

## Parsningsalgoritmer

### OH-serie: Dependensparsning

<http://stp.lingfil.uu.se/~matsd/uv/uv09/pa/>



UPPSALA  
UNIVERSITET

Mats Dahllöf  
Institutionen för lingvistik och filologi  
Maj 2009

1

## Dependensgraf

Givet en mängd etiketter,  $L$ , är en dependensgraf  $D = (W, A)$ , där

- $W = w_0 \dots w_{n-1}$  (sekvens av förekomster)
- $w_i < w_j$  om  $i < j$  (linjär ordning)
- $A$  är en mängd av etiketterade bågar  $(w_i, l, w_j)$ , där  $w_i, w_j \in W$  och  $l \in L$ .  
 $w_i$  är huvud till  $w_j$ , säger vi.
- Om  $i \neq j$ , så  $w_i \neq w_j$  (p.g.a. förekomstkravet). Nivre (2008) använder heltal direkt som noder.

3

## Riktad väg

Givet en dependensgraf  $D = (W, A)$

och två noder  $a$  och  $b$  ( $a \in W$  och  $b \in W$ )

så utgör en riktad väg mellan  $a$  och  $b$  en icke-tom delmängd av bågar  $\{(a, l_1, w_{i_1}), (w_{i_1}, l_2, w_{i_2}), \dots, (w_{i_{k-1}}, l_k, w_{i_k})\} \subseteq A$ , sådan att  $w_{i_k} = b$ .

Alltså, för specialfall...

Vid kortast tänkbara väg:  $\{(a, l_1, b)\} \subseteq A$ .

Vid näst kortast tänkbara väg:  $\{(a, l_1, w_{i_1}), (w_{i_1}, l_2, b)\} \subseteq A$ .

Vid tre snäpp:  $\{(a, l_1, w_{i_1}), (w_{i_1}, l_2, w_{i_2}), (w_{i_2}, l_3, b)\} \subseteq A$ .

5

## Projektivitet

Projektivitet innebär att varje delträd utgör en kontinuerlig delsträng.

Ett dependensträd  $D = (W, A)$  är projektivt om det gäller generellt för alla bågar  $(w_i, l, w_j) \in A$  att

- OM vi har en nod:  $w_k \in W$   
som ligger under bågen, d.v.s.  $i < k < j$  eller  $j < k < i$
- SÅ finns det en riktad väg från (huvudet)  $w_i$  till  $w_k$ .

7

## Dependensanalys (repetition)

- Dependenser: relationer mellan två ord, ett överordnat (huvud) och ett underordnat (dependent).
- Dependenserna kan vara etiketterade (ofta med funktionella begrepp från traditionell grammatik).
- Satslösning ger trädstrukturer med ordförekomster som noder. Inga frasnoder, alltså.
- Fraser ges indirekt genom de delträd som har ett visst ord som rotnod.

2

## Vänster- eller högerbåge?

- Dependensrelationernas riktning: huvud till dependent.
- **Vänsterbåge** (linjärt sett):  
Om  $(w_i, l, w_j) \in A$  och  $w_j < w_i$ ,  
så är huvudet till höger och bågen löper åt vänster.
- **Högerbåge** (linjärt sett):  
Om  $(w_i, l, w_j) \in A$  och  $w_i < w_j$ ,  
så är huvudet till vänster och bågen löper åt höger.

4

## Välformade dependensträd

En dependensgraf  $D = (W, A)$  är ett välformat träd om

- Det finns en rot  $r \in W$ , sådan att det inte finns någon  $(w_i, l, r) \in A$ .
- Det precis en riktad väg mellan (roten)  $r$  och varje annan nod.

6

## En typ av grammatik

Låt oss tänka oss att vi har en grammatik som ger oss satser av typerna  $T \stackrel{\leftarrow}{\Leftarrow} U$  och  $T \stackrel{\rightarrow}{\Rightarrow} U$  som vi tolkar så här:

- $T \stackrel{\leftarrow}{\Leftarrow} U$ : Det är tillåtet med  $(w_j, l, w_i) \in A$  då  $i < j$  (vänsterbåge) och  $w_j$  och  $w_i$  är förekomster av symbolerna  $T$  resp.  $U$ .  
(Underordnad före överordnad, typ *Hon ser...*)
- $T \stackrel{\rightarrow}{\Rightarrow} U$ : Det är tillåtet med  $(w_i, l, w_j) \in A$  då  $i < j$  (högerbåge) och  $w_i$  och  $w_j$  är förekomster av symbolerna  $T$  resp.  $U$ .  
(Överordnad före underordnad, typ *... ser honom.*)

8



## (d) Högerbågesreduktion

- Från:  $(s_1 \dots s_l s_0, w_1 \dots w_{l-1}, A)$ .
- Till:  $(s_1 \dots s_l, w_1 \dots w_{l-1}, A \cup \{(s_l, l, s_0)\})$ .  
( $s_0$  "otillgängligt" därefter, p.g.a. projektiviteten.)
- Villkor:
  - Minst två noder på stacken (som synes) och
  - $T : U : RA : l$  enligt grammatiken, där  $T$  och  $U$  är typerna för förekomsterna  $s_l$  respektive  $s_0$ .

17

## Orakel

- Ett orakel är en funktion från konfigurationer till transitioner.
- Ett orakel kan alltså göra att vi kan tillämpa ett transitionssystem på ett deterministiskt sätt.
- Metafor: Oraklet sitter inne med mer grammatikkunskaper än transitionssystemet.

19

## Systemets egenskaper

- Detta system blir icke-deterministiskt. (Varför?)
- Detta system kan ge alla projektiva dependensträd, om vi struntar i grammatikvillkoren och etiketter. (Det är *fullständigt* m.a.p. den klassen av strukturer.)
- Detta system kan bara ge projektiva dependensträd. (Det är *sunt* m.a.p. den klassen av strukturer.)

18

## Orakel

- Givet ett projektivt dependensträd kan vi, om vi antar att "arc-standard"-systemet producerat det, rekonstruera vilka konfigurationer och transitioner som lett fram till trädet.
- Givet en korpus av sådana träd kan vi ta fram en samling av "korrekta" konfiguration-transitions-par.
- Med sådana data kan vi konstruera orakel med hjälp av maskininlärning. (Och vi kan utvärdera dem med testdata av samma typ.)
- Detta ger en robust "grammatikfri" ingång till parsning (se Nivres artiklar).

20