

Comparative Introduction to Lexicalist Syntactic Theories

**Lascarides & Copestake: Default
Representation in Constraint-based
Frameworks**

Emilia Ellsiepen
millaL7@web.de

Outline

- Motivation of the use of defaults in general
- Requirements on default unification
- Informal overview of YADU
- Definitions for TDFS and default unification
- Linguistic examples

Why defaults?

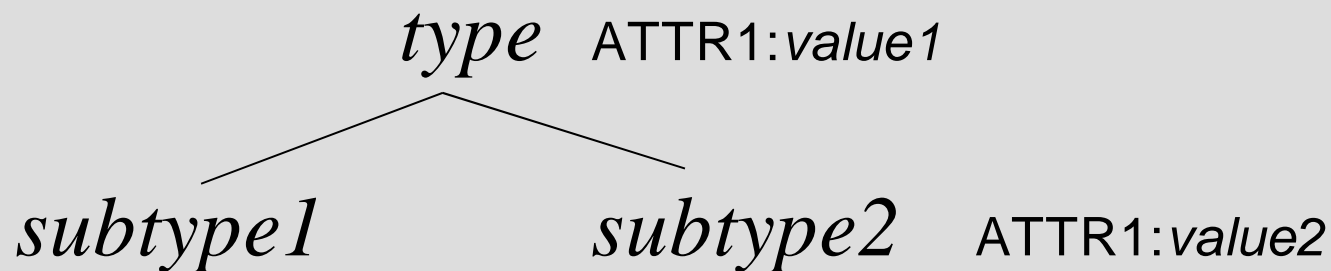
What is a default?

- A generalization with exceptions
=> they are defeasible

Why do we need defaults for linguistic description?

- To be more elegant

What do defaults in a type hierarchy look like?



Where value1 does not subsume value2

An example: past of verbs

Three generalizations:

- Past is formed with suffix *-ed* *defeasible*
- Past participle has the same form as past *defeasible*
- Passive participle has the same form as past participle *indefeasible*

Exceptions:

- Past is formed differently e.g. *sleep-slept-slept*
- Past participle is different from past e.g. *speak-spoke-spoken*

What should default unification do?

- Should make default inheritance possible
- Shouldn't supplant monotonic unification
- Should show similar behaviour as monotonic unification

Criteria

1. Non-default information can be distinguished from default unification and is always preserved
2. Default unification behaves like monotonic unification in the cases where monotonic unification would succeed
3. Default unification never fails unless there is conflict in non-default information
4. Default unification returns a single result, deterministically
5. Default unification can be described using a binary, order independent operation
6. Defaults can be given a precedence ordering such that more specific information overrides less specific information

Problems with previous definitions

Based on Kaplan's priority union Kaplan'87:

- Assymmetric

Young and Rounds'93:

- No precedence between defaults based on specificity
- Used logic is not easy to extend to allow for this

PDU:

- Extension of Young&Rounds using conditional logic
- Meets criterion 6: can impose ordering constraints in terms of specificity
- Fails on reentrancy

Hierarchy for verbs with default and non-default information

[**verb**
PAST : / 1
PASTP : 2 / 1
PASSP : 2 / 1]

|

[**regverb**
PAST : /+ed]

|

[**pst-t-vb**
PAST : /+t]

Two kinds of information incorporated: default and non-default information

- Left of / non-default information
- Right of / default information
- Omit non-default if it is \top
- Omit / default, if it is equal to non-default

Hierarchy of Typed Default Feature Structures (TDFSs)

$$\left[\begin{array}{l} \text{verb} \\ \text{PAST} : \top \\ \text{PASTP} : \boxed{2} \\ \text{PASSP} : \boxed{2} \end{array} \right] / \left\{ \left\langle \left[\begin{array}{l} \text{PAST} : \boxed{1} \\ \text{PASTP} : \boxed{1} \end{array} \right], \text{verb} \right\rangle, \right. \\ \left. \left\langle \left[\begin{array}{l} \text{PAST} : \boxed{1} \\ \text{PASSP} : \boxed{1} \end{array} \right], \text{verb} \right\rangle \right\}$$

|

$$\left[\begin{array}{l} \text{regverb} \\ \text{PAST} : \top \end{array} \right] / \left\{ \left\langle \left[\text{PAST} : +ed \right], \text{regverb} \right\rangle \right\}$$

|

$$\left[\begin{array}{l} \text{pst-t-verb} \\ \text{PAST} : \top \end{array} \right] / \left\{ \left\langle \left[\text{PAST} : +t \right], \text{pst-t-verb} \right\rangle \right\}$$

