A Strictly Lexicalized Approach to Phonology

Markus Walther, University of Düsseldorf[1]

Motivation

IN syntax and semantics, both computational and otherwise, we have long observed a tendency to lexicalize grammatical information. In effect, this move integrates components of the grammar, their logical separation notwithstanding. Hence, potential interface problems between formally disparate components are avoided. In the context of Finnish syntax Karttunen (1986) demonstrates, for instance, that by using a categorial unification grammar one may dispense with a separate phrase structure rule component at the expense of enriched lexical items. One trivial combinatory schema survives, which facilitates the local combination with adjacent structure.

König (1994) in addition emphasizes that viewing the lexicon as the sole repository of grammatical information results in ◇ *conceptual simplicity*: Structures may be described en bloc in the lexicon, no splitting between components is necessary. Taking lexical items as the overall starting point gives a uniform basis for generalization and schematization. ◇ *Procedural efficiency* stems from the fact that only constraints which can be lexically accessed through input items need to be considered during processing. This has the potential benefit of yielding a major reduction in search space. If the total number of structures activated by a finite input is also finite, termination is guaranteed.

In her paper König then develops a fully lexicalized HPSG-like grammar for German, amounting to an „intensional description of the set of legal string–syntax–semantics triples“ formalized in the Comprehensive Unification Formalism (CUF, Dörre/Dorna 1993). Clearly, however, what is lacking in modern constraint-based grammar is genuine phonological information, since mere orthographical string representations are unsuited for a number of realistic tasks such as interfacing with spoken language frontends or modelling prosodic constraints within syntax (e.g. for English heavy NP shift, cf. Bird 1991).

Previous work in the Declarative Phonology paradigm (DP, Scobbie 1991) is usually compatible with HPSG assumptions, possessing the same monotonic and non-derivational stance as well as the fundamental description–object distinction, while being lexicalized to various degrees. Scobbie himself sees a major motivation for DP in that it solves the 'Interaction Problem' which plagues derivational multi-component approaches. However, integrating existing proposals into a full HPSG grammar typically requires one of the following:

- the heterogenous combination of constraint solvers for typed feature structures (TFS) and finite state automata (FSA) for modelling phonology. Unfortunately, Bird (1991) in proposing such a setup did not describe how to do this in practice and available TFS implementations currently lack support for efficient computation with operations such as FSA intersection. Therefore at present the computational consequences of such a move remain unclear.[2] Also it is hard to see how finite-state approaches could facilitate a declarative coupling of phonetics and phonology (as pursued e.g. in Walther/Kröger 1994), where

[1]The author is grateful to Mark Ellison for very helpful comments and discussion based on an earlier version of this paper. Many thanks also to Petra Barg, Jim Kilbury, Suzanne Wolting and two anonymous reviewers. All remaining errors are my own. Correspondence should be directed to: Markus Walther, Seminar für Allgemeine Sprachwissenschaft, Heinrich-Heine-Universität, Universitätsstr. 1, D-40225 Düsseldorf, Germany.
Electronic mail: walther@ling.uni-duesseldorf.de

[2]While the calculus of operations over finite state automata could be simulated within many current grammar formalisms, efficiency considerations would mandate interfacing such formalisms to an external FSA constraint solver. As far as we know, it is an open research question how to achieve efficient coupling of such disparate constraint solvers as well as how to optimize FSA-enriched TFS grammars.

arithmetic computation is required to describe the concrete temporal structure of phonetic events, instantiating abstract temporal relationships with specific millisecond values.

