DEMONSTRATIO MATHEMATICA
Vol. XXXVII No 2 2004

Amit Surana, Asok Ray

MEASURE OF REGULAR LANGUAGES

Abstract. This paper reviews and extends the recent work on signed real measure of regular languages within a unified framework. The language measure provides total ordering of partially ordered sets of sublanguages of a regular language to allow quantitative evaluation of the controlled behavior of deterministic finite state automata under different supervisors. The paper presents a procedure by which performance of different supervisors can be evaluated based on a common quantitative tool. Two algorithms are provided for computation of the language measure and their equivalence is established along with a physical interpretation from the probabilistic perspective.

1. Introduction

Deterministic finite-state automata (DFSA) can be represented by a regular languages [2] [5] and are usually capable of capturing the symbolic behavior of physical plants. The concept of discrete-event supervisory control, based on a DFSA plant model, was proposed in the seminal paper of Ramadge and Wonham [8]. The (controlled) sublanguages of the plant language could be different under different supervisors that satisfy their own respective specifications. Such a partially ordered set of sublanguages requires a quantitative measure for total ordering of their respective performance. To address this issue, Wang and Ray [12] formulated a signed measure of regular languages followed by Ray and Phoha [9] who constructed a vector space of formal languages and defined a metric based on the total variation measure of the language. This paper reviews these publications on language measure for discrete-event supervisory control within a unified framework and presents certain clarifications and extensions.

The signed real measure for a DFSA is constructed based on assignment of an event cost matrix and a characteristic vector. Two techniques for

Key words and phrases: Formal Languages, Performance Measure, Discrete Event Supervisory Control.
This work has been supported in part by Army Research Office (ARO) under Grant No. DAAD19-01-1-0646; and NASA Glenn Research Center under Grant No. NNC04GA49G.

language measure computation have been recently reported. While the first technique [12] leads to a system of linear equations whose (closed form) solution yields the language measure vector, the second technique [9] is a recursive procedure with finite iterations. A sufficient condition for finiteness of the signed measure has been established in both cases.

In order to induce total ordering on the measure of different sublanguages of a plant language under different supervisors, it is implicit that same strings in different sublanguages must be assigned the same measure. This is accomplished by a quantitative tool that assigns an event cost matrix and a characteristic vector for language measure computation. The clarifications and extensions presented in this paper are intended to enhance development of systematic analytical techniques for synthesis of discrete-event supervisory control systems. For example, Fu et al. [3][4] have proposed unconstrained optimal control of regular languages where state-based optimal control policies are synthesized by selectively disabling controllable events to maximize performance indices based on a measure of the controlled plant language.

The paper is organized in six sections including the present introductory section. Section 2 briefly describes the language measure, introduces the notations, and presents the procedure by which the performance of different supervisors can be compared based on a common quantitative tool. Section 3 discusses two alternative methods for computing language measure. Section 4 illustrates usage of the language measure for construction of metric spaces of formal languages and synthesis of optimal discrete-event supervisors. Section 5 addresses issues regarding physical interpretation of the event cost used in the language measure. The paper is summarized and concluded in Section 6 along with recommendations for future research.

2. Language measure concept

Following the terminology of Ramadge and Wonham [8], let $G_i \equiv \langle Q, \Sigma, \delta, q_i, Q_m \rangle$ be a trim (i.e., accessible and co-accessible) finite-state automaton model that represents the discrete-event dynamics of a physical plant where $Q = \{q_1, q_2, \cdots, q_n\}$ is the (finite) set of states with q_i being the initial state; $\Sigma = \{\sigma_1, \sigma_2, \cdots, \sigma_m\}$ is the (finite) alphabet of events; $\delta : Q \times \Sigma \rightarrow Q$ is the (possibly partial) function of state transitions; and $Q_m \equiv \{q_{m_1}, q_{m_2}, \cdots, q_{m_l}\} \subseteq Q$ is the (non-empty) set of marked states (known as accepted states in the computer science literature [2] [5]) $q_{m_k} = q_j$ for some $j \in \mathcal{I} \equiv \{1, \cdots, n\}$.

We have followed the notation of Ramadge and Wonham [8] in formulating the structure of DFSA G_i, which allows the state transition function δ

to be a partial function. This approach differs from the classical definition of DFSA [2] [5], where δ must be a total function. The rationale for making δ a partial function is to account for physical constraints of inhibiting certain events at selected states and also to accommodate modelling uncertainty as discussed later in Section 5. However, by adding a dump state to G_i, the partial function δ can be extended to a total function leading to the classical description of DFSA.

Let Σ^* be the Kleene closure of Σ, i.e., the set of all finite-length strings made of the events belonging to Σ as well as the empty string ε that is viewed as the identity of the monoid Σ^* under the operation of string concatenation, i.e., $\varepsilon s = s = s\varepsilon$. The extension $\delta^* : Q \times \Sigma^* \to Q$ is defined recursively in the usual sense [2] [5] [8].

DEFINITION 2.1. *The language $L(G_i)$ generated by a DFSA G initialized at the state $q_i \in Q$ is defined as:*

$$L(G_i) = \{s \in \Sigma^* \mid \delta^*(q_i, s) \in Q\}. \tag{2.1}$$

Since the state transition function δ is (possibly) a partial function, we allow $L(G_i) \subseteq \Sigma*$. If δ is a total function,then Definition 2.1 necessitates that the generated language $L(G_i) = \Sigma*$.

DEFINITION 2.2. *The language $L_m(G_i)$ marked by a DFSA G_i initialized at the state $q_i \in Q$ is defined as:*

$$L_m(G_i) = \{s \in \Sigma^* \mid \delta^*(q_i, s) \in Q_m\}. \tag{2.2}$$

DEFINITION 2.3. *For every $q_i, q_k \in Q$, let $L(q_i, q_k)$ denote the set of all strings that, starting from the state q_i, terminate at the state q_k, i.e.,*

$$L(q_i, q_k) = \{s \in \Sigma^* \mid \delta^*(q_i, s) = q_k\}. \tag{2.3}$$

In order to obtain a quantitative measure of the marked language, the set Q_m of marked states is partitioned into Q_m^+ and Q_m^-, i.e., $Q_m = Q_m^+ \cup Q_m^-$ and $Q_m^+ \cap Q_m^- = \emptyset$. The positive set Q_m^+ contains all *good* marked states that one would desire to reach, and the negative set Q_m^- contains all *bad* marked states that one would not want to terminate on, although it may not always be possible to completely avoid the *bad* states while attempting to reach the *good* states. From this perspective, each marked state is characterized by an assigned real value that is chosen based on the designer's perception of the state's impact on the system performance.

