A Pregroup Analysis of Japanese Causatives *

Kumi Cardinal

Keio University, Shonan-Fujisawa Campus,
Graduate School of Media and Governance
5322 Endo, Fujisawa-shi, Kanagawa 252-8520, Japan
cardinal@sfc.keio.ac.jp

Abstract. We explore a computational algebraic approach to grammar via pregroups. We examine how the structures of Japanese causatives can be treated in the framework of a pregroup grammar. In our grammar, the dictionary assigns one or more syntactic types to each word and the grammar rules are used to infer types to strings of words. We developed a practical parser representing our pregroup grammar, which validates our analysis.

Keywords: pregroup grammar, Japanese causatives.

1. Introduction

Japanese causatives have been analyzed from a number of points of view: transformational, lexical, and movement approaches among others. Here, I propose a computational method for analyzing the different types of causative constructions. Our analysis is based on the notion of pregroup grammar, which has been developed as an algebraic tool to recognize grammatically well-formed sentences in natural languages (Lambek, 1999).

This paper is organized as follows: section 2 introduces the pregroup formalism; section 3 describes the properties and characteristics of Japanese causatives; section 4 presents the analysis of Japanese causatives based on pregroup grammar; and finally, section 5 discusses about the implementation of a practical parser.

2. Pregroup grammar

The main idea of a pregroup grammar, which is shared with other categorial grammars, is to assign one (or more) *compound types* to each word of the language and to check the grammaticality of sentences by doing a calculation on strings of types.

We construct the compound types in the following way: we begin with a partially ordered set $(A, \rightarrow)$ of basic types, the partial order being denoted by the arrow. From the basic types we build simple types by taking adjoints or repeated adjoints. Thus from the basic type a, we obtain the simple types

$$\ldots, a^{ll}, a^{l}, a, a^{r}, a^{rr}, \ldots$$

Compound types are strings of simple types. We then form the *free pregroup* generated by A, whose elements are compound types.

The only computations required are contractions $a^{l}a \rightarrow 1$, $aa^{r} \rightarrow 1$, and expansions $1 \rightarrow aa^{l}$, $1 \rightarrow a^{r}a$, where a is a simple type.

One can extend the operation $()^l$ and $()^r$ to compound types, by defining

$$1^l = 1 = 1^r,$$
$$(a \cdot b)^l = b^l \cdot a^l,$$
$$(a \cdot b)^r = b^r \cdot a^r.$$

The symbol 1 stands for the empty string of types and will be usually omitted, as will be the dot that stands for multiplication.

What makes free pregroups particularly suitable for computation is the following observation:

SWITCHING LEMMA (Lambek, 1999) *When showing that* $x_1 \cdots x_m \rightarrow y_1 \cdots y_n$ *for simple types* x_i *and* y_i*, one may assume without loss of generality that all contractions* $a^l a \rightarrow 1$ *and* $aa^r \rightarrow 1$ *precede all expansions* $1 \rightarrow aa^l$ *and* $1 \rightarrow a^r a$.

The importance of the *Switching Lemma* is that it offers a deciding procedure, by a sequence of contractions, when a string of simple types can be rewritten as a single simple type. If expansions were needed in addition to contractions, we could for instance compute a number of expansions, followed by a number of contractions, followed by another series of expansions, and continue and so on in an ever-ending process.

So, for the purpose of sentence verification, expansions are not needed, but only contractions, combined with some rewriting by the partial order of *A*. Expansions are useful for theoretical purposes, for example to prove the following:

$$a^{rl} = a = a^{lr},$$
$$a \rightarrow b \Rightarrow b^l \rightarrow a^l,$$
$$a \rightarrow b \Rightarrow b^r \rightarrow a^r.$$

3. The structure of causative verbs

In Japanese, the causative is formed by attaching the bound causative morpheme *–(s)ase* to the stem of a verb. If the verb to which it attaches ends in a consonant, the initial consonant *s* of the morpheme is deleted.

In (1b), the causative morpheme is attached to a transitive verb stem, *kak* ‘write’. Hanako, the original agent of the transitive verb is designated with the dative particle *ni*.

(1)

a. *Hanako ga tegami o kaita.*
Hanako NOM letter ACC wrote
‘Hanako wrote a letter.’

b. *Taroo ga Hanako ni tegami o kak-ase-ta.*
Taro NOM Hanako DAT letter ACC write-CAUS-PAST
‘Taro made Hanako write a letter.’

