DEMONSTRATIO MATHEMATICA
Vol. XXXVIII No 3 2005

Ishanu Chattopadhyay, Asok Ray

A COMPLEX MEASURE FOR LINEAR GRAMMARS

Abstract. The signed real measure of regular languages, introduced and validated in recent literature, has been the driving force for quantitative analysis and synthesis of discrete-event supervisory (DES) control systems dealing with finite state automata (equivalently, regular languages). However, this approach relies on memoryless state-based tools for supervisory control synthesis and may become inadequate if the transitions in the plant dynamics cannot be captured by finitely many states. From this perspective, the measure of regular languages needs to be extended to that of non-regular languages, such as Petri nets or other higher level languages in the Chomsky hierarchy. Measures for non-regular languages has not apparently been reported in open literature and is an open area of research. As a step toward achieving this goal, this paper introduces a complex measure of *linear context free grammars* (*LCFG*) that belong to the class of non-regular languages. The proposed complex measure becomes equivalent to the signed real measure, reported in recent literature, if the *LCFG* is degenerated to a regular grammar.

1. Introduction

Finite state automata (FSA) (equivalently, regular languages) have been widely used to model and synthesize supervisory control laws for discrete-event plants [4] because the task of discrete-event supervisory (DES) control synthesis becomes mathematically tractable and computationally efficient due to simplicity of regular languages. According to the paradigm of DES control, a finite-state automaton (e.g., the discrete-event model of a physical plant) is a language generator whose behavior is constrained by the supervisor to meet a given specification. The (controlled) sublanguage of the plant behavior could be different under different supervisors that satisfy their own respective specifications. Such a partially ordered set of sublanguages re-

1991 *Mathematics Subject Classification:* 93A05.
Key words and phrases: Language Measure; Discrete Event Supervisory Control; Linear Grammars.
This work has been supported in part by the U.S. Army Research Laboratory and the U.S. Army Research Office (ARO) under Grant No. DAAD19-01-1-0646; and NASA Glenn Research Center under Grant No. NNC04GA49G.

quires a quantitative measure for total ordering of their respective performance. To address this issue, a signed real measures of regular languages has been reported in literature [7] to provide a mathematical framework for quantitative comparison of controlled sublanguages of the unsupervised plant language. This measure provides a total ordering of the sublanguages of the unsupervised plant and formalizes a procedure for synthesis of DES controllers for finite state automaton plants, as an alternative to the procedure of Ramadge and Wonham [4]. Optimal control of finite state automata has been recently have been reported [5] based on the language measure and formalizes quantitative analysis and synthesis of DES control laws. The approach is state-based and the language measure parameters are identified from physical experiments or simulation on a deterministic finite state automaton (DFSA) model of the plant [8]. However, using memoryless state-based tools for supervisory control synthesis may suffer serious shortcomings if the details of transitions cannot be captured by finitely many states. This problem has been partially circumvented by Petri nets that can accommodate certain classes of non-regular languages [3] in the Chomsky hierarchy [2]. There is apparently no quantitative tool for supervisory control synthesis of Petri nets compared to what are available for finite state automata [5]. Hence, there is a need for developing measures of non-regular languages as a quantitative tool of supervisory control synthesis for discrete-event systems that cannot be represented by regular languages. Toward achieving this goal, the first step is to construct measure(s) of non-regular languages where the state-based approach [7] may not be applicable.

This paper first shows that the measure of a regular language proposed in [7] is equivalent to that of the regular grammar, without referring to states of the automaton. Then, it extends the signed real measure of regular languages to a complex measure for the class of non-regular languages [1], generated by the (deterministic) *linear context free grammar* (*LCFG*) that is a subclass of deterministic pushdown automata (DPDA) [2]. The signed real measure is extended to a complex measure over the real field, where the multiplication operation of complex numbers is different from the conventional one. In this case, the complex space over the real field degenerates to the union of a pair of one-dimensional real spaces instead of being isomorphic to the two-dimensional real space. The extended complex-valued language measure, formulated in this paper, is potentially applicable to analysis and synthesis of DES control laws where the plant model could be represented by an *LCFG*.

The paper is organized in five sections including the present one. Section 2 briefly introduces the basic concepts, notations and background materials for formal languages. Section 3 discusses the measure of regular grammars and shows its equivalence with that of regular languages. Section 4 extends the measure to linear grammars and the concept is elucidated with an example in Section 4. The paper is summarized and concluded in Section 5 with recommendations for future research.

2. Concepts and notations

This section introduces notations and background materials for formal languages [2] along with definitions of key concepts.

DEFINITION 2.1. A context free grammar (CFG) is a 4-tuple $\Gamma=(V,T,P,S)$, where V and T are mutually disjoint (i.e., $V \cap T = \emptyset$) finite sets of variables and terminals, respectively; and P is a finite set of productions by which strings are derived from the start symbol S. Each production in P is of the form $v \rightarrow \alpha$, where $v \in V$ and $\alpha \in (V \cup T)^*$.

REMARK 2.1. The language generated by a grammar Γ consists of all strings obtained from legal (i.e., permissible) productions beginning with the start symbol.

DEFINITION 2.2. A regular grammar is a CFG (V,T,P,S) where every production in P takes exactly one of the following two alternative pairs of forms (i.e., there are either right derivations or left derivations but not both):

$$(1) \qquad \begin{cases} v \rightarrow \alpha w \\ v \rightarrow \alpha \end{cases} \text{or} \quad \begin{cases} v \rightarrow w\alpha \\ v \rightarrow \alpha \end{cases}$$

where $v, w \in V$ and $\alpha \in T \cup \{\epsilon\}$; and ϵ is the empty string.