- TDFS the tail containing the default (defeasible) information and the type on which this default information is defined

Two operations on TDFSs

- \sqcap combines two TDFSs to get a new one containing the information of both
- Usual unification on infeasible part
 - Set union of tails, removing inconsistent information

DefFS computes a default TFS given a TDFS by unifying the infeasible information on the left hand side with as specific information on the right hand side as possible

Result of applying:



DefFS

$$\left[\begin{array}{l} \mathbf{verb} \\ \mathbf{PAST} : \top \\ \mathbf{PASTP} : \boxed{2} \\ \mathbf{PASSP} : \boxed{2} \end{array} \right] / \left\{ \left\langle \left[\begin{array}{l} \mathbf{PAST} : \boxed{1} \\ \mathbf{PASTP} : \boxed{1} \end{array} \right], \mathbf{verb} \right\rangle, \right. \\ \left. \left\langle \left[\begin{array}{l} \mathbf{PAST} : \boxed{1} \\ \mathbf{PASSP} : \boxed{1} \end{array} \right], \mathbf{verb} \right\rangle \right\} \quad \left[\begin{array}{l} \mathbf{verb} \\ \mathbf{PAST} : \boxed{1} \\ \mathbf{PASTP} : \boxed{1} \\ \mathbf{PASSP} : \boxed{1} \end{array} \right]$$

$$\left[\begin{array}{l} \mathbf{regverb} \\ \mathbf{PAST} : \top \\ \mathbf{PASTP} : \boxed{2} \\ \mathbf{PASSP} : \boxed{2} \end{array} \right] / \left\{ \left\langle \left[\begin{array}{l} \mathbf{PAST} : \boxed{1} \\ \mathbf{PASTP} : \boxed{1} \end{array} \right], \mathbf{verb} \right\rangle, \right. \\ \left\langle \left[\begin{array}{l} \mathbf{PAST} : \boxed{1} \\ \mathbf{PASSP} : \boxed{1} \end{array} \right], \mathbf{verb} \right\rangle, \\ \left. \left\langle \left[\mathbf{PAST} : +\mathbf{ed} \right], \mathbf{regverb} \right\rangle \right\} \quad \left[\begin{array}{l} \mathbf{regverb} \\ \mathbf{PAST} : \boxed{1} +\mathbf{ed} \\ \mathbf{PASTP} : \boxed{1} \\ \mathbf{PASSP} : \boxed{1} \end{array} \right]$$

$$\left[\begin{array}{l} \mathbf{pst-t-verb} \\ \mathbf{PAST} : \top \\ \mathbf{PASTP} : \boxed{2} \\ \mathbf{PASSP} : \boxed{2} \end{array} \right] / \left\{ \left\langle \left[\begin{array}{l} \mathbf{PAST} : \boxed{1} \\ \mathbf{PASTP} : \boxed{1} \end{array} \right], \mathbf{verb} \right\rangle, \right. \\ \left\langle \left[\begin{array}{l} \mathbf{PAST} : \boxed{1} \\ \mathbf{PASSP} : \boxed{1} \end{array} \right], \mathbf{verb} \right\rangle, \\ \left\langle \left[\mathbf{PAST} : +\mathbf{ed} \right], \mathbf{regverb} \right\rangle, \\ \left. \left\langle \left[\mathbf{PAST} : +\mathbf{t} \right], \mathbf{pst-t-vb} \right\rangle \right\} \quad \left[\begin{array}{l} \mathbf{pst-t-verb} \\ \mathbf{PAST} : \boxed{1} +\mathbf{t} \\ \mathbf{PASTP} : \boxed{1} \\ \mathbf{PASSP} : \boxed{1} \end{array} \right]$$

Motivation for tails: Order Independence

Non-Commutativity: if we have a default structure on the one side, a non-default structure on the other side

