A Pushdown Machine for Context-Free Tree Translation

Johannes Osterholzer*

Institute of Theoretical Computer Science
Technische Universität Dresden

Abstract. We identify a syntactic restriction of *synchronous context-free tree grammars*. The notion of (linear and non-deleting) *pushdown extended top-down tree transducers* is introduced and we prove that the transformations of the former coincide with those of the latter.

1 Introduction

In syntax-based machine translation of natural languages, an input sentence s is translated by applying a tree transformation to a parse tree ξ of s, given a grammar for the input language. The transformation is often performed by formalisms such as extended top-down tree transducers (XTT) [7, 5]. These, however, can not capture certain phenomena that occur in natural language [4].

Hence a number of more powerful formalisms has been introduced, among those *synchronous context-free tree grammars (SCFTG)* [6]. They can be considered as context-free tree grammars where both an input and output tree are derived synchronously (cf. Fig. 1). Synchronous derivation leaves open the problem of *parsing* a given input tree. But it may prove beneficial, just as for string languages, to research formalisms which take parsing into account. Hence, we propose a formalism with a unidirectional derivation semantics, called *pushdown extended top-down tree transducers (PDXTT)*. These can be understood as extended top-down tree transducers where the finite state control is equipped with a tree pushdown storage [7, 2] that allows the recognition of an input context-free tree language, or as (extended input) pushdown tree automata [3] with output.

The class of transformations that are computed by linear and non-deleting PDXTT will be proven to coincide with the class of transformations of *simple* SCFTG, which are a certain syntactic restriction of SCFTG. Actually, this is a generalization of the characterization of simple syntax-directed translation schemata by pushdown transducers [1] to tree transformations.

2 Synchronous Context-Free Tree Grammars

Context-free tree grammars (CFTG) are a generalization of CFG to trees. Roughly, a CFTG $\mathcal{G}$ consists of two ranked alphabets Σ and N of *terminal* resp. *nonterminal symbols.* The productions of $\mathcal{G}$ allow to rewrite a nonterminal A of rank k *within* a tree into a tree over $N \cup \Sigma$, i.e., in $T_{N\cup\Sigma}(X_k)$, cf. [7].

* Partially supported by DFG Graduiertenkolleg 1763 (QuantLA)

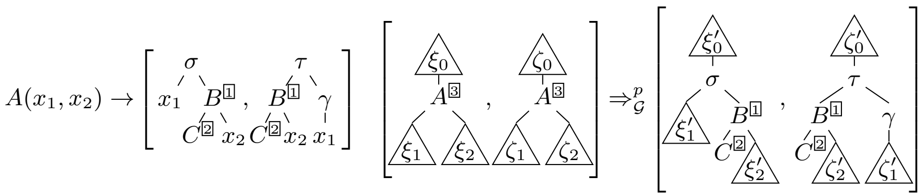

Fig. 1. An SCFTG production p and its application to a sentential form

The right-hand sides of the productions of *SCFTG* now are *pairs* of such trees, such that there is a bijection between the contained nonterminal symbols, and both trees are linear and non-deleting in a common set of variables. The sentential forms of SCFTG are also pairs of trees over $N \cup \Sigma$, where the nonterminals are correlated as above. A production may only be applied to such a correlated pair of nonterminals. For an example, see Fig. 1, where the bijective relation between the nonterminal symbols A resp. B is denoted by a boxed superscript number (an *index*), and ξ_0' etc. are just the result of a consistent renaming of these indices in ξ_0 etc. to avoid clashes. This SCFTG transforms a tree ξ into ζ if both ξ and ζ contain only terminals and the pair $[\xi, \zeta]$ is derived from an initial pair of nonterminals in a finite number of steps.

Intuitively, a production is *simple* if both the input and the output tree from its right-hand side exhibit the same call structure of nonterminal symbols and variables: for every occurring nonterminal $A^{\boxed{i}}$ of rank k, and for every $j \in \{1, \ldots, k\}$, the sets of indexed nonterminals and variables contained in the j-th child subtree of $A^{\boxed{i}}$ must be equal in the right-hand side's input and output component. Hence, the right-hand side of the production p in Fig. 1 is simple. In contrast, the right-hand side $\big[D^{\boxed{1}}(D^{\boxed{2}}(x_1)), D^{\boxed{2}}(D^{\boxed{1}}(x_1))\big]$ is not simple, since $D^{\boxed{1}}$ dominates $D^{\boxed{2}}$ in the input, but not in the output tree. The right-hand side $\big[A^{\boxed{1}}(x_1, x_2), A^{\boxed{1}}(x_2, x_1)\big]$ is not simple either, since x_1 appears as the nonterminal's first argument in the input, but as the second one in the output. An SCFTG is called simple if all its productions are simple.

