Intersecting Multiargument Feature Specifications

Sebastian Bank & Daniela Henze*

Abstract

Multiargument agreement poses a problem for the algorithmic learning of inflection: While the invariant morphosyntactic features of a marker can often be computed by intersecting the features of the agreed-with head from all of the markers' occurrences, this technique fails to capture the invariants that are at hand wherever markers agree with varying or even multiple heads. We thus provide an inventory of possible types of morphosyntactic feature specifications that allows to capture all feature invariants to be found in transitive agreement. As we define the intersection operation and subset relations for them, they can be employed in any learning algorithm by means of simple set semantics.

1. Introduction

Algorithmic models of learning inflectional systems essentially extract form-meaning-mappings (markers, lexemes) from a given pool of training data. The training data contain full word forms or sentences paired with their particular meaning and the learning algorithm identifies invariant form-meaning correspondences. If some formal invariant always corresponds to the meaning components of *one specific head* (e.g. a sound sequence/affix always occurring when the subject has a certain person value), its meaning can be captured by determining the invariance in that single head and recording it together with some information that identifies the head (a case feature). As the meaning of say an agreement head is typically represented by an unstructured set of morphosyntactic features, the invariant meaning for different occurrences of that head can be computed by simply intersecting the corresponding feature sets (e.g. Pertsova 2007). However, in systems of multi-argument agreement, markers inflect for *varying* or perhaps even *multiple heads*: For instance, markers may have an 'absolutive' alignment/case (e.g. occur with first person

* The research documented here belongs to the DFG project *Micro- and Macro-variation: Hierarchy Effects in Kiranti and Broader Algic* (TR 521-3). For the thought provoking discussion leading to this paper, we thank Corinna Handschuh.

2 in agreement, 247–265
Sebastian Bank, Doreen Georgi & Jochen Trommer (eds.)
Linguistische Arbeits Berichte 88, Universität Leipzig 2010

intransitive subjects and transitive objects), neutral alignment (occur if any argument is plural), or exhbit some portmanteau-like distribution (first person subject when the object is plural). In learning algorithms using the sketched meaning calculation technique (single head feature intersection plus case features), this sort of meaning invariance cannot be recognized. As we show in section 2, it is not only inadequate to handle such cases with a more complex case feature system (one cannot avoid to adapt the learner), but it also fails to capture the full set of feature invariants that are logically possible.

To lift these restrictions, we give a different solution where the information, to which head(s) given features actually refer is represented by an independent, higher-level component that supports plain set operations: an inventory of *feature specifications*. In section 3 we formally define the range of possible feature specifications that can be built upon intransitives and (mono-)transitives and provide the rules for intersecting and subset testing them. A learning algorithm can build and intersect the feature specifications for all transitive and intransitive occurrences of a marker (or a marker candidate) and thereby compute the most specific invariant, that can be formulated by the presence of feature values from its occurrences. By the subset relation, all possible generalized variants of this maximal specification can then be retrieved. Furthermore, this relation allows to arrange feature specifications into implicational hierarchies of informativity.

2. Learning Inflection

On an abstract level, an inflection learning algorithm (or 'learner') is a function which is given a set of form-meaning pairs (the input, or 'text') from which it produces a set of 'markers' or 'lexemes' (a lexicon) that provides enough information (for a grammar) to identify the right form given the meaning and the right meaning given the form of any item in the input. As non-trivial learners generate lexicons that contain less items than the original input, their effort lies in detecting correspondences between form and meaning properties that apply to multiple input items (generalizations). The challenge hence lies in mapping formal invariants (phonemic features) to meaning invariants (morphosyntactic features) and vice versa. Departures from plain one-to-one correspondences like synonymy (free variation) and particularly homonymy (syncretism) complicate this task. In the following, we will confine ourselves to the meaning side and exemplify, how to generalize a single meaning from

different occurrences of a form and show what problems arise with multi-argument agreement. Hence, we can safely ignore the interdependence with the identification of unique forms (e.g. segmenting phoneme sequences into affix strings or identifying prosodic invariances like umlaut) for the moment and leave this problem to concrete algorithms.