Non-Associativity if we don't keep track of the unifications:

$$\mathbf{t_1} \sqsubset \mathbf{t_2} \sqsubset \mathbf{t_3}$$

$$\mathbf{a} \sqcap \mathbf{b} = \perp$$

$$\mathbf{d} \sqcap \mathbf{b} = \perp$$

$$\mathbf{a} \sqcap \mathbf{d} = \mathbf{c}$$

$$\text{TFS}_1: \left[\begin{array}{l} \mathbf{t_1} \\ \mathbf{F} : / \mathbf{a} \end{array} \right]$$

$$\text{TFS}_2: \left[\begin{array}{l} \mathbf{t_2} \\ \mathbf{F} : / \mathbf{b} \end{array} \right]$$

$$\text{TFS}_3: \left[\begin{array}{l} \mathbf{t_3} \\ \mathbf{F} : / \mathbf{d} \end{array} \right]$$

$$\text{TFS}_1 \hat{\sqcap} \text{TFS}_2 = \left[\begin{array}{l} \mathbf{t_1} \\ \mathbf{F} : / \mathbf{a} \end{array} \right]$$

$$(\text{TFS}_1 \hat{\sqcap} \text{TFS}_2) \hat{\sqcap} \text{TFS}_3 = \left[\begin{array}{l} \mathbf{t_1} \\ \mathbf{F} : / \mathbf{c} \end{array} \right]$$

$$\text{TFS}_2 \hat{\sqcap} \text{TFS}_3 = \left[\begin{array}{l} \mathbf{t_2} \\ \mathbf{F} : / \mathbf{b} \end{array} \right]$$

$$\text{TFS}_1 \hat{\sqcap} (\text{TFS}_2 \hat{\sqcap} \text{TFS}_3) = \left[\begin{array}{l} \mathbf{t_1} \\ \mathbf{F} : / \mathbf{a} \end{array} \right]$$

Definition of a TDFS

Definition 5: Typed Default Feature Structures

A typed default feature structure defined on a set of features Feat , a type hierarchy $\langle \text{Type}, \sqsubseteq \rangle$ and a set of indices N is a tuple $\langle I, T \rangle$ where:

- I is a typed feature structure $\langle Q, r, \theta, \delta \rangle$ on a set of features Feat , a type hierarchy $\langle \text{Type}, \sqsubseteq \rangle$ and a set of indices N , as defined in Definition 2 (i.e., it's a rooted directed acyclic graph);
- T is a tail: that is, it is a set of pairs, where:
 - the first member of the pair is an atomic FS, as defined in (Carpenter 1993); that is, a single path, or a path equivalence; and
 - the second member of the pair is a type;

and

$$\langle F, t \rangle \in T, I \sqcap F \neq \perp$$

Definition of $\overset{\langle \rangle}{\sqcap}$

Definition 7: $\overset{\langle \rangle}{\sqcap}$

Let $F_1 =_{def} I_1/T^1$ and $F_2 =_{def} I_2/T^2$ be two TDFSs, and let $F_{12} =_{def} F_1 \overset{\langle \rangle}{\sqcap} F_2$. Furthermore, assume $F_{12} =_{def} I_{12}/T^{12}$. Then I_{12} and T^{12} are calculated as follows:

1. The Infeasible Part:

$$I_{12} = I_1 \sqcap I_2$$

That is, the infeasible TFS is the unification of the infeasible parts of the arguments.

2. The Tail T^{12} :

$$T^{12} =_{def} (T^1 \cup T^2) \setminus Bot^{12}$$

where Bot^{12} are all the elements T of $T^1 \cup T^2$ where the atomic FS $\wp_{fs}(T)$ is incompatible with I_{12} . That is:

$$Bot^{12} = \{T \in (T^1 \cup T^2) : \wp_{fs}(T) \sqcap I_{12} = \perp\}$$

Credulous default unification

Definition 9: (Carpenter 1993)

The result of credulously adding the default information in G to the strict information in F is given by:

$$F \overset{\leftarrow}{\sqcap}_c G = \{F \sqcap G' : G' \sqsupseteq G \text{ is maximally specific wrt the subsumption hierarchy such that } F \sqcap G' \text{ is defined}\}$$

Extended version (works on sets of atomic FSs):

$$F_1 \overset{\leftarrow}{\sqcap}_{ca} \{G_1, \dots, G_n\} = \{F_1 \sqcap F_2 : F_2 \text{ is the unification of a maximal subset of } \{G_1, \dots, G_n\} \text{ such that } F_1 \sqcap F_2 \text{ is defined}\}$$

Definition 10: Credulous Default Unification $\overset{\leftarrow}{\sqcap}_{cs}$ on Sets