REMARK 2.2. The generated language for a deterministic finite state automaton ($DFSA$) is a regular language [2].

DEFINITION 2.3. A linear grammar is a CFG (V,T,P,S) where every production in P takes one of the following forms:

$$(2) \qquad v \rightarrow \alpha w; \quad v \rightarrow w\alpha; \quad v \rightarrow \alpha$$

where $v, w \in V$ and $\alpha \in T \cup \{\epsilon\}$.

REMARK 2.3. In view of Remark 2.1 and Definition 2.3, the set of production rules P in a linear grammar $\Gamma = (V,T,P,S)$ can be modified as $\tilde{\Gamma} = (\tilde{V},T,\tilde{P},S)$ by augmenting the set V of variables as $\tilde{V} \equiv V \cup A$ and by updating the set P of production rules by $\tilde{P}$. That is, $\varphi = (v \rightarrow \alpha) \in P$ is replaced by $\tilde{\varphi} = (v \rightarrow a\alpha) \in \tilde{P}$, where $v \in V$, $\alpha \in T \cup \{\epsilon\}$ and $a \in A$. This is analogous to the trim operation in regular languages [4].

REMARK 2.4. The modified grammar $\tilde{\Gamma} = (\tilde{V},T,\tilde{P},S)$ is a superset of the original grammar $\Gamma = (V,T,P,S)$ in the sense that it contains the generated language of $\Gamma = (V, T, P, S)$ and, in addition, has productions of the type $v \to aw$. The production $A \to \epsilon$ is added to P in each step of the modification; and $v \to \epsilon$ must exist $\forall v \in V$.

REMARK 2.5. Regular grammars have only right (or left) derivations with a single variable. In contrast, linear grammars include both right and left derivations with a single variable. This is precisely what allows the linear grammars to model a certain class of non-regular languages. For example, a production rule of the type $V \longrightarrow \sigma V_1$ is fundamentally different form a production $V \longrightarrow V_1\sigma$. If V_1 is subsequently replaced by a symbol σ_1, the order of generation of σ and σ_1 is maintained in the derived string in the first case and it is reversed in the second. Note that if $V \longrightarrow V_1\sigma$ is the first right linear production rule used in a particular derivation, then σ is necessarily the last terminal in the derived string.

A geometric approach is adopted in this paper to deal with the presence of both right and left derivations in an *LCFG*. The possible non-causality of derivations is handled by introducing the following notions: *imaginary transitions, generated path, path mapping function, and the event plane.*

Generation of a symbol through the right linear production is denoted by an imaginary transition as opposed to a symbol generated by a left linear production which is denoted by a real transition. An imaginary transition is labelled by the prefix i with the generated symbol. For example, $V \longrightarrow V_1\sigma$ implies $i\sigma$ has occurred while $V \longrightarrow \sigma V_1$ implies simply σ has occurred. The concept of real and imaginary transitions facilitates the notion of a generated path.

DEFINITION 2.4. A generated path $\lambda \in (\{\epsilon, i\} \times \Sigma)^\star$ is the sequential order of transitions (real or imaginary) in any particular derivation.

For example, if a derivation proceeds sequentially through the production rules $V_1 \longrightarrow V_2\sigma_1$, $V_2 \longrightarrow \sigma_2 V_3$, $V_3 \longrightarrow V_4\sigma_3$, $V_4 \longrightarrow \sigma_4 V_5$, then the generated path $\lambda \equiv i\sigma_1\sigma_2 i\sigma_3\sigma_4$.

DEFINITION 2.5. The set of all paths, generated by an *LCFG* Γ, is denoted as $\mathcal{P}_\Gamma \in 2^{(\{\epsilon,i\}\times\Sigma)^\star}$.

A given generated path λ corresponds to a particular derived string. However, in non-regular grammars, a single string may be derived through more than one path. That is, there exists a surjective mapping from the set of all generated paths by an *LCFG* to the set of all derived strings (i.e., the language generated by the grammar). In general, the map may not be injective. However, if the *LCFG* is regular, then this map becomes injective and hence bijective.

DEFINITION 2.6. The path mapping function $\wp : \mathcal{P}_\Gamma \longrightarrow L(\Gamma)$ is defined as follows: For any path $\lambda \in \mathcal{P}_\Gamma$, the corresponding derived string is obtained by concatenating all the symbols generated by real transitions followed by a concatenation of the ones generated by imaginary transitions in a reversed order.

For example, $\wp(i\sigma_1\sigma_2 i\sigma_3\sigma_4) \longmapsto \sigma_2\sigma_4\sigma_3\sigma_1$. In regular grammars, the absence of imaginary transitions implies that $\wp$ is the identity map.

An example of the right invariant relation is the well-known Nerode equivalence relation ($\mathcal{N}$) on a language L which is defined as follows [2]:

$$\forall\, x, y \in L,\ x\mathcal{N}y,\ \text{if}\ \forall\, u \in \Sigma^*,\ xu \in L \Leftrightarrow yu \in L. \tag{3}$$

A language L is regular if and only if there exists a Nerode equivalence relation of finite index [2]. Applying the notion of Nerode equivalence on $\mathcal{P}_\Gamma$, it follows that if two paths λ_1, λ_2 are generated through production rule sequences $\{P_{11}, P_{12}, \cdots, P_{1n}\}$ and $\{P_{21}, P_{22}, \cdots, P_{2m}\}$ such that the final variable on the righthand side of the derivation is identical (say V_k) in the two cases, then $\lambda_1\mathcal{N}\lambda_2$. This follows immediately from noting that if λ is a path initiating from V_k, then both $\lambda_1\lambda$ and $\lambda_2\lambda$ are elements of $\mathcal{P}_\Gamma$, and if $\lambda \in 2^{(\{\epsilon,i\}\times\Sigma)^*}$ cannot be generated from the variable V_k, then $\lambda_1\lambda$, $\lambda_2\lambda \notin \mathcal{P}_\Gamma$. This observation suggests that, at least in an *LCFG*, variables convey the same meaning as states in the context of regular languages.

