

Natuurlijke taal interfaces

toets

vrijdag, 24 december 2010
docenten: Kata Balogh & Joris Borgdorff

Opdracht 1

Neem de volgende zinnen.

- (1) a. *Competent women and men hold all the good jobs in the firm.*
b. *Jan loopt naar de bank.*
c. *A professor talked to every student.*

- (a) Van welke soort ambiguïteit zijn deze zinnen een voorbeeld?
- (b) Leg alledrie soorten ambiguïteiten uit op basis van deze zinnen.
- (c) Geef de verschillende vertalingen in Predikaat-logica van de zin in (1c).

Opdracht 2

Neem de volgende zin.

- (2) *John and every teacher brought a laptop.*

- (a) Geef de boom-structuur van de zin op basis van onderstaande de herschrijf-regels.
- (b) Op basis van de onderstaande vertaal-regels (bij elk herschrijf-regel) geef voor elke knoop in de boom de semantische vertaling. Geef stap voor stap aan, hoe je bij elke knoop de semantische vertaling kan uitrekenen van de vertaling op basis van de gedeeltes.
- (c) Leg het verschil uit van de twee vertalingen van 'John' ($j / \lambda P.P(j)$) en van de twee vertalingen van 'teacher' ($TE / \lambda x.TE(x)$).

Herschrijfregels en vertaal-regels

$S \rightarrow NP VP$	$S' = NP'(VP')$
$VP \rightarrow V$	$VP' = V'$
$VP \rightarrow V NP$	$VP' = V'(NP')$
$NP \rightarrow DET N$	$NP' = DET'(N')$
$NP \rightarrow PN$	$NP' = PN'$
$NP \rightarrow NP CONJ NP$	$NP' = \lambda Q[CONJ'(NP'_1(Q))(NP'_2(Q))]$
$DET \rightarrow \text{every}$	$DET' = \lambda S.\lambda P.\forall x(S(x) \rightarrow P(x))$
$DET \rightarrow a$	$DET' = \lambda S.\lambda P.\exists x(S(x) \wedge P(x))$
$CONJ \rightarrow \text{and}$	$CONJ' = \lambda A\lambda B(A \wedge B)$
$PN \rightarrow \text{John}$	$j / \lambda P.P(j)$
$N \rightarrow \text{teacher}$	$TE / \lambda x.TE(x)$
$N \rightarrow \text{laptop}$	$LT / \lambda x.LT(x)$
$V \rightarrow \text{brought}$	$\lambda P.\lambda x.P(\lambda y.BR(x, y))$

Opdracht 3

Neem de volgende vraag.

(3) *Which boy does John see?*

- (a) Geef de boom-structuur van de zin op basis van de onderstaande herschrijf-regels.
- (b) Op basis van de onderstaande vertaal-regels (bij elk herschrijf-regel) geef voor elke knoop in de boom de semantische vertaling. Geef stap voor stap aan, hoe je bij elke knoop de semantische vertaling kan uitrekenen van de vertaling op basis van de gedeeltes.

$S \rightarrow \text{WHQ}$	$S' = \text{Question}[\text{WHQ}]$
$\text{WHQ} \rightarrow \text{WHNP YNQ}[+t]$	$\text{WHQ}' = \text{WHNP}'(\lambda \text{trvar. YNQ}')$
$\text{WHNP} \rightarrow \text{WHDET N}$	$\text{WHNP}' = \text{WHDET}'(\text{N}')$
$\text{YNQ}[-t] \rightarrow \text{does NP}[-t] \text{VP}[-t]$	$\text{YNQ}' = \text{NP}'(\text{VP}')$
$\text{YNQ}[+t] \rightarrow \text{does NP}[-t] \text{VP}[+t]$	$\text{YNQ}' = \text{NP}'(\text{VP}')$
$\text{VP}[-t] \rightarrow \text{V NP}[-t]$	$\text{VP}' = \lambda x. \text{NP}'(\lambda z. \text{V}'(x, z))$
$\text{VP}[+t] \rightarrow \text{V NP}[+t]$	$\text{VP}' = \lambda x. \text{NP}'(\lambda z. \text{V}'(x, z))$
$\text{NP}[-t] \rightarrow \text{PN}$	$\text{NP}' = \text{PN}'$
$\text{NP}[+t] \rightarrow \epsilon$	$\text{NP}' = \lambda P. P(\text{trvar})$
$\text{WHDET} \rightarrow \text{which}$	$\lambda S. \lambda P. \lambda y. (S(y) \wedge P(y))$
$\text{PN} \rightarrow \text{John}$	$j / \lambda P. P(j)$
$\text{N} \rightarrow \text{boy}$	$\text{BOY} / \lambda x. \text{BOY}(x)$
$\text{V} \rightarrow \text{see}$	$\lambda P. \lambda x. P(\lambda y. \text{SEE}(x, y))$

Opdracht 4

- (a) Vertaal de volgende zinnen in propositionele tijdslogica (gebruik de operatoren **P, H, F, G**). Geef de sleutel voor de propositionele variabelen die je gebruikt. Het gedeelte in (iii) tussen haakjes dient ter verheldering van de temporele structuur, het hoeft niet te worden vertaald.
- (b) Definieer een Kripke-model voor propositionele tijdslogica, waar de zin (i) waar is.
 - (i) Anna werd verliefd op Peter, en sindsdien houdt ze altijd en eeuwig van hem.
 - (ii) Anna wilde altijd taalkunde gaan studeren, maar ze deed het nooit.
 - (iii) (De docent vertelde Anna dat) zij nooit haar diploma zou halen op deze manier.