Let \mathcal{F}_1 be a set of TFSs $\{F_1, \dots, F_n\}$, and \mathcal{G}_2 a set of atomic FSs. Then

$$\mathcal{F}_1 \overset{\leftarrow}{\sqcap}_{cs} \mathcal{G}_2 = \{F_1 \overset{\leftarrow}{\sqcap}_{ca} \mathcal{G}_2, \dots, F_n \overset{\leftarrow}{\sqcap}_{ca} \mathcal{G}_2\}$$

Definitions of Specificity Partition and *DefFS*

Let T be a tail. Then μ_1, \dots, μ_m is a Specificity Partition of T if:

$$T = \mu_1 \cup \dots \cup \mu_m$$

and

$$\begin{aligned} \mu_1 &= \{ \langle \phi, t \rangle \in T : \forall \langle \phi', t' \rangle \in T, t' \not\sqsubseteq t \} \\ \mu_i &= \{ \langle \phi, t \rangle \in T : \exists \langle \phi', t' \rangle \in \mu_{i-1} \text{ where } t' \sqsubseteq t \text{ and} \\ &\quad \forall \langle \phi'', t'' \rangle \in T \setminus (\mu_1 \cup \dots \cup \mu_{i-1}), t'' \not\sqsubseteq t \} \quad 2 \leq i \leq m \end{aligned}$$

Definition 12: The Operation *DefFS*

Let F be a TDFS I/T . Then

$$DefFS(F) = \sqcup((I \overset{\leftarrow}{\prod}_{cs} \wp_{fs}(\mu_1)) \overset{\leftarrow}{\prod}_{cs} \dots \overset{\leftarrow}{\prod}_{cs} \wp_{fs}(\mu_n))$$

where $\langle \mu_1, \dots, \mu_n \rangle$ is a specificity partition on T .

Linguistic examples I: english modals

- Usually english modals combine with verb phrases in base form
- ought requires to-infinitive
- Otherwise ought behaves like other modals
- - (6)
 - a. I ought to finish this paper.
 - b. * I ought finish this paper.
 - c. You oughtn't to disagree with the boss.
 - d. Ought we to use this example?
 - e. You are not to argue.
- => make the head value of the first element of the modal's COMPS list base by default

English modals cntd

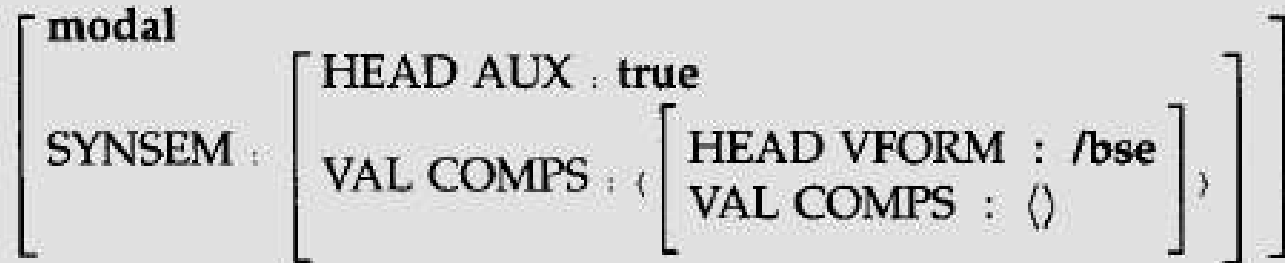


Figure 7
Constraint on type **modal**.

- Notation: /bse in the TFS just means, there is a winning element in the tail, which contains this information
- SYNSEM HEAD AUX *true* is non-default information, which holds for all objects of Type modal