In the sequel, the terms ***state*** and ***variable*** are used interchangeably as they convey similar meaning in the present context; the same applies to the terms ***terminals*** and ***events***. Note that the context-free nature of LCFG implies each variable can be rewritten by the specified production rules, irrespective of where the variable occurred. This is precisely the kind of Markov property that associates variables with states and terminals with events.

DEFINITION 2.7. The event mapping $\eta : \Sigma \to \mathbb{Z}$ is a function that maps the event alphabet into the set of integers. Let $\Sigma = \{\sigma_1, \cdots, \sigma_k, \cdots, \sigma_m\}$, then

$$\eta(\sigma_k) = k. \tag{4}$$

DEFINITION 2.8. The *event plane* can be viewed as the complex plane itself on which the trajectory of the discrete-event system is reconstructed as the strings are generated. The transitions $S \to \sigma_k v_i$ and $S \to v_i\sigma_k$ transfer the state located at the origin $(0,0)$, to $(\eta(\sigma_k), 0)$ and $(0, \eta(\sigma_k))$, respectively. Thus, there exists two possible directions in which the same event σ_k may cause transition from the same state v_i depending on whether the event is causing an imaginary transition or a real transition. Figure 1 illustrates the above concept of event plane in the formulation of LCFG measure. In

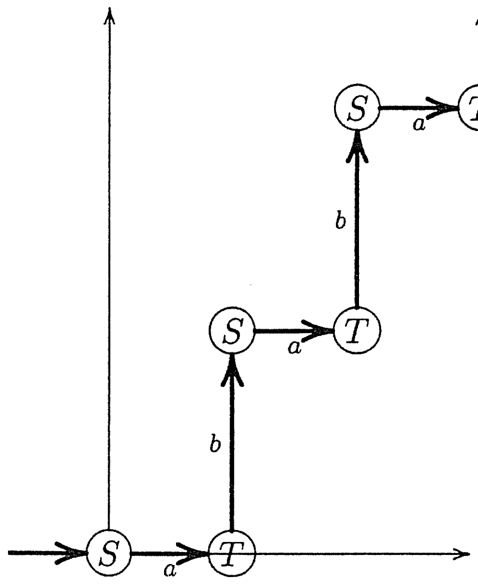

Fig. 1. Derivations on the event plane for $L = \{a^n b^n : n \in \mathbb{N}\}$ having the linear grammar $S \rightarrow aT; T \rightarrow Sb$

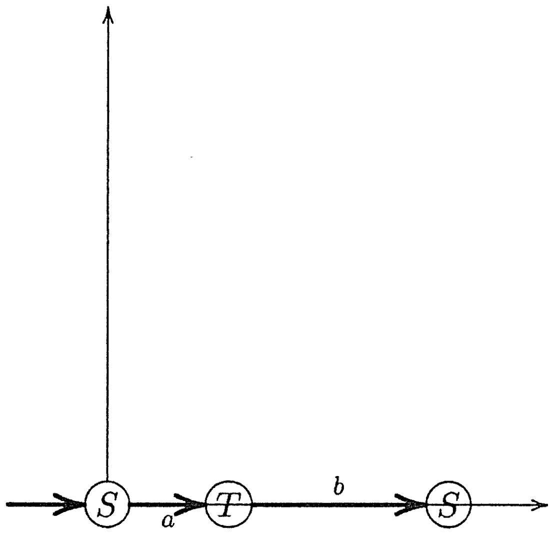

Fig. 2. Derivations on the event plane for $L = \{(ab)^\star\}$ having the linear grammar $S \rightarrow aT; T \rightarrow bS$

contrast, the linear grammar in Figure 1 degenerates to the regular grammar in Figure 2 and it shows how the event plane collapses to the real line. That is, the derivations always occur along the real axis of the event plane in Figure 2.

3. Measure of regular grammars

This section first introduces the concept of regular-grammar-based measures and then shows its equivalence to that of recently reported state-based measure [7]. In essence, the concept of the state-based language measure is reformulated in terms of regular grammars, followed by construction of the measure. While detailed proofs of the supporting theorems are given in [2], sketches of the proofs that are necessary for developing the underlying theory are presented here.

THEOREM 3.1. *If L is a regular language, then there is a regular grammar Γ such that either $L = L(\Gamma)$ or $L = L(\Gamma) \cup \{\epsilon\}$.*

Proof. Let $G = (Q, \Sigma, \delta, q_i, A)$ be an FSA. Let us construct the grammar Γ with $V = Q$ and $T = \Sigma$. The set of productions is constructed as follows:

$$\forall q_i, q_j \in Q,\ s_r \in \Sigma, \begin{cases} Add\ q_i \rightarrow s_r q_j\ if\ \delta(q_i, s_r) = q_j \\ Add\ q_i \rightarrow s_r\ if \delta(q_i, s_r) \in A. \end{cases}$$ ∎

THEOREM 3.2. *If Γ is a regular grammar, then $L(\Gamma)$ is a regular language.*

Proof. Let $\Gamma = (V, T, P, S)$ be a regular grammar. Let us construct a (possibly nondeterministic) finite state automaton G that exactly accepts the language $L(\Gamma)$. Specifically, let $G = (V \cup \{W\}, T, \delta, S, \{W\})$ where W is the only marked state and δ is defined as follows:

$$\delta(v_i, s_r) \equiv \delta_1(v_i, s_r) \cup \delta_2(v_i, s_r) \cup \delta_3(v_i, s_r)$$

$$\text{where } \begin{cases} \delta_1(v_i, s_r) = v_j, & \text{if } v_i \to s_r v_j \in P \\ \delta_2(v_i, s_r) = \{W\}, & \text{if } v_i \to s_r \in P \\ \delta_3(v_i, s_r) = \phi, & \text{otherwise.} \end{cases}$$ ∎

Remark 3.1. It follows from Theorem 3.2 and Theorem 3.1 that a language L is regular iff there is a regular grammar Γ such that either $L = L(\Gamma)$ or $L = L(\Gamma) \cup \{\epsilon\}$. Therefore, there is a regular grammar for every finite state automaton that exactly generates the language of the regular grammar and vice versa.