• the use of computationally expensive recursive relations for iterative or unbounded phonological processes (Carpenter/Mastroianni 1994). Even if only tail-recursion is employed, it is unclear how to gain much benefit from optimization techniques such as partial evaluation in such a setting. While inherent nondeterminism in those parts of morphophonemic descriptions pertaining to the edges of morphemes may not be resolvable until run-time combination with adjacent structure occurs, morpheme-medially enough context is often available to allow for compile-time evaluation of constraints tied to these sites, resulting in run-time determinism. As it appears, this locality of nondeterminism cannot be exploited when partially evaluating the conjunction of morphemic descriptions with phonological generalizations modelled as unidirectional, global iterations over entire phonological strings.[3]

The present paper consequently attempts to explore what is needed to practically integrate phonology with the lexicon in the strict sense. Our goal is to stay entirely within conventional typed feature logic (i.e., CUF) and to avoid the use of recursive definitions, employing a single concatenative operation to combine phonological descriptions. It is expected that by applying optimization techniques from logic programming such as partial evaluation one may then hope for a much more efficient treatment of realistic phonological descriptions than what has hitherto been realizable.

The leading idea of strictly lexicalized phonology (henceforth SLP) will be to enrich segmental descriptions to constrain their neighbourhood such that the resulting chain of local constraint interactions yields well-formed surface structures.

Self-Syllabifying Segments

THE strict lexicalization of non-iterative alternations that rely on a fixed span of surrounding context has never been a major problem. For example, the productive consonantal alternations in German

(1) Honi[ç]-Honi[g]e ‘honey’ sg. – pl.

can be formalized with reference to at most the immediate left neighbour of a segment (see the lexicalist treatment in Walther/Wiese 1995). But what about the apparently iterative, unbounded process of predicting syllabic roles from a string of segmental representations, known as syllabification? Can we break down this task into local constraints interacting in a chain of segments? To be sure, syllabic roles are cross-linguistically of overwhelming importance in conditioning phonological phenomena. In the example above, two of the three generalizations involved, namely ‘Final Devoicing’ and ‘/g/ Spirantization’ depend on syllable structure.

Now, a closer look at the declarative approach to syllabification of Walther (1992) reveals that the constraints proposed therein are already mostly local, so we will take that approach as our starting point.

The idea is to maximally exploit a widely assumed order on the relative prominence of segment classes, sonority, for the task at hand. Typically, when we graph prominence curves over syllables, triangular shapes peaking at the syllable nucleus result, whereas syllable boundaries are at troughs in the curve. Like many others we use the following sonority ordering:[4]

(2) obstruents < nasals < laterals < r-sounds < high vowels < non-high vowels

As shown in Walther (1992), we may break down prominence curves into a sequence of rises and falls between immediately adjacent segments. The constraint *sbs1* in (4) maps such sonority differences to values *rise*, *fall* of a prom feature. Next, a binary decomposition

[3] Put another way, iterative or recursive relations directly model the implicitly understood universal quantification over segmental positions that guides phonological constraint application. The approach to be presented in contrast ‘unfolds’ the universal quantification by pushing constraints into individual segments, taking advantage of the finite length of segmental string tokens.

[4] Formally, simple tabulations of the relations $<, >, =$ over segment classes – encoded as CUF types – are employed. The tabulations take the form of functional dependencies.

of the four syllabic roles *onset*, *nucleus*, *coda* and ambisyllabic *codaonset* in terms of the two simplex types *ons* and *nocoda* gives us a way to talk about *partial* syllabic roles. This can be expressed with CUF propositional type formulae:

$$(3)\quad \begin{array}{lcrcr} onset & \leftrightarrow & ons & \wedge & no\_coda \\ nucleus & \leftrightarrow & \neg ons & \wedge & no\_coda \\ coda & \leftrightarrow & \neg ons & \wedge & \neg no\_coda \\ codaonset & \leftrightarrow & ons & \wedge & \neg no\_coda \end{array}$$

Attaching compositionally defined partial syllabic roles such as ¬ *ons*, ¬ *nocoda* to sonority rises and falls by way of constraint *sbs2* in (4), the local interaction of adjacent segmental syllabification constraints already yields the correct syllable structure assignment in the base case.