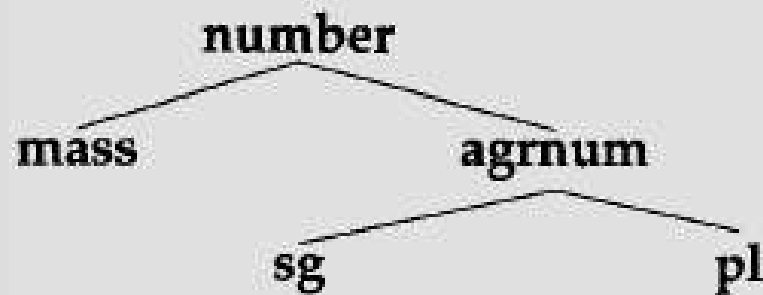
English modals ctnd

Description for <i>could</i> :	$\left[\begin{array}{l} \text{modal} \\ \text{ORTH : could} \end{array} \right]$
Structure for <i>could</i> :	$\left[\begin{array}{l} \text{modal} \\ \text{ORTH : could} \\ \text{SYNSEM : } \left[\begin{array}{l} \text{HEAD AUX : true} \\ \text{VAL COMPS : } \langle \left[\begin{array}{l} \text{HEAD VFORM : bse} \\ \text{VAL COMPS : } \langle \rangle \end{array} \right] \rangle \end{array} \right] \end{array} \right]$
Description for <i>ought</i> :	$\left[\begin{array}{l} \text{modal} \\ \text{ORTH : ought} \\ \text{SYNSEM VAL COMPS : } \langle \left[\text{HEAD VFORM : inf} \right] \rangle \end{array} \right]$
Structure for <i>ought</i> :	$\left[\begin{array}{l} \text{modal} \\ \text{ORTH : ought} \\ \text{SYNSEM : } \left[\begin{array}{l} \text{HEAD AUX : true} \\ \text{VAL COMPS : } \langle \left[\begin{array}{l} \text{HEAD VFORM : inf} \\ \text{VAL COMPS : } \langle \rangle \end{array} \right] \rangle \end{array} \right] \end{array} \right]$

Figure 9
Examples of lexical description for modal verbs.

Linguistic examples III: Agreement and semantic plurality

- Intuition: sg/pl agreement usually, but not always follows the semantics
- Three different values for number, mass, sg,pl:



- Two attributes:
 - PLMOD encodes semantic value of number
 - INST AGR NUM encodes agreement value of number

