Defence Research and Development Canada
Recherche et développement pour la défense Canada

DEFENCE R&D DÉFENSE

Non-Self-Embedding Context-Free Grammars for Electronic Warfare

Fred A. Dilkes and Nikita Visnevski

Defence R&D Canada – Ottawa
TECHNICAL MEMORANDUM
DRDC Ottawa TM 2004-157
October 2004

Canada

Non-Self-Embedding Context-Free Grammars for Electronic Warfare

Fred A. Dilkes
Defence R&D Canada - Ottawa

Nikita Visnevski
McMaster University

Defence R&D Canada - Ottawa

Technical Memorandum

DRDC Ottawa TM 2004-157

October 2004

Abstract

It is possible to describe the behaviour of many multi-function radar systems in terms of context-free grammars. These grammars, although potentially natural devices for the description of the radar system, generally lead to greater computation complexity compared with simpler, regular grammars. Given a particular context-free grammar, it may or may not be possible to construct an equivalent regular grammar that describes the same behaviour. This memo is concerned with deciding whether or not such a grammar can be constructed, and presents and algorithm for generating the corresponding finite automaton.

Résumé

Il est possible de décrire le comportement de nombreux systèmes radar multifonctions à partir de grammaires acontextuelles. Bien que ces grammaires puissent offrir des moyens naturels de décrire le système radar, elles accroissent en général la complexité des calculs par rapport aux grammaires régulières, plus simples. Pour une grammaire acontextuelle donnée, il peut être possible ou non de construire une grammaire régulière équivalente qui décrive un même comportement. Le présent document vise à déterminer si une telle grammaire peut ou non être construite, et présente un algorithme pour la production de l'automate fini correspondant.

This page intentionally left blank.

Executive summary

Formal language theory has been proposed as a potentially valuable tool in electronic intelligence. The dynamics of many complex radar systems may have a concise representation in terms of a linguistic structure known as a “phase-structured grammar”. The latter are flexible mathematical constructions that may describe a wide variety of intricate waveform behaviours. These may be applied to both electronic intelligence and electronic support.

It is conjectured by the authors that all languages of interest that may describe the behaviour of threat radar systems belong to the category of “regular languages”. Every regular language can be described by a “regular grammar” and, conversely, every regular grammar describes a regular language. However, for some complex radar systems, a simple description in terms of a regular grammar may involve a large number of nontrivial states that are awkward to enumerate.

It may occur that some systems have a much more natural description in terms of a more generalized form of grammar known as a “context-free grammar”. The drawback of context-free grammars lies in the fact that their signal processing algorithms have greater time and memory requirements than do those of regular grammars. Although most context-free grammars do not describe regular languages there are some cases in which a context-free grammar is not an regular grammar but nonetheless still generates a regular language.

As a result, given a particular context-free grammar, it may or may not be possible to construct an equivalent regular grammar that generates the same language. It is known that a sufficient condition for such methods to exist is that the grammar be “non-self-embedding”.

This memo answers the following questions: If a radar system has a natural description in terms of a context-free grammar then how is one to (i) verify that the grammar is non-self-embedding, and therefore describes a regular language, and (ii) derive a regular grammar that describes that language?

Fred A. Dilkes, Nikita Visnevski. 2004. Non-Self-Embedding Context-Free Grammars for Electronic Warfare. DRDC Ottawa TM 2004-157. Defence R&D Canada - Ottawa.

Sommaire

On a proposé de faire appel à la théorie des langages formels comme outil potentiellement utile dans le domaine du renseignement électronique. La dynamique de nombreux systèmes radar complexes pourrait être représentée de manière concise selon une structure linguistique appelée “ grammaire syntagmatique ”. Les grammaires syntagmatiques sont des constructions mathématiques souples qui peuvent décrire une grande variété de comportements d’onde compliqués. Elles peuvent s’appliquer à la fois au renseignement électronique et au soutien électronique.

Les auteurs conjecturent que tous les langages d’intérêt qui peuvent décrire le comportement de systèmes radar ennemis font partie de la catégorie des langages réguliers. Tout langage régulier peut être décrit par une grammaire régulière et, inversement, toute grammaire régulière décrit un langage régulier. Toutefois, dans le cas de certains systèmes radars complexes, une simple description selon une grammaire régulière peut impliquer un grand nombre d’états non triviaux qu’il est malaisé d’énumérer.

Il peut arriver que certains systèmes peuvent être décrites de façon beaucoup plus naturelle selon une forme de grammaire plus généralisée appelée “ grammaire acontextuelle ”. L’inconvénient des grammaires acontextuelles tient au fait que leurs algorithmes de traitement des signaux exigent plus de temps et de mémoire que ceux des grammaires régulières. Bien que la plupart des grammaires acontextuelles ne décrivent pas des langages réguliers, il y a des cas où une grammaire acontextuelle n’est pas une grammaire régulière, mais génère néanmoins un langage régulier.

Par conséquent, pour une grammaire acontextuelle donnée, il peut être possible ou non de construire une grammaire régulière équivalente qui génère le même langage. On sait qu’une condition suffisante pour l’existence de ces méthodes est que la grammaire ne fasse pas appel à l’auto-enchâssement.

Le présent document répond aux questions suivantes : si un système radar a une description naturelle selon une grammaire acontextuelle, comment peut-on (i) s’assurer que la grammaire ne fait pas appel à l’auto-enchâssement, et que, par conséquent, elle décrit un langage régulier, et (ii) construire une grammaire régulière qui décrit ce langage ?

Fred A. Dilkes, Nikita Visnevski. 2004. Non-Self-Embedding Context-Free Grammars for Electronic Warfare. DRDC Ottawa TM 2004-157. R & D pour la défense Canada - Ottawa.

Table of contents

List of figures

1 Introduction

The framework of formal languages [1] and their stochastic variants [2] has been proposed as a potential tool for modeling the behaviour of radar systems in "electronic support" (ES) and "electronic intelligence" (ELINT) [3, 4, 5, 6].

It is conjectured by the authors that all languages of interest that may describe the behaviour of threat radar systems belong to the category of "regular languages" (RLs). Every RL can be described by a "regular grammar" (RG) and, conversely, every RG describes an RL. Regular grammars are closely related state-machine descriptions such as "nondeterministic finite automata" (NFA) [1] in the non-stochastic case and "hidden Markov models" [7] in the stochastic case. However, for some complex radar systems, a simple description in terms of a regular grammar may involve a large number of nontrivial states that are awkward to enumerate.