A. Formulation of regular grammar measures

This section follows the same construction procedure as in [7] because there exists a one-to-one-correspondence between the state set Q of an automaton and the variable set V of the corresponding regular grammar. The same holds true for the alphabet set Σ and the terminal T of the regular grammar. The notion of marked states as well as that of good and bad marked states translates naturally to this framework. The variable set V can be partitioned into sets of marked variables V_m and non-marked variables $V - V_m$ and the set V_m is further partitioned into good and bad marked variables as V_m^+ and V_m^- [7].

Definition 3.1. The language $L(\Gamma_i)$ generated by a context free grammar (CFG) Γ_i initialized at state $v_i \in V$ is defined as:

$$L(\Gamma_i) = \{s \in \Sigma^* \mid \text{there is a derivation of s from } \Gamma_i\}. \tag{5}$$

Definition 3.2. The marked language $L_m(\Gamma_i)$ generated by a CFG Γ_i, initialized at state $v_i \in V$, is defined as:
$L_m(\Gamma_i) = \{s \in \Sigma^* \mid$ there is a derivation of s from Γ_i which terminates on a marked variable$\}$.

Definition 3.3. For every $v_i, v_k \in V$, the set of all strings that, starting from v_i, terminate on v_k is defined as the language $L(v_i, v_k)$. That is, $L(v_i, v_k) = \{s \in \Sigma^* \mid$ there is a derivation of s from v_i that terminates on $v_k\}$.

Definition 3.4. The characteristic function $\chi : V \to [-1,1]$ is defined in exact analogy with the state based approach [7]:

$$\forall v_i \in V, \qquad \chi(v_i) \in \begin{cases} [-1,0), & v \in V_m{}^- \\ \{0\}, & v \notin V_m \\ (0,1], & v \in V_m{}^+ \end{cases}$$

and thus χ assigns a signed real weight to each of the sublanguages $L(v_i, v)$.

Similar to the measure of regular languages [7], the characteristic vector is denoted as: $\mathbf{X} = [\chi_1\ \chi_2\ \cdots\ \chi_n]^T$, where $\chi_j \equiv \chi(v_j)$, is called the $\mathbf{X}$-vector. The j-th element χ_j of $\mathbf{X}$-vector is the weight assigned to the corresponding terminal state v_j. Hence, the $\mathbf{X}$-vector is also called the state weighting vector in the sequel.

The marked language $L_m(\Gamma_i)$ consists of both good and bad event strings that, starting from the initial state v_i, lead to V_m^+ and V_m^-, respectively. Any event string belonging to the language $L^0(\Gamma_i) = L(\Gamma_i) - L_m(\Gamma_i)$ terminates on one of the non-marked states belonging to $V - V_m$; and L^0 does not contain any one of the good or bad strings. The regular languages $L(\Gamma_i)$ and $L_m(\Gamma_i)$ can be expressed as:

$$L(\Gamma_i) = \bigcup_{v_k \in V} L(v_i, v_k) = \bigcup_{k=1}^{n} L(v_i, v_k), \tag{6}$$

$$L_m(\Gamma_i) = \bigcup_{v_k \in Q_m} L(v_i, v_k) = L_m^+(\Gamma_i) \cup L_m^-(\Gamma_i) \tag{7}$$

where the sublanguage $L(v_i, v_k) \subseteq \Gamma_i$, having the initial state v_i, is uniquely labelled by the terminal state $v_k, k \in \mathcal{I}$ and $L(v_i, v_j) \cap L(v_i, v_k) = \emptyset\ \forall j \neq k$; and $L_m^+ \equiv \bigcup_{v \in V_m^+} L(v_i, v)$ and $L_m^- \equiv \bigcup_{v \in V_m^-} L(v_i, v)$ are good and bad sublanguages of $L_m(\Gamma_i)$, respectively. Then, $L^0(\Gamma_i) = \bigcup_{v \notin V_m} L(v_i, v)$ and $L(\Gamma_i) = L^0(\Gamma_i) \cup L_m^+(\Gamma_i) \cup L_m^-(\Gamma_i)$.

Now we construct a signed real measure $\mu : 2^{L(\Gamma_i)} \to \mathbf{R} \equiv (-\infty, +\infty)$ on the σ-algebra $K = 2^{L(\Gamma_i)}$. The construction is exactly equivalent to that for the state-based automata [7]. With the choice of this σ-algebra, every singleton set made of an event string $\omega \in L(\Gamma_i)$ is a measurable set, which qualifies itself to have a numerical quantity based on the above decomposition of $L(\Gamma_i)$ into L^0, L^+, and L^-, respectively called null, positive, and negative sublanguages. The event costs are defined below.