2.1. The Meaning of a Marker

A small fragment of the English pronoun paradigm serves as a simple example:

(1) *Determining the invariant meaning of a syncretic marker by intersection*

a.

	sg	**pl**
1	I	we
2	you	you

b.

form	occurrences	invariance
I	[1,sg]	[1,sg]
we	[1,pl]	[1,pl]
you	[2,sg],[2,pl]	[2]

The paradigm in (1a) contains the *four* distinct form-meaning pairs representing the input (*I*-1SG, *we*-1PL, *you*-2SG, *you*-2PL), while the table in (1b) records the meanings associated with all occurrences for each of the *three* distinct forms. The last column with the invariant meaning of the markers is computed by intersecting the feature sets from their occurrences. In the case of *I* and *we* there is exactly one occurrence, so the (trivial) 'invariance' is *identical* to the meaning of that single occurrence. In case of the syncretic *you* the intersection of [2,sg] and [2,pl] yields the underspecified meaning [2] for the marker. This meaning is both *necessary* and *sufficient* for this form in the current paradigm:

(2) a. *you* → [2] b. *you* ← [2]

All instances of *you* have a second person meaning (2a), and it is the only marker, that is second person, so every time there is a second person feature in the meaning, the form *you* occurs (2b). As all computed meanings are necessary and sufficient (↔) for their form, the result (3) is a valid lexicon for the given data (i.e. it allows to derive all forms from their meaning and vice versa).

(3) a. *I* ↔ [1,sg] b. *we* ↔ [1,pl] c. *you* ↔ [2]

To quickly examine the impact of different syncretism types on the intersected meaning and the relation between the calculated meaning and the occurrence of the form, we next look at two abstract examples naming the forms A to G:

(4) *Different marker distributions and their invar. meaning by intersection*

a. (i)

	sg	**pl**
1	A	D
2	B	D
3	C	C

(ii)

	sg	**pl**
1	E	F
2	E	G
3	G	G

b.

form	occurrences	invariance	relation
A	[+1,+sg]	[+1,+sg]	=
B	[+2,+sg]	[+2,+sg]	=
C	[+3,+sg],[+3,+pl]	[+3]	↔
D	[+1,+pl],[+2,+pl]	[−3,+pl]	↔
E	[+1,+sg],[+2,+sg]	[−3,+sg]	↔
F	[+1,+pl]	[+1,+pl]	=
G	[+2,+pl],[+3,+sg],[+3,+pl]	[−1]	→

For the non-sycretic markers A, B and G, the 'calculated' meaning is again identical to the meaning of their single occurrence (=).[1] For the syncretic markers with a 'rectangular' distribution (C, D and E), the intersected meaning is both necessary and sufficient (↔) for their occurrence: As the syncretism field of C spans over all number values, it is captured by the calculated full neutralization/underspecification of number, while for D and E negative feature values, feature decomposition or other means of expressing partial information (non-third person) have to be employed. For the remaining, default-like distributions like the 'L-shaped' one of G, the intersection only yields their necessary but not sufficient features (→): Every occurrence of G is second or third person (5a), but in the non-first person there is also marker E (5b), so an implication from non-first-person to G (5c) does not hold (is not true for all cases):

(5) a. $G \rightarrow [-1]$ b. $E \vee G \leftarrow [-1]$ c. $G \not\leftarrow [-1]$

[1]Note that such trivial invariance is both necessary and sufficient by definition.

To derive the right forms from the calculated meaning of this marker, it thus has to be ensured that only E and not G ends up as the form for the second person singular case (where in principle both their meanings match). So the component (the grammar) that picks the matching marker entry from the lexicon for a given form or meaning needs to be adapted/explicated: It may adhere to some provided (extrinsic) precedence ordering (e.g. E blocks G), and/or resolve the competition between multiple matching marker entries by their (intrinsic) properties (like e.g. the subset principle does). With such adjustments, the information computed in (4b) again ends up as valid lexicon for the given data.

2.2. Multiargument Agreement

So far, we silently assumed that all forms in the input covary with the properties (morphosyntactic features) of the same single syntactic head. Hence in the training data, there was no need to explicitly specify which head was exactly meant (e.g. subject or object) – the task for the leaning algorithm was to solely determine meaning invariance *within* that single head. With transitive agreement though – markers possibly agree with different or even multiple heads –, the input items need to specify which head has which features for each form. We will try to use the common case features S (intransitive subject), A (transitive subject), and P (transitive object) to represent this information within the feature sets. Consider the following abstract intransitive (6a) and transitive (6b) paradigm parts (subjects features notated in the rows, object features in the columns) with the markers A to F:

(6) a.

S	
1	A
2	B
3	C

b.

A→P	1	2	3
1	A	B	EF
2	AB	B	BDF
3	AC	BCD	CF

The meaning associated with the single occurrence of the only non-syncretic form E (and its identical invariant) is represented as follows:

(7)

form	occurrences	invariance
E	[A,+1][P,+3]	[A,+1][P,+3]

Note that this notates E as a true portmanteau marker specifying a person feature value for both transitive subject and object: The specification [A,+1][P,+3] represents a logical conjunction of the statements 'the transitive subject is first person' and 'the object is third person'.[2] The invariant meaning for the transitive-only marker F, may be calculated by separately intersecting the subjects' and the object features found in all its occurrences:

(8)

form	occurrences	invariance
F	[A,+1][P,+3], [A,+2][P,+3], [A,+3][P,+3]	[A][P,+3]

As the marker's occurrence is not sensitive to the subject's features, the invariant meaning doesn't specify any substantial features (is underspecified) for the subject: The transitive subject case feature [A] in the result doesn't refer to any (person) feature and should thus rather be ignored. The result represents the statement that the transitive object is third person, which is both necessary and sufficient for F.