The Japanese causative is well known for having two types, the so-called *o*-causative and the ni-causative. When the causative morpheme is attached to an intransitive verb, as in (2), the original agent of the verb stem can take either the accusative *o* or the dative *ni*.

(2)

a. *Hanako ga aruita.*
Hanako NOM walked.
‘Hanako walked.’

b. *Taroo ga Hanako o aruk-ase-ta.*
Taro NOM Hanako ACC walk-CAUS-PAST
‘Taro made Hanako walk.’

c. *Taroo ga Hanako ni aruk-ase-ta.*
Taro NOM Hanako DAT walk-CAUS-PAST
'Taro let Hanako walk.'

Kuno (1973), Shibatani (1973), and Kitagawa (1974), among others, have commented on the semantics difference between the *o*-causative and the *ni*-causative. *O*-causatives have been characterized as coercive causatives whereas *ni*-causatives have been referred to as non-coercive causatives. Note that this choice of *o* or *ni* exists only if the verb to which *–(s)ase* is attached is intransitive. If the verb is transitive, the original agent of the verb can appear only with the dative *ni*. This constraint against having two or more accusative cases in the same clause is called the Double-*o* Constraint (Harada, 1973).

4. Computational approach to causative verbs

We shall assign one ore more (compound) types to Japanese words and verify whether a given string of words is a grammatical sentence by performing a calculation in the pregroup.

First, we introduce the following basic types:

π = pronoun;
$\tilde{n}$ = proper name;
n = noun;
s_i = sentence,
i = 1 for the present tense;
i = 2 for the past tense.
c_1 = nominative complement;
c_3 = dative complement;
c_4 = accusative complement;
c_6 = ablative complement.

Concepts describing the grammatical constructions of the language fragment help to choose the basic types and the ordering. However, in selecting basic types, we are primarily interested in their algebraic effectiveness, whether or not they correspond to traditional grammatical categories.

We also postulate $n \rightarrow \tilde{n} \rightarrow \pi$.

We then assign the following types to some representative verbs:

aruku 'walk':	$c_1^r s_1$
kaku 'write':	$c_4^r c_1^r s_1$
otiru 'drop':	$c_6^r c_1^r s_1$
aruk-ase-ru 'walk-CAUS-PRES':	$c_4^r c_1^r s_1$, $c_3^r c_1^r s_1$
kak-ase-ru 'write-CAUS-PRES':	$c_4^r c_3^r c_1^r s_1$
oti-sase-ru 'drop-CAUS-PRES':	$c_6^r c_4^r c_1^r s_1$

As Harada (1973) notes, *ni*-causatives are possible only when the causee holds control over the action that he/she performs. Since 'to drop' is usually not considered as a self-controllable action, the causative verb *otisaseru* has type $c_6^r c_4^r c_1^r s_1$ but not $c_6^r c_3^r c_1^r s_1$[1].

The typing in the pregroup formalism assumes that the verb is the central point of the sentence. The type of the verb may change depending on the order of the complements.

[1] In the context such as where the causer is a film director and the causee an actor, we can imagine that the film director makes the actor/actress act as he/she directs. In this case, we could consider the verb 'to drop' as a self-controllable action and the causative verb *otisaseru* would also take the type $c_6^r c_3^r c_1^r s_1$.

However, to keep our analysis within the scope of this paper, we will consider only the most common word order and we will ignore cases of scrambling.

(3)

a. *Hanako ga tegami o kaita.*
$\tilde{n}\ (\pi^r c_1)\ n\,(\pi^r c_4)\ (c_4{}^r c_1{}^r s_2) \rightarrow s_2$
Hanako NOM letter ACC wrote
'Hanako wrote a letter.'

b. *Taroo ga Hanako ni tegami o kak-ase-ta.*
$\tilde{n}\,(\pi^r c_1)\ \tilde{n}\ \ (\pi^r c_3)\ n\ \ (\pi^r c_4)\ (c_4{}^r c_3{}^r c_1{}^r s_2) \rightarrow s_2$
Taro NOM Hanako DAT letter ACC write-CAUS-PAST
'Taro made Hanako write a letter.'