Definition 3.5. The event cost of the regular grammar Γ_i is defined as: $\tilde{\pi} : \Sigma^* \times V \to [0,1]$ such that $\forall v_i \in V$, $\forall \sigma_j \in \Sigma$, $\forall s \in \Sigma^*$,

(1) $\tilde{\pi}[\sigma_j, v_i] \equiv \tilde{\pi}_{ij} \in [0,1)$; $\sum_j \tilde{\pi}_{ij} < 1$;

(2) $\tilde{\pi}[\sigma_j, v_i] = 0$ if $\nexists v_i \to \sigma_j v_k \in P$, where P is the set of production rules; and $\tilde{\pi}[\epsilon, v_i] = 1$;
(3) $\tilde{\pi}[\sigma_j \omega, v_i] = \tilde{\pi}[\sigma_j, v_i]\ \tilde{\pi}[\omega, v_k]$, where $v_i \to \sigma_j v_k \in P$.

DEFINITION 3.6. The state transition cost of the regular grammar Γ_i is defined as: $\pi : V \times V \to [0,1)$ such that $\forall v_i, v_j \in V$, $\pi[v_i, v_j] = \sum_{\sigma \in \Sigma : \exists v_i \to \sigma v_j \in P} \tilde{\pi}[\sigma, v_i] \equiv \pi_{ij}$ and $\pi_{ij} = 0$ if $\{\sigma \in \Sigma : v_i \to \sigma v_j\} \cap P = \emptyset$. The $n \times n$ state transition cost matrix, called the $\mathbf{\Pi}$-matrix, is defined as:

$$\mathbf{\Pi} = \begin{bmatrix} \pi_{11} & \pi_{12} & \cdots & \pi_{1n} \\ \pi_{21} & \pi_{22} & \cdots & \pi_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \pi_{n1} & \pi_{n2} & \cdots & \pi_{nn} \end{bmatrix}.$$

DEFINITION 3.7. The signed real measure μ of every singleton string set $S = \{s\} \in 2^{L(\Gamma_i)}$ where $s \in L(v_i, v)$ is defined as $\mu(S) \equiv \tilde{\pi}(s, v_i)\chi(v)$. The signed real measure of the sublanguage $L(v_i, v) \subseteq L(\Gamma_i)$ is defined as

$$\mu(L(v_i, v)) = \Big(\sum_{s \in L(v_i, v)} \tilde{\pi}[s, v_i] \Big) \chi(v). \tag{8}$$

The signed real measure of the language of a regular grammar Γ_i initialized at a state $v_i \in V$, is defined as:

$$\mu_i \equiv \mu(L(\Gamma_i)) = \sum_{v \in \Gamma} \mu(L(v, v_i)). \tag{9}$$

The language measure vector, denoted as: $\boldsymbol{\mu} = [\mu_1\ \mu_2\ \cdots\ \mu_n]$, is called the $\boldsymbol{\mu}$-vector.

Based on the reasoning of the state based approach [7], it follows that:

$$\mu_i = \sum_j \pi_{ij} \mu_j + \chi_i. \tag{10}$$

In vector form, $\boldsymbol{\mu} = \mathbf{\Pi} \boldsymbol{\mu} + \mathbf{X}$ whose solution is given by:

$$\boldsymbol{\mu} = (\mathbf{I} - \mathbf{\Pi})^{-1} \mathbf{X}. \tag{11}$$

REMARK 3.2. The matrix $\mathbf{\Pi}$ is a contraction operator [5] and hence $(\boldsymbol{I} - \mathbf{\Pi})$ is invertible. So, the $\boldsymbol{\mu}$-vector in Eq. (10) is uniquely defined.

4. Language measure for linear grammars

This section extends the concept of language measure to linear grammars (V, T, P, S) that are a generalization of regular grammars [2].

A. Linear grammar measure construction

It follows from Definition 2.6 that, given a path ω in the language, the generated string is obtained by the path mapping $\wp(\omega)$. The approach of this paper is to construct a measure of the set of all such paths rather than the measure of strings. This is necessary since any attempt to apply the Myhill-Nerode theorem on a linear non-regular language results in an infinite number of equivalence classes (i.e., the Nerode equivalence relation is not of finite index). However, as we will argue shortly, the language of all paths ($\mathcal{P}_\Gamma$) is regular and hence a right invariant relation of finite index exists on $\mathcal{P}_\Gamma$. Then, it follows from the reasoning in [7] that a well-defined signed real measure exists on $\mathcal{P}_\Gamma$. In this section, we will construct a complex measure on $\mathcal{P}_\Gamma$ allowing one to differentiate between the real and imaginary transitions and is thus more intuitive in the case of linear non-regular grammars. Moreover, it is argued that the defined complex measure coincides for the $\mathcal{P}_\Gamma$ and $L(\Gamma)$.

LEMMA 4.1. *If there exists a complex measure ϑ on $\mathcal{P}_\Gamma$ with the σ-algebra $2^{\mathcal{P}_\Gamma}$, i.e., if $(\mathcal{P}_\Gamma, 2^{\mathcal{P}_\Gamma}, \vartheta)$ is a well-defined measure space then there exists a measure μ on $L(\Gamma)$ with the σ-algebra $2^{L(\Gamma)}$ such that*

$$\vartheta(\mathcal{P}_\Gamma) = \mu(L(\Gamma)) \tag{12}$$

Proof. Let $(\mathcal{P}_\Gamma, 2^{\mathcal{P}_\Gamma}, \vartheta)$ be a well-defined measure space. Let us define a map $\psi : L(\Gamma) \longrightarrow \mathbb{C}$ as follows:

$$\forall\, \omega \in L(\Gamma),\ \psi(\omega) = \sum_{\lambda \in \wp^{-1}(\omega)} \vartheta(\lambda). \tag{13}$$

