Departing from Pāṇini for good reasons

GÉRARD HUET

Abstract

Pāṇini's Aṣṭādhyāyī, and its refinements brought about by the trimuni tradition, is the unchallenged gold standard of Sanskrit correct usage. It brings under one comprehensive system a complete grammar of the language, dealing with phonetics, morphology, syntax and semantics. A competent Sanskrit speaker may in principle justify any meaningful enunciation in the language by constructing a sequence of grammar rules and lexicon accesses that will yield its phonetic realization under the intended meaning. This fact is not questioned here. However, the use of 'meaningful' and 'meaning' in the precise statement above is essential. It assumes not just that the enunciation be meaningful, but that the speaker knows its meaning, and may refer to it in the process of grammatical justification. This observation has led to numerous discussions in the literature [Cardona, Kiparsky, Houben, Scharf among others] arguing that the grammar is not usable simply as a set of independent modules operating across the various 'levels' of phonetics, morphology, syntax and semantics. This raises a challenge to the proper design and implementation of a mechanical Aṣṭādhyāyī simulator, since interaction with a human operator is necessary, not just for lexicon access, but also for the validation of semantic conditions.

Worse still is the problem of using the Pāṇinian tradition for the design of a mechanical Sanskrit analyzer, able for instance to do semi-automatic annotation of Sanskrit corpora, since part-of-speech tagging and even segmentation of sentences (*sandhiviccheda*) poses challenges in the absence of the intended meaning. Morphology is hopelessly interwoven with syntax, if only because compounds have an unbounded number of components, and thus full lexicons must operate at a level of morphemes and not just words. This induces computational complexity problems, whose solution demands a different organization of the grammatical processes. It is just not feasible to somehow regard Aṣṭādhyāyī as a generating device, whose inversion would yield a parsing algorithm.

This paper illustrates the necessary change of methodology on three precise points, concerning the analysis of compounds. First, Pāṇini explains compound formation as a recursive process at the level of inflected words (padas). You may form a new pada by joining together two padas. Thus the word *ātmanepadam* is obtained by glueing *ātmane* and *padam*, or more precisely *ātman*-s_1 and *pada*-s_2, where suffixes s_1 and s_2 are the corresponding morphological markers. For instance, s_1 expresses the dative case, so that *ātmanepadam* may be glossed as "word for self". Similarly, the compound *devakulam* is obtained by glueing *deva*-s_1 and *kula*-s_2, where s_1 expresses the genitive case, consistently with its ṣaṣṭītatpuruṣa status issued from the meaning of its gloss as the non-compound substantive phrase *devasya kulam* i.e. "god's house". Here, however, a process of erasure of markers (lopa) operates to ultimately erase s_1, and leaves us with the final phonemic realization *devakulam*, and not **devasyakulam*. This process is optional, and thus both *devakulam* and the so-called aluk compound *ātmanepadam* are derivable under a unique morphological process of samāsa formation. This is part of the formal beauty of Pāṇini's grammar, namely its brevity (lāghava). However, if one wanted to reverse this process in a computational parser, we would have to un-erase so to speak all morphological markers from initial segments of compounds, in order to synthetize not only *devasya*, but all possible forms of stem *deva* in the 3 numbers, 3 genders, and 7 cases, that is 63 forms, corresponding to the 63 potential paraphrases of compound *devakulam*. This is clearly computationally untractable, and not needed anyway. Thus the recursion on compounding padas ought to be replaced by linear recursion on base stems (pratipādikās), and the iic. bare form *deva*- must be lexicalized as a morpheme usable for regular compound formation, replacing the non-determinism search branching factor of 63 to a deterministic search for a single form. In the case of aluk compounds, which are the exception rather than the rule, we may lexicalize them, recognizing the fact that aluk compound formation is not productive in the language.

Another issue arises form the fact that the binary rule of compound formation corresponds to a binary tree structure, namely the phrase structure of its paraphrase. Thus e.g. the stem *baddhapadmāsanastha* is analysed as ((*baddha*-(*padma*-*āsana*))-*stha*) "he who stands in the locked lotus position". This binary tree structure ((A-(B-C))-D) is one among 5 ways of forming a binary tree with 4 leaves, or equivalently of parenthesizing an expression with 4 components. This decomposition arises from our understanding of the meaning of this compound, which gives us the dependencies between components yielding this unique factorization. In the absence of knowledge of this meaning, the phonemic realization of a compound with $n + 1$

components could possibly lead to C_n possible interpretations, where C_n is the n-th Catalan number, a combinatorial function that is exponential in n. Thus *pravaranṛpamukuṭamaṇimarīcimañjarīcayacarcitacaraṇayugalaḥ*, a compound found in Pañcatantra, even after sandhi segmentation, leads potentially to C_{10} = 16796 interpretations. We propose to decouple dependency analysis, a semantic process, from the purely morphological formation of linear compounds or *pre-compounds*, of the form $I_1 < I_2 < ... < F$, where I_1, I_2,... are iic. morphemes (bare stems) and F is an inflected non-compound form. We may thus interpret the above compound as the unique pre-compound: *pravara<nṛ<pa<mukuṭa<maṇi<marīci<mañjarī<caya< carcita<caraṇa<yugalaḥ*, recognizable by a simple terminal recursion within a finite-state lexicon-driven lexer. Finally, we propose to ignore the exocentric (bahuvrīhi) status of compounds during segmentation-tagging considered as a preliminary pass creating a linear structure, further analyzable by more semantic processes such as kāraka analysis in a separate second-level independent module. This removes one more exponential explosion.

On these principles we have built an experimental Sanskrit parser for the classical language, which is able to recognize long compounds such as the above, which we segment in only 16 interpretations, with a second semantic role analysis phase pruning out all of them except the intended solution. Our computational processes are not Paninian in the sense of being able to synthesize the exact sequence of rules from the Aṣṭādhyāyī necessary to derive a given sentence, but we claim that they are sufficient to analyze a fair proportion of the classical corpus, given a root lexicon covering its vocabulary, and precise enough to be usable by students of the language as a computer-aided reader assistant, and ultimately by philologists as an editing tool.