$$(4)\quad sbs1 := (\begin{bmatrix} \textsf{left}\,|\,\textsf{self}\,|\,\textsf{seg} > (\boxed{1}) \\ \textsf{self} \begin{bmatrix} \neg\, initial & \\ \textsf{seg} & \boxed{1} \\ \textsf{prom} & fall \end{bmatrix} \end{bmatrix} \vee \begin{bmatrix} \textsf{left}\,|\,\textsf{self}\,|\,\textsf{seg} < (\boxed{2}) \\ \textsf{self} \begin{bmatrix} \neg\, initial & \\ \textsf{seg} & \boxed{2} \\ \textsf{prom} & rise \end{bmatrix} \end{bmatrix} \vee \begin{bmatrix} \textsf{self}\ initial \\ \textsf{left}\ nil \end{bmatrix})$$

$$sbs2 := (\begin{bmatrix} \textsf{self} & \neg final \wedge ons \\ \textsf{right}\,|\,\textsf{self} & \begin{bmatrix} no\_coda \wedge regular \\ \textsf{prom}\ rise \end{bmatrix} \end{bmatrix} \vee \begin{bmatrix} \textsf{self} & \neg final \wedge \neg ons \\ \textsf{right}\,|\,\textsf{self} & \begin{bmatrix} \neg no\_coda \wedge regular \\ \textsf{prom}\ fall \end{bmatrix} \end{bmatrix} \vee \begin{bmatrix} \textsf{self} & \neg final \\ \textsf{right}\,|\,\textsf{self} & \neg regular \end{bmatrix} \vee \begin{bmatrix} \textsf{self} & final \\ \textsf{right} & nil \end{bmatrix})$$

So far we have not accounted for adjacent segments with equal sonority, which introduce plateaus into the triangular prominence curve. Straightening out the plateaus differs for non-nasal sonorants and the rest: whereas for the first group correct syllabification is obtained by reversing the last prominence direction[5], obstruents and nasals must continue the prominence direction of their left neighbour. The reader may verify that this gives the right results by looking at e.g. *sträubst* [ʃtʁɔɪpst] *'resist' 2sg pres ind* and *Thea* [teeaa] *(proper name)* with equal sonority stretches /ʃt/-/pst/-/eeaa/. This extension of base syllabification is incorporated into constraint *sbs* in (5), which forms part of every phonological segment definition. Note that strictly local constraint interaction again proved sufficient to handle equal sonority stretches, despite their unboundedness in principle.

$$(5)\quad sbs := sbs2 \wedge (sbs1 \vee \begin{bmatrix} \textsf{left}\,|\,\textsf{self} & \begin{bmatrix} \textsf{seg} & < (lateral) \wedge = (\boxed{1}) \\ \textsf{prom} & \boxed{2} \end{bmatrix} \\ \textsf{self} & \begin{bmatrix} \textsf{seg} & \boxed{1} \\ \textsf{prom} & \boxed{2} \end{bmatrix} \end{bmatrix} \vee \begin{bmatrix} \textsf{left}\,|\,\textsf{self} & \begin{bmatrix} \textsf{seg} & >= (lateral) \wedge = (\boxed{1}) \\ \textsf{prom} & \boxed{2} \end{bmatrix} \\ \textsf{self} & \begin{bmatrix} \textsf{seg} & \boxed{1} \\ \textsf{prom} & \neg \boxed{2} \end{bmatrix} \end{bmatrix})$$

It is found necessary to place additional constraints on the periphery of syllabification domains, namely to classify *initial* and *final* positions via types, and specify respective prominence directions and partial syllabic roles. This motivates the predicates *word,last* below.