c. **Taroo ga Hanako o tegami o kak-ase-ta.*
$\tilde{n}\ (\pi^r c_1)\ \tilde{n}\ \ (\pi^r c_4)\ n\ \ (\pi^r c_4)\ (c_4{}^r c_3{}^r c_1{}^r s_2)$
Taro NOM Hanako ACC letter ACC write-CAUS-PAST

The calculations are performed in the following steps. We take as an example the string of types corresponding to the sentence (3a).

$$\begin{aligned}
\tilde{n}\,(\pi^r c_1)\ n\,(\pi^r c_4)\ (c_4{}^r c_1{}^r s_2) \quad &= \quad (\tilde{n}\,\pi^r)\ c_1\ (n\,\pi^r)\ (c_4 c_4{}^r)\ c_1{}^r\ s_2 \\
&\rightarrow \quad 1 \bullet c_1 \bullet 1 \bullet 1 \bullet c_1{}^r s_2 \\
&= \quad c_1 c_1{}^r s_2 \\
&\rightarrow \quad 1 \bullet s_2 \\
&= \quad s_2
\end{aligned}$$

Note that we used the partial order $\tilde{n} \rightarrow \pi$ and $n \rightarrow \pi$ for the contractions $\tilde{n}\,\pi^r \rightarrow \pi\ \pi^r \rightarrow 1$ and $n\,\pi^r \rightarrow \pi\ \pi^r \rightarrow 1$.

(4)

a. *Hanako ga aruita.*
$\tilde{n}\ \ (\pi^r c_1)\ \ (c_1{}^r s_2) \rightarrow s_2$
Hanako NOM walked
'Hanako walked.'

b. *Taroo ga Hanako o aruk-ase-ta.*
$\tilde{n}\ \ (\pi^r c_1)\ \tilde{n}\ \ \ (\pi^r c_4)\ (c_4{}^r c_1{}^r s_2) \rightarrow s_2$
Taro NOM Hanako ACC walk-CAUS-PAST
'Taro made Hanako walk.'

c. *Taroo ga Hanako ni aruk-ase-ta.*
$\tilde{n}\ \ (\pi^r c_1)\ \tilde{n}\ \ \ (\pi^r c_3)\ (c_3{}^r c_1{}^r s_2) \rightarrow s_2$
Taro NOM Hanako DAT walk-CAUS-PAST
'Taro let Hanako walk.'

In (5), the noncausative verb *otiru* 'to drop' allows either an animate or an inanimate subject. However, the causative counterpart *otisaseru* 'cause to drop' requires an animate object, as shown by the examples in (6).

(5)

a. *Taroo ga saka kara otita.*
$\tilde{n}\,(\pi^r c_1)\ \ n\,(\pi^r c_6)\ (c_6{}^r c_1{}^r s_2) \rightarrow s_2$
Taro NOM hill from fell
'Taro fell from the hill.'

b. *Hon ga tana kara otita.*

$n\ (\pi^r c_1)\ n\ (\pi^r c_6)\ (c_6{}^r c_1{}^r s_2) \rightarrow s_2$
book NOM shelf from fell
'The book dropped off the shelf.'

(6)

a. *Ziroo ga Taroo o saka kara oti-sase-ta.*
$\tilde{n}\ (\pi^r c_1)\ \tilde{n}\ (\pi^r c_4)\ n\ (\pi^r c_6)\ (c_6{}^r c_4{}^r c_1{}^r s_2) \rightarrow s_2$
Ziro NOM Taro ACC hill from fall-CAUS-PAST
'Ziro caused Taro to fall from the hill.'

b. **Ziroo ga hon o tana kara oti-sase-ta.*
$\tilde{n}\ (\pi^r c_1)\ n\ (\pi^r c_4)\ n\ (\pi^r c_6)\ (c_6{}^r c_4{}^r c_1{}^r s_2) \rightarrow s_2$
Ziro NOM book ACC shelf from fall-CAUS-PAST
'Ziro caused the book to drop from the shelf.'

The sentence (6a) with the animate causee is grammatical; the causee (Taro) is understood to have fallen of his own accord, and the causer (Ziro) caused this to happen indirectly. The sentence (6b) is ungrammatical because the causee, *hon* 'book', being inanimate, cannot 'drop on his own accord'.

Our current choice of typing accepts the ungrammatical sentence (6b). We must revise our types and perhaps introduce new types so that our grammar rejects the sentence (6b).