3 Pushdown Extended Top-Down Tree Transducers

In contrast to the productions of SCFTG, the rules of PDXTT are asymmetric, and permit a state-based rewriting of input into output trees. Just as for XTT, every rule allows to match the input tree with a context of finite but arbitrary height. Their right-hand sides are trees, at whose frontiers the state-based rewriting may continue on the remaining subtrees of the input. Unlike XTT however, the derivation process of PDXTT is controlled by a tree pushdown. Thus, a rule can additionally check the top symbol of the tree pushdown for the current input tree, and push further information that controls the derivation of the remaining subtrees. A PDXTT $\mathcal{M}$ transforms ξ into ζ if ζ consists entirely of output symbols and if it is a normal form of $\langle q_0, \gamma_0, \xi \rangle$ with regard to $\Rightarrow_{\mathcal{M}}$, where q_0 and γ_0 are the initial state, resp. initial pushdown symbol, of $\mathcal{M}$.

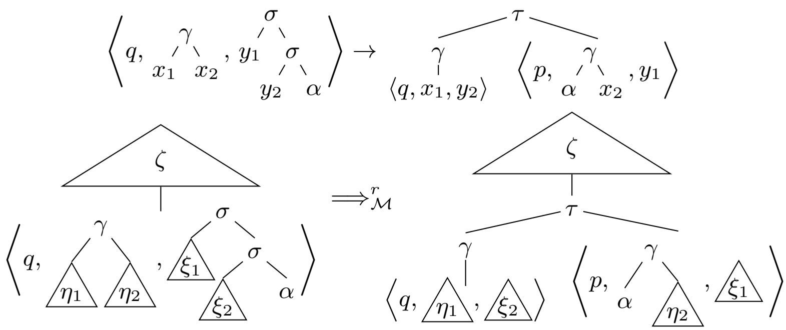

Fig. 2. A PDXTT rule r and its application

For an example rule r and its application to a configuration, see Fig. 2. The tree ζ has already been produced as output, while the input (sub)tree $\sigma(\xi_1, \sigma(\xi_2, \alpha))$ has yet to be rewritten by state q. Since the tree pushdown is of the form $\gamma(\eta_1, \eta_2)$, r can be applied, producing some output, with the remaining inputs ξ_1 and ξ_2 marked for processing, controlled by the pushdowns η_1 and η_2, where moreover $\gamma(\alpha, x_1)$ has been pushed onto η_2.

If, for every rule of a PDXTT $\mathcal{M}$, each variable x_i and y_j on its left-hand side appears exactly once on its right-hand side, then $\mathcal{M}$ is *linear and non-deleting.* Rule r in Fig. 2 is of this form.

Theorem 1. *Let τ be a tree transformation. The following are equivalent:*

1. *There is a simple SCFTG s.t. τ is its transformation.*
2. *There is a linear and nondeleting PDXTT s.t. τ is its transformation.*

Proof. By a close correspondence between simple SCFTG in (a certain) normal form and linear and nondeleting one-state PDXTT in (another) normal form.

References

1. A. V. Aho and J. D. Ullman. *The Theory of Parsing, Translation, and Compiling.* Prentice-Hall, 1972.
2. J. Engelfriet and H. Vogler. Pushdown Machines for the Macro Tree Transducer. *Theoret. Comput. Sci.*, 42(3):251–368, 1986.
3. I. Guessarian. Pushdown Tree Automata. *Math. Syst. Theory*, 16(1):237–263, 1983.
4. L. Kallmeyer. *Parsing Beyond Context-Free Grammars.* Springer, 2010.
5. A. Maletti, J. Graehl, M. Hopkins, and K. Knight. The Power of Extended Top-Down Tree Transducers. *SIAM J. Comput.*, 39(2):410–430, 2009.
6. M.-J. Nederhof and H. Vogler. Synchronous Context-Free Tree Grammars. In *Proc. of TAG+11*, pages 55–63, 2012.
7. W. C. Rounds. Mappings and Grammars on Trees. *Theory Comput. Syst.*, 4(3):257–287, 1970.