It may occur that some systems have a much more natural description in terms of "context-free grammars" (CFGs) [1]. The complete set of all RGs forms a sub-category of the larger category of CFGs. Although most CFGs do not describe RLs there are some cases in which a CFG is not an RG but nonetheless still generates a RL. As a result, given a particular CFG, it may or may not be possible to construct an equivalent RG that generates the same language.

The drawback of CFGs lies in the fact that their signal processing algorithms have greater time and memory requirements than do those of RGs. The optimal algorithm for computing the overall probability that a stochastic CFG with N nonterminal symbols will generate a given string of length L is known as the "inside algorithm" [8] whose has a time complexity of $O(N^3L^3)$ and a memory complexity of $O(NL^2)$. By contrast, the the "forward algorithm" [7, 9, 4] is the optimal solution for strings generated with hidden Markov models and exhibits a time complexity of $O(NL^2)$ and a memory complexity of $O(NL)$.

This memo attempts to address the following questions: If a radar system has a natural description in terms of a context-free grammar then how is one to (i) verify that the radar language is regular, and (ii) derive a regular grammar that describes that language? It is known that a sufficient, but not necessary, condition for such methods to exist is that the CFG be "non-self-embedding" (NSE).

This document is organized as follows. The concepts of languages and grammars, including the definitions of context-free, regular and non-self-embedding grammars, are reviewed in Section 2; the reader who is already familiar with these concepts may safely skip directly to the subsequent section. In Sections 3 and 4 describe a framework within which the questions of interest can be evaluated. The construction of an algorithm for determining a suitable regular language is presented in Section 5.

2 Languages and Grammars

This section contains a review of some elements of the theory of formal languages and grammars. A more complete and rigorous treatment of deterministic grammars can be found in [1]. Stochastic grammars are discussed in [2, 8, 9].

2.1 Deterministic formal languages

The framework begins with a finite set of symbols known as the "vocabulary", denoted by $\mathcal{T}$. A "string" is any finite sequence of symbols in the vocabulary. The set of all strings over a set of symbols $\mathcal{T}$ is denoted by $\mathcal{T}^*$. The number of symbols contained in the string α is called the "length" of the string, and is denoted by $|\alpha|$. For example, a sequence $\alpha = (a_1 a_2 \cdots a_L)$ is a string in $\mathcal{T}^*$ of length $|\alpha| = L$, as long as each component a_m, $m = 1 \ldots L$ is an element of $\mathcal{T}$. There is a unique "empty string" denoted by $\varepsilon \in \mathcal{T}^*$ whose length vanishes, $|\varepsilon| = 0$.

A "deterministic formal language" $\mathcal{L} \subseteq \mathcal{T}^*$ is some particular set of strings over a specified vocabulary $\mathcal{T}$. This notion is very broad, and potentially very complex as the language may contain an infinite number of different strings. Nevertheless, it is often possible to describe languages of practical interest in terms of a finite set of rules.

As a simple example, consider the language over the vocabulary $\mathcal{T} = \{a, b, c\}$ consisting of all finite repetitions of the sequence *bcac*, including the null string:

$$\begin{aligned} \mathcal{L} &= \{\varepsilon, bcac, bcacbcac, \ldots\} \\ &= \{(bcac)^n | n = 0, 1, 2, 3, \ldots\}. \end{aligned} \tag{1}$$

This language is quite repetitive and may represent a signal with a degree of stability. The individual strings do not contain much information and it is trivial to determine whether or not a string is in this language.

2.2 Context-Free Grammars and Languages

For the ES application, the language would represent all possible combination of sequences that a radar could ever execute, from power-up to shutdown. This set is effectively infinite even for the simplest radar system. For electronically agile radar systems, the language can be quite sophisticated and does not have a straightforward description like that shown in (1). A general mechanism is required to define the language associated with complex radar systems.

Many formal languages of interest can be represented using a constructive mathematical formalism originally developed by Chomsky [10] known as a "phase-structured grammar". A complete treatment of this construction can be found in [1]. Here, it suffices to restrict attention to a particular subclass known as a "context-free grammar".

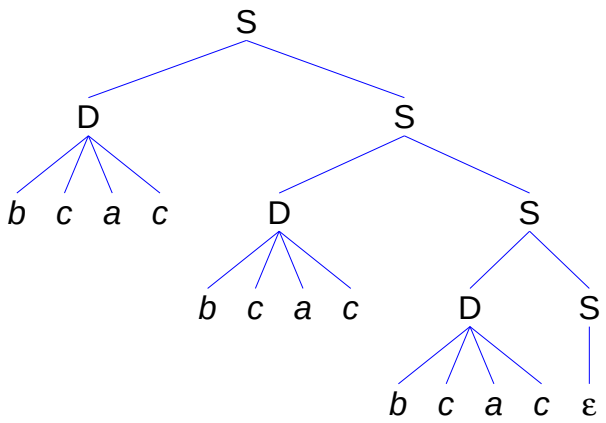


Figure 1: *A parse tree for the derivation* $S \stackrel{*}{\Rightarrow} bcacbcacbcac$.

.

Definition 1*A "context-free grammar" (CFG) is a 4-tuple* $\mathcal{G} = (\mathcal{N}, \mathcal{T}, \mathcal{R}, S)$. *The sets* $\mathcal{T}$ *and* $\mathcal{N}$ *are disjoint collections of symbols known respectively as "terminals" and "nonterminals";* $\mathcal{T} \cap \mathcal{N} = \emptyset$. *The set* $\mathcal{R} \subset \mathcal{N} \times (\mathcal{T} \cup \mathcal{N})^*$ *is known as the set of "replacement rules". The symbol* $S \in \mathcal{N}$ *is called the "starting symbol".*

If $X \in \mathcal{N}$ and $\Omega \in (\mathcal{T} \cup \mathcal{N})^*$ then the statement $(X, \Omega) \in \mathcal{R}$ may be written $X \to \Omega$ which is read to mean "X can be replaced by Ω."

To illustrate the connection between grammars and formal languages, consider an example grammar $\mathcal{G}$ prescribed by

$$\mathcal{G} = \left(\begin{array}{ll} \mathcal{N} = \{S, D\}, & \mathcal{T} = \{a, b, c\}, \\ \mathcal{R} = \{S \to DS, S \to \varepsilon, D \to bcac\}, & S \end{array} \right). \tag{2}$$

This grammar is a generative model for the language (1). It is possible, for example, to derive the example string *bcacbcacbcac* from the starting symbol S by invoking a sequence of replacements represented by the graph in Figure 1. Such a graph is known as a "parse tree". At each branch, one of the nonterminals is replaced using one of the rules in $\mathcal{R}$. The figure shows three instances of the replacement $S \to DS$, three of $D \to bcac$ and one of $S \to \varepsilon$.

