Cyclic Evaluation

# Rule ordering

## Old idea that grammatical rules have to be ordered

### Passive

#### NP1 V NP2 => NP2 was V+ed by NP1

### Reflexive

#### NP V NP => NP V Xself (if NP = NP and both NPs are in the same clause

### Pass > Refl

#### John likes John => John was liked by John => John was liked by himself

### Refl > Pass

#### John likes John => John likes himself => himself was liked by John

### conclusion

#### passive must precede reflexivisation

# Cycles

## Problem

### John believe Bill like Bill

#### pass > refl

##### J bel [Bill like Bill] => Bill was belived [ to like Bill] by J => Bill was bel [ to like him] by J

#### refl > pass

##### J bel [ B like B] => J bel [B like himself] => B was bel [to like himself] by John

#### Conclusion

##### reflexivisation must precede passive!!!

## Solution – cyclical application

### rules apply in order to most embedded S first then to next S in same order

#### [S1 J bel [S2 B like B]]

##### no passive on S2

##### reflex on S2 = [S1 J bel [S2 B like himself]]

##### passive on S1 = [S1 B was bel [S2 to like himself] by J]

#### Alternatively

##### passive on S2 = [S1 J bel [S2 B to be liked by B]]

##### refl on S2 = [S1 J bel [S2 B to be liked by himself]]

##### passive in S1 = [S1 B was bel [to be liked by himself] by J]

# Cyclic movement in 70s

## The notion of cyclicity came to dominate the theory of bounding – the limitation of distance in movement

## In subjacency it is encoded in the notion of a bounding node: nodes identifying the measuring out of distance of a movement

### no more than one bounding node can be crossed by any one movement

#### who do[S you think [ t [S he likes t]]]

#### \* who do [S you wonder [ why [S he likes t]]]

# Cyclicity in Alignment Syntax

## ... who John likes

### sPp, sAp, whPp, whAp

#### sAp > whAp

#### no side switching, so

##### sPp > sAp and whPp > whAp

## ... who John thinks likes Bill

### whPp<int>, whAp<int>

### but if sAp > whAp<int> we would expect

#### ... John thinks who likes Bill

### if whAp<int> > sPp we would expect

#### ... John who thinks likes Bill

## solution:

### candidates are evaluated in several cycles, each cycle concentrates on the relationships given with respect to a single predicate

### but instead of the traditional ‘bottom’ up cyclicity, the data indicates that the dominating predicates are attended to first

#### superordinate predicate = predicate which takes another as its argument

#### subordinate predicate = predicate which is an argument of another

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| P1 cycle | sPp | sAp | whPp<int> | whAp<int> | oFp |
| ... S P1<int> Wh P2 Obj |  |  | \*! |  |  |
| ... S wh P1<int> P2 Obj |  | \*! |  |  |  |
| =>... Wh S P1<int> P2 Obj |  |  |  | \* |  |
| => ...Obj Wh S P1<int> P2  |  |  |  | \* |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| P2 cycle | sPp | sAp | whPp<int> | whAp<int> | oFp |
| =>... Wh S P1<int> P2 Obj |  | \*\* |  |  |  |
| ...Obj Wh S P1<int> P2  |  | \*\* |  |  | \*! |

## Gaspar’s Mop-up cycle

### coordination of clauses with gaps follows general word order principles:

### John likes – but Mary hates garlic

#### the object follows both predicates

### John likes garlic but – hates pepper

#### the subject precedes both predicates

### Garlic, John likes but – Mary hates

#### topic object precedes both predicates

### languages with different word orders show a similar pattern

#### SOV =

##### S O V and S – V

##### S O V and – O V

#### VSO

##### V – O and V S O

##### V S – and V S O

### Gaspar analysed this as follows

#### cycles of evaluation would order arguments with respect to relevant predicates, but the order of the two clauses is unfixed by these cycles

#### The last cycle deals with unfulfilled requirements – a predicate which lacks a subject or an object it otherwise requires will attempt to satisfy this requirement with respect to some other subject or object

##### S V – and S V O

###### verb with gapped object has an object following it

##### S V O and S V --

###### verb with gapped object has no following object

##### first candidate wins