$$(6)\quad word := \begin{bmatrix} \textsf{self} \begin{bmatrix} no\_coda \wedge initial \wedge regular \\ \textsf{prom}\ rise \end{bmatrix} \end{bmatrix}$$

$$last(\boxed{Seg}) := sbs \wedge \begin{bmatrix} \textsf{self} \begin{bmatrix} \neg ons \wedge final \\ \textsf{seg}\ \boxed{Seg} \end{bmatrix} \end{bmatrix}$$

$$is(\boxed{Seg}, \boxed{More}) := sbs \wedge \boxed{1} \begin{bmatrix} \textsf{self} & \begin{bmatrix} regular \\ \textsf{seg}\ \boxed{Seg} \end{bmatrix} \\ \textsf{right} & \boxed{More} \begin{bmatrix} \textsf{left}\ \boxed{1} \end{bmatrix} \end{bmatrix}$$

$$segeln := is(z, is(e, is(e, is(g, is(l, last(n)))))) \wedge word$$

Finally, if we can provide a means to glue segments together in concatenative fashion, the result will be a p h o n o l o g i c a l string with superimposed syllabic roles. While for morphological segmentation during parsing a recursive predicate may well be necessary, a simple non-recursive combination (*is*) suffices for describing lexical entries, depicted in (6) together with an example application. Note the segment-wise conjoined syllabification constraint *sbs*.[6]

Apart from SLP-style syllabification, backwards reference is quite generally found necessary in phonological context specifications. Thus the above introduction of a kind of doubly linked lists encoded in feature structures

[5]Reversal is effected by negation w.r.t. the { *fall*, *rise* } domain, see ¬ $\boxed{2}$ in (5).

[6]While Ellison (1993 & p.c.) identifies concatenation as a non-logical, procedural operation, his arguments rely on the assumption that individual concatenands are non-extensible strings, hence could not constitute surface-true descriptions of portions of, say, a larger domain such as utterance or word. Note that in contrast the *is*/2 segmental concatenation predicate in (6) deals with extensible strings. Separate constraints such as *last*/1 ensure non-extensibility beyond a given domain if desired.

is well motivated.[7] While **self**, **right** correspond to the usual 'first element–rest list' encoding of lists, **left** together with a reasonable $\boxed{1}$ [right | left $\boxed{1}$] identity (and vice versa) gives us the desired behaviour, resulting in cyclic feature structures. These are permitted in several contemporary constraint formalisms, including CUF.

Of course, the above presentation would be empirically incomplete for German in a number of respects, namely the proper treatment of multiple syllabification domains, e.g. within compounds, phonotactic restrictions, non-ambisyllabic sonority troughs and syllable length restrictions. However, we do not expect major difficulties in adapting the solutions of Walther (1992) to the strictly lexicalized setting.

Schwa in German Infinitives

IN search of a promising test case for SLP, let us turn to schwa distribution in German infinitive endings. The most recent study we know of is the treatment in Neef (1994). For lack of space we cannot discuss his interesting analysis here, except to note that it is not formalized and differs in its account of the morphological influence on schwa.

Schwa, consistently marked in the orthography by <e>, must be treated in a morpheme-based phonological lexicon because of its alternation with zero in many contexts, especially inflection. Though all but two infinitives (*tun,sein*) have schwa in their final syllable, its position varies from the final infinitive marker *-n* itself to stem-internal segments.[8] Some typical examples are given in (7).

(7)	seg[ə]ln	lieb[ə]n	bech[ɐ]rn	atm[ə]n
	'sail'	*'love'*	*'drink heavily'*	*'breathe'*

The relevance of infinitival schwa for SLP comes from its dual nature as a mostly phonological phenomenon with a twist of morphological influence, as we shall see shortly. The analysis will not only provide fresh insight into the linguistic facts, but in addition offers a clean integration of lexical-morphological and phonological information with no interaction problems whatsoever.