If a specified CFG admits a parse tree whose root node is $X \in \mathcal{N}$ and whose yield is $\Omega \in (\mathcal{T} \cup \mathcal{N})^*$ then one says that "Ω can be derived from X" or simply $X \stackrel{*}{\Rightarrow} \Omega$. In this notation, Figure 1 implies that

$$S \stackrel{*}{\Rightarrow} bcacbcacbcac\,.$$

The language $\mathcal{L}(\mathcal{G})$ described by a grammar $\mathcal{G}$ is the set of strings of terminals that can be derived from S.

Definition 2*A "context-free language" (CFL) is a language that can be described by a context-free grammar.*

2.3 Regular Grammars

A very special subclass of context-free grammars is the case of "regular grammars" (RGs). There are two different types of regular grammars.

Definition 3*A CFG $\mathcal{G} = (\mathcal{N}, \mathcal{T}, \mathcal{R}, S)$ is "right-linear" if each production rule in $\mathcal{R}$ is either of the form $X \to \alpha Y$ or $X \to \alpha$ where $X, Y \in \mathcal{N}$ and $\alpha \in \mathcal{T}^*$. Similarly, a CFG $\mathcal{G} = (\mathcal{N}, \mathcal{T}, \mathcal{R}, S)$ is "left-linear" if each production rule in $\mathcal{R}$ is either of the form $X \to Y\alpha$ or $X \to \alpha$. A CFG is a "regular grammar" (RG) if it is either left-linear or right-linear or both.*

Every RG is a CFG but the converse is not true.

Definition 4*A "regular language" (RL) is a language that can be described by a regular grammar.*

2.4 Non-Self-Embedding Grammars

Another important subcategory of context-free grammars is that of "non-self-embedding" grammars, first identified by Chomsky [11].

Definition 5*A context-free grammar $\mathcal{G} = (\mathcal{N}, \mathcal{T}, \mathcal{R}, S)$ is called "self-embedding" if $\exists X \in \mathcal{N}$ such that $\mathcal{G}$ admits a derivation of the form $X \stackrel{*}{\Rightarrow} \Omega X \Gamma$ in which $\Omega, \Gamma \in (\mathcal{T} \cup \mathcal{N})^* \backslash \{\varepsilon\}$. A CFG is called "non-self-embedding" (NSE)*[1] *if it is not self-embedding.*

In other words, a non-self-embedding grammar has the property that no nonterminal may reproduce itself with a nonempty prefix and postfix.

Every regular grammar is non-self-embedding. It is known [11] that every non-self-embedding CFG describes a regular language. Consequently, any NSE grammar can be replaced by an equivalent RG that generates the same language.

3 Order Relations and Equivalence Classes

In following sections, an attempt is made to address the following weaker version of the questions outlined in the introduction: given a CFG, (i) how is one to verify whether or

[1] In the terminology of [12], and NSE context-free grammar is known as a "strongly regular" grammar.

not it is NSE and (ii) if it is NSE, then how is one to construct a representative RG? Answers to these questions have appeared in in at least two similar but seemingly independent sources [12] and [13]. The results are paraphrased, compiled and summarized here.

The construction begins with a binary relation $\mathcal{E}$ defined over the set of nonterminals $\mathcal{N}$. This relation is defined by

$$\mathcal{E} \equiv \left\{ (X,Y) \in \mathcal{N} \times \mathcal{N} \middle| \exists \Omega, \Gamma \in (\mathcal{T} \cup \mathcal{N})^*, (X \to \Omega Y \Gamma) \in \mathcal{R} \right\}. \tag{3}$$

Consistent with common notational practice, $X \mathcal{E} Y$ is equivalent to $(X,Y) \in \mathcal{E}$.

From the relation $\mathcal{E}$ we can construct an order relation $\precsim$ defined to be the "reflexive and transitive closure" [1, 14] of $\mathcal{E}$. In other words, $\precsim$ is the minimal relation for which satisfies the reflexivity, transitivity and closure properties, respectively given by (i) $X \precsim X$, (ii) $X \precsim Y$ and $Y \precsim Z$ implies that $X \precsim Z$, and (iii) $X \mathcal{E} Y$ implies that $X \precsim Y$ for $\forall X, Y, Z \in \mathcal{N}$.

In terms of grammatical derivations, the order relation $\precsim$ is equivalent to

$$X \precsim Y \text{ if and only if } \exists \Omega, \Gamma \in (\mathcal{T} \cup \mathcal{N})^* \text{ such that } X \stackrel{*}{\Rightarrow} \Omega Y \Gamma \tag{4}$$

where $\stackrel{*}{\Rightarrow}$ indicates the existence of a derivation sequence associated with the given CFG $\mathcal{G}$. Reflexivity requires that this this include a trivial derivation $X \stackrel{0}{\Rightarrow} X$ whose complete parse tree consists of exactly one node X and no productions so that $X \precsim X$ for $\forall X \in \mathcal{N}$.

The order relation $\precsim$ gives rise to a natural equivalence relation $\sim$ on $\mathcal{N}$

$$X \sim Y \text{ if and only if } X \precsim Y \text{ and } Y \precsim X, \text{ for } \forall X, Y \in \mathcal{N}.$$

The equivalence relation can be used to partition $\mathcal{N}$ into a set of equivalence classes. The equivalence class containing $X \in \mathcal{N}$ is denoted by

$$[X] = \{ Y \in \mathcal{N} | Y \sim X \}.$$

The distinct equivalence classes are pairwise disjoint so that $[X] = [Y]$ if and only if $X \sim Y$, otherwise $[X] \cap [Y] = \emptyset$. The set of all equivalence classes

$$\tilde{\mathcal{N}} \equiv \{ [X] | X \in \mathcal{N} \}$$

covers the entire set of nonterminals

$$\bigcup_{[X] \in \tilde{\mathcal{N}}} [X] = \mathcal{N},$$

and is known as a finite "partition" of the set $\mathcal{N}$.

The order relation $\precsim$ on $\mathcal{N}$ naturally induces an order relation $\preceq$ on $\tilde{\mathcal{N}}$ defined by

$$[X] \preceq [Y] \text{ if and only if } X \precsim Y.$$

This notion is independent of the particular nonterminals used to represent $[X]$ and $[Y]$ since $X \precsim Y$ implies that $X' \precsim Y'$ for all $X' \in [X]$ and $Y' \in [Y]$. This is a "partial order relation" [14] on the set of equivalence classes since it satisfies the reflexivity, transitivity and antisymmetry conditions.

The following lemma is straightforward to demonstrate [13].