DEFINITION 2.4. *The characteristic function $\chi : Q \to$ [-1, 1] that assigns a signed real weight to a state-based sublanguage $L(q_i, q)$, having each of its* *strings terminating on the same state q, is defined as:*

$$\forall q \in Q, \quad \chi(q) \in \begin{cases} [-1,0), & q \in Q_m^- \\ \{0\}, & q \notin Q_m \\ (0,1], & q \in Q_m^+. \end{cases} \tag{2.4}$$

The state weighting vector, denoted by $\mathbf{X} = [\chi_1\ \chi_2\ \cdots\ \chi_n]^T$, is called the $\mathbf{X}$-vector. The j^{th} element χ_j of $\mathbf{X}$-vector is the weight assigned to the corresponding state q_j.

In general, the marked language $L_m(G_i)$ consists of both *good* and *bad* strings that, starting from the initial state q_i, respectively lead to Q_m^+ and Q_m^-. Any event string belonging to the language $L^0(G_i) \equiv L(G_i) - L_m(G_i)$ leads to one of the non-marked states belonging to $Q - Q_m$ and $L^0(G_i)$ does not contain any one of the *good* or *bad* strings. Partitioning Q_m into the positive set Q_m^+ and the negative set Q_m^- leads to partitioning of the marked language $L_m(G_i)$ into a positive language $L_m^+(G_i)$ and a negative language $L_m^-(G_i)$. Based on the equivalence classes defined in the Myhill-Nerode Theorem [2], the regular languages $L(G_i)$ and $L_m(G_i)$ can be expressed as:

$$L(G_i) = \cup_{q\in Q} L(q_i, q) = \cup_{k=1}^{n} L(q_i, q_k), \tag{2.5}$$

$$L_m(G_i) = \cup_{q\in Q_m} L(q_i, q) = L_m^+(G_i) \cup L_m^-(G_i) \tag{2.6}$$

where the sublanguage $L(q_i, q_k) \subseteq L(G_i)$ is uniquely labelled by the state $q_k, k \in \mathcal{I} \equiv \{1, \cdots, n\}$ and $L(q_i, q_k) \cap L(q_i, q_j) = \emptyset\ \forall k \neq j$; and $L_m^+(G_i) \equiv \cup_{q\in Q_m^+} L(q_i, q)$ and $L_m^-(G_i) \equiv \cup_{q\in Q_m^-} L(q_i, q)$ are *good* and *bad* sublanguages of $L_m(G_i)$, respectively. Then, $L^0(G_i) = \cup_{q\notin Q_m} L(q_i, q)$ and $L(G_i) = L^0(G_i) \cup L_m^+(G_i) \cup L_m^-(G_i)$.

Now we construct a signed real measure $\mu : 2^{L(G_i)} \to \mathbf{R} \equiv (-\infty, \infty)$ on the $\sigma-$algebra $M = 2^{L(G_i)}$. With this choice of $\sigma-$algebra, every singleton set made of an event string $s \in L(G_i)$ is a measurable set, which allows its quantitative evaluation based on the above state-based decomposition of $L(G_i)$ into null (i.e., L^0), positive (i.e., L^+), and negative (i.e., L^-) sublanguages. Conceptually similar to the conditional probability, each event is assigned a cost based on the state at which it is generated.

DEFINITION 2.5. *The event cost of the DFSA G_i is defined as a (possibly partial) function $\tilde{\pi} : \Sigma^* \times Q \to [0,1]$ such that $\forall q_j \in Q$, $\forall \sigma_k \in \Sigma$, $\forall s \in \Sigma^*$,*

(1) $\tilde{\pi}[\sigma_k, q_j] \equiv \tilde{\pi}_{jk} \in [0,1)$; $\sum_k \tilde{\pi}_{jk} < 1$;
(2) $\tilde{\pi}[\sigma, q_j] = 0$ *if* $\delta(q_j, \sigma)$ *is undefined;* $\tilde{\pi}[\epsilon, q_j] = 1$;
(3) $\tilde{\pi}[\sigma_k s, q_j] = \tilde{\pi}[\sigma_k, q_j]\ \tilde{\pi}[s, \delta(q_j, \sigma_k)]$.

A simple application of the induction principle to part(3) in above Definition shows $\tilde{\pi}[st, q_j] = \tilde{\pi}[s, q_j]\tilde{\pi}[t, \delta^*(q_j, s)]$. The condition $\sum_k \tilde{\pi}_{jk} < 1$ provides a sufficient condition for the existence of the real signed measure as discussed in Section 3. Additional comments on the physical interpretation of the event cost are provided in Section 5.

DEFINITION 2.6. *The state transition cost of a DFSA G is defined as a function* $\pi : Q \times Q \to [0,1)$ *such that* $\forall q_i, q_j \in Q$, $\pi[q_i, q_j] = \sum_{\{\sigma \in \Sigma : \delta(q_i,\sigma) = q_j\}} \tilde{\pi}[\sigma, q_i] \equiv \pi_{ij}$ *and* $\pi_{ij} = 0$ *if* $\{\sigma \in \Sigma : \delta(q_i, \sigma) = q_j\} = \emptyset$. *The* $n \times n$ *state transition cost matrix is defined as:*

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

and is referred to as the $\mathbf{\Pi}$*-matrix in the sequel.*

DEFINITION 2.7. *The signed real measure* μ *of every singleton string set* $\{s\} \in 2^{L(G_i)}$, *where* $s \in L(q_i, q)$, *is defined as:* $\mu(\{s\}) \equiv \tilde{\pi}(s, q_i)\chi(q)$ *implying that*

$$\forall s \in L(q_i, q), \qquad \mu(\{s\}) \begin{cases} = 0, & q \notin Q_m \\ > 0, & q \in Q_m^+ \\ < 0, & q \in Q_m^-. \end{cases} \tag{2.7}$$

