

Coordination
Ling 233B 2/25/02

1. Two alternative glue approaches
 - Paths as resources
 - Dependency consumption (c.f. control)

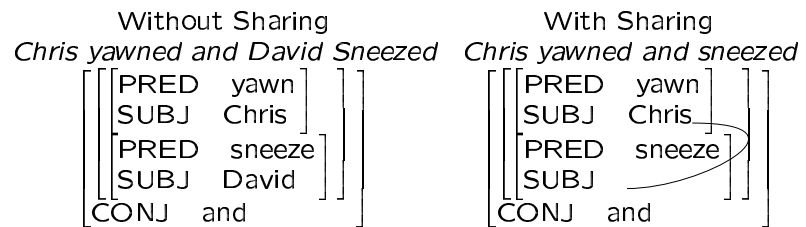
2. Multiple, V, and NP coordination

1

2

Coordination in f-structure

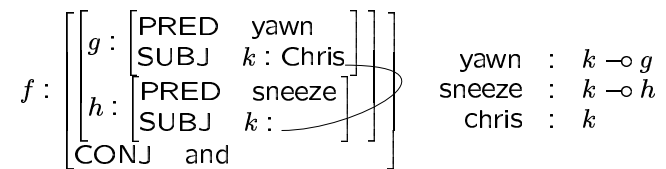
The Basic Problem with Coordination



3

Similar to control:

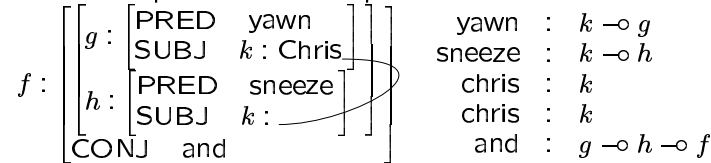
— Multiple dependencies on a single resource



4

1. Kehler et al 1995

- Treat paths through f-structure as resources
- Re-entrant path leads to duplication of resources



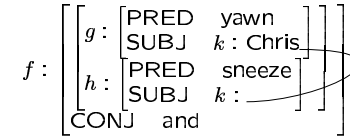
2. Adapt analysis of resource sharing in control

- | | | |
|---|---|---|
| yawn | : | $k \multimap g$ |
| sneeze | : | $k \multimap h$ |
| chris | : | k |
| $\lambda P, Q, x. \text{and}(P(x), Q(x))$ | : | $(k \multimap g) \multimap (k \multimap h) \multimap (k \multimap f)$ |

Worked Example

chris : k
 yawn : $\langle g \text{ SUBJ} \rangle \multimap g$
 sneeze : $\langle h \text{ SUBJ} \rangle \multimap h$
 and : $g \multimap h \multimap f$

- Two R-resources: $R(g, \text{SUBJ}, k)$, $R(h, \text{SUBJ}, k)$
- Axiom: $\lambda X, y. X :!(\forall F, G, P. G_\sigma \multimap !(R(F, P, G) \multimap \langle F \ P \rangle_\sigma \multimap X))$
- First application:
chris : $\langle g \text{ SUBJ} \rangle$
- Second application:
chris : $\langle h \text{ SUBJ} \rangle$
- Can now derive
 $\text{and}(\text{yawn}(\text{chris}), \text{sneeze}(\text{chris}))$



- Represent relation between attribute and its value by an R-relation
 $R(g, \text{SUBJ}, k)$ $R(k, \text{SUBJ}, k)$
- Provide an axiom for promoting R-relations to resources
 - $!(\forall F, G, P, X. G_\sigma \multimap X \multimap !(R(F, P, G) \multimap \langle F \ P \rangle_\sigma \multimap X))$
 - If the set of paths (ending in) G has meaning X , then a resource can be introduced for each (one-step) path ending in G .
 - ! (of-course): turns off resource-accounting for anything within its scope.*

*Axiom presented in old-style glue, since Curry-Howard proof term associated with ! is complicated.

A man yawned and sneezed

a-man : $(k \multimap H) \multimap H$
 yawn : $\langle g \text{ SUBJ} \rangle \multimap g$ $R(g, \text{SUBJ}, k)$
 sneeze : $\langle h \text{ SUBJ} \rangle \multimap h$ $R(h, \text{SUBJ}, k)$
 and : $g \multimap h \multimap f$

- Introduce assumption, $X : k$
- Two applications of axiom:
 $X : \langle g \text{ SUBJ} \rangle$
 $X : \langle h \text{ SUBJ} \rangle$
- Can now derive
 $\text{and}(\text{yawn}(X), \text{sneeze}(X)) : f$
- Discharge assumption and scope quantifier
 $\text{exists}(X, \text{man}(X), \text{and}(\text{yawn}(X), \text{sneeze}(X)))$
- Cannot derive:
 $\text{and}(\text{exists}(X, \text{man}(X), \text{yawn}(X)), \text{exists}(X, \text{man}(X), \text{yawn}(X)))$