Lemma 1*Every context-free grammar that contains exactly one equivalence class is a regular grammar if and only if it is non-self-embedding.*

3.1 Production Graphs

The pair $(\mathcal{N}, \mathcal{E})$ defines a "directed graph" [1, 14] whose nodes are identified with nonterminal symbols $\mathcal{N}$ and whose directed edges are given by the relation $\mathcal{E}$. In [13], such a construction is known as a "production graph" for the grammar $\mathcal{G}$. The relation $X \precsim Y$ is equivalent to the statement that the production graph admits a directed path from node X to node Y. The equivalence classes $[X]$ represent a partitioning of the production graph $(\mathcal{N}, \mathcal{E})$ into "strongly connected components" [14].

The significance of this analogy is that efficient algorithms for the identification of strongly connected components of directed graphs are known (see for example Tarjan [15]), and can be applied to determine the equivalence class partitions directly. In addition, simpler brute-force approaches exist [13] and are very easy to implement.

It should be cautioned that the production graph described in this subsection is not a state machine and should not be confused with a finite automaton since there is no meaningful association between the graph edges $\mathcal{E}$ and the vocabulary $\mathcal{T}$. Consequently, the production graph is not a complete representation of all of the information in the original grammar and the mapping $\mathcal{G} \mapsto (\mathcal{N}, \mathcal{E})$ is not one-to-one. Indeed, most CFGs do not admit an exact state machine representation of any kind. As will be illustrated in Section 5, those exceptional CFGs that can be represented using state machines will normally yield a state space Q that is much larger than the original set of nonterminal symbols $\mathcal{N}$.

3.2 Example

The preceding concepts can be illustrated by the following example. Suppose that a grammar $\mathcal{G} = (\mathcal{N}, \mathcal{T}, \mathcal{R}, S)$ has the following symbols

$$\mathcal{N} = \{S, A, B, C, D, E\}, \tag{5a}$$
$$\mathcal{T} = \{a, b, c, d\}, \tag{5b}$$

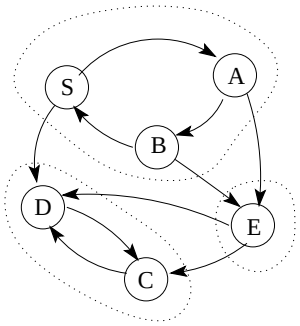


Figure 2: *The production graph of* $\mathcal{G}$

and the following production rules

$$\mathcal{R} = \left\{ \begin{array}{l} S \rightarrow DA, \\ A \rightarrow bEaB, \\ B \rightarrow aE, \\ B \rightarrow S, \\ C \rightarrow bD, \\ D \rightarrow daC, \\ D \rightarrow a, \\ E \rightarrow D, \\ E \rightarrow Cc \end{array} \right\}. \tag{5c}$$

The production graph $(\mathcal{N}, \mathcal{E})$ for this grammar is illustrated in Figure 2. It is straightforward to see that the partition consists of exactly three equivalence classes $\tilde{\mathcal{N}} = \{[S],[C],[E]\}$ given by $[S] = \{S,A,B\}$, $[C] = \{C,D\}$ and $[E] = \{E\}$. The partition is indicated in Figure 2 by dotted lines enclosing the appropriate nodes.

The complete order relation of distinct elements of $\tilde{\mathcal{N}}$ is given by $[S] \preceq [E]$, $[S] \preceq [C]$ and $[E] \preceq [C]$.

4 Composition and Decomposition of CFGs

In general, the order relation $\preceq$ is not guaranteed to represent a total ordering since there may exist a pairs of equivalence classes $([X],[Y])$ that do not satisfy either $[X] \preceq [Y]$ or $[Y] \preceq [X]$. Nevertheless, there must exist a permutation of the equivalence classes that respects the partial ordering; such an arrangement is known as a "linear extension" [14]. Let n be the number of distinct equivalence classes in $\tilde{\mathcal{N}}$. Let $(\mathcal{N}_0, \mathcal{N}_1, \cdots, \mathcal{N}_{n-1})$ be an ordering of these equivalence classes, arranged so that $\mathcal{N}_i \preceq \mathcal{N}_j$ only if $i \leq j$ and $\mathcal{N}_i = \mathcal{N}_j$ if and only if $i = j$.

Hereafter, it is assumed that the grammar $\mathcal{G}$ has no useless nonterminal symbols so that every nonterminal can be reached from S. This implies that $[S] \preceq [A]$ for every class $[A] \in \tilde{\mathcal{N}}$ and guarantees that $\mathcal{N}_0 = [S]$.

4.1 Decomposition of Grammars

Let us suppose that each equivalence class $\mathcal{N}_i$, $i = 0 \ldots n-1$ we define a new context-free grammar $\mathcal{G}_i = (\mathcal{N}_i, \mathcal{T}_i, \mathcal{R}_i, S_i)$ whose terminals and productions are described as follows:

$$\begin{aligned} \mathcal{T}_i &\equiv \mathcal{T} \cup \bigcup_{j=i+1}^{n} \mathcal{N}_j \,, \\ \mathcal{R}_i &\equiv \left\{ (X \to \Omega) \in \mathcal{R} \middle| X \in \mathcal{N}_i, \Omega \in (\mathcal{T} \cup \mathcal{N})^* \right\}, \\ S_i &\in \mathcal{N}_i \,. \end{aligned}$$

The starting symbol for the first equivalence class is defined to be $S_0 \equiv S$. For all other grammars the starting symbol is an arbitrary element of the appropriate nonterminal set. If a grammar with a particular start symbol is required, then it is denoted by $\mathcal{G}_{i;X} = (\mathcal{N}_i, \mathcal{T}_i, \mathcal{R}_i, X)$ for any $X \in \mathcal{N}_i$. In this way, the original grammar $\mathcal{G}$ is used to generate a set of sub-grammars $\{\mathcal{G}_i | i = 0, \ldots, n-1\}$.