Thus an event string terminating on a *good* (*bad*) marked state has a *positive* (*negative*) measure and one terminating on a non-marked state has *zero* measure. It follows from Definition 2.7 that the signed measure of the sublanguage $L(q_i, q) \subseteq L(G_i)$ of all events, starting at q_i and terminating at q, is:

$$\mu(L(q_i, q)) = \Big(\sum_{s \in L(q_i, q)} \tilde{\pi}[s, q_i] \Big) \chi(q). \tag{2.8}$$

DEFINITION 2.8. *Given a DFSA* $G_i \equiv \langle Q, \Sigma, \delta, q_i, Q_m \rangle$ *the cost* ν *of a sublanguage* $K \subseteq L(G_i)$ *is defined as the sum of the event cost* $\tilde{\pi}$ *of individual strings belonging to* K*:*

$$\nu(K) = \sum_{s \in K} \tilde{\pi}[s, q_i]. \tag{2.9}$$

DEFINITION 2.9. *The signed real measure of the language of a DFSA* G_i *initialized at a state* $q_i \in Q$, *is defined as:*

$$\mu_i \equiv \mu(L(G_i)) = \sum_{q \in Q} \mu(L(q_i, q)). \tag{2.10}$$

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

It follows immediately from above definition that $\mu(L(q_i, q)) = \nu(L(q_i, q))\chi(q)$. It has been proved in [9] [12] that under the condition $\sum_k \tilde{\pi}_{jk} < 1$ of Definition 2.5, the signed real language measure μ converges. Further the total variation measure $|\mu|$ of μ has also been shown to be finite [9].

In the above setting, the role of the language measure in DES control synthesis is explained below:

A discrete-event non-marking supervisor S restricts the marked behavior of an uncontrolled (i.e., unsupervised) plant G_i such that $L_m(S/G_i) \subseteq L_m(G_i)$. The uncontrolled marked language $L_m(G_i)$ consists of *good* strings leading to Q_m^+ and *bad* strings leading to Q_m^-. A controlled language $L_m(S/G_i)$ based on a given specification of the supervisor S may disable some of the *bad* strings and keep some of the *good* strings enabled. Different supervisors $S_j : j \in \{1, 2, \ldots, n_s\}$ for a DFSA G_i achieve this goal in different ways and generate a partially ordered set of controlled sublanguages $\{L_m(S_j/G_i) : j \in \{1, 2, \ldots, n_s\}\}$. The real signed measure μ provides a precise quantitative comparison of the controlled plant behavior under different supervisors because the set $\{\mu(L_m(S_j/G_i)) : j \in \{1, 2, \ldots, n_s\}\}$ is totally ordered.

In order to realize the above goal, the performance of different supervisors has to be evaluated based on a common quantitative tool. Let $G \equiv \langle Q_1, \Sigma, \delta_1, q_{11}, Q_{m1}\rangle$ denote the uncontrolled plant and $S \equiv \langle Q_2, \Sigma, \delta_2, q_{21}, Q_{m2}\rangle$ denote the supervisor with corresponding marked language $L_m(G)$ and $L_m(S)$, respectively. Then $L_m(G)$ denotes the uncontrolled plant language, $L_m(S)$ is language of control specification and $L_m(G) \cap L_m(S)$ is the controlled sublanguage under the supervisor S. Let $C \equiv \langle Q, \Sigma, \delta, q_1, Q_m\rangle$ where, $Q = Q_1 \times Q_2$, $q_1 = (q_{11}, q_{21})$, $Q_m = \{(p, q) | p \in Q_{m1}$ and $q \in Q_{m2}\}$ and the transition function δ is defined by the formula

$$\delta((p, q), \sigma) = (\delta_1(p, \sigma), \delta_2(q, \sigma)), \tag{2.11}$$

$(\forall p \in Q_1,\quad q \in Q_2$ and $\sigma \in \Sigma)$. Then, based on the established results from automata theory [5] and supervisory control [8], we conclude that C accepts the language $L_m(G) \cap L_m(S)$.

It follows from the above discussion that the extension δ^* satisfies: $\forall s \in \Sigma^*$

$$\delta^*((p, q), s) = (\delta_1^*(p, s), \delta_2^*(q, s)). \tag{2.12}$$

Also $L(G)$ is partitioned by $L(q_{11}, q_{1j})$ $1 \le j \le n_1$ where $|Q_1| = n_1$. With the above construction each of $L(q_{11}, q_{1j})$ is further partitioned by $L(q_{11}, q_{1j}) \cap L(q_{21}, q_{2k})$, $1 \le k \le n_2$ where $|Q_2| = n_2$. Thus for any $q_{1j} \in Q_{m1}$ the set of strings which is retained in $L_m(G) \cap L_m(S)$ is given by $L(q_{11}, q_{1j}) \cap$

$(\cup_{q_{2k}\in Q_{m2}} L(q_{21}, q_{2k}))$. A supervisor which retains maximum possible strings corresponding to $q_{1j} \in Q_{m1}^{+}$ while discarding as many strings as possible corresponding to $q_{1j} \in Q_{m1}^{-}$ would give a higher measure and hence a better performance. The construction above shows immediately how the event cost and characteristic function assigned to the uncontrolled plant can be used as a quantitative tool with which the performance of different supervisors can be evaluated and compared. The following procedure indicates how this can be accomplished explicitly:

DEFINITION 2.10. *Let G, S and C be defined as above. Let G represent the unsupervised plant and $\tilde{\pi}_1$ be the event cost and χ_1 be the characteristic function. Then for the DFSA C which represents the controlled sublanguage, $\tilde{\pi}$ is defined as:*

$$\tilde{\pi}[\sigma, (q_{1i}, q_{2j})] = \tilde{\pi}_1[\sigma, q_{1i}] \tag{2.13}$$

$\forall \sigma \in \Sigma$ and $\forall i, j$ s.t. $1 \leq i \leq n_1$, $1 \leq j \leq n_2$.

