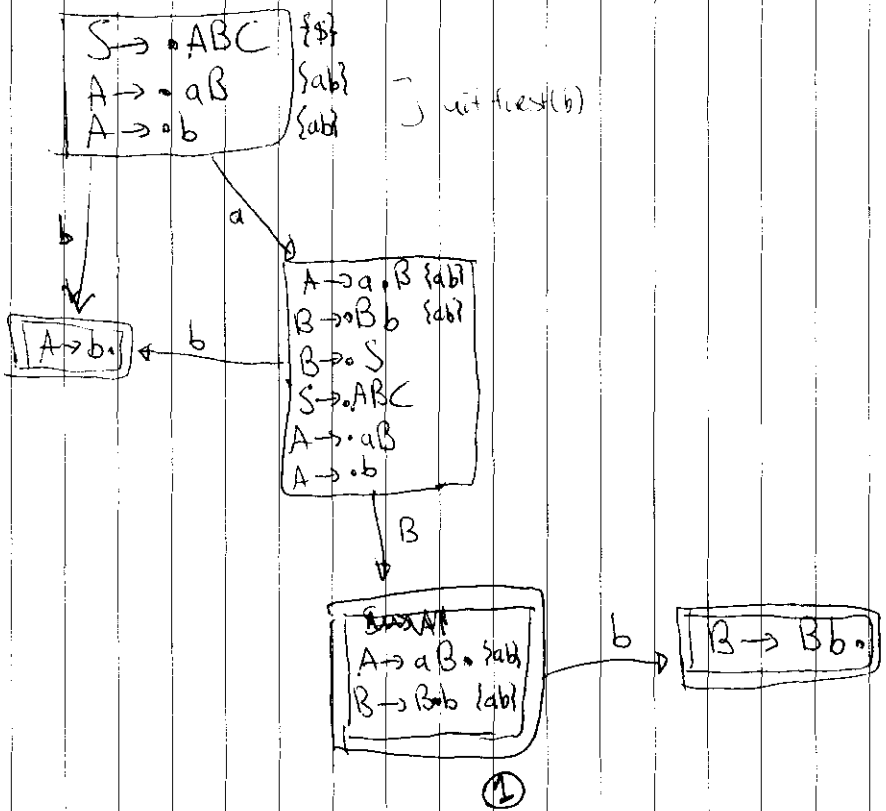
LR(0) zou staat diagram (gedeelte van)

niet LR(0), want er is een shift/reduce conflict op toestand (A → aB• , B → B•b) by ①

Ook niet SLR(1), want b ∈ FOLLOW(A), dus nog steeds een shift/reduce conflict.

LR(1): { } / follow sets v regels toevoegen → nog steeds een shift/reduce conflict.

conflicten proberen op te lossen heeft geen zin, want B ⇒* Sb* en S ⇒* αBβ dus er zijn geen enkele geldige parse trees mogelijk en de taal die beschreven wordt is ∅


S → •ABC {$}
A → •aB {ab}
A → •b {ab}
wie first(b)
a
b
A → a•B {ab}
B → •Bb {ab}
B → •S
S → •ABC
A → •aB
A → •b
A → b•
b
B
A → aB• {ab}
B → B•b {ab}
b
B → Bb•
1

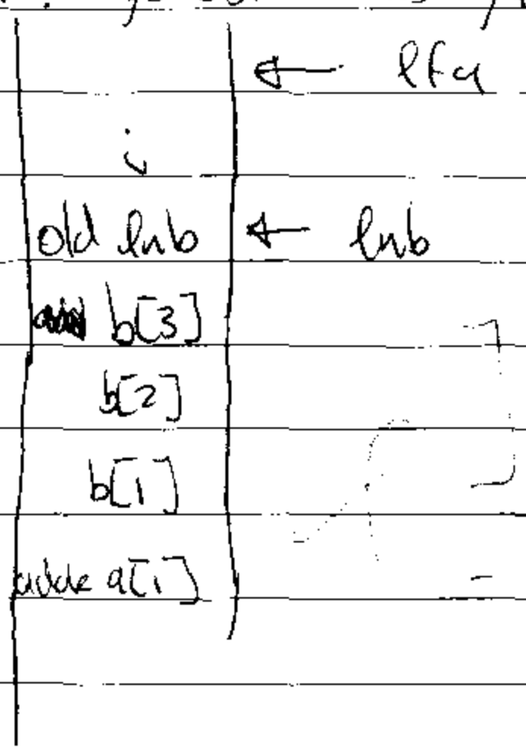
Naam:
Adres:
Postcode en
Woonplaats:

Studentnummer:
Studierichting:
Jaar van eerste inschrijving:

