

A New Method for Compiling Parallel Replacement Rules

Anssi Yli-Jyrä and Kimmo Koskenniemi

¹ Language Research Service, CSC Scientific Computing Ltd., Finland

² Department of General Linguistics, University of Helsinki, Finland
`firstname.lastname@helsinki.fi`

Kempe and Karttunen [1] have presented a method that compiles a set of parallel conditional replacement (rewriting) rules into a finite-state transducer. Other, simpler methods exist for single rules or for rules of a restricted type, but they can be used only in restricted situations.

We introduce a new compilation method that is simpler, general, and more optimizable and extensible than the previous solutions. We have already implemented it in our subversion of the Stuttgart FS tool, and in addition, tested even the directed and scattered variants of the method with XFST. The method allows expressing rules that were not previously possible. The method uses parameters that allow for obtaining directional, gradient, scattered, or disjunctively ordered replace or markup rules as its extensions and optimizations in several special cases. The full technical report, further articles on the parameters and pointers to implementations are collected at <http://www.ling.helsinki.fi/~aylijyra/replace/>.

Most prior methods for simple and parallel replacement rely heavily on the use of brackets that indicate the (non)occurrences of substrings in strings, and a separated concatenation closure that make changes in bracketed replace regions. In contrast, our method is considerably simpler since it packs the set of rules and their context conditions into a single language representing all the rules. The packing eliminates the need for computing Kaplan and Kay's if-then idioms [2] or Mohri and Sproat's [3] marking transducers. As a byproduct, the set of possible rules is closed under the Boolean operations, which enables expressing disjunctive ordering and negative default rules.

The orientation of contexts is expressed in traditional methods by ordering the algorithm-encapsulated transducers differently under the composition operation. In our method, the contexts are symbol-pair languages whose unknown sides can be left under-specified by the user or the rule compiler to account for left-, right-, down- or up-ward contexts [4].

In most previous methods, the user specifies possible replacement centers as pairs of input and output languages. In contrast, our method

assumes, along with van Noord and Gerdemann [5], that changes in replacement centers are specifiable with a transducer.

The prior methods unanimously exclude, along with Kaplan and Kay (KK) [2], the part of string already rewritten from further rewriting. This assumption makes rules $xax \rightarrow xbx$ and $a \rightarrow b/x_x$ nonequivalent, although they have been traditionally considered as notational variants of each other. In contrast, Generalized Two-Level Grammars [6] provides a relaxed interpretation for optional replacement that actually treats both forms of rules equivalently. Our current work extends these grammars with a correct interpretation for obligatory rules.

The double arrow rules of two-level grammars would lead to a too strict, local, interpretation of obligatoriness. For this reason, we define obligatory rules by first applying the rules as optimal and then ordering the obtained mappings according to the matching [7] replace regions. This is implemented in following steps:

1. insertion of a pair of brackets to the alphabet; embracing centers with such brackets; banning the brackets elsewhere; – to conform to the KK style rewriting, we also ban internal brackets within the centers;
2. interpretation of replace rules and their oriented contexts, respectively, as context restriction rules and two-level contexts;
3. extraction of the 0-free domain [2] from the generalized two-level grammar [6]; its worsening by manipulation [7] of brackets;
4. constraining the domain of the previous regular relation with the complement of the worsened language; removal of the brackets and 0's in both domain and range sides.

References

1. Kempe, A., Karttunen, L.: Parallel replacement in finite state calculus. In: 16th COLING 1996, Proc. Conference. Volume 2., Copenhagen, Denmark (1996) 622–627
2. Kaplan, R.M., Kay, M.: Regular models of phonological rule systems. *Computational Linguistics* **20**(3) (1994) 331–378
3. Mohri, M., Sproat, R.: An efficient compiler for weighted rewrite rules. In: 34th ACL 1996, Proc. Conference, Santa Cruz, CA, USA (1996) 231–238
4. Karttunen, L.: The replace operator. In: 33th ACL 1995, Proceedings of the Conference, Cambridge, MA, USA (1995) 16–23
5. Gerdemann, D., van Noord, G.: Transducers from rewrite rules with backreferences. In: 9th EACL 1999, Proceedings of the Conference. (1999) 126–133
6. Yli-Jyrä, A., Koskenniemi, K.: Compiling generalized two-level rules and grammars. In: Proceedings of FinTAL 2006. LNAI (2006)
7. Gerdemann, D., van Noord, G.: Approximation and exactness in Finite-State Optimality Theory. In Eisner, J., Karttunen, L., Thériault, A., eds.: SIGPHON 2000, Finite State Phonology. (2000)