The χ-vector is defined as:

$$\chi((q_{1i}, q_{2j})) = \chi_1(q_{1i})\mathcal{I}(q_{2j}) \tag{2.14}$$

where $\mathcal{I}(\cdot)$ is the indicator function defined as:

$$\mathcal{I}(q) = \begin{cases} 1 & q \in Q_{m2} \\ 0 & q \notin Q_{m2} \end{cases}. \tag{2.15}$$

Let $s \in L((q_{11}, q_{21}), (q_{1i}, q_{2j}))$. Since $q_{1i} = \delta^*(q_{11}, s)$, by Definition 2.7, $\mu(\{s\}) = \tilde{\pi}_1[s, q_{11}]\chi_1(q_{1i})$ for the unsupervised (i.e., open loop) plant. For the supervised (i.e., closed loop) plant, we have $\mu(\{s\}) = \tilde{\pi}[s, (q_{11}, q_{21})]\chi((q_{1i}, q_{2j})) = \tilde{\pi}_1[s, q_{11}]\chi_1(q_{1i})\mathcal{I}(q_{2j})$ by Equations 2.13 and 2.14. In other words, if no event in the string s is disabled by the supervisor, then $\mu(\{s\})$ remains unchanged; otherwise, $\mu(\{s\}) = 0$. Thus, Definition 2.10 guarantees that same strings in different controlled sublanguages of a plant language $L(G_i)$ are assigned the same measure. Hence, the performance of different supervisors can be compared with a common quantitative tool.

Finally to conclude, it should be noted that while the domain (i.e., $2^{L(G_i)}$) of the language measure μ is partially ordered, its range which is a subset of $\mathbf{R}$ becomes totally ordered. The set $L(G_i)$ with the σ-algebra, $2^{L(G_i)}$, forms a measurable space. In principle, any measure μ can be defined on this measurable space to form a measure space (i.e., the triple $\langle L(G_i), 2^{L(G_i)}, \mu\rangle$). The choice of the signed language measure as given by Definitions 2.7 and 2.9, has been the motivated by the fact that it may serve as a performance measure and hence should have a physical significance in the DES controller synthesis. Moreover, defining the measure in this way also leads to simple computational procedures as discussed in the next section and further elaborated later in Section 5.

3. Language measure computation

Various methods of obtaining regular expressions for DFSAs are reported in Hopcroft [2], and Martin [5] and Drobot [1]. While computing the measure of a given DFSA, the same event may have different significance when emanating from different states. This requires assigning (possibly) different costs to the same event defined on different states. Therefore, it is necessary to obtain a regular expression which explicitly yields the state-based event sequences. In order to compute the language measure we transform these procedures of evaluating regular expression from symbolic equations to algebraic ones. We present the following two methods [12], [9] for language measure computation.

3.1. Method I: Closed form solution[12]. This section presents a closed-form method to compute the language measure via inversion of a square operator.

DEFINITION 3.1. *Let* $L_i \equiv L_m(G_i), i \in \{1, \cdots, n\}$, *denote the regular expression representing the marked language of an n-state DFSA* $G_i = (Q, \Sigma, \delta, q_i, Q_m)$ *where* q_i *is the initial state.*

DEFINITION 3.2. *Let* σ_j^k *denote the set of event(s)* $\sigma \in \Sigma$ *that is defined on the state* q_j *and leads to the state* $q_k \in Q$, *where* $j, k \in \{1, \cdots, n\}$, *i.e.,* $\delta(q_j, \sigma) = q_k, \forall \sigma \in \sigma_j^k \subseteq \Sigma$.

Then, given a DFSA $G_i = (Q, \Sigma, \delta, q_i, Q_m)$ with $|Q| = n$, the procedure to obtain the system equation by a set of regular expressions L_i of the marked language $L_m(G_i), i \in \{1, \cdots, n\}$, as follows:

$$\forall q_i \in Q, \qquad L_i = \sum_j R_{i,j} + \mathcal{E}_i, \qquad i \in \{1, \cdots, n\} \tag{3.1}$$

where $\forall i$, $R_{i,j}$ is defined as:

1. If $\exists\ \sigma \in \Sigma$, such that $\delta(q_i, \sigma) = q_j \in Q, j \in \{1, \cdots, n\}$, then $R_{i,j} = \sigma_i^j L_j$, otherwise, $R_{i,j} = \emptyset$.
2. If $q_i \in Q_m$, $\mathcal{E}_i = \epsilon$, otherwise, $\mathcal{E}_i = \emptyset$.

The set of symbolic equations may be written as:

$$L_i = \sum_j \sigma_i^j L_j + \mathcal{E}_i. \tag{3.2}$$

The above system of symbolic equations can be solved using a result given below, which is illustrated through an example.

LEMMA 3.1. *Let u, v be two known regular expressions and r be an unknown regular expression that satisfies the following algebraic identity:*

$$r = ur + v. \tag{3.3}$$

Then, the following relations are true:

(1) $r = u^*v$ is a solution to equation 3.3
(2) If $\epsilon \notin u$, then $r = u^*v$ is the unique solution to equation 3.3.

The proof of Lemma 3.1 is given in [1] [9].