Neef extracted 367 phonological types out of a representative corpus of approx. 17000 infinitive tokens, which also form the basis of this analysis. Now, a striking first observation when syllabifying those types using the SLP approach is that for a very large percentage we can predict both the correct occurrence and position of schwa from schwa-less segmental strings alone. Specifically, alternating schwa in syllabified forms surfaces as a consonant which gets the *nucleus* syllabic role by way of a sonority peak. This correctly reflects the variability of vocalic schwa, in that – depending on speech rate, style and other pragmatic factors – all such infinitival schwas may actually be pronounced as syllabic consonants, e.g. syllabic [l,n,ɐ] in (7).[9] Phonetic evidence suggests that, at least in this context, we can regard both syllabic and more vocalic forms of schwa as linguistically controlled phonetic interpretation of invariant phonological structure.

While syllabification alone derives the bulk of infinitive schwas, the rest can be captured by a simple generalization about their word-final sonority shape:[10]

(8) infinitive-final *-n* is exceptionally the nucleus of a syllable, if prominence

[7] See Ellison (1994) for a similar proposal couched in modal logic. Mark Ellison (p.c.) has integrated this proposal into his TRDL formalism for finite-state description of phonological analyses.

[8] Actually, the verbal inflectional marker *-n : [+V, +pl, -2]* displays identical behaviour w.r.t. schwa.

[9] Under an appropriate model of prosodic-phonetic interpretation, [ɐ] is the realization of nucleus-/ʁ/. Neef claims (p.94) that forms such as *atmen, widmen* must *obligatorily* be pronounced with an overt schwa. In our judgment, these may not only be pronounced without overt vocalic schwa between the bilabial and syllabic alveolar nasal, but quite often even with nasal release of the labial.

[10] To be precise, this excludes the exceptional *tun 'to do',sein 'to be'* mentioned above, which must be dealt with separately in everybody's account. The only other class of attested shapes not yet covered is given by forms like <*busserln*> */bUsʁln/ 'to kiss around'* (p.c. Martin Neef), characterised by a second consonantal nucleus to the left of falsely syllabic *-n*. In the example just given, /ʁ,n/ would be nuclei, /l/ would be an onset. Our informants, however, classify *all* such forms as of dialectal origin, presumably Bavarian. Note that it *is* straightforward to extend (8) to cover these marginal cases by imposing the additional precondition that the antepenultimate segment does not form a consonantal nucleus.

falls at least twice at the end of the word.

Typical forms which fall under the above generalization are *mal[ə]n, fahr[ə]n, seh[ə]n.* This morphological schwa condition (*msc*) is straightforwardly reflected in (9):

(9) $$inf\_n := last(n) \wedge (msc \vee notmsc)$$

$$msc := \begin{bmatrix} \mathsf{left}\,|\,\mathsf{self} & \begin{bmatrix} ons \\ \mathsf{prom}\ fall \end{bmatrix} \\ \mathsf{self} & \begin{bmatrix} nucleus \wedge \neg regular \\ \mathsf{prom}\ fall \end{bmatrix} \end{bmatrix}$$

For brevity we have omitted the definition of *notmsc*, which covers the 'elsewhere' case where the precondition of (8) is not satisfied, marking this scenario as *regular*. Note that *is* was defined in (6) to include position-wise syllabification.

The clear advantage of the present proposal is that the set of possible interactions of disjunctively specified constraints can now be precomputed to a large extent at lexicon definition time. We would argue that, apart from non-recursiveness, this crucially hinges on having the right level of constraint 'granularity' for maximal interaction. Obviously for our test case, the context spans of fine-grained syllabification and infinitive constraints agree well, so partial evaluation of their conjunction can in a sense 'fold' morphologically conditioned exceptional syllabification right into the final syllabification constraint itself. In general, we envisage most morpheme-internal constraints to become deterministic, so that these can be fully evaluated at lexicon definition time, leaving just constraints which reference the periphery or have to 'look' across boundaries for dynamic combination computation.[11]

Nonlocality

NO one would be satisfied with a proposal for computational phonology which had principled difficulty in dealing with global or 'action-at-a-distance' phenomena. This has been the foremost area of interest in Autosegmental Phonology (AP, Goldsmith 1990). AP received its primary motivation from the lucid analysis of tonal phenomena, where tone-bearing units are not coextensive with segments, but tones nevertheless interact as if they were adjacent. Instead of presenting another case study for tone, a generic way to make arbitrary non-adjacent 'autosegments' see each other with only local computation will be described below. The method proposed does not involve any kind of explicit search.