4.1.The attribute

Pregroup grammar handles features such as agreement and number by a proliferation of basic types. Consider the incorrect sentence *I sleeps* where the pronoun *I* has type π_1 and the verb *sleeps* the type $\pi_3{}^r s$, where π_3 stands for the third person singular. The sentence is incorrect since there is no agreement between the subject and the verb.

(7) *I sleeps
$\pi_1\ (\pi_3{}^r s)$

However, if the attribute is dropped, the typing is interpreted as a subject followed by an intransitive verb, and therefore the sentence *I sleeps* is accepted.

(8) *I sleeps
$\pi\ (\pi^r s) \rightarrow s$

French nouns and adjectives vary in gender and number. Degeilh et al. (2005) introduce the basic type n_{gn} to denote a complete noun phrase, depending on its gender g and number n. They postulate $n_{gn} \rightarrow n$. The type of a noun is indexed by g, which stands for 1 = masculine or 2 = feminine, and by n, where $n = 1$ means singular and $n = 2$ means plural. Nouns are count nouns or mass nouns; count nouns have type c_{gn} and mass nouns have type m_{gn}.

Consider, for example, the noun phrase *une étudiante* 'a student', where the (feminine) article *une* has type $n_{21}c_{21}{}^l$ and the (feminine) noun *étudiante* has type c_{21}.

(9) une étudiante
$(n_{21}c_{21}{}^l)\ c_{21} \rightarrow n_{21}$

We can treat the animacy restriction of sentences such as (6) in a similar way that Degeilh et al. (2005) did to handle the gender and number attributes in the French noun phrases. Adding the feature *animate* to our grammar will resolve the problem. We introduce the new basic types

n_a and $\tilde{n}_a$ where $a = 1$ means animate, and $a = 2$ means inanimate. Further, we postulate $n_a \rightarrow n$ and $\tilde{n}_a \rightarrow n$. The basic types π and c_4 will also be indexed by a : π_a and c_{4a}. We also postulate $n_a \rightarrow \pi_a$, $\tilde{n}_a \rightarrow \pi_a$, $\pi_a \rightarrow \pi$ and $c_{4a} \rightarrow c_4$.

(10)

a. *Ziroo ga Taroo o saka kara oti-sase-ta.*
$\tilde{n}\ (\pi^r c_1)\ \tilde{n}_1\ (\pi_1{}^r c_{41})\ n\ (\pi^r c_6)\ (c_6{}^r c_{41}{}^r c_1{}^r s_2) \rightarrow s_2$
Ziro NOM Taro ACC hill from fall-CAUS-PAST
'Ziro caused Taro to fall from the hill.'

b. **Ziroo ga hon o tana kara oti-sase-ta.*
$\tilde{n}\ (\pi^r c_1)\ n_2\ (\pi_2{}^r c_{42})\ n\ (\pi^r c_6)\ (c_6{}^r c_{41}{}^r c_1{}^r s_2)$
Ziro NOM book ACC shelf from fall-CAUS-PAST
'Ziro caused the book to drop from the shelf.'

Although the types c_{42} and c_{41} differ only from one index, they are as different as n and c_{41}. The sequence $c_{42}c_{41}{}^r$ does not contract to 1, and so the string of types corresponding to the sentence *Ziroo ga hon o tana kara otisaseta* cannot reduce to the simple type s_2.

The following is another example similar to *oti-sase-ru* 'cause to fall'. The intransitive verb *agaru* 'rise' allows both animate and inanimate subject, but the causative verb *agar-ase-ru* 'cause to rise' requires an animate object.

(11)

a. *Taroo ga butai ni agatta.*
$\tilde{n}\ (\pi^r c_1)\ n\ (\pi^r c_3)\ (c_3{}^r c_1{}^r s_2) \rightarrow s_2$
Taro NOM stage on rose
'Taro rose onto the stage.'

b. *Maku ga agatta.*
$n\ (\pi^r c_1)\ (c_1{}^r s_2) \rightarrow s_2$
curtain NOM rose
'The curtain rose.'