EXAMPLE 3.1. Let $\Sigma = \{a, b\}$, $Q = \{1, 2, 3\}$, the initial state is 1 the sole marked state is 2 in Figure 3.1. Let the set of linear algebraic equations representing the transitions at each state of the DFSA be:

$$\begin{cases} L_1 = a_1^1 L_1 + b_1^2 L_2 \\ L_2 = a_2^1 L_1 + b_2^3 L_3 + \epsilon \\ L_3 = a_3^1 L_1 + b_3^2 L_2 \end{cases} \tag{3.4}$$

where the 'forcing' term ϵ is introduced on the right side of the i-th equation whenever $q_i \in Q_m, i \in \mathcal{I}$. By application of Lemma 3.1, the regular expression for the marked language $L_m(G_1)$ is:

$$L_m(G_1) \equiv L_1 = (a_1^1)^* b_1^2 (a_2^1 (a_1^1)^* b_1^2 + b_2^3 a_3^1 (a_1^1)^* b_1^2 + b_2^3 b_3^2)^*$$

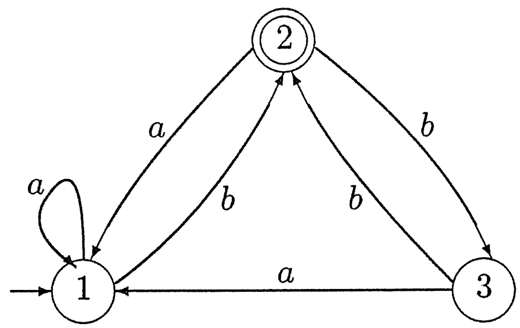

Fig. 1. Finite State Machine for Example 3.1

However instead of obtaining regular expressions, we can compute language measure directly by transforming this set of equations into a system of linear equations. This is based on the following result.

THEOREM 3.1. *The language measure of the symbolic equations 3.2 is given by*

$$\mu_i = \sum_j \pi_{ij} \mu_j + \chi_i. \tag{3.5}$$

Proof. Following Equation 3.1 and Definition 2.4:

$$\forall i \in \mathcal{I}, \qquad \mu(\mathcal{E}_i) = \begin{cases} \chi_i & \text{if } \mathcal{E}_i = \epsilon \\ 0 & \text{otherwise.} \end{cases} \tag{3.6}$$

Therefore, each element of the vector $\mathbf{X} = [\chi_1 \ \chi_2 \ \cdots \ \chi_n]^T$ is the forcing function in Equations 3.2 and 3.5. Starting from the state q_i, the measure of the language $L_i \equiv L_m(G_i)$ (see Definition 3.1)

$$\begin{aligned} \mu_i = \mu(L_i) &= \mu\Big(\sum_j \sigma_i^j L_j + \mathcal{E}_i\Big) \\ &= \mu\Big(\sum_j \sigma_i^j L_j\Big) + \mu(\mathcal{E}_i) \\ &= \sum_j \mu(\sigma_i^j L_j) + \mu(\mathcal{E}_i) \\ &= \sum_j \mu(\sigma_i^j)\mu(L_j) + \mu(\mathcal{E}_i) \\ &= \sum_j \pi(\sigma_i^j)\mu(L_j) + \mu(\mathcal{E}_i) \\ &= \sum_j \pi_{ij}\mu(L_j) + \mu(\mathcal{E}_i) \\ &= \sum_j \pi_{ij}\mu_j + \chi_i. \end{aligned}$$

The third equality in the above derivation follows from the fact that $\mathcal{E}_i \cap \sigma_i^j L_j = \emptyset$. It is also true that

$$\forall j \neq k, \qquad \sigma_i^j L_j \cap \sigma_i^k L_k = \emptyset \tag{3.7}$$

since each string in $\sigma_i^j L_j$ starts with an event in σ_i^j while each string in $\sigma_i^k L_k$ starts from an event in σ_i^k and $\sigma_i^j \bigcap \sigma_i^k = \emptyset$, as G_i is a DFSA. This justifies the fourth equality. $\mu(\sigma_i^j L_j) = \mu(\sigma_i^j)\mu(L_j)$ follows from Definition 2.8 and the fact that $\mu(L(q_i, q)) = \nu(L(q_i, q))\chi(q)$. Therefore, by Definitions 2.6 and 3.2, $\mu(\sigma_i^j L_j) = \pi[q_i, q_j] \ \mu(L_j) = \pi_{ij} \ \mu(L_j)$. ∎

In vector notation, Equation 3.5 in Theorem 3.1 is expressed as: $\boldsymbol{\mu} = \mathbf{\Pi}\boldsymbol{\mu} + \mathbf{X}$ whose solution is given by:

$$\boldsymbol{\mu} = (\mathbf{I} - \mathbf{\Pi})^{-1}\mathbf{X} \tag{3.8}$$

provided that the matrix $\mathbf{I} - \mathbf{\Pi}$ is invertible. This will also guarantees the existence of $\boldsymbol{\mu}$. We have the following important result.

THEOREM 3.2. *Given DFSAs* $G_i \equiv \langle Q, \Sigma, \delta, q_i, Q_m \rangle$, $1 \leq i \leq n$ *with the state transition cost matrix* $\mathbf{\Pi}$. *Then the matrix* $(\mathbf{I} - \mathbf{\Pi})$ *is an invertible bounded linear operator and* $\boldsymbol{\mu} \in \mathbf{R}^n$.

Proof. It follows from Definitions 2.5 and 2.6 that the induced norm $||\mathbf{\Pi}||_\infty \equiv max_i \sum_j \pi_{ij} = 1 - \theta$ where $\theta \in (0, 1)$. Then $(\mathbf{I} - \mathbf{\Pi})$ is invertible and is a bounded linear operator with the norm $||\mathbf{I} - \mathbf{\Pi}||_\infty \leq \theta^{-1}$ [6]. It then follows immediately from Equation 3.8 that $\boldsymbol{\mu} \in \mathbf{R}^n$ ■

COROLLARY 3.1. *The language measure vector* μ *is bounded as:* $||\mu||_\infty \leq \theta^{-1}$ *where* $\theta = (1 - ||\mathbf{\Pi}||_\infty)$.

Proof. The proof follows by applying the norm inequality property and Theorem 3.2 to Equation 3.8 and the fact that $||\mathbf{X}||_\infty \leq 1$ by Definition 2.4. ■

Definitions 2.5 and 2.6 provide a sufficient condition for the language measure μ of the DFSA G to be finite. Alternatively, necessary and sufficient conditions for convergence of μ which are based on certain properties of nonnegative matrices are given in [7]. A closed-form algorithm to compute a language measure based on the above procedure can be summarized as follows:

(1) For a given $G_i \equiv \langle Q, \Sigma, \delta, q_i, Q_m \rangle$, obtain the characteristic vector $\mathbf{X}$ and the event cost function $\tilde{\pi}$ (Definition 2.5).
(2) Generate the $\mathbf{\Pi}$ matrix (Definition 2.6).
(3) Compute the language measure vector $\boldsymbol{\mu} \longleftarrow (\mathbf{I} - \mathbf{\Pi})^{-1}\mathbf{X}$ using Gaussian elimination.
(4) μ_i, the ith element of $\boldsymbol{\mu}$-vector is the measure of the marked language of the DFSA G_i.