The remaining markers all exhibit distributions, where the invariant features span over different heads. To recognize such generalizations via intersections we need to both adapt the case feature system and the intersection of multi-head feature structures: Additional case features or a feature decomposition must account for all possible combinations/alignments of cases (SA, SP, AP, SAP) and the intersection needs to consider the different possible ways of combining feature specifications with multiple heads.[3] For example for a marker that occurs in the two cases represented by (9a), to retain both invariant features (first person 'nominative' and singular 'absolutive'), the case features are extended by the highlighted combined features in (9b) and then the results of both ways of intersecting the intransitive (9b-i) with one of the transitive heads from (9b-ii) are combined (by logical conjunction) to form (9c):

(9)
a. (i) [S,+1,+sg] (ii) [A,+1,+pl][P,+3,+sg]
b. (i) [S,**SA**,**SP**,+1,+sg] (ii) [A,**SA**,+1,+pl][P,**SP**,+3,+sg]
c. [SA,+1][SP,+sg]

[2] Models with a more restricted format for notating the meaning of a marker might translate such an invariant to a specification using a contextual restriction (e.g. either 'first person transitive subject in the context of second person object' or the other way around) or also completely avoid specifying more than one heads' features by using independent machinery like blocking rules or impoverishment to implement such distributions.

[3] Note that we leave out some of the combined case features in the following for readability.

It may be worth to point out, that such additions that allow to recognize markers of all possible case alignments may technically not be necessary in systems with single-argument agreement: If all markers obey one fixed alignment for the language, this can be implemented by initially having the right combined case features on the syntactic side. Yet, in systems with multi-argument agreement, we typically face markers of different alignment types, so this complexity demands for additional machinery to find all invariant features.

Coming back to the remaining markers from (6), the introduced changes to the case features and the intersection operation for the learner suffice to calculate the invariant features, that are both necessary and sufficient for the occurrence of the absolutive first person marker A:

(10)

form	occurrences	invariance
A	[S,SA,SP,+1], [A,SA,+1][P,SP,+1], [A,SA,+2][P,SP,+1], [A,SA,+3][P,SP,+1]	[SA][SP,+1]

Marker A occurs if and only if the intransitive subject or transitive object (SP) is first person. The nominative (SA) third person distribution of marker C is fully captured as well:

(11)

form	occurrences	invariance
C	[S,SA,SP,+3], [A,SA,+3][P,SP,+1], [A,SA,+3][P,SP,+2], [A,SA,+3][P,SP,+3]	[SA,+3][SP]

Marker B exhibits the distribution of a marker, that is underspecified for case (or has a neutral alignment): It occurs as soon as any argument is second person. This is another type of invariant that spans over different heads. To capture it, it is again crucial to intersect each head of a specification with each head of the other specifications that it is intersected with.

(12)

form	occurrences	invariance
B	[S,SA,SP,+2], [A,SA,+1][P,SP,+2], [A,SA,+2][P,SP,+1], [A,SA,+2][P,SP,+2] [A,SA,+2][P,SP,+3], [A,SA,+3][P,SP,+2]	[+2][]

While this delivers the necessary and sufficient invariants for the kinds of markers we have shown so far, it also has to be considered quite inadequate: The different nature of the case features on the one and the substantial features on the other side is not accounted for in the representation, but rather results

in additional complications (e.g. to ignore feature sets that solely consist of case features). From a logical point of view, the case features implement higher grade *meta* information on the substantial features, as the case determines the scope to which the substantial information applies (to which head(s)). Furthermore, it is undesirable for learning algorithms to employ a special new intersection technique for feature specifications from transitive agreement (the need to intersect all possible combinations). So these specifics needed to account for multi-argument agreement are better factored out into an independent component, that provides the same simple set-like interface that learning algorithms use for cases with a single head only.

Ultimately, the sketched system fails to fully capture the distribution of markers like D. Being a 'symmetric' portmanteau, D occurs in transitives where one argument is second and the other is third person:

(13)

form	occurrences	invariance
D	[A,+3][P,+2], [A,+2][P,+3]	[+3][+2]

For markers with such distributions the necessary and sufficient invariant is, that one (transitive) head has one property and the *non-identical* remaining head has another one. The crucial point is, that there is no occurrence, where both properties are found on the same head, which of course can be the case, if the properties are compatible – e.g. a person and a number feature or negative feature values. The representation in (13) doesn't capture the non-identity requirement of the two heads: Following the intersection operation used so far, intersecting [+2][+3] with [+3,+2] would yield [+2][+3] again (i.e. [+2][+3] is a subset of/implied by [+3,+2] in this notation). Without case features, there is hence no semantic difference between a two-head feature specification ([x][y]) and a single-head specification with the features of both ([x,y]) in the current notation. To disambiguate these cases, the system would need to be further extended to notate (non-)identity of heads and the intersection operation again needed to be adapted to that. The notations we develop in the next section will account for all this while providing the same clean interface for all cases.