Agreement and semantic plurality: hierarchy

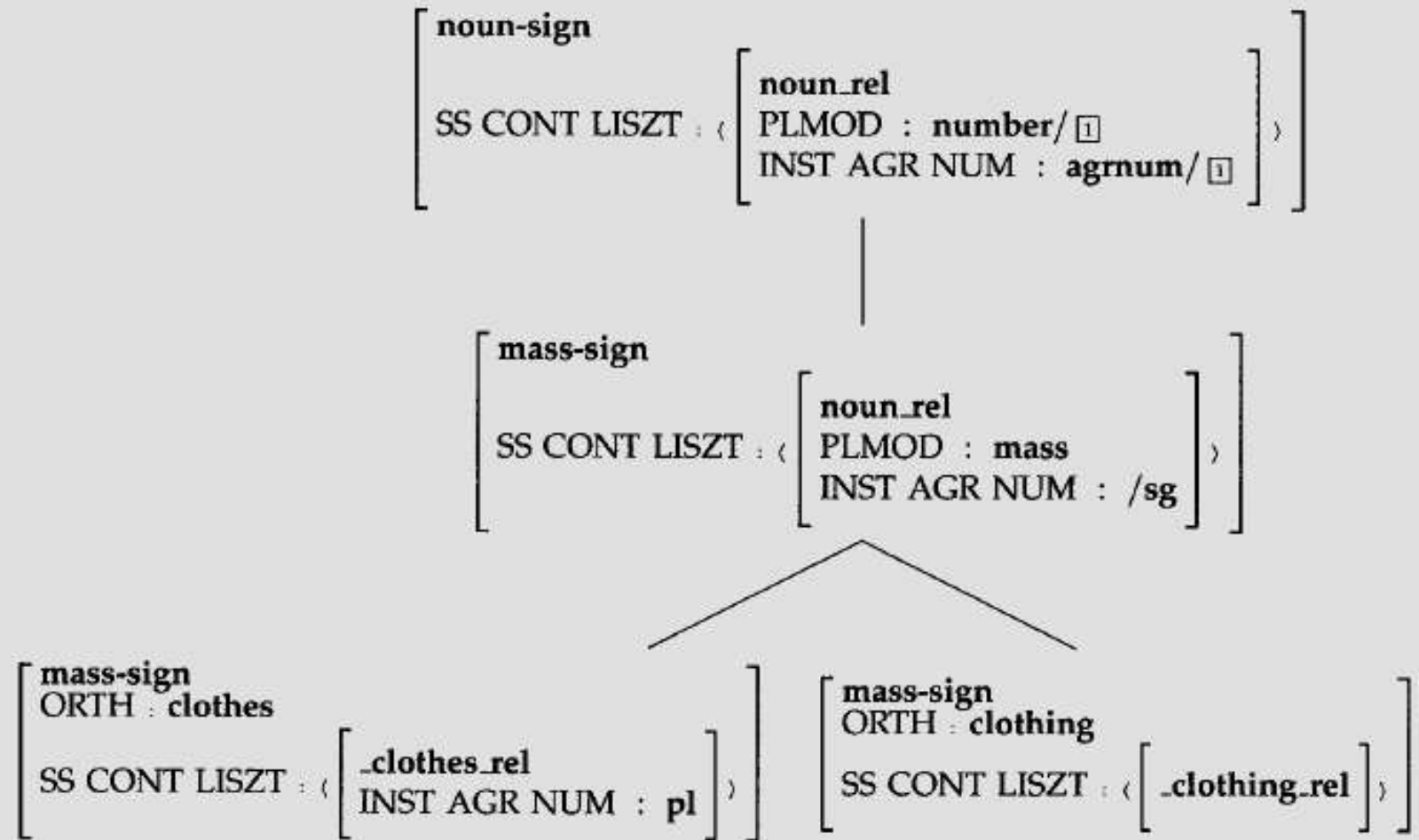


Figure 16

Relating agreement and semantics for nouns (SYNSEM is abbreviated SS).

Agreement and semantic plurality ctnd

$\left[\begin{array}{l} \text{mass-sign} \\ \text{ORTH : clothes} \\ \\ \text{SYNSEM CONT LISZT : } \left(\begin{array}{l} \text{_clothes_rel} \\ \text{PLMOD : mass} \\ \text{INST AGR NUM : pl} \end{array} \right) \end{array} \right]$
$\left[\begin{array}{l} \text{mass-sign} \\ \text{ORTH : clothing} \\ \\ \text{SYNSEM CONT LISZT : } \left(\begin{array}{l} \text{_clothing_rel} \\ \text{PLMOD : mass} \\ \text{INST AGR NUM : sg} \end{array} \right) \end{array} \right]$

Figure 17
Expanded and *DefFilled* lexical signs.

- Default reentrancy of PLMOD and INST ARG NUM on noun-sign doesn't survive on either mass-sign

- /sg survives on clothing, is overridden on clothes

Linguistic examples IV: persistent defaults, lexicon and pragmatics

Transitive verbs in intransitive use:

- (9) a. John drinks all the time.
b. John drinks alcohol all the time.

- (10) a. We've already eaten.
b. We've already eaten a meal.

- (11) a. I spent yesterday afternoon baking.
b. I spent yesterday afternoon baking cookies, cakes, or bread.
(as opposed to ham, apples, or potatoes, for example)

- Both have similar meanings because of implicit object

Persistent defaults ctnd.

Default information can be overridden by pragmatics:

- (12) As long as we're baking anyway, we may as well do the ham now too.
(due to Silverstein, cited in Fillmore [1986])

=> Default which persists as a default beyond the
lexicon

Persistent defaults ctnd.

ORTH : bake
 SYNSEM : [

 VAL COMPS : (

 [

 OPT : true

 HEAD : noun

 VAL COMPS : {}

 CONT : [

 INDEX : \boxed{y}

 LISZT : $\boxed{1}/p \left\langle \left[\begin{array}{l} \text{flour-based_rel} \\ \text{INST : } \boxed{y} \end{array} \right] \right\rangle$

 .

]

]

 CONT LISZT : $\left\langle \left[\begin{array}{l} \text{_bake_rel} \\ \text{ARG1 : } \boxed{x} \\ \text{ARG2 : } \boxed{y} \end{array} \right] \right\rangle \oplus \boxed{1}$

]

]

Summary

- Defaults are useful for elegant linguistic descriptions (english verb past tense, english modals, agreement and semantic plurality)
- Some phenomena can only be described using defaults (transitive vs intransitive with implicit default object)
- YADU is a default unification, which
 - distinguishes default from non-default information
 - is order independent
 - Is deterministic
 - Allows for a precedence ordering of the defaults

Reference: Lascarides, A. and A. Copestake (1999).
Default Representation in Constraint-based
Frameworks. *Computational Linguistics* 25(1), 55-
105.