Bladnr.: 2/3
Tentamen: Vertaalbouw
Datum: 09-02-2007
Naam docent: Jongejan

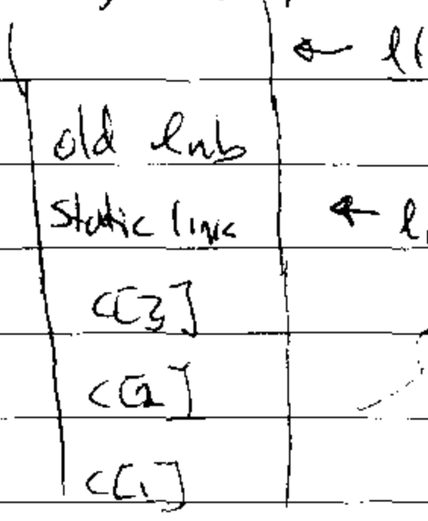
2 a

AR van set_cell
PN=1, niet genest dus geen static chain



b

AR van p
PN=2 (genest), dus static chain erby



entry:
M(lfa) = anW
M(lfa+1) = lnb
lnb = lfa
lfa = lfa + 2

ext:
lfa = lnb
lnb = M(lfa+1)
Return

C1

$R_0 = M(M(lnb)+1)$; i

$R_1 = M(lnb - 4 + R_0)$; c[i], -3 adder c[i] -1 onderliggens] -4

$R_1 = R_1 + 2$

$R_2 = M(lnb) - 4 + R_0$; adres b[i]

$M(R_2) = R_1$

C2

env = lnb

$M(lfa) = M(lnb-3)$; push b[1]

$M(lfa+1) = M(lnb-2)$; push b[2]

$M(lfa+2) = M(lnb-1)$; push b[3]

$lfa = lfa + 3$

call p

$lfa = lfa - 3$

10

C3

$R_0 = M(lnb - 4 + M(lnb+1))$; b[i]

$R_0 = R_0 \times 2$

$R_1 = M(lnb - 4)$; adres a, want a by referene

$M(R_1 - 4 + M(lnb+1)) = R_0$

C4 (set-all (a,4))

$M(lfa) = gp$; push adres a[1]

$M(lfa+1) = M(gp)$; push a[1]

$M(lfa+2) = M(gp+1)$; push a[2]

$M(lfa+3) = M(gp+2)$; push a[3]

$lfa = lfa + 4$

call set-all

$lfa = lfa - 4$

Naam:
Adres:
Postcode en
Woonplaats:

Studentnummer:
Studierichting:
Jaar van eerste inschrijving:

Bladnr.: 3 13
Tentamen: Vertaalbouw
Datum: 09-02-2007
Naam docent: Jorge Jan

3a

het is een manier LL(1) parser die in plaats van een expliciete stack de callstack gebruikt.
voor het matchen van een nonterminal N wordt gekeken naar de FIRST sets van de alternatieven (en mogelijk follow set(N) als $\varepsilon \in$ FIRST(N)) om een alternatief te voorspellen. als dan bv $N \rightarrow aAbC$ voorspelt is wordt naar alle (non)terminals gezocht dmv recursie:
match(a); pA(), match(b); pC()
met pX de procedure voor de nonterminal X, en match(a) een functie die een 'a' als volgende character/token in de input eist en deze verwijdert.

b

procedure Pdecls
if sym ∈ FIRST(Pdecls)
Pdecl(); pDecls();
else if sym ∈ Follow(Pdecls)
Empty() (* skip *)
else
error("can't match Pdecls")
end
end

procedure Pcall
match(callsym); match(id); match(lpar)
Arglst();
match(rpar);
end

procedure Arg
if sym = int OR sym = character
nextsym
else if sym ∈ FIRST(Arg)
Pcall();
else
error("can't match Arg")
end
end

10

c

```
procedure Pdecls (keys)
  delete (FIRST(Pdecls) ∪ keys ∪ FOLLOW(Pdecls))
  if sym ∈ FIRST(Pdecls)
    Pdecl(keys ∪ FIRST(Pdecls))
    Pdecls(keys)
  end
  // else niks ~~...~~ ct Empty(), maar die doet niks
end

procedure Pcall (keys)
  ~~...~~
  match (callsym, keys ∪ {lpar, rpar, id} ∪ FIRST(Arglst))
  match (id, keys ∪ {lpar, rpar} ∪ FIRST(Arglst))
  match (~~...~~ lpar, keys ∪ {rpar} ∪ FIRST(Arglst))
  Arglst (keys ∪ {rpar})
  match (~~...~~ rpar, keys)
end

procedure Arg(keys)
  delete (keys ∪ FIRST(Arg))
  if ~~...~~ sym ∈ First(Pcall)
    Pcall(keys)
  else if sym = character
    nextsym
  else // korte keuze voor default productie.
    match(int ~~...~~ keys)
  end
end
```

de parameter 'keys' is hier steeds van het type tsymbolset. De eerste regel van bovenstaande procedures zou moeten zijn:

```
procedure X (keys: tsymbolset)
```