3. Feature Specifications for Multiargument Agreement

The specifications we define serve to capture all possible invariants to be found in the feature values for intransitives and monotransitives. They basically

state the presence of features values in the three heads under consideration: intransitive subject (S), transitive subject (A), and transitive object (P). While feature *values* represent true statement on the properties of a head (14a) and feature *sets* refer to logical conjunctions of the included statements (14b), feature *specifications* represent statements on the scope of such statements (14c).

(14) *Feature values, sets and specifications with their equivalent statements*

a. +1 = The head is first person.
−du = The head has non-dual number.

b. [+1,−du] = The head is first person and has non-dual number.

c. $_{SA}$[+1,−du] = For the intransitive or transitive subject it is true that it is first person and has non-dual number.

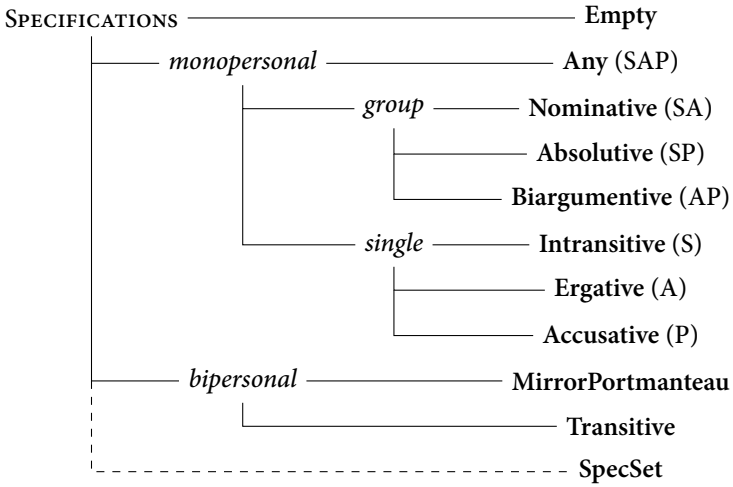
A learner can build a specification for every single occurrence of a form and then intersect these. The resulting specification (or specification set) then represents the feature properties, that are true for all occurrences of the form – its necessary condition. By the provided semantics for the implication relation, it can easily be determined, if the result is also sufficient or if additional information is needed to derive all forms from the calculated meaning. The implication can furthermore be used to choose among different generalized but still necessary meanings.

As the specifications determine the scope of given (substantial) feature values, we define them as functions taking (none, one, or two) non-empty sets of feature values as arguments. The single specification without an argument is the trivial case of an empty specification:

(15) **Empty**() ⇔ [ ]$_{\varnothing}$
This is the empty specification.

Specifications that take one argument are *monopersonal*. They express that the given features are always present on one specific head (S, A, P), across a specific group of two heads (SA, SP, AP), or on any of the three heads (SAP). Specifications with two arguments are *bipersonal*. As they simultaneously postulate the presence of features for two distinct heads, they can't be found in intransitives. They serve to represent the full specification of transitives and may also be used to describe the meaning of all kinds of portmanteau markers. This gives the basic distinctions for a simple type hierarchy of the different kinds of specifications we will define:

(16)



All instances of specifications support the basic operation of intersection (∩) yielding the logically strongest specification the intersected items have in common – or the empty specification in case they have nothing in common:

(17) a. Intransitive([+1]) ∩ Ergative([+1]) = Nominative([+1])
b. Nominative([+3,−pl]) ∩ Accusative([+3,+pl]) = Any([+3])
c. Intransitive([+2]) ∩ Transitive([+3],[+2]) = Absolutive([+2])
d. Transitive([+2,+pl],[+3,+pl]) ∩ Any([+1,−pl]) = Empty()

The intersection – as well as the strength comparison – is based on an implication (or subset) relation (→) between specifications. Together with the intersection of the feature sets given as function arguments, this defines the computations in (17). By the implication relation, specifications are arranged into a partial order. This allows for equivalence checking (two way implication), checking if a marker is to be inserted in a given environment (implication from environment to marker specification), and finally to construct hierarchies of possible generalizations of the meaning for a marker:

(18) a. MirrorPortmanteau([+1],[+pl]) ↔ MirrorPortmanteau([+pl],[+1])
b. Transitive([+1],[+3]) → Nominative([+1])
c. Ergative([+pl])
→ Nominative([+pl]), Biargumentive([+pl])
→ Any([+pl])

Furthermore, if multiple markers compete for insertion, this comparison may contribute to determining the specificity relations between them – e.g. instead of defining arbitrary specificities for case features. In (18c) for instance, the first specification is the most specific one – it implies both specifications on the second line, which both imply the most generic last one.