Don't duplicate resources: rather, consume dependencies on resources

yawn : $k \multimap g$
 sneeze : $k \multimap h$
 chris : k
 $\lambda P, Q, x. \text{and}(P(x), Q(x))$: $(k \multimap g) \multimap (k \multimap h) \multimap (k \multimap f)$

$$\frac{\lambda P, Q, x. \text{and}(P(x), Q(x)) : (k \multimap g) \multimap (k \multimap h) \multimap (k \multimap f) \quad \text{yawn} : k \multimap g}{\frac{\lambda Q, x. \text{and}(\text{yawn}(x), Q(x)) : (k \multimap h) \multimap (k \multimap f) \quad \text{sneeze} : k \multimap h}{\lambda x. \text{and}(\text{yawn}(x), \text{sneeze}(x)) : (k \multimap f) \quad \text{chris} : k}}{\text{and}(\text{yawn}(\text{chris}), \text{sneeze}(\text{chris})) : f}$$

1. Resource Duplication

- Uses ! (of-course) modality
 Extends fragment of linear logic used (computational consequences?)
- Resource duplication inappropriate for raising/control
David seemed/wanted to leave

2. Dependency consumption

- Some kind of polymorphism required
Chris caught and cooked a fish:
 Two shared e-type arguments
Chris wanted and managed to leave:
 Shared e-type and t-type arguments

$$\frac{\dots : (k \multimap g) \multimap (k \multimap h) \multimap (k \multimap f) \quad \text{yawn} : k \multimap g}{\frac{\lambda \dots : (k \multimap h) \multimap (k \multimap f) \quad \text{sneeze} : k \multimap h}{\lambda x. \text{and}(\text{yawn}(x), \text{sneeze}(x)) : (k \multimap f) \quad \lambda P. \text{exists}(X, \text{man}(X), P(X)) : (k \multimap H) \multimap H}}{\text{exists}(X, \text{man}(X), \text{and}(\text{yawn}(X), \text{sneeze}(X))) : f}$$

Cannot derive:
 $\text{and}(\text{exists}(X, \text{man}(X), \text{yawn}(X)), \text{exists}(X, \text{man}(X), \text{yawn}(X)))$

Extending fragment of linear logic increases complexity of glue proof search

But, axiom using ! is really a mechanism for generating additional (optional) premises

Perhaps axiom can somehow be embedded in mechanism for extracting premises.

Still a problem is some premises are optional:

- Without optional premises, must prove

$$\Gamma \vdash f$$

- With optional premises, must prove

$$\Gamma, \theta \vdash f \otimes \theta$$

Non-determinism about which premises to include in θ

- Do not want to duplicate resources in raising:
Chris seemed to leave
seem(leave(chris))
- Nor for property versions of control
Chris wanted to leave
want(chris,leave)
- Axiom leading to resource duplication is optional (due to !)
 - So can preserve previous treatment of raising and control
 - But will end up with optionally duplicated resources (θ) that must be discarded.

1. Coordination with more than two conjuncts
2. V coordination
3. NP coordination

- Resource duplication is a general approach
 - In case of n shared paths, n resources get duplicated
 - Accounts for e.g. V coordination for all subcat frames
Chris caught & cooked a fish, Chris wanted & managed to leave.
- Dependency consumption requires polymorphism to be general
 - Instead of
 $\lambda P, Q, x. \text{and}(P(x), Q(x)) : (k \multimap g) \multimap (k \multimap h) \multimap (k \multimap f)$
which covers just one case
 - $\lambda P, Q, \vec{X}. \text{and}(P(\vec{X}), Q(\vec{X})) : (\vec{\alpha} \multimap g) \multimap (\vec{\alpha} \multimap h) \multimap (\vec{\alpha} \multimap f)$
Where $\vec{\alpha}$ is a sequence of antecedents
 - Polymorphism used in categorial treatments of conjunction
May be able to compile polymorphic schema to individual cases

Chris ate, drank, and (or) slept

One conjunction, but three conjuncts

1. Kehler et al: *and* produces extra optional premises
 $\lambda P, Q. \text{and}(P, Q) : ![(\uparrow \in)_{\sigma} \multimap \uparrow_{\sigma} \multimap \uparrow_{\sigma}]$

2. Extra premises introduced by comma, or by c-structure.

In implementation, easiest to trigger extra premises non-lexically from each coordinated f-structure node (next slide)

S Coordination

```
sem_template(scoord1, _, _,
  and: [eq: [p: ['$^', '^'], '%conj'],
        not: [fprecedes: ['^', '_']],
        eq: [p: ['%conj', 'COORD-LEVEL'], 'S']],

  &(X, sigma(^) +~> X -* sigma('%conj') +~> X )
).

sem_template(scoord2, _, _,
  and: [eq: [p: ['$^', '^'], '%conj'],
        not: [not: [fprecedes: ['^', '_']]],
        eq: [p: ['%conj', 'COORD-LEVEL'], 'S'],
        eq: [p: ['%conj', 'CONJ'], '%bool']],

  &(X, &(Y,
    (sigma(^) +~> X @ sigma('%conj') +~> Y)
    -*
    sigma('%conj') +~> ['%bool', X, Y]))
).

lex_sem(_, _, 'V', scoord1).
lex_sem(_, _, 'V', scoord2).
```