(12)

a. *Ziroo ga Taroo o butai ni agar-ase-ta.*
$\tilde{n}\ (\pi^r c_1)\ \tilde{n}_1\ (\pi_1{}^r c_{41})\ n\ (\pi^r c_3)\ (c_3{}^r c_{41}{}^r c_1{}^r s_2) \rightarrow s_2$
Ziro NOM Taro ACC stage on rise-CAUS-PAST
'Ziro caused Taro to rise onto the stage.'

b. **Ziroo ga maku o agar-ase-ta.*
$\tilde{n}\ (\pi^r c_1)\ n_2\ (\pi_2{}^r c_{42})\ (c_{41}{}^r c_1{}^r s_2)$
Ziro NOM curtain ACC rise-CAUS-PAST
'Ziro caused the curtain to rise.'

5. Parsing with pregroups

We implemented a parser to test and judge the accuracy of the proposed approach. We followed the algorithm presented in Degeilh et al. (2005), which solves the decision problem of the theory of pregroups as well as recognition by a pregroup grammar in time proportional to the cube of the length of the input string.

Buszkowski (2001) has shown that pregroup grammars are weakly equivalent to context-free grammars. Context-free parsing algorithms such as the ones proposed by Younger (1967) and Earley (1970) also have complexity n^3, however, the context-free grammar associated with a pregroup grammar would include the whole dictionary in its set of rules. Furthermore, context-

free algorithms use a constant factor which must bound the number of symbols and rules of the grammar. In Degeilh et al.'s algorithm, however, the constant depends on a bound for the number of types per word and a bound for their length, so there is no need to restrict oneself to finite dictionaries.

Before presenting their algorithm, let me introduce some technical definitions.

Definition 1. (Degeilh et al., 2005)

i) Let V be a non-empty set, A a partially ordered set and P the free pregroup generated by A. A *dictionary of vocabulary V with types in P* is a map D from V to the set of subsets of P.

ii) A dictionary D is *bounded* if there are constants k and l such that for every word $v \in V$ the set $D(v)$ has at most k elements, and each type in $D(v)$ has at most length l. A dictionary D is locally finite if $D(v)$ is finite for all $v \in V$. It is said to be finite if the sets V, A, and $D(v)$ are finite.

iii) A *type-assignment* for a string $v_1 \ldots v_n$ of elements in V is a sequence $t_1 \ldots t_n$ of types in P such that $t_i \in D(v_i)$, for $i = 1,\ldots, n$.

iv) Let a be a simple type in P. A string $v_1 \ldots v_n$ of elements in V is a-grammatical if it has a type assignment $t_1 \ldots t_n$ such that $t_1 \ldots t_n \rightarrow a$. A sequence $v_1 \ldots v_n$ is grammatical if it is a-grammatical for some simple type a.

A type checking algorithm provides a solution to the problem of grammaticality for every dictionary D in which the sets $D(v)$ are finite. A type assignment algorithm provides $v_1 \ldots v_n$ with associated strings of types from the dictionary. Enumerating all the possible type assignments would not be very efficient: if k_i is the number of elements in $D(v_i)$ then there are $k_1 k_2 \ldots k_n$ different type assignments for $v_1 \ldots v_n$. Degeilh et al.'s recognition algorithm combines type assignment and type checking.

The intuitive idea underlying the algorithm is as follows: we process the string of symbols W = $v_1 \ldots v_n$ from left to right, proceeding by stages. At each stage, we choose a symbol v_i represented by its index i, a type t in $D(v_i)$ and a position p in t. We examine the simple type(s) placed just to the left of this position in some type assignment and store it (them) in the memory, where they are kept as a 'left parenthesis' waiting to be contracted with a simple type that might come later. Moreover, each of them could also be a 'right parenthesis' to some earlier simple type, that is, a 'left parenthesis' ready for contraction. In this case, the two types are contracted; this means that the 'left parentheses' awaiting contraction at the earlier stage become available again, that is, they are stored in the memory at the present stage. This defines a function Nlp_{DW} on the set of stages. For further details regarding the *Nlp* function, please refer to Degeilh et al. (2005).

5.1.Experiment

Sentences of Japanese are written without word boundaries. The use of a morphological analyzer is thus required to identify words and their grammatical category. This preprocessing phase was realized by the use of the Japanese morphological analyzer system Mecab (http://mecab.sourceforge.net/). Mecab's output then served as input to our pregroup parser.