3.1. Monopersonal Specifications

The strongest specifications referring to a single argument simply state, that the given features are found on a single out of the three possible heads:

(19) a. **Intransitive(x)** ⇔ $[\ldots]_{\mathrm{S}}$
The intransitive subject has the feature(s) *x*.
b. **Ergative(x)** ⇔ $[\ldots]_{\mathrm{A}}$
The transitive subject has the feature(s) *x*.
c. **Accusative(x)** ⇔ $[\ldots]_{\mathrm{P}}$
The transitive object has the feature(s) *x*.

The next subtype consists of the three possibilities to form a less specific combined case out of two of them:

(20) a. **Nominative(x)** ⇔ $[\ldots]_{\mathrm{SA}}$
The intransitive subject or transitive subject has the feature(s) *x*.
b. **Absolutive(x)** ⇔ $[\ldots]_{\mathrm{SP}}$
The intransitive subject or transitive object has the feature(s) *x*.
c. **Biargumentive(x)** ⇔ $[\ldots]_{\mathrm{AP}}$
The transitive subject or transitive object has the feature(s) *x*.

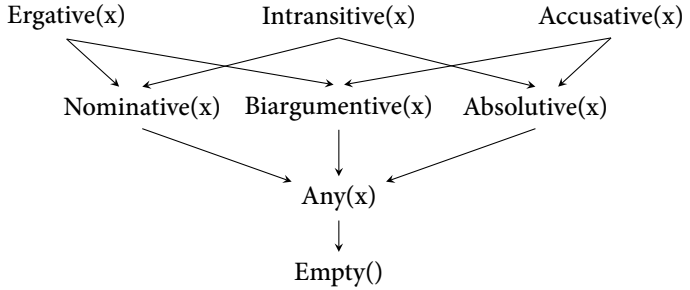
The weakest monopersonal specification finally states, that the features are found on any of the three heads:

(21) **Any(x)** ⇔ $[\ldots]_{\mathrm{SAP}}$
There is an argument that has the feature(s) *x*.

Because (20) and (21) are built by combining the cases from (19) and the empty specification is implied by all specifications, it is easy to identify the exact implication relations between the specification types hitherto defined. Their rules of intersection can then be briefly described with a graph showing these implications: If two specifications from (22) are intersected, the resulting

specification type is the nearest one, that can be reached by going down from both (summing the steps). The feature values result from plain intersection.

(22)



If both specifications have the same type, the resulting type trivially doesn't change – it's the nearest one, e.g. (23a). If one of them is (directly or indirectly) implied by the other one, the resulting type is of the implied, less specific one (23b).[4] All remaining cases are combinations of nodes from one of the two uppermost 'tiers' in (22).

(23) a. Ergative([+1,+pl]) ∩ Ergative([+pl]) = Ergative([+pl])
b. Intransitive([+2]) ∩ Any([+2,−pl]) = Any([+2])
c. Ergative([+pl]) ∩ Intransitive([+3,+pl]) = Nominative([+pl])
d. Nominative([+3,−pl]) ∩ Accusative([+3,−pl]) = Any([+3,−pl])

By these relations, intersecting Ergative(x) with Accusative(x) yields Biargumentive(x), because it is the only one directly reachable from both. If this specification type is removed from (22), the nearest crossing point defining the result would be Any(x) instead.

3.2. Bipersonal Specifications

For monotransitives, there are two possible ways to specify feature invariants for both heads at once: Either it is fully specified, to which argument the two feature sets belong, or it is left undetermined for both.[5] The former is captured

[4] In fact, the behaviour of identical and indirectly implied items relate to the reflexivity and the transitivity of the implication relation.

[5] A specification system accounting for ditransitives also needed to define the three possibilities of only specifying the exact head for one feature set in tripersonal specifications and furthermore additional bipersonal specifications for the possible groupings of three heads.

by a logical conjunction of an Ergative(x) and an Accusative(x) specification whose arguments are referring to different heads:

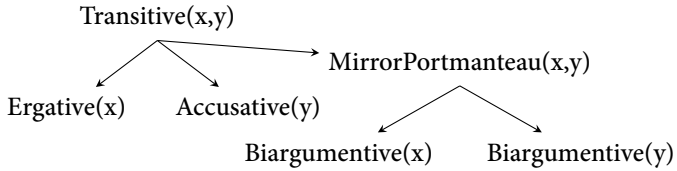
(24) **Transitive(x,y)** ⇔ $[[\ldots][\ldots]]_{\mathbf{A\neq P}}$
The transitive subject has the feature(s) *x*
and the transitive object has the feature(s) *y*.