Note that ψ is well-defined because $(\mathcal{P}_\Gamma, 2^{\mathcal{P}_\Gamma}, \vartheta)$ is a measure space and the mapping function $\wp$ is surjective (see Definition 2.6). Since ψ induces a complex measure [6] $\mu : 2^{L(\Gamma)} \longrightarrow C$ as:

$$\forall\, \Omega \subseteq L(\Gamma),\ \mu(\Omega) = \sum_{\omega \in \Omega} \psi(\omega) \tag{14}$$

on the space $L(\Gamma)$, it follows that

$$\mu(L(\Gamma)) = \sum_{\omega \in L(\Gamma)} \psi(\omega) \tag{15}$$

$$= \sum_{\lambda \in \wp^{-1}(L(\Gamma))} \vartheta(\lambda) \tag{16}$$

$$= \vartheta(\mathcal{P}_\Gamma). \tag{17}$$

■

REMARK 4.1. The construction of Lemma 4.1 is a natural way of inducing a measure on the quotient space of a measure space.

LEMMA 4.2. *For a linear grammar Γ, the path language $\mathcal{P}_\Gamma$ is regular.*

Proof. The statement follows from the fact that $\mathcal{P}_\Gamma$ is a language on the alphabet $\Sigma \sqcup i\Sigma$ and can be generated by a left-linear grammar with production rules of the type $S \longrightarrow \sigma V$ or $S \longrightarrow i\sigma V$. ∎

We continue with our explicit construction as follows:
Let $\Upsilon \equiv \{x : x = a + ib \text{ and } a \in [0,1], b \in [0,1]\}$, i.e., Υ is the closed unit square on the complex plane C. Let a binary operator $\star : C \times C \to C$ be defined as: $(a + ib) \star (c + id) = ac + ibd \ \forall a, b, c, d \in R$. The identity for the operator $\star$ is $1 + i$ since $\forall z \in C, z \star (1 + i) = z$ and if $z = a + ib$ with $a \neq 0$ and $b \neq 0$, then there exist a a unique $z^{-1} \in C$ such that $z^{-1} = \frac{1}{a} + i\frac{1}{b}$. That is, $z \star z^{-1} = (1 + i)$. The operator $\star$ can be extended to multi-dimensional cases by $\star : C^{n\times m} \times C^{m\times l} \to C^{n\times l}$ as follows:

If $\mathcal{A} \in C^{n\times m}$, $\mathcal{B} \in C^{m\times l}$, then $\mathcal{A} \star \mathcal{B} = \mathcal{C} \in C^{n\times l}$ in the sense that $c_{ij} = \Sigma_{k=1}^{n} a_{ik} \star b_{kj}$. Further, if $\mathcal{A} = A_r + iA_{im}$ and $\mathcal{B} = B_r + iB_{im}$ where the pairs (A_r, B_r) and (A_{im}, B_{im}) denote the real and imaginary parts of the matrices $\mathcal{A}$ and $\mathcal{B}$, respectively, it follows that $\mathcal{A} \star \mathcal{B} = A_r B_r + iA_{im}B_{im}$.

The identity for the above $\star$ operation is $(1 + i)I$ where I is the standard identity matrix of dimension $n \times n$. Let us denote the identity for the $\star$ operation by:

$$\mathcal{I} = (1 + i)I \tag{18}$$

where $\mathcal{A} \star \mathcal{I} = \mathcal{I} \star \mathcal{A} = \mathcal{A} \quad \forall \mathcal{A} \in C^{n\times n}$.

REMARK 4.2. The inverse of a matrix $\mathcal{A} \in C^{n\times n}$ under the $\star$ operation, if it exists, is given as:

$$\mathcal{A}^{-\mathbb{1}} = (\mathcal{A}_r + i\mathcal{A}_{im})^{-\mathbb{1}} = {\mathcal{A}_r}^{-1} + i{\mathcal{A}_{im}}^{-1} \tag{19}$$

that is different from the standard inverse $\mathcal{A}^{-1}$. Notice that both real $(\mathcal{A}_r)$ and imaginary $(\mathcal{A}_{im})$ parts of the matrix $\mathcal{A}$ must be individually invertible in the usual sense for existence of $\mathcal{A}^{-\mathbb{1}}$.

In view of the $\star$ operator, definitions of the language measure parameters, χ, $\tilde{\pi}$ and π, are generalized as follows:

DEFINITION 4.1. The characteristic function $\chi : V \to \Upsilon$ is defined as:

$$\forall v \in V, \ \ \chi(v) = k(1 + i) \tag{20}$$

$$\text{where} \begin{cases} k \in [-1, 0) \ if \ v \in {V_m}^- \\ k = 0 \qquad\ \ if \ v \notin V_m \\ k \in (0, 1] \ \ \ if \ v \in {V_m}^+. \end{cases} \tag{21}$$

The characteristic value χ assigns a complex weight to a language $L(v,u)$ that, starting at the variable v, ends at the variable u. A real weight, in the range of -1 to +1, is assigned to each state as it was done in the case of real grammars, and then this weight is made complex by multiplying (in the usual sense) with $1+i$.