The j-th element of the i-th row of the $(\mathbf{I} - \mathbf{\Pi})^{-1}$ matrix, denoted as ν_i^j, is the language measure of the DFSA with the same state transition function δ as G_i and having the following properties: (i) the initial state is q_i; (ii) q_j is the only marked state; and (iii) the χ-value of q_j is equal to 1. Thus, $\mu_i \equiv \mu(L_m(G_i))$ is given by $\mu_i = \sum_j \nu_i^j \chi_j$. Numerical evaluation of the language measure of G_i requires Gaussian elimination of the single variable μ_i involving the real square matrix $(\mathbf{I} - \mathbf{\Pi})$. As such the computational complexity of the language measure algorithm is polynomial in the number of states.

3.2. Method II: Recursive solution [9]. This section presents a second method to compute the language measure using a recursive procedure based on concept of Kleene's theorem [5] which shows that a language accepted by a DFSA is regular. It also yields an algorithm to recursively construct the regular expression of its marked language instead of the the closed form solution in Method I.

DEFINITION 3.3. *Given q_i, $q_k \in Q$, a non-empty string p of events (i.e. $p \neq \epsilon$) starting from q_i and terminating at q_k is called a path. A path p from q_i to q_k is said to pass through q_j if $\exists\ s \neq \epsilon$ and $t \neq \epsilon$ such that $p = st$; $\delta^*(q_i, s) = q_j$ and $\delta^*(q_j, t) = q_k$.*

DEFINITION 3.4. *A path language p_{ik}^j is defined to be the set of all paths from q_i to q_k, which do not pass through any state q_r for $r > j$. The path language p_{ik} is defined to be the set of all paths from q_i to q_k. Thus, the language $L(q_i, q_k)$ is obtained in terms of the path language p_{ik} as:*

$$L(q_i, q_k) = \begin{cases} p_{ii} \cup \{\epsilon\} & \text{if} \quad k = i \\ p_{ik} & \text{if} \quad k \neq i \end{cases}$$

$$\Rightarrow \nu(L(q_i, q_k)) = \begin{cases} \nu(p_{ii}) + 1 & \text{if} \quad k = i \\ \nu(p_{ik}) & \text{if} \quad k \neq i \end{cases}.$$

Every path language p_{ik}^j is a regular language and subset of $L(G_i)$. As shown in [9], following recursive relation holds for $0 \leq j \leq n-1$:

THEOREM 3.3. *Given a $G_i \equiv \langle Q, \Sigma, \delta, q_i, Q_m \rangle$, the following recursive relation holds for $1 \leq j \leq n-1$*

$$p_{lk}^0 = \{\sigma \in \Sigma : \delta(q_l, \sigma) = q_k\}, \tag{3.9}$$

$$p_{lk}^{j+1} = p_{lk}^j \cup p_{l,j+1}^j (p_{j+1,j+1}^j)^* p_{j+1,k}^j. \tag{3.10}$$

Proof. Since the states are numbered form 1 to n in increasing order, $p_{lk}^0 = \{\sigma \in \Sigma : \delta(q_l, \sigma) = q_k\}$ follows directly form the state transition map $\delta : Q \times \Sigma \rightarrow Q$ and Definition 3.4. Given $p_{lk}^j \subseteq p_{lk}^{j+1}$, let us consider the set $p_{lk}^{j+1} - p_{lk}^j$ in which each string passes through q_{j+1} in the path from q_l to q_k and no string must pass through q_m for $m > (j+1)$. Then, it follows that $p_{lk}^{j+1} - p_{lk}^j = p_{l,j+1}^j p_{j+1,k}^{j+1}$ where $p_{j+1,k}^{j+1}$ can be expanded as: $p_{j+1,k}^{j+1} = (p_{j+1,j+1}^j p_{j+1,k}^{j+1}) \cup p_{j+1,k}^j$ that has a unique solution: by Lemma 3.1 because $\epsilon \notin p_{j+1,j+1}^j$ based on Definition 3.4. Therefore,

$$p_{lk}^{j+1} = p_{lk}^j \cup p_{l,j+1}^j (p_{j+1,j+1}^j)^* p_{j+1,k}^j. \qquad \blacksquare$$

Based on three lemmas proved below, the above relations can be transformed into an algebraic equation conceptually similar to Theorem 3.1 of Method I. Along with the procedure to compute the language measure it is established that $\sum_{j=1}^n \pi_{ij} < 1, \forall i$ is sufficient for finiteness of μ.

LEMMA 3.2. $\nu((p_{kk}^0)^*(\cup_{j \neq k} p_{kj}^0)) \in [0, 1)$.

Proof. Following Definition 2.5, $\nu(p^0_{kk}) \in [0,1)$. Therefore by convergence of geometric series,

$$\nu((p^0_{kk})^*(\cup_{j\neq k} p^0_{kj})) = \frac{\sum_{j\neq k}\nu(p^0_{kj})}{1-\nu(p^0_{kk})} \in [0,1).$$

because $\sum_j \nu(p^0_{kj}) < 1 \Rightarrow \sum_{j\neq k}\nu(p^0_{kj}) < 1-\nu(p^0_{kk})$. ∎

LEMMA 3.3. $\nu(p^j_{j+1,j+1}) \in [0,1)$.