The latter then is a conjunction of two Any(x) or rather Biargumentive(x) specifications for distinct heads:

(25) **MirrorPortmanteau(x,y)** ⇔ $[[\ldots][\ldots]]_{\mathbf{AP\neq AP}}$
There is an argument that has the feature(s) *x*
and there is a different argument that has the feature(s) *y*.

Thus, the order of the two arguments is significant for Transitive(x,y), while it is insignificant by definition for MirrorPortmanteau(x,y) – as already stated in (18a). As both bipersonal specifications contain two monopersonal specifications and (24) is clearly a more specific version of (25), their implication relations are as follows:

(26)



This already suggests, how the intersection of a bipersonal and a monopersonal (or empty) specification *M* is to be computed: *M* is intersected with the included specifications for both arguments and then both results must be combined, which we will notate as union (∪):

(27) a. **Transitive(x,y)** ∩ **M(z)** :=
(Ergative(x) ∩ M(z)) ∪ (Accusative(y ∩ M(z))
b. **MirrorPortmanteau(x,y)** ∩ **M(z)** :=
(Biargumentive(x) ∩ M(z)) ∪ (Biargumentive(y) ∩ M(z))

The union is needed to retain the feature invariants from both heads at once:

(28) a. Transitive([+1,+pl],[+3,−pl]) ∩ Absolutive([+1,−pl]) =
Any([+1]) ∪ Absolutive([−pl])

b. Transitive([+1,+pl],[+3,−pl]) ∩ Intransitive([+1,−pl]) =
Nominative([+1]) ∪ Absolutive([−pl])

Furthermore this allows to unambiguously state all possible underspecified assignments of multiple feature values onto the given heads – same head, different heads, and finally same or different head:

(29) Any([+2,+pl]) ∩ MirrorPortmanteau([+2],[+pl]) =
Any([+2]) ∪ Any([+pl])

The intersection of two bipersonal specifications is again computed by the union of two possible ways to combine them (straight and crossed). The result only contains another bipersonal specification, if (either way) both arguments have a common subset at the same time.

(30) a. **Transitive(x_1,y_1)** ∩ **Transitive(x_2,y_2)** :=
Straight(x_1∩x_2,y_1∩y_2) ∪ Crossed(x_1∩y_2,y_1∩x_2)
b. **MirrorPortmanteau(x_1,y_1)** ∩ **Bipersonal(x_2,y_2)** :=
Crossed(x_1∩x_2,y_1∩y_2) ∪ Crossed(x_1∩y_2,y_1∩x_2)
c. **Straight(x,y)** :=
Transitive(x,y) if $x \neq \varnothing$ and $y \neq \varnothing$
Ergative(x) if $x \neq \varnothing$ and $y = \varnothing$
Accusative(y) if $x = \varnothing$ and $y \neq \varnothing$
Empty() if $x = \varnothing$ and $y = \varnothing$
d. **Crossed(a,b)** :=
MirrorPortmanteau(a,b) if $a \neq \varnothing$ and $b \neq \varnothing$
Biargumentive(a) if $a \neq \varnothing$ and $b = \varnothing$
Biargumentive(b) if $a = \varnothing$ and $b \neq \varnothing$
Empty() if $a = \varnothing$ and $b = \varnothing$

Due to the implication from Transitive(x,y) to MirrorPortmanteau(x,y), the intersection can only yield the former specification type if all intersected items are of this logically stronger type.

(31) a. Transitive([+3,+sg],[+2,+pl]) ∩ Transitive([+3,+pl],[+1,+pl]) =
Transitive([+3],[+pl])
b. Transitive([+sg],[+pl]) ∩ MirrorPortmanteau([+1,+sg],[+2,+pl]) =
MirrorPortmanteau([+sg],[+pl])

c. Transitive([+1,+du],[+3,+pl]) ∩ Transitive([+2,+pl],[+1,+pl]) =
MirrorPortmanteau([+1],[+pl]) ∪ Accusative([+pl])

The rules for implications between bipersonal specifications are already implicitly defined by their intersection instructions: A Transitive(x,y) specification implies another one if both matched argument pairs are in the subset (or implication) relation, while a MirrorPortmanteau(x,y) is also implied by another bipersonal specification if the intersection holds with crossed arguments:

(32) a. **Transitive(x_1,y_1)** → **Transitive(x_2,y_2)** iif
$x_1 \rightarrow x_2$ and $y_1 \rightarrow y_2$
b. **Bipersonal(x_1,y_1)** → **MirrorPortmanteau(x_2,y_2)** iif
($x_1 \rightarrow x_2$ and $y_1 \rightarrow y_2$) or ($x_1 \rightarrow y_2$ and $y_1 \rightarrow x_2$)

As mentioned earlier, the mirror (or symmetric) portmanteau specification is restricted to the cases, where it is undecidable, which head the given features exactly refer to. Thus, a specification like (33) with a common non-empty subset of the two feature sets is not well-formed – the common subset mandatory refers exactly to the transitive subject and object. In such cases, the common subset has to be 'factored out' into a separate (more specific) Transitive(x,y) specification, or some implication relations may not be recognized:

(33) *MirrorPortmanteau([+3,+pl],[+3]) ⇔
Transitive([+3],[+3]) ∪ Biargumentive([+3,+pl])

Similarly, symmetric portmanteaus whose two arguments could not refer to a single head simply because they are incompatible can't differ from the combination of two Biargumentive(x) specifications and have to be converted to represent this logically weaker status:

(34) *MirrorPortmanteau([−pl],[+pl]) ⇔
Biargumentive([−pl]) ∪ Biargumentive([+pl])

The same finally holds for symmetric portmanteau specifications with two feature sets, that are compatible but only, because all values are non-orthogonal and thus compatible by definition (e.g. negative features of a single category):

(35) *MirrorPortmanteau([−1],[−3]) ⇔
Biargumentive([−1]) ∪ Biargumentive([−3])

With these special cases in mind – they ultimately depend on definitions in the concrete feature system used – all implicational relations between specifications are defined. This can be used to create redundancy-free sets of feature specifications.

3.3. Specification Sets

As seen in the last section, any intersection with a bipersonal specification yields a set of specifications which retains feature invariants for both heads. So far, this has been represented by the union operator. The implication relation allows us to define a potentially more compact representation of such unions:

(36) **SpecSet(…)** $\Leftrightarrow \{\ldots\}_{\not\Rightarrow} \Leftarrow$ **Spec$_1$** $\cup$ **Spec$_2$** … $\cup$ **Spec$_n$**
Set of logically conjuncted pairwise non-implying specifications

The environments (or the set of paradigm cells) matched by a SpecSet() is the environment matched by all of its members (the intersection of their cell sets). A SpecSet() without members is equivalent to the Empty() specification, and a single specification is equivalent to a singleton SpecSet() with this specification as its only member. Being pairwise non-implying removes all redundancy – the sets are reduced to the logically independent minimum:

(37) Transitive([+1,+sg],[+3,+pl]) ∩ Transitive([+1,+sg],[+3,+sg]) =
Transitive([+1,+sg],[+3]) ∪ Biargumentive([+sg]) =
SpecSet(Transitive([+1,+sg],[+3]))

Furthermore, we can straightforwardly perform intersection and implication checking with more than two specifications by defining them on SpecSets:

(38) a. **SpecSet(a_1,a_2,…,a_n)** ∩ **SpecSet(b_1,b_2,…,b_m)** :=
SpecSet(a_1∩b_1,a_1∩b_2,…,a_1∩b_m,a_2∩b_1,a_2∩b_2,…,a_n∩b_m)
b. **SpecSet(a_1,a_2,…,a_n)** → **SpecSet(b_1,b_2,…,b_m)** iif
for every b ∈ b_1,b_2,…,b_m there is an a ∈ a_1,a_2,…,a_n such that a → b

As by (38a) sets of specifications are intersected by a union of the intersections of all possible combinations of pairs (the Cartesian product), the Transitive(x,y) specification can now also be expressed by an equivalent SpecSet containing both included monopersonal specifications:

(39) Transitive(x,y) ↔ {Ergative(x),Accusative(y)}$_{\leftrightarrows}$

While this makes the definition of the transitive specification as primitive notion formally redundant, this crucially does not hold for the symmetric portmanteau: A specification set of its included monopersonal specifications does not include the restriction, that both have to refer to distinct heads. Thus, this set is only a logically weaker, implied but not equivalent version of the symmetric portmanteau:

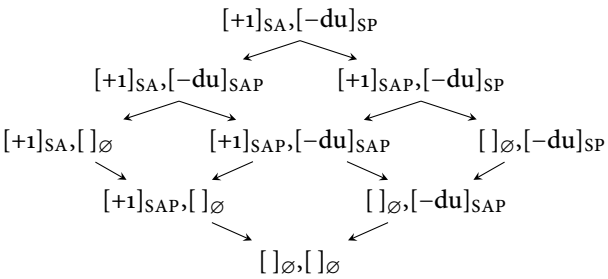
(40) MirrorPortmanteau(x,y) → {Biargumentive(x),Biargumentive(y)}$_{\leftrightarrows}$