Opdracht 5

Zinnen bevatten naast woorden ook leestekens, een bekende is de komma. Een mogelijk gebruik van de komma is die van een opsomming als in de zin "Jan, Piet, Kees en Klaas lopen." Als we de opbouw van dit type zin willen analyseren, voor opsommingen van willekeurige lengte, zouden we de volgende (vrij minimale) syntax kunnen voorstellen:

```

s --> np, vp
np --> npsum, coord, noun
npsum --> noun
npsum --> npsum, komma, npsum
vp --> iv

```

waar **komma** een komma representeert. Deze syntax is vervolgens geïmplementeerd in prolog, inclusief semantiek.

- (a) Leg uit waarom de uitwerking van deze syntax in prolog problemen zal geven.
- (b) Schrijf een syntax die in prolog wel zal werken en leg uit waarom.

Natuurlijke taal interfaces

tentamen: 24 december 2010, 13-15, SciencePark G2.10

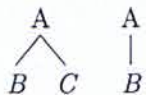
onderwerpen en cheat-sheet

1. Ambigüiteit (soorten, problemen voor NLP)
2. Type-theory, lambda-abstractie en lambda-conversie
3. Compositionele semantiek
 - herschrijf-regels en vertaal-regels
 - vertalingen van:
4. Gegeneraliseerde kwantoren
5. Kripke-modellen
 - propositionele tijdslogica
 - modale predikaatlogica
 - possible worlds semantiek
6. Semantiek van vragen en antwoorden
7. Anaphora en discourse anaphora (soorten, problemen voor NLP)

1 Compositionele semantiek

herschrijf-regel: $A \rightarrow B C$ met vertaal-regel: $A' = B'(C')$

herschrijf-regel: $A \rightarrow B$ met vertaal-regel: $A' = B'$



2 Type theory

basis types: e, t

e : uitdrukkingen die naar entiteiten verwijzen

t : uitdrukkingen die naar waarheidswaarden verwijzen (proposities)

De verzameling van types (T) gedefinieerd als:

T is de kleinste verzameling zodanig dat

(i) $e, t \in T$

(ii) als $a, b \in T$, dan $\langle a, b \rangle \in T$

Functionele applicatie

Type $\langle a, b \rangle$ is een uitdrukking, die als toegepast is op een uitdrukking van type a , een uitdrukking van type b oplevert

$\Rightarrow [\alpha_{\langle a, b \rangle}(\beta_a)]_b$

3 Lambda-abstractie en lambda-conversie

Syntaxis:

als we hebben een open propositie (i) (met een ongebonden variabele), dan kunnen we die variabele binden met \forall, \exists (ii) of met de lambda-operator: λ (iii).

- (i) $P(x)$
- (ii) $\forall x.P(x), \exists x.P(x)$
- (iii) $\lambda x.P(x)$

Semantiek:

$\lambda x.P(x) = \{x \mid P(x)\} \Rightarrow$ de verzameling van individuen, die de eigenschap P hebben

Lambda-conversie:

$$(\lambda x_a(\phi))(\alpha_a) = \phi[x/\alpha]$$

Functionele applicatie:

$$[(\lambda x_a(\phi))_{\langle a,b \rangle}(\alpha_a)]_b$$

4 Kripke-models

De propositionale taal wordt uitgebreid met operatoren, bv. operator: O .

Toegevoegd aan een formula ϕ , levert een nieuwe formula $O\phi$

$$O\phi, \neg O\phi, O\neg\phi, O\phi \wedge \psi, OO\phi \text{ etc.}$$

Een (propositionale) Kripke-model M bestaat uit:

- K : een niet lege verzameling van contexten
- R : twee-plaatsige relatie op K – bereikbaarheidsrelatie
- V : waardering, die aan elke propositie p in elke context $k \in K$ een waarheidswaarde $V_k(p)$ toekent.

Temporele logica

Als M is een Kripke-model, T een verzameling van tijdstippen ($T = \{t_1, \dots, t_n\}$) en R is de eerder-dan relatie ($<$), dan

- $V_{M,t}(G\phi) = 1$ iff voor alle $t' \in T$ zodat $tRt' : V_{M,t'}(\phi) = 1$
- $V_{M,t}(F\phi) = 1$ iff voor minstens een $t' \in T$ zodat $tRt' : V_{M,t'}(\phi) = 1$
- $V_{M,t}(H\phi) = 1$ iff voor alle $t' \in T$ zodat $t'Rt : V_{M,t'}(\phi) = 1$
- $V_{M,t}(P\phi) = 1$ iff voor minstens een $t' \in T$ zodat $t'Rt : V_{M,t'}(\phi) = 1$

Modale logica

Als M is een Kripke-model, W een verzameling van mogelijke werelden ($W = \{w, \dots, w_n\}$) en R is de toegankelijkheids relatie ($wRw' : w'$ is toegankelijk van w), dan

- $V_{M,w}(\Box\phi) = 1$ iff voor alle $w' \in W$ zodat $wRw' : V_{M,w'}(\phi) = 1$
- $V_{M,w}(\Diamond\phi) = 1$ iff voor minstens een $w' \in W$ zodat $wRw' : V_{M,w'}(\phi) = 1$