Proof. The path $p^j_{j+1,j+1}$ may contain at most j loops, one around each of the states $q_1, q_2\cdots, q_j$. If the path $p^j_{j+1,j+1}$ does not contain any loop, then $\nu(p^j_{j+1,j+1}) \in [0,1)$ because $\forall s \in p^j_{j+1,j+1}$, $\nu(s) < 1$ and each of s originates at state $j+1$. Next suppose there is a loop around q_l and that does not contain any other loop; this loop must be followed by one or more events σ_k generated at q_l and leading to some other states q_m where $m \in \{1,\cdots,j+1\}$ and $m \neq l$. By Lemma 3.2, $\nu(p^j_{j+1,j+1}) \in [0,1)$. Proof follows by starting from the innermost loop and ending with all loops at q_j. ∎

LEMMA 3.4.

$$\nu((p^j_{j+1,j+1})^*) = \frac{1}{1-\nu(p^j_{j+1,j+1})} \in [1,\infty). \tag{3.11}$$

Proof. Since $\nu(p^j_{j+1,j+1}) \in [0,1)$ from Lemma 3.3, $\nu((p^j_{j+1,j+1})^*) = \frac{1}{1-\nu(p^j_{j+1,j+1})} \in [1,\infty)$ ∎

Finally we come to the main result of this section which is stated as the following theorem.

THEOREM 3.4. *Given a DFSA $G_i \equiv \langle Q,\Sigma,\delta,q_i,Q_m\rangle$ the following recursive result holds for $0 \le j \le n-1$:*

$$\nu(p^{j+1}_{lk}) = \nu(p^j_{lk}) + \frac{\nu(p^j_{l,j+1})\nu(p^j_{j+1,k})}{1-\nu(p^j_{j+1,j+1})}. \tag{3.12}$$

Proof.

$$\begin{aligned}
\nu(p^{j+1}_{lk}) &= \nu(p^j_{lk} \cup p^j_{l,j+1}(p^j_{j+1,j+1})^* p^j_{j+1,k}) \\
&= \nu(p^j_{lk}) + \nu(p^j_{l,j+1}(p^j_{j+1,j+1})^* p^j_{j+1,k}) \\
&= \nu(p^j_{lk}) + \nu(p^j_{l,j+1})\nu((p^j_{j+1,j+1})^*)\nu(p^j_{j+1,k}) \\
&= \nu(p^j_{lk}) + \frac{\nu(p^j_{l,j+1})\nu(p^j_{j+1,k})}{1-\nu(p^j_{j+1,j+1})}
\end{aligned}$$

where second step follows from fact that $p^j_{lk} \cap p^j_{l,j+1}(p^j_{j+1,j+1})^* p^j_{j+1,k} = \varnothing$. The third step follows from Definition 2.8 and the last step is a result of Lemma 3.4. ∎

Based on the above result, a recursive algorithm to compute a language measure is summarized as:

(1) For a given $G_i \equiv \langle Q, \Sigma, \delta, q_i, Q_m \rangle$, obtain $\mathbf{X}$ (characteristic vector) and $\tilde{\pi}$ (event cost matrix),
(2) Compute the Π matrix (Definition 2.6),
(3) $\nu(p_{lk}^0) \longleftarrow \pi_{lk}$ for $1 \leq l, k \leq n$,
(4) for j=0 to n-1
    for l=1 to n
        for k=1 to n
        $\nu(p_{lk}^{j+1}) = \nu(p_{lk}^j) + \frac{\nu(p_{l,j+1}^j)\nu(p_{j+1,k}^j)}{1-\nu(p_{j+1,j+1}^j)}$
        end
    end
  end
(5) Calculate $\nu(L(q_i, q_k))$ from $\nu(p_{ik})$ using Definition 3.4,
(6) $\mu_i \longleftarrow \sum_{q \in Q_m} \nu(L(q_i, q))\chi(q)$ is the measure of marked language of DFSA, G_i.

Since there are only three *for* loops, the computational complexity of this method is polynomial in number of states of DFSA, same as that for Method I.

4. Usage of the language measure

The two methods of language measure computation, presented in Section 3, have the same computational complexity, $\mathcal{O}(n^3)$, where n is the number of states of the DFSA. However, each of these two methods offer distinct relative advantages in specific contexts. For example, the recursive solution in Section 3.2 might prove very useful for construction of executable codes in real time applications, while the closed form solution in Section 3.1 is more amenable for analysis and synthesis of decision and control algorithms. The following two subsections present usage of the language measure for construction of metric spaces of formal languages and synthesis of optimal discrete-event supervisors.

4.1. Vector space of formal languages. The language measure can be used to construct a vector space of sublanguages for a given DFSA $G_i \equiv \langle Q, \Sigma, \delta, q_i, Q_m \rangle$. The total variation measure $|\mu|$ [10] induces a metric on this space, which quantifies the distance function between any two sublanguages of $L(G_i)$.

PROPOSITION 4.1. *Let $L(G_i)$ be the language of a DFSA $G_i = (Q, \Sigma, \delta, q_i, Q_m)$. Let the binary operation of exclusive-OR $\oplus : 2^{L(G_i)} \times 2^{L(G_i)} \rightarrow 2^{L(G_i)}$ be defined as:*

$$(K_1 \oplus K_2) = (K_1 \cup K_2) - (K_1 \cap K_2) \tag{4.1}$$

$\forall K_1, K_2 \subseteq L(G_i)$. *Then* $(2^{L(G_i)}, \oplus)$ *is a vector space over the Galois field* $GF(2)$.

Proof. It follows from the properties of exclusive-OR that the algebra $(2^{L(G_i)}, \oplus)$ is an Abelian group where $\emptyset$ is the zero element of the group and the unique inverse of every element $K \subseteq 2^{L(G_i)}$ is K itself because $K_1 \oplus K_2 = \emptyset$ if and only if $K_1 = K_2$. The associative and distributive properties of the vector space follows by defining the scalar multiplication of vectors as: $0 \otimes K = \emptyset$ and $1 \otimes K = K$. ∎

The collection of singleton languages made from each element of $L(G_i)$ forms a basis set of vector space $(2^{L(G_i)}, \oplus)$ over $GF(2)$. It is shown below, how "Total Variation" of signed measure μ can be used to define a metric on above vector space.

PROPOSITION 4.2. *Total variation measure* $|\mu|$ *on* $2^{L(G_i)}$ *is given by* $|\mu|(L) = \sum_{s \in L} |\mu(\{s\})|\ \forall L \subseteq L(G_i)$.