The general setting presupposed in much AP work is that segments (or subparts of them) that possess a property $\mathcal{P}$ (e.g. consonanthood, vocalicness, tone-bearingness, nucleushood, labiality) project onto a $\mathcal{P}$-tier, whereas not-$\mathcal{P}$ segments do not project. A tier is a multiset of phonological elements (called autosegments) ordered by immediate precedence. Projection proceeds from a distinguished *skeletal* tier (whose $\mathcal{P}$ may be assumed empty, i.e. always true) and preserves transitive precedence relationships.

A closer look reveals that it suffices to compute new immediate adjacency relations per $\mathcal{P}$-tier, if these can be somehow overlaid on the skeletal tier.[12]

It is actually very simple to achieve this with only local constraints. We need to define two new features **left**$\mathcal{P}$, **right**$\mathcal{P}$ per tier for segmental positions, and then the predicates *P_tier_next, P_tier_last* in (10) will do the job if conjoined to constraints *is,last* of (6). Again, as immediate adjacency should be symmetrical, entirely analogous definitions of *P_tier_previous, P_first*, not shown here, are needed to complete the picture.

(10) $$P\_tier\_next := \begin{bmatrix} \mathsf{self}\ final \end{bmatrix} \vee$$

$$\begin{bmatrix} \mathsf{self} & \begin{bmatrix} \neg final \\ \mathsf{right}\mathcal{P}\ \boxed{1} \end{bmatrix} \\ \mathsf{right}\,|\,\mathsf{self}\ \boxed{1} & \begin{bmatrix} \mathsf{seg}\ \mathcal{P} \end{bmatrix} \end{bmatrix} \vee \begin{bmatrix} \mathsf{self} & \begin{bmatrix} \neg final \\ \mathsf{right}\mathcal{P}\ \boxed{2} \end{bmatrix} \\ \mathsf{right}\,|\,\mathsf{self} & \begin{bmatrix} \mathsf{seg} & \neg\mathcal{P} \\ \mathsf{right}\mathcal{P} & \boxed{2} \end{bmatrix} \end{bmatrix}$$

$$P\_last := \begin{bmatrix} \mathsf{self}\,|\,\mathsf{right}\mathcal{P}\ nil \end{bmatrix} \wedge P\_tier\_previous$$

The second disjunct in (10) says that 'if my right segment has the sought-after property

[11] While finite-state approaches can go even further in composing a finite-state lexicon with the set of phonological constraints, run-time FSA combination operations such as intersection still appear to be necessary for dealing with *productivity*, i.e. unknown morphemes in case of parsing.

[12] As a note to the specialist, in a monotonic logic-based formalization this implies that the awkward mechanism of 'tier conflation' is neither needed nor possible, a welcome formal restriction. Phonology may remain monostratal w.r.t. to the skeletal tier despite an arbitrary fixed number of *virtual* tiers.

$\mathcal{P}$,[13] I can establish having found the next element which is $\mathcal{P}$-wise immediately adjacent by coindexing its place with my own next element 'pointer' right$\mathcal{P}$ $\boxed{1}$'. Otherwise, one must defer identification of a $\mathcal{P}$-bearing element by sharing the adjacent next element pointers (right$\mathcal{P}$ $\boxed{2}$), with identification to be decided further to the right. The right word periphery is handled by *P_last*.