This finally allows us to compute intersections on arbitrary sets of specifications. This is the way to capture all feature invariants found in the different occurrences of a marker:

(41) ⋂{Intransitive([+1,−du,−pl]),
Intransitive([+1,−du,+pl]),
Transitive([+1,−du,−pl],[+3,−du,+pl]),
Transitive([+1,+du,−pl],[+3,−du,+pl]),
Transitive([+1,−du,+pl],[+3,−du,+pl])} =
{Nominative([+1]),Absolutive([−du])}$_{\leftrightarrows}$

The partial order of the implication relation allows to build a hierarchy of possible generalizations of such a complex specification set. Here, every 'tier' of equally ordered specification sets is generated by applying generalization of exactly one member of the mother node:

(42)



If applied to specifications with multiple features, the generalization hierarchy – apart from type generalization – of course also includes the different possibilities to delete exactly one feature information from the mother node. Note that (42)

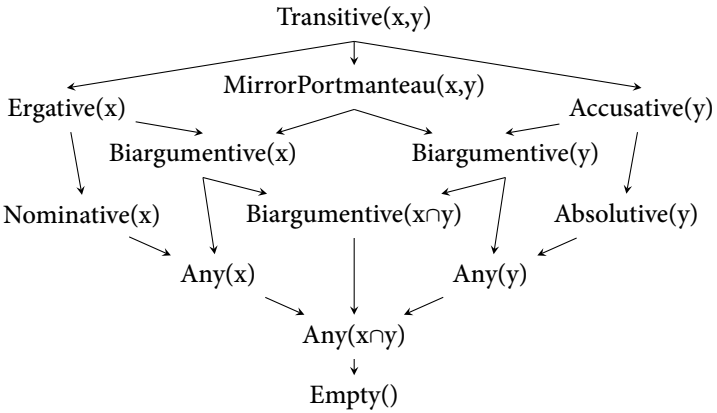
does not contain the specification $[+1,-du]_{SAP}$, because this invariant does not hold, as it is contradicted by Transitive([+1,+du,−pl],[+3,−du,+pl]) from (41). This may be clarified by another example for the implication relations between a monopersonal specification and a specification set built from a partition of its features – also see (29):

(43) Any(+1,+2,−sg,+pl) → {Any(+1),Any(+2,−sg),Any(+pl)}$_{\leftrightarrows}$

The atomic specification is logically stronger, because it is only fulfilled, if all features are found on the same head, while this is sufficient but not necessary for the invariant formalized by the weaker specification set.

Finally, (44) provides the missing full visualization of the possible generalization paths for bipersonal specifications:

(44)



4. Summary

Virtually any algorithm of learning is based on the identification of invariants. On a very abstract level, learning inflectional systems can be summarized as search for bidirectional mappings of formal invariants and meaning invariants. In case of inflectional markers, their meaning is typically represented as sets of feature values for the morphosyntactic contrasts that are distinguished in a given language. With multiple of such unstructured sets from different occurrences of a marker, the invariant meaning is easily determined by intersection. However, if an inflectional system with the potential to agree with different or even multiple syntactic heads is to be learned, the representation of the meaning has

to reflect this structure and the intersection operation has to be adapted to handle this complexity.

In section 2, we showed, that this can partly be achieved by adding plain case features to the feature sets for the different heads the agreement system is sensitive to. However, it is quite hard – and for certain cases impossible – to capture all feature invariants that can be found this way and it almost always takes additional machinery or definitions. Furthermore, the different nature of 'case' information, defining the scope for the other substantial features is not reflected in the representation this way.

To overcome these shortcomings, we provided a well defined formalism that represents all possible feature invariants to be found in intransitives and monotransitives. With these tools, a learning algorithm can simply stick to the intersection operation on the meaning representations for different occurrences of a marker. Crucially, they don't restrict the algorithm to a certain type of invariants to be found: By traversing the implication hierarchy in search of the relevant invariant, an algorithm might introduce any bias towards specific types of specifications on its own – e.g. preferring monopersonal over bipersonal markers or grouped monopersonal specifications that conform to the general case alignment of the given language. If specific types of specifications are to be excluded *per se* – e.g. the rather unusual Biargumentive or MirrorPortmanteau specification – they can be removed from the implicational hierarchy of the specification types. Retaining the implicational paths they spun, the intersection operation then simply yields the maximal specific invariant, that can be formulated without the removed specification types. Having said that, it may be interesting to search for exactly these kinds of feature invariants just because they look unusual at first sight: either to confirm their possibility or to work out a theory that accounts for their non-occurrence or rarity.

References

Pertsova, Katya (2007). *Learning Form-Meaning Mappings in Presence of Homonymy*. PhD thesis: University of California, Los Angeles.