Sanskrit Shallow Parsing

Gérard Huet
INRIA Paris-Rocquencourt

April 24, 2013

Abstract

We describe and justify a shallow parser for Classical Sanskrit. It performs a simplified *kāraka* analysis on a Sanskrit text, where *sandhi viccheda* and morphological analysis have already been performed. The only semantic roles considered are agent [*kartā*] and patient/goal [*karman*], and thus the only lemmas playing a role are the ones labeled with cases Nominative, Accusative, and Instrumental. The algorithm proceeds in several steps. First the input is pre-processed by tools transducers, where the tools correspond to grammatical words such as "ca" and "saha". This permits e.g. to recognize coordinated items, and to synthesize compound lemmas accounting for the corresponding nominal phrase. Then the predicate item of the sentence is selected, whether verbal or nominal, and its governance [*ākāṅkṣā*] is assessed. Then lemmas matching the requirements are selected. Missing roles incur penalty. Then extra lemmas are attempted to be merged with selected lemmas, providing they agree (attempting to recognize nominal phrases by concord). Orphan nominative lemmas incur further penalty. Extra oblique lemmas are considered as adverbs. All possible segmentation solutions are given a penalty, computed as the minimum penalty of all expansions of their possible morphological multitags. Only segmentations of minimal penalty are retained.

This shallow parser is fast and robust. It filters out typically 90% of the wrong segmentations. It is insensitive to dislocations, and thus works on poetry as well as on prose. It has the important advantage of not requiring compound expansion, even in known cases of *asamartha* compounds, thus limiting the complexity of sentence analysis in the presence of compounding. Its main limitation is its poor handling of anonymous/ellipsed agents, suggesting it should be applied at the discourse rather than sentence level. This crude shallow parser is not usable as a stand-alone analyser, since on one hand it assumes *sandhi*-analysed input, and on the other hand it is too crude to synthesize a phrase structure tree or dependency graph. But it may be put to use as a filtering to more sophisticated parsers, in a divide-and-conquer manner.