DEFINITION 4.2. The event cost of the LCFG Γ is defined as a function $\tilde{\pi} : (\Sigma \sqcup i\Sigma)^* \times V \rightarrow \Upsilon$ such that $\forall v_k, v_\ell \in V$, $\forall \sigma_j \in \Sigma$,

(1) $\tilde{\pi}[\sigma_j, v_k] \equiv \mathbb{R}e(\tilde{\pi}_{kj}) \in [0,1)$; $\sum_j \mathbb{R}e(\tilde{\pi}_{kj}) < 1$;
(2) $\tilde{\pi}[i\sigma_j, v_k] \equiv \mathbb{I}m(\tilde{\pi}_{kj}) \in [0,1)$; $\sum_j \mathbb{I}m(\tilde{\pi}_{ij}) < 1$;
(3) $\tilde{\pi}[\sigma_j, v_k] = i$ if $\nexists v_k \rightarrow \sigma_j v_\ell \in P$;
(4) $\tilde{\pi}[i\sigma_j, v_k] = 1$ if $\nexists v_k \rightarrow v_\ell \sigma_j \in P$;
(5) $\tilde{\pi}[\epsilon, v_\ell] = 1 + i$;
(6) $\tilde{\pi}[\tau\omega, v_k] = \tilde{\pi}[\tau, v_k] \star \tilde{\pi}[\omega, v_\ell]$

where $\tau \in \Sigma \sqcup i\Sigma$; $\omega \in (\Sigma \sqcup i\Sigma)^*$; and $v_k \rightarrow \tau v_l$ if $\tau \equiv \sigma$ and $v_k \rightarrow v_\ell \tau$ if $\tau \equiv i\sigma$.

DEFINITION 4.3. The state transition cost of the LCFG Γ is defined as $\pi : V \times V \rightarrow \Upsilon$ such that $\forall v_k, v_j \in V$,

$$\pi[v_k, v_j] = \sum_{\substack{\sigma_\ell \in \Sigma: \\ \exists v_k \rightarrow \sigma_\ell v_j \in P}} \mathbb{R}e(\tilde{\pi}[\sigma_\ell, v_k]) + i \sum_{\substack{\sigma_\ell \in \Sigma: \\ \exists v_k \rightarrow v_j \sigma_\ell \in P}} \mathbb{I}m(\tilde{\pi}[\sigma_\ell, v_k]) \equiv \pi_{kj} \tag{22}$$

and $\pi_{kj} = 0$ if $\{\sigma \in \Sigma : v_k \rightarrow \sigma v_j \text{ or } \sigma \in \Sigma : v_k \rightarrow v_j \sigma\} \cap P = \emptyset$. The $n \times n$ complex-valued state transition $\mathbf{\Pi}$-matrix is defined as:

$$\mathbf{\Pi} = \begin{bmatrix} \pi_{11} & \pi_{12} & \cdots & \pi_{1n} \\ \pi_{21} & \pi_{22} & \cdots & \pi_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \pi_{n1} & \pi_{n2} & \cdots & \pi_{nn} \end{bmatrix}.$$

DEFINITION 4.4. Let Γ_k be a linear grammar, initialized at a state $v_k \in V$. The complex measure μ of every singleton path set $\Lambda = \{\lambda\} \in 2^{\mathcal{P}_{\Gamma_k}}$ is defined as: $\mu(\Lambda) \equiv \tilde{\pi}(\lambda, v_k) \star \chi(v)$. The complex measure μ of every singleton string set $\Omega = \{\omega\} \in 2^{L(\Gamma_k)}$ is defined as: $\mu(\Omega) \equiv \sum_{\lambda:\wp(\lambda)=\omega} \tilde{\pi}(\lambda, v_k) \star \chi(v)$. Then, the complex measure of the sublanguage $L(v_k, v) \subseteq L(\Gamma_k)$ of all strings terminated at the state $v \in V$ is defined as:

$$\mu(L(v_k, v)) = \Big(\sum_{\omega \in L(v_k,v)} \tilde{\pi}[\omega, v_k] \Big) \star \chi(v). \tag{23}$$

Complex measure of the language of the linear grammar Γ_k is defined as:

$$\mu_k \equiv \mu(L(\Gamma_k)) = \sum_{v \in \Gamma} \mu(L(v_k, v)). \tag{24}$$

The language measure vector, denoted as $\boldsymbol{\mu} = [\mu_1\ \mu_2\ \cdots\ \mu_n]^T$, is called the $\boldsymbol{\mu}$-vector.

B. Computation of the μ-vector

This subsection presents a procedure to formulate the complex measure of $\mathcal{P}_\Gamma$ and by Lemma 4.1 it coincides with the complex measure for $L(\Gamma_k)$. Now,

$$\mathcal{P}_{\Gamma_k} = (\cup_j \sigma_k{}^j \mathcal{P}_{\Gamma_j}) \cup_k \varepsilon_k$$

where the null event ε_k is defined as

$$\varepsilon_k = \begin{cases} \epsilon, & \text{if self loop at } v_i \\ \emptyset, & \text{otherwise.} \end{cases}$$

The above expression formalizes the fact that the set of paths from a state v_k is exactly equal to the union of the sets of paths obtained by looking at the first event and then considering all possible legal paths thereafter. Hence, if the first event is σ_ℓ and the current state changes to v_j, then the set of all paths thereafter is exactly equal to $\mathcal{P}_{\Gamma_j}$. The expression is structurally identical to that given for $DFSA$ in [7] with the understanding that the event $\sigma_\ell{}^j$ can be either real or imaginary.