Here, I confined my attention to a minuscule portion of Japanese grammar, focusing on causative constructions. But as the syntactical analysis will become more exhaustive, the introduction of new types will be needed. Hence, the data must be organized in an efficient way; grammatical categories and their corresponding types were organized into a dictionary.

We designed the dictionary so that a sequence of words is a grammatical construction if and only if one of the corresponding strings of types reduces to a basic type, say α. In the case where a word has more than one type, it is sufficient that one of the possible choices yields a string reducing to the basic type α.

Every dictionary that respects this equivalence is said to be *correct* (it recognizes only grammatical constructions) and *complete* (it recognizes all grammatical constructions of the language fragment) (Degeilh et al., 2005).

A correct and complete dictionary satisfies the following robustness properties (Degeilh et al., 2005):

i) Assigning new types to words:
Suppose a word *w* has type β and we also give it type α such that $\alpha \rightarrow \beta$, then every string of words that is accepted using β for *w* is also accepted using α.

ii) Extensions by new basic types:
One can extend a given set *A* of basic types, by declaring new types and adding inequalities involving the new types, thus obtaining a larger set of basic types *A′*. Then the free pregroup *P′* generated by *A′* includes the free pregroup *P* generated by *A*.

This signifies that we can increase the language fragment by adding new types to the dictionary, without having to repeat verification of correctness and completeness performed before the extension.

6. Conclusion

We proposed a computational approach based on pregroup grammar to analyze causative constructions in Japanese. We introduced basic types and have assigned types to words which allowed us to account for the linguistic data, recognizing grammatically well-formed sentences.

Using the pregroup algorithm proposed by Degeilh et al. (2005), we developed a parser to confirm the validity and the efficiency of our grammar. Furthermore, it would be easy to add new types in the dictionary to cover other grammatical constructions.

Pregroup grammars are particularly well suited for investigating the computational aspect of language processing. The grammar rules are universal: they are the same whatever the language, only the dictionary changes. Moreover, pregroup grammars gain in expressive power by the introduction of a higher number of basic types.

The pregroup formalism may not be as expressive as other formalisms such as the HPSG framework, as it does not provide any structure for the phonological and semantic information, however, it enables us to parse sentences by means of simple calculations. Furthermore, in analyzing a sentence, we go from left to right, imitating the way a human hearer might proceed.

References

Buszkowski, W. 2001. Lambek Grammars Based on Pregroups. In P. de Groote, G. Morrill, and C. Retoré, ed., *Logical Aspects of Computational Linguistics*, 95-109. Springer LNAI 2099.

Cardinal, K. 2002. *An Algebraic Study of Japanese Grammar*. Master's thesis, McGill University.

Degeilh , S. and Preller, A. 2005. Efficiency of Pregroups and the French Noun Phrase. *Journal of Logic, Language and Information*, 14, 423-444.

Earley, J. 1970. An Efficient Context-Free Parsing Algorithm. *Communications of the AMC*, 13(2), 94-102.

Harada, S.I. 1973. Counter-Equi NP Deletion. *Research Institute of Logopedics and Phoniatrics, Annual Bulletin, University of Tokyo*, 7.

Kitagawa. C. 1974. Case Marking and Causativization. *Papers in Japanese Linguistics*, 4, 43-57.

Kuno, S. 1973. *The Structure of the Japanese Language*. No. 3 of Current Studies in Linguistics. The MIT Press, Cambridge, Massachusetts.

Lambek, J. 1999. Type Grammar Revisited. In A. Lecomte, F. Lamarche, and G. Perier, ed., *Logical Aspects of Computational Linguistics*, pp. 1-27. Springer LNAI 1582.

Miyagawa, S. 1980. Complex Verbs and the Lexicon. In Coyote Papers, 1. University of Arizona, Tucson.

Miyagawa, S. 1984. Blocking and Japanese Causatives. *Lingua*, 64, 177-207.

Miyagawa, S. 1986. Restructuring in Japanese. In T. Imai and M. Saito, ed., *Issues in Japanese Linguistics*. Foris, Dordrecht, 1986.

Miyagawa, S. 1989. Structure and Case Marking in Japanese, volume 22 of *Syntax and Semantics*. Academic Press, San Diego.

Shibatani, M. 1973. Semantics of Japanese Causativization. Foundation of Language, 9, 327-373.

Younger, D. 1967. Recognition and Parsing of Context-Free Languages in Time n3. Information and control, 10(2).