The NSE property of the sub-grammars $\{\mathcal{G}_i | i = 0, \ldots, n-1\}$ is inherited from that of the original grammar $\mathcal{G}$ according to the following lemma [13]

Lemma 2 *$\mathcal{G}$ is self-embedding if and only if at least one of the grammars $\mathcal{G}_i$ is self-embedding.*

Combining Lemma 2 with Lemma 1 it immediately follows that

Lemma 3 *$\mathcal{G}$ is NSE if and only if each grammar in the set $\{\mathcal{G}_i | i = 0, \ldots, n-1\}$ is regular.*

4.2 Composition of Grammars

A concise relationship between $\mathcal{G}$ and $\{\mathcal{G}_i | i = 0, \ldots, n-1\}$ can be established by defining the following operation [13]:

Definition 6 *Suppose that $\mathcal{G}_1 = (\mathcal{N}_1, \mathcal{T}_1, \mathcal{P}_1, S_1)$ and $\mathcal{G}_2 = (\mathcal{N}_2, \mathcal{T}_2, \mathcal{P}_2, S_2)$ are any two context-free grammars such that $\mathcal{N}_1 \cap \mathcal{N}_2 = \emptyset$ and $\mathcal{N}_1 \cap \mathcal{T}_2 = \emptyset$. Under these circumstances, the "composition" of the two grammars $\mathcal{G} = \mathcal{G}_1 \circ \mathcal{G}_2$ is defined by $\mathcal{G} = (\mathcal{N}, \mathcal{T}, \mathcal{R}, S)$ where $\mathcal{N} = \mathcal{N}_1 \cup \mathcal{N}_2$, $\mathcal{T} = (\mathcal{T}_1 \cup \mathcal{T}_2) \backslash \mathcal{N}_2$, $\mathcal{R} = \mathcal{R}_1 \cup \mathcal{R}_2$ and $S = S_1$.*

It is straightforward to show that this composition operation is associative, however it is not commutative. From the construction of $\{\mathcal{G}_i | i = 0, \ldots, n-1\}$, in Subsection 4.1 it immediately follows that

$$\mathcal{G} = \mathcal{G}_0 \circ \mathcal{G}_1 \circ \cdots \circ \mathcal{G}_{n-1} \,.$$

4.3 Example

Referring to the example in Subsection 3.2, we see the order relation has only one linear extension since $[S] \preceq [E] \preceq [C]$. The resulting decomposition of grammars is has the following components

$$\mathcal{G}_0 = \left(\begin{array}{rcl} \mathcal{N}_0 & = & \{S,A,B\}, \\ \mathcal{T}_0 & = & \{a,b,c,d,C,D,E\}, \\ \mathcal{R}_0 & = & \left\{ \begin{array}{l} S \rightarrow DA, \\ A \rightarrow bEaB, \\ B \rightarrow aE, \\ B \rightarrow S \end{array} \right\}, \\ S_0 & = & S \end{array} \right) \tag{6a}$$

$$\mathcal{G}_{1;E} = \left(\begin{array}{rcl} \mathcal{N}_1 & = & \{E\}, \\ \mathcal{T}_1 & = & \{a,b,c,d,C,D\}, \\ \mathcal{R}_1 & = & \left\{ \begin{array}{l} E \rightarrow D, \\ E \rightarrow Cc \end{array} \right\}, \\ S_1 & = & E \end{array} \right) \tag{6b}$$

$$\mathcal{G}_{2;X} = \left(\begin{array}{rcl} \mathcal{N}_2 & = & \{C,D\}, \\ \mathcal{T}_2 & = & \{a,b,c,d\}, \\ \mathcal{R}_2 & = & \left\{ \begin{array}{l} C \rightarrow bD, \\ D \rightarrow daC, \\ D \rightarrow a \end{array} \right\}, \\ S_{2;X} & = & X \end{array} \right), \tag{6c}$$

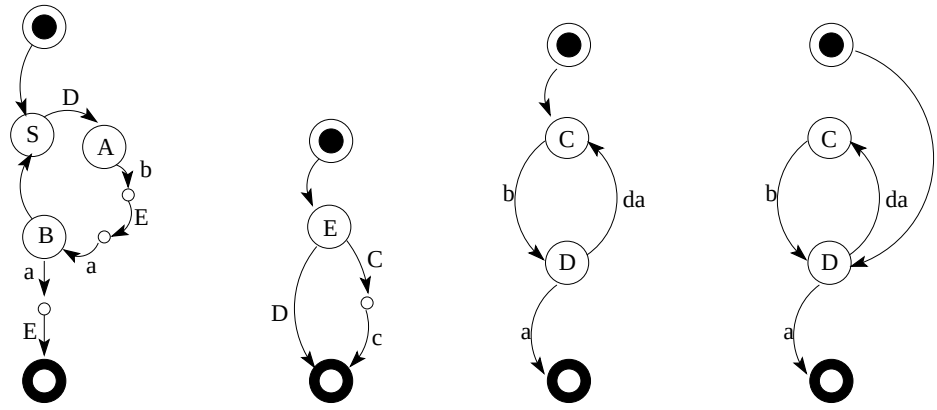
where, in (6c), X may represent either of the nonterminals contained in $\mathcal{N}_2$.

It is straightforward to see that each of the grammars $\mathcal{G}_0$, $\mathcal{G}_1$ and $\mathcal{G}_2$ is right-linear and therefore regular. It follows from Lemma 3 that $\mathcal{G}$ must be non-self-embedding and consequently describes a regular language.

5 Construction of a Finite State Automaton

If a particular CFG is determined to be non-self-embedding, then it must describe a regular language. In turn, any regular language can be described by right-linear grammar or, equivalently [1] by a "nondeterministic finite automaton".

Definition 7 *A "nondeterministic finite automaton" (NFA) is a 5-tuple $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ with the following components. The set Q is a collection of "states". The set Σ is the collection of "input symbols". The set $\delta \subseteq Q \times \Sigma^* \times Q$ is the "transition relation". The state $q_0 \in Q$ is the "initial" state. The set $F \subseteq Q$ is a collection of "terminal" or "accepting" states.*



(a) NFA $\mathcal{A}_0$ (b) NFA $\mathcal{A}_{1;E}$ (c) NFA $\mathcal{A}_{2;C}$ (d) NFA $\mathcal{A}_{2;D}$

Figure 3: *Component finite automata.*

Note that this definition allows for transitions in δ to be associated with strings in Σ^* (including the null string ε) rather than being restricted to singleton symbols Σ, as more conventional [1]. It is always straightforward to convert such an automaton to an equivalent automaton in which $\delta \subseteq Q \times \Sigma \times Q$ by inserting or merging states as appropriate.