Hence,

$$\begin{aligned} \mu(\mathcal{P}_{\Gamma_k}) &= \mu((\cup_j \sigma_k^j \mathcal{P}_{\Gamma_j}) \cup_k \varepsilon_k) = \mu(\cup_j \sigma_k^j \mathcal{P}_{\Gamma_j}) + \mu(\varepsilon_k) \\ &= \sum_j \mu(\sigma_k{}^j \mathcal{P}_{\Gamma_j}) + \chi(v_k) \\ &= \sum_j \pi_{kj} \star \mu(\mathcal{P}_{\Gamma_j}) + \chi(v_k). \end{aligned} \tag{25}$$

The first three steps in Eq. (25) follow from the fact that if the first symbol for two paths is different, then the paths cannot be identical. However, the generated strings may still be the same. The fourth step follows from property (6) of the $\tilde{\pi}$ function in Definition 4.2. The final step trivially follows from Definition 4.4 of the measure. In vector form, the complex measure μ is given by:

$$\begin{aligned} \boldsymbol{\mu}(L(\Gamma_k)) \equiv \boldsymbol{\mu}(\mathcal{P}_{\Gamma_k}) &= \mathbf{\Pi} \star \mu + \mathbf{X} = (\mathcal{I} - \mathbf{\Pi})^{-\mathbb{1}} \star \mathbf{X} \\ &= ((\mathbf{I} - \mathbb{R}e\mathbf{\Pi}) + i(\mathbf{I} - \mathbb{I}m\mathbf{\Pi}))^{-\mathbb{1}} \star \mathbf{X} \\ &= (\mathbf{I} - \mathbb{R}e\mathbf{\Pi})^{-1}\mathbb{R}e\mathbf{X} + i(\mathbf{I} - \mathbb{I}m\mathbf{\Pi})^{-1}\mathbb{I}m\mathbf{X} \end{aligned} \tag{26}$$

where $\mathbb{R}e$ and $\mathbb{I}m$ refer to the real and imaginary parts of the matrices, respectively; and existence of the matrix inverses is guaranteed by the following conditions:

$$\sum_j \mathbb{R}e(\tilde{\pi}_{kj}) < 1; \quad \sum_j \mathbb{I}m(\tilde{\pi}_{kj}) < 1 \ \ \forall k.$$

A simple example is presented below to illustrate how the complex μ is computed for an *LCFG.*

C. Example

Let a language L generate all strings of the type $\{a^n b^n : n \geq 0\}$ over the alphabet $\Sigma = \{a, b\}$. The non-regular language L can be generated by the grammar $\{v \rightarrow avb|\epsilon\}$ that can be rewritten as: $v_1 \rightarrow av_2$; and $v_2 \rightarrow v_1 b$. The resulting event costs (see Definition 4.2) are expressed in the matrix form as:

$$\widetilde{\mathbf{\Pi}} = \begin{bmatrix} p & 0 \\ 0 & iq \end{bmatrix}$$

where the parameters p and q can be identified from the experimental time series data of the system dynamics [8]. The state transition cost matrix (see Definition 4.3) is then obtained as:

$$\mathbb{R}e\mathbf{\Pi} = \begin{bmatrix} 0 & p \\ 0 & 0 \end{bmatrix} \text{ and } \mathbb{I}m\mathbf{\Pi} = \begin{bmatrix} 0 & 0 \\ q & 0 \end{bmatrix}.$$

Assigning characteristic values (i.e., weights) of the two states v_1 and v_2 to be $(1 + i)\chi_1$ and $((1 + i)\chi_2$ respectively, the complex measure vector of the language L is evaluated as:

$$\mu(L) = \left\{ \begin{matrix} (\chi_1 + p\chi_2) + \mathrm{i}\chi_1 \\ \chi_2 + \mathrm{i}(\chi_2 + q\chi_1) \end{matrix} \right\}.$$

5. Summary, conclusions, and future research

This paper introduces the notion of a quantitative measure of non-regular languages [1], generated by *linear context free grammars (LCFG)* that belong to the low end of Chomsky hierarchy [2]. It shows that the measure of regular languages, reported in [7], can be obtained by its generating regular grammar without referring to states of the automaton. Then, the paper extends the signed real measure to a complex measure for the class of non-regular languages, generated by *LCFG.* The extended language measure is potentially applicable to quantitative analysis and synthesis of discrete-event supervisory (DES) control systems, where the plant model of a complex dynamical system is not restricted to be a finite state machine.

The quantitative measure of linear context free grammars, introduced in this paper, is a first step toward development of such measures for higher level languages in the Chomsky hierarchy, such as Petri nets. Further research is needed for the next step in two main directions:

- Extension of the notion of the event plane to reveal geometric properties of formal languages on which little work is reported.
- Construction of closed-form expressions for quantitative measure of context free languages.

References

[1] I. Chattopadhyay, A. Ray, and X. Wang, *A Complex Measure of Non-Regular Languages for Discrete-Event Supervisory Control*, Proceedings of American Control Conference, Boston, MA (2004), pp. 5120–5125.

[2] J. E. Hopcroft, R. Motwani, and J. D. Ullman, *Introduction to Automata Theory, Languages, and Computation*, 2nd ed., Addison-Wesley, 2001.

[3] J. O. Moody and P. J. Antsaklis, *Supervisory Control of Discrete Event Systems Using Petri Nets*, Kluwer Academic, 1998.

[4] P. J. Ramadge and W. M. Wonham, *Supervisory control of a class of discrete event processes*, SIAM J. Control Optim. 25 (1987), no. 1, 206–230.

[5] A. Ray, J. Fu, and C. Lagoa, *Optimal supervisory control of finite state automata*, Internat. J. Control 77 (2004), no. 12, 1083–1100.

[6] W. Rudin, *Real and Complex Analysis*, 3rd ed., McGraw Hill, New York, 1988.

[7] A. Surana and A. Ray, *Measure of regular languages*, Demonstratio Math. 37 (2004), no. 2, 485–503.

[8] X. Wang, A. Ray and A. Khatkhate, *On-line Identification of Language Measure Parameters for Discrete Event Supervisory Control*, Applied Mathematical Modelling 29 (2005), no. 6, 597–613.

THE PENNSYLVANIA STATE UNIVERSITY
UNIVERSITY PARK, PA 16802, USA
E-mails: ixc128@psu.edu
axr2@psu.edu

Received May 24, 2004; revised version December 31, 2004.