Consequently, it is now easy to write SLP-style definitions which place constraints on e.g. the previous labial segment though there may be an a priori unbounded number of intervening non-labials.[14]

Conclusion

THE development of SLP is still in its infancy. We have seen how to enrich phonological segments so as to make them self-syllabifying through strictly local constraint interactions, thereby eliminating a source of recursive definitions once thought indispensible. A case study of schwa/zero alternation in German infinitive endings demonstrated that SLP is well suited to describe morphophonological interaction in a clean fashion. It was highlighted how sufficiently fine-grained constraint 'granularity' favoured by SLP-style modelling helps to exploit constraint optimization techniques to the fullest.

In conjunction with inheritance-based lexicons, the segmentally localized constraints can be tied to particular subclasses, thereby further attenuating the need for defaults or diacritics, which are often needed only to prevent coarse-grained phonological constraints from applying too generally.

Finally, there is ample room for actual analyses of 'action-at-a-distance' phenomena using the SLP treatment of nonlocality. As SLP at present relies on concatenation, an investigation into nonconcatenative, prosodic morphology is also warranted. Preliminary work suggests that SLP can deal with it in concatenative fashion by making use of constraint-governed optionality in descriptions plus a concept of optimality.

The above analyses have been fully computer-implemented and are available from the author upon request.

References

Bird, S. (1992): Finite-State Phonology in HPSG. In: *Proceedings of COLING-92*, Nantes, pp 74–80.

Dörre, J. / Dorna, M. (1993): CUF - a formalism for linguistic knowledge representation. Deliverable R.1.2A, DYANA 2.

Carpenter, B. / Mastroianni, M. (1994): Constraint-based Morpho-phonology. In: S. Bird (ed.), *Proceedings of the 1st ACL SIGPHON Workshop*, Las Cruces, pp 13-24.

Ellison, T. Mark (1993): Concatenation vs conjunction in constraint based phonology. In: T. M. Ellison, J. M. Scobbie (eds.): Computational Phonology. Edinburgh Working Papers in Cognitive Science 8, Center for Cognitive Science, University of Edinburgh, 1–18.

Ellison, T. Mark (1994): Constraints, Exceptions and Representations. In: S. Bird (ed.), *Proceedings of the 1st ACL SIGPHON Workshop*, Las Cruces, pp 25–32.

Goldsmith, J. (1990): Autosegmental and Metrical Phonology. Basil Blackwell, Oxford.

Karttunen, L. (1986): Radical Lexicalism. In: Mark Baltin/Anthony Kroch (eds.),*Alternative Conceptions of Phrase Structure*, University of Chicago Press, pp 43–65.

König, E. (1994): A study in grammar design. Arbeitspapiere des SFB 340, Nr. 54, University of Stuttgart.

Neef, M. (1994): Wortdesign. Unpublished PhD thesis, University of Cologne.

Scobbie, J.M. (1991): Towards Declarative Phonology. In: Bird, S. (ed.), *Declarative Perspectives on Phonology*, pp 1-26. Centre for Cognitive Science, Edinburgh.

Walther, M. (1992): Deklarative Silbifizierung in einem constraint-basierten Grammatikformalismus. Master's Thesis, University of Stuttgart.

Walther, M./Kröger, B. J. (1994): Phonologie-Phonetikkopplung in einem constraintbasierten gesturalen Modell. In: Harald Trost (ed.), Proceedings KONVENS '94, Österreichische Gesellschaft für Artificial Intelligence, Vienna.

Walther, M./Wiese, R. (1995): Deklarative versus prozedurale Modellierung von Konsonantenalternationen im Deutschen. In: *Linguistische Berichte* 157, pp 175-185.

[13] Without loss of generality we assume the tier-defining property $\mathcal{P}$ to be encodable in propositional logic, which underlies the type system of CUF. This is straightforward to achieve with the example properties mentioned above.

[14] Such non-local reference would be necessary for modelling an alternation in the Ethio-Semitic language Tigrinya spoken in Eritrea. A slight complication arises when asking for the initial or final element of a given tier: if that element has property $\mathcal{P}$, we are done, but if not, we must dereference once through right$\mathcal{P}$/left$\mathcal{P}$ to access the correct element.