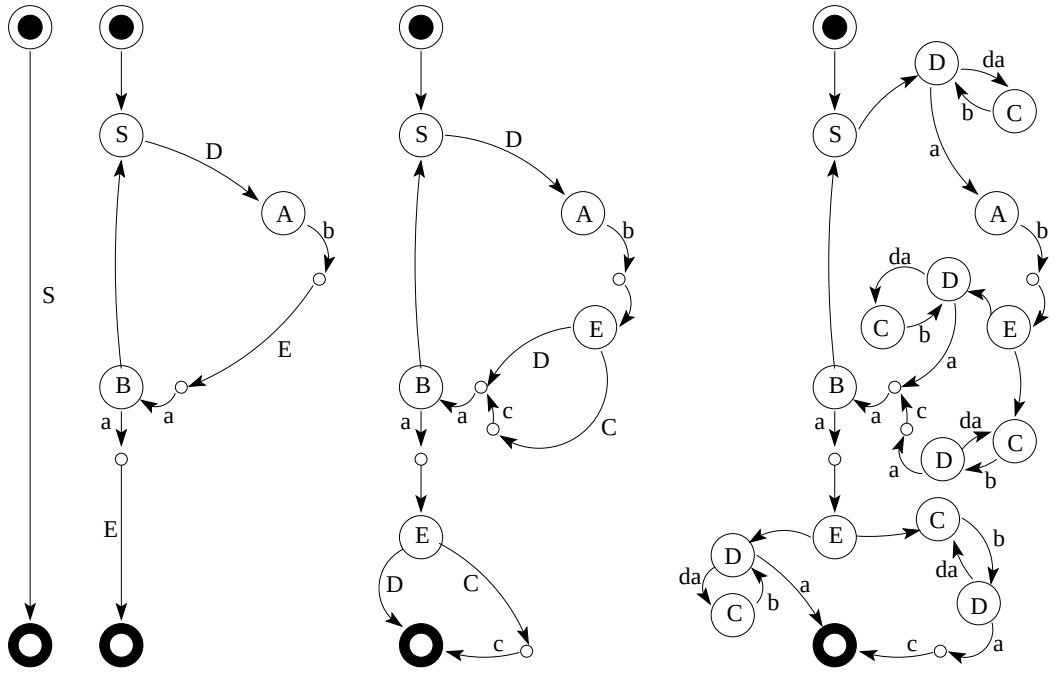
5.1 Example

Before producing an algorithm to determine the NFA from the NSE grammar, it is instructive to consider the construction for the example in Subsections 3.2 and 4.3.

The technique is most easily understood by first constructing, for each grammar $\mathcal{G}_{i;X}$ in the decomposition, an intermediate NFA, denoted by $\mathcal{A}_{i;X} = (Q_i, \Sigma_i, \delta_{i;X}, q_{0;i}, F_i)$. Automata corresponding to grammars (6a) and (6b) are respectively shown in Figures 3(a) and 3(b). Initial states $q_{0;i}$ are illustrated by a solid dot and terminal states in F_i are represented by a solid ring. The grammars of (6c) are represented by two automata shown in Figures 3(c) and 3(d) corresponding to $\mathcal{G}_{2;C}$ and $\mathcal{G}_{2;D}$ respectively.

The construction of an NFA $\mathcal{A}_{i;X}$ from a right-linear grammar $\mathcal{G}_{i;X}$ is a very straightforward exercise. In each case, the set of input symbols is associated with the set of terminal symbols of the corresponding grammar, $\Sigma_i \equiv \mathcal{T}_i$. The set of states Q_i includes a disjoint union of one initial state $q_{0;i}$, exactly one accepting state in F_i and the set of nonterminals $\mathcal{N}_i$. The transition relation $\delta_{i;X}$ is constructed from the regular production rules $\mathcal{R}_i$ and includes the transition $(q_{0;i}, \varepsilon, X)$. The only subtlety here is that each automata has been constructed in such a way so that its transitions are all associated either with strings in $\mathcal{T}^*$ or with nonterminals in $\mathcal{N}$; no transition includes both terminals and nonterminals so that $\delta_{i;X} \subseteq (Q \times \mathcal{T}^* \times Q) \cup (Q \times (\mathcal{T}_k \backslash \mathcal{T}) \times Q)$. This is accomplished by adding additional states to Q_i, illustrated by small circles, in order to isolate the transitions associated with nonterminal symbols.

The complete automata is shown to be constructed in four stages, illustrated in Figure 4.



(a) (b) NFA for $\mathcal{G}_0$ (c) NFA for $\mathcal{G}_0 \circ \mathcal{G}_1$ (d) NFA for $\mathcal{G}_0 \circ \mathcal{G}_1 \circ \mathcal{G}_2$

Figure 4: *Nondeterministic finite automata.*

The construction begins with a simple finite automata shown in Figure 4(a). The automata has only two states and one transition associated with the symbol *S*, from the initial state to the terminal state. The automaton in Figure 4(b) is obtained from the graph in Figure 4(a) by replacing the transition associated with the output symbol *S* by the corresponding intermediate state machine in Figure 3(a). The head of the original transition in Figure 4(a) is identified with the initial state of Figure 3(a), while the terminal state of the intermediate machine is identified with the tail of the original transition. The resulting graph in Figure 4(b) is actually identical to Figure 3(a) and can be considered to be the automata associated with the grammar $\mathcal{G}_0$.

Figure 4(c) is obtained from Figure 4(b) by replacing any transition associated with the output symbol E with an instance of the state machine in Figure 3(b). As before, the head of the old transition is identified with the initial symbol of Figure 3(b). The terminal state of Figure 3(b) is identified with the tail of the old transition. Figure 4(b) can be interpreted as an automata associated with the grammar $\mathcal{G}_0 \circ \mathcal{G}_1$. Notice in this graph that there are several nodes labeled with the nonterminal symbol E. The labels in these diagrams are not unique identifiers for the states and are shown here only for clarity of presentation; in general, the mapping from states in the resulting NFA to nonterminals in the original context-free grammar is many-to-one.

Figure 4(d) is the final automata for the grammar $\mathcal{G} = \mathcal{G}_0 \circ \mathcal{G}_1 \circ \mathcal{G}_2$. It is obtained from Figure 4(c) by replacing *C*-transitions with Figure 3(c) and *D*-transitions with Figure 3(d).

procedure createNFA
  $\mathcal{Q} \leftarrow \emptyset$; $\delta \leftarrow \emptyset$; $\Sigma \leftarrow \mathcal{T}$
  $q_0 \leftarrow$ addNewState
  $f \leftarrow$ addNewState
  $F \leftarrow \{f\}$
  insertNFA(q_0, S, f)
end

Figure 5: *Procedure* "createNFA" *creates a finite automaton.*

procedure addNewState
  $q \leftarrow$ createNewState
  $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{q\}$
  return q
end

Figure 6: *Procedure* "addNewState" *creates a fresh new state and adds it to the automaton without modifying the transition functions.*

As before, the head and tail of the old transition are associated respectively with the initial and terminal states the relevant automata in Figure 3.