17

1. Kehler et al: Covered by their existing machinery
Additional argument resources to verbs (OBJ, OBJth, COMP...) duplicated
2. Dependency consumption:
Need to simulate polymorphism

18

V Coordination: Simulating Polymorphism

- For transitive verbs
 $\lambda P, Q, x, y. \text{and}(P(x, y), Q(x, y)) :$
 $[(\uparrow \text{SUBJ}) \otimes (\uparrow \text{OBJ}) \multimap \uparrow] \multimap$
 $[((\in \uparrow) \text{SUBJ}) \otimes ((\in \uparrow) \text{OBJ}) \multimap (\in \uparrow)] \multimap$
 $[((\in \uparrow) \text{SUBJ}) \otimes ((\in \uparrow) \text{OBJ}) \multimap (\in \uparrow)]$
- Need to generalize across various subcat frames
- Either case by case
- Or by exploiting re-entrancy
 $\lambda P, Q, x, y. \text{and}(P(x, y), Q(x, y)) :$
 $\forall A. [(\uparrow \text{SUBJ}) \otimes A \multimap \uparrow] \multimap$
 $[((\in \uparrow) \text{SUBJ}) \otimes A \multimap (\in \uparrow)] \multimap$
 $[((\in \uparrow) \text{SUBJ}) \otimes A \multimap (\in \uparrow)]$
 where A ranges over all second arguments
- Need another entry for 3-argument verbs

19

NP Coordination

General scheme for semantics of NP_1, NP_2, \dots and/or NP_n

```
exists(X, and(group(X),
  and/or(member(NP1, X),
  and/or(member(NP2, X), ...
  ... (member(NPn, X))...))),
  P(X))
```

E.g. *Tom, Dick and Harry sneezed*

```
exists(X, and(group(X),
  and(member(tom, X),
  and(member(dick, X),
  member(harry, X)))),
  forall(Y, member(Y, X),
  sneeze(Y)))
```

20

A man or a woman sneezed

```
exists(X, and(group(X),
  or(exists(M, man(M), member(M,X)),
    exists(W, woman(W), member(W,X)))),
  forall(Y, member(Y,X),
    sneeze(Y)))
```

(We do not want readings where the conjunct quantifiers take wide scope over the conjunction).

Glue Version of Analysis

```
sem_template(npcoord1,_,_,
  eq: [p: ['COORD-LEVEL'], 'NP']

  &(R, &(S, &(H,
    (&(X1, ('VAR'(sigma('^))->X1 -* 'RESTR'(sigma('^))+>[R,X1]))
    @ &(X2, (sigma('^)->X2 -* sigma(H)+>[S,X2])))
    -* sigma(H)+>
      [exists, X\[and,[group,X],[R,X]], X\[S,X]])))
).

sem_template(npcoord2,_,_,
  and: [eq: [p: ['$', '^'], '%conj'],
    not: [fprecedes: [^, _]],
    eq: [p: ['%conj'], 'COORD-LEVEL'], 'NP'],
    eq: [p: ['%conj'], 'COORD-FORM'], '%bool']],

  &(Quant,
    &(H, &(S, &(Y, sigma('^) -> Y -* sigma(H)+> [S,Y])
      -* sigma(H)+> [Quant,S]))
    -*
    &(X, 'VAR'(sigma('%conj')) -> X
      -* 'RESTR'(sigma('%conj')) +> [Quant,Y1\[member,Y1,X]]))
).
```

No man and no woman sneezed

Bad

```
exists(X, and(group(X),
  and(not(exists(M, man(M), member(M,X)),
    not(exists(W, woman(W), member(W,X)))))),
  forall(Y, member(Y,X),
    sneeze(Y)))
```

Better

```
not(
  exists(X, and(group(X),
    or(exists(M, man(M), member(M,X)),
      exists(W, woman(W), member(W,X)))),
    forall(Y, member(Y,X),
      sneeze(Y)))
)
```

Glue Analysis (contd.)

```
sem_template(npcoord3,_,_,
  and: [eq: [p: ['$', '^'], '%conj'],
    not: [not: [fprecedes: [^, _]],
    eq: [p: ['%conj'], 'COORD-LEVEL'], 'NP'],
    eq: [p: ['%conj'], 'COORD-FORM'], '%bool']],

  &(Quant,
    &(H, &(S, &(Y, sigma('^) -> Y -* sigma(H)+> [S,Y])
      -* sigma(H)+> [Quant,S]))
    -*
    &(P,
    &(X, 'VAR'(sigma('%conj')) -> X -* 'RESTR'(sigma('%conj')) +> [P,X])
    -*
    &(X1, 'VAR'(sigma('%conj')) -> X1 -*
      'RESTR'(sigma('%conj')) +>
        ['%bool', [P,X1], [Quant,Y1\[member,Y1,X1]]]))
    ).
```