Finally, it should be noted that grammars in (6) are all right-linear. If any of the grammars $\mathcal{G}_i$ had been left-linear, then the corresponding component NFA could have been constructed, according to [1], by reversing the right-hand-side of all of every production rule in $\mathcal{P}_i$, constructing the NFA for the resulting right-linear grammar, reversing the orientation of every transition and exchanging the identification of initial and terminal states. Once these are constructed, their insertion into the full automaton for $\mathcal{G}$ follows exactly the same procedure illustrated above.

5.2 Nederhof's algorithm

The illustration shown in Subsection 5.1 proceeds in a "breadth-first" fashion meaning that the entire NFA for each composition of the form $\mathcal{G}_0 \circ \mathcal{G}_1 \circ \cdots \circ \mathcal{G}_k$ is computed before computing the NFA for the subsequent composition $\mathcal{G}_0 \circ \mathcal{G}_1 \circ \cdots \circ \mathcal{G}_k \circ \mathcal{G}_{k+1}$. Nederhof's algorithm proceeds instead in a recursive "depth-first" fashion where each nonterminal transition is resolved completely into terminal transitions before the next nonterminal transition is considered. A variation on this algorithm is given in Figures 5, 6 and 7.

The primary difference between the algorithm presented here, and that of Nederhof [12] is the treatment of so-called "nonrecursive" nonterminals. A nonterminal $X \in \mathcal{N}$ is said to be "recursive" if the production graph $(\mathcal{N}, \mathcal{E})$ admits a non-trivial directed path from X to X. If a nonterminal X is not recursive, then is guaranteed to form its own singleton equivalence class $[X] = \{X\}$. In the example of Subsection 3.2, E is the only nonterminal

procedure insertNFA($q_\triangleleft, \alpha, q_\triangleright$)
  if $\alpha \in \mathcal{T}^*$ **then** $\delta \leftarrow \delta \cup \{(q_\triangleleft, \alpha, q_\triangleright)\}$; **end**; **return**
  $\exists (A \in \mathcal{N}), (\beta_\triangleleft, \beta_\triangleright \in (\mathcal{N} \cup \mathcal{T})^*)$ **such that** $\alpha = \beta_\triangleleft A \beta_\triangleright$
  if $\beta_\triangleleft = \varepsilon$ **then**
    $\tilde{q}_\triangleleft \leftarrow q_\triangleleft$
  else
    $\tilde{q}_\triangleleft \leftarrow$ addNewState
    insertNFA($q_\triangleleft, \beta_\triangleleft, \tilde{q}_\triangleleft$)
  end
  if $\beta_\triangleright = \varepsilon$ **then**
    $\tilde{q}_\triangleright \leftarrow q_\triangleright$
  else
    $\tilde{q}_\triangleright \leftarrow$ addNewState
    insertNFA($\tilde{q}_\triangleright, \beta_\triangleright, q_\triangleright$)
  end
  $\exists \mathcal{N}_k \in \tilde{\mathcal{N}}$ **such that** $A \in \mathcal{N}_k$
  for $\forall B \in \mathcal{N}_k$, **do** $q[B] \leftarrow$ addNewState **end**
  if isRightLinear($\mathcal{G}_k$) **then**
    for $\forall (C \rightarrow X_1 \cdots X_{m-1} X_m) \in \mathcal{P}_k$ **do**
      if $X_m \notin \mathcal{N}_k$ **then**
        insertNFA($q[C], X_1 \cdots X_m, \tilde{q}_\triangleright$)
      else % $X_m \in \mathcal{N}_k$
        insertNFA($q[C], X_1 \cdots X_{m-1}, q[X_m]$)
      end
      $\delta \leftarrow \delta \cup \{(\tilde{q}_\triangleleft, \varepsilon, q[A])\}$
    end
  else % isLeftLinear($\mathcal{G}_k$)
    for $\forall (C \rightarrow X_1 X_2 \cdots X_m) \in \mathcal{P}_k$ **do**
      if $X_1 \notin \mathcal{N}_k$ **then**
        insertNFA($\tilde{q}_\triangleright, X_1 \cdots X_m, q[C]$)
      else % $X_1 \in \mathcal{N}_k$
        insertNFA($q[X_1], X_2 \cdots X_{m-1}, q[C]$)
      end
      $\delta \leftarrow \delta \cup \{(q[A], \varepsilon, \tilde{q}_\triangleright)\}$
    end
  end
end

Figure 7: *Procedure* "insertNFA($q_\triangleleft, \alpha, q_\triangleright$)" *inserts a component of a NFS starting at state* $q_\triangleleft \in \mathcal{Q}$ *and ending at state* $q_\triangleright \in \mathcal{Q}$ *whose output corresponds to* $\alpha \in (\mathcal{T} \cup \mathcal{N})^*$.

that is not recursive. As as a consequence, there is exactly one transition leading into each state labeled by E in Figure 4(d), and that transition is always associated with the null string ε. The head and tail of this transition can be merged trivially into one state. Unlike the algorithm presented here, Nederhof's approach economizes on the number of states in the final NFA by treating such nonterminals as special cases.

6 Conclusion

Radar models based on phrase-structured grammars have potential applications in electronic warfare. In some cases, the most natural grammatical framework in which an ELINT analyst may describe a radar system is through a context-free grammar. Unfortunately, the complexity of signal processing algorithms for such grammars is much higher than that of the more straightforward regular grammars.

It is conjectured that many or most context-free grammars that describe radar models are non-self-embedding and therefore describe regular languages. The determination of whether or not a given context-free grammar $\mathcal{G} = (\mathcal{N}, \mathcal{T}, \mathcal{R}, S)$ is non-self-embedding consists of four steps:

1. Use the production rules $\mathcal{R}$ to construct the edge relation $\mathcal{E}$ of a directed graph $(\mathcal{N}, \mathcal{E})$ using (3).

2. Determine the strongly connected components (i.e. equivalence classes) of $(\mathcal{N}, \mathcal{E})$ using an algorithm such as the one due to Tarjan [15].

3. Decide, for each equivalence class $\mathcal{N}_i$, whether or not the associated grammar G_i is regular (i.e. either left-linear or right-linear).

4. Invoke Lemma 3 to determine whether $\mathcal{G}$ is non-self-embedding.

If a grammar is thus determined to be non-self-embedding then a non-deterministic finite automaton description of the language can be computed using the algorithm described in Section 5. The resulting automaton can, in turn, be converted into an equivalent regular grammar.

References

1. Hopcroft, J. E., Motwani, R., and Ullman, J. D. (2001). Introduction to Automata Theory, Languages and Computation, 2nd ed. Addison-Wesley.

2. Fu, K. S. (1974). Syntactic Methods in Pattern Recognition, New York, NY: Academic Press.

3. Dilkes, F. A. (2004). Linguistic Radar Modeling for Electronic Support: Pulse Processing for Stochastic Regular Grammars. In *Proceedings of 2005 Workshop on Defence Applications of Signal Processing*, Midway, Utah. (in press).

4. Lavoie, P. (2001). Hidden Markov Modeling for Radar Electronic Warfare. (DREO TM 2001-127). Defence R&D Canada – Ottawa. UNCLASSIFIED.

5. Nayar, S. and Smithers, D. (1995). Unpublished Report.

6. Visnevski, N., Krisnamurthy, V., Haykin, S., Currie, B., Dilkes, F., and Lavoie, P. (2003). Multi-Function Radar Emitter Modelling: A Stochastic Discrete Event System Approach. In *Proceedings of the 42nd IEEE Conference on Decision and Control (CDC)*, Maui, Hawaii, USA.

7. Rabiner, L. and Juang, B.-H. (1993). Fundamentals of Speech Recognition, New Jersey: Prentice Hall.

8. Lari, K. and Young, S. J. (1990). The Estimation of Stochastic Context-Free Grammars using the Inside-Outside Algorithm. *Computer Speech and Language*, **4**, 35–56.

9. Jelinek, F., Lafferty, J. D., and Mercer, R. L. (1992). Basic Methods of Probabilistic Context Free Grammars. In Laface, P. and DeMori, R., (Eds.), *Speech Recognition and Understanding*, pp. 345–360. Berlin: Springer.

10. Chomsky, N. (1956). Three Models for the Description of Language. *IRE Transactions on Information Theory*, **IT-2**(3), 113–124.

11. Chomsky, N. (1959). A Note on Phrase Structure grammars. *Information and Control*, **2**, 393–395.

12. Nederhof, M.-J. (2000). Regular Approximations of CFLs: A Grammatical View. In Bunt, H. and Nijholt, A., (Eds.), *Advances in Probabilistic and other Parsing Technologies*, Ch. 12, pp. 221–241. Kluwer Academic Publishers.

13. Anselmo, M., Giammarresi, D., and Varricchio, S. (2003). Finite Automata and Non-self-Embedding Grammars. In Champarnaud, J.-M. and Maurel, D., (Eds.), *Implementation and Application of Automata, 7th International Conference, CIAA 2002, Tours, France, July 3-5, 2002, Revised Papers*, Vol. 2608 of *Lecture Notes in Computer Science*, pp. 47–56. Springer.

14. Weisstein, E. W.. MathWorld (Online). From MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com/ (July 26, 2004).

15. Tarjan, R. (1972). Depth-First Search and Linear Graph Algorithms. *SIAM Journal of Computing*, **1**(2), 146–160.

DOCUMENT CONTROL DATA

(Security classification of title, body of abstract and indexing annotation must be entered when document is classified)

1. ORIGINATOR (the name and address of the organization preparing the document. Organizations for whom the document was prepared, e.g. Centre sponsoring a contractor's report, or tasking agency, are entered in section 8.)

Defence R&D Canada - Ottawa
3701 Carling Avenue, Ottawa ON, CANADA, K1A 0Z4

2. SECURITY CLASSIFICATION (overall security classification of the document including special warning terms if applicable).

UNCLASSIFIED

3. TITLE (the complete document title as indicated on the title page. Its classification should be indicated by the appropriate abbreviation (S,C,R or U) in parentheses after the title).

Non-Self-Embedding Context-Free Grammars for Electronic Warfare

4. AUTHORS (Last name, first name, middle initial. If military, show rank, e.g. Doe, Maj. John E.)

Dilkes, Fred A. ; Visnevski, Nikita

5. DATE OF PUBLICATION (month and year of publication of document)

October 2004

6a. NO. OF PAGES (total containing information. Include Annexes, Appendices, etc).

24

6b. NO. OF REFS (total cited in document)

15

7. DESCRIPTIVE NOTES (the category of the document, e.g. technical report, technical note or memorandum. If appropriate, enter the type of report, e.g. interim, progress, summary, annual or final. Give the inclusive dates when a specific reporting period is covered).

Technical Memorandum

8. SPONSORING ACTIVITY (the name of the department project office or laboratory sponsoring the research and development. Include address).

Defence R&D Canada - Ottawa
3701 Carling Avenue, Ottawa ON, CANADA, K1A 0Z4

9a. PROJECT OR GRANT NO. (if appropriate, the applicable research and development project or grant number under which the document was written. Specify whether project or grant).

13EL13

9b. CONTRACT NO. (if appropriate, the applicable number under which the document was written).

10a. ORIGINATOR'S DOCUMENT NUMBER (the official document number by which the document is identified by the originating activity. This number must be unique.)

DRDC Ottawa TM 2004-157

10b. OTHER DOCUMENT NOs. (Any other numbers which may be assigned this document either by the originator or by the sponsor.)

11. DOCUMENT AVAILABILITY (any limitations on further dissemination of the document, other than those imposed by security classification)

(X) Unlimited distribution
( ) Defence departments and defence contractors; further distribution only as approved
( ) Defence departments and Canadian defence contractors; further distribution only as approved
( ) Government departments and agencies; further distribution only as approved
( ) Defence departments; further distribution only as approved
( ) Other (please specify):

12. DOCUMENT ANNOUNCEMENT (any limitation to the bibliographic announcement of this document. This will normally correspond to the Document Availability (11). However, where further distribution beyond the audience specified in (11) is possible, a wider announcement audience may be selected).

Full unlimited announcement

13. ABSTRACT (a brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), (R), or (U). It is not necessary to include here abstracts in both official languages unless the text is bilingual).

It is possible to describe the behaviour of many multi-function radar systems in terms of context-free grammars. These grammars, although potentially natural devices for the description of the radar system, generally lead to greater computation complexity compared with simpler, regular grammars. Given a particular context-free grammar, it may or may not be possible to construct an equivalent regular grammar that describes the same behaviour. This memo is concerned with deciding whether or not such a grammar can be constructed, and presents and algorithm for generating the corresponding finite automaton.

14. KEYWORDS, DESCRIPTORS or IDENTIFIERS (technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be selected from a published thesaurus. e.g. Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus-identified. If it not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title).

Electronic Intelligence
Electronic Surveillance
Electronic Warfare
Pattern Recognition
Pulse Processing
Radar
Grammar
Non-self embedding grammar

Defence R&D Canada

Canada's leader in defence
and national security R&D

R & D pour la défense Canada

Chef de file au Canada en R & D
pour la défense et la sécurité nationale

www.drdc-rddc.gc.ca