Proof. The proof follows from the fact that $\Sigma_k |\mu(L_k)|$ attains its supremum for the finest partition of L which consists of the individual strings in L as elements of the partition. ∎

COROLLARY 4.1. *Let* $L(G_i)$ *be a regular language for a DFSA* $G_i \equiv \langle Q, \Sigma, \delta, q_i, Q_m \rangle$. *For any* $K \in 2^{L(G_i)}$, $|\mu|(K) \leq \theta^{-1}$ *where* $\theta = 1 - ||\mathbf{\Pi}||_\infty$ *and* $\mathbf{\Pi}$ *is the state transition cost matrix of the DFSA.*

Proof. The proof follows from Proposition 4.2 and Corollary 3.1. ∎

DEFINITION 4.1. *Let* $L(G_i)$ *be a regular language for a DFSA* $G_i \equiv \langle Q, \Sigma, \delta, q_i, Q_m \rangle$. *The distance function* $d : 2^{L(G_i)} \times 2^{L(G_i)} \to [0, \infty)$ *is defined in terms of the total variation measure as:*

$$d(K_1, K_2) = |\mu|((K_1 \cup K_2) - (K_1 \cap K_2)) \tag{4.2}$$

$\forall K_1, K_2 \subseteq L(G_i)$.

The above distance function $d(\cdot, \cdot)$ quantifies the difference between two supervisors relative to the controlled performance of the DFSA plant.

PROPOSITION 4.3. *The distance function defined above is a pseudo-metric on the space* $2^{L(G_i)}$

Proof. Since the total variation of a signed real measure is bounded [10], $\forall K_1, K_2 \subseteq L(G_i)$, $d(K_1, K_2) = |\mu|(K_1 \oplus K_2) \in [0, \infty)$. Also by Definition 4.1 $d(K_1, K_2) = d(K_2, K_1)$. The remaining property of the triangular inequality follows from the inequality $|\mu|(K_1 \oplus K_2) \leq |\mu|(K_1) + |\mu|(K_2)$ which is based on the fact that $(K_1 \oplus K_2) \subseteq (K_1 \cup K_2)$ and $|\mu|(K_1) \leq |\mu|(K_2)\ \forall K_1 \subset K_2$. ∎

The pseudo-metric $|\mu| : 2^{L(G_i)} \to [0, \infty)$ can be converted to a metric of the space $(2^{L(G_i)}, \oplus)$ by clustering all languages that have zero total variation measure as the null equivalence class $N = \{K \in 2^{L(G_i)} : |\mu|(K) = 0\}$. This procedure is conceptually similar to what is done for defining norms in the L_p spaces. In that case N contains all sublanguages of $L(G_i)$, which terminate on non-marked states starting from the initial state, i.e. $N = \varnothing \cup (\cup_{q \notin Q_m} L(q, q_i))$. In the sequel, $|\mu|(\cdot)$ is referred to as a metric of the space $2^{L(G_i)}$. Thus, the metric $|\mu|(\cdot)$ can be generated from $d(\cdot, \cdot)$ as: $|\mu|(K) = d(K, J)\ \forall K \in 2^{L(G_i)}\ \forall J \in N$. Unlike the norms on vector spaces defined over infinite fields, the metric $|\mu|(\cdot)$ for the vector space $(2^{L(G_i)}, \oplus)$ over $GF(2)$ is not a functional. This interpretation of language as a vector and associating a metric to quantify distance between languages, can have significant advantage in many respects.

4.2. Optimal control of regular languages. The (signed) language measure μ could serve as the performance index for synthesis of an optimal control policy (e.g.,[11]) that maximizes the performance of a controlled sublanguage. The salient concept is succinctly presented below.

Let $\mathcal{S} \equiv \{S^0, S^1, \cdots, S^N\}$ be a set of supervisory control policies for the unsupervised plant automaton G where S^0 is the null controller (i.e., no event is disabled) implying that $L(S^0/G) = L(G)$. Therefore the controller cost matrix $\mathbf{\Pi}(S^0) = \mathbf{\Pi}^0$ that is the $\mathbf{\Pi}$-matrix of the unsupervised plant automaton G. For a supervisor $S^k, k \in \{1, 2, \cdots, N\}$, the control policy is required to selectively disable certain controllable events so that the following (elementwise) inequality holds: $\mathbf{\Pi}^k \equiv \mathbf{\Pi}(S^k) \leq \mathbf{\Pi}^0$ and $L(S^k/G) \subseteq L(G), \forall S^k \in \mathcal{S}$. The task is to synthesize an optimal cost matrix $\mathbf{\Pi}^* \leq \mathbf{\Pi}^0$ that maximizes the performance vector $\mu^* \equiv [\mathbf{I} - \mathbf{\Pi}^*]^{-1}\mathbf{X}$, i.e., $\mu^* \geq \mu^k \equiv [\mathbf{I} - \mathbf{\Pi}^{\mathbf{k}}]^{-1}\mathbf{X}\ \forall\ \mathbf{\Pi}^{\mathbf{k}} \leq \mathbf{\Pi}^0$ where the inequalities are implied elementwise. The research work in this direction is in progress and some of the results are reported in recent publications [3], [4].

5. Event Cost: A probabilistic interpretation

The signed real measure (Definition 2.9) for a DFSA is based on the assignment of the characteristic vector and the event cost matrix. As stated earlier, the characteristic function is chosen by the designer based on his/her perception of the states' impact on system performance. On the other hand, the event cost is an intrinsic property of the plant. The event cost $\tilde{\pi}_{jk}$ is conceptually similar to the state-based conditional probability as in Markov Chains, except for the fact that it is not allowed to satisfy the equality condition $\sum_k \tilde{\pi}_{jk} = 1$. (Note that $\sum_k \tilde{\pi}_{jk} < 1$ is a requirement for convergence of the language measure.) The rationale for this strict inequality is explained below.

Since the plant model is an inexact representation of the physical plant, there exist unmodelled dynamics to account for. This can manifest itself either as unmodelled events that may occur at each state or as unaccounted states in the model. Let Σ_{uj} denote the set of all unmodelled events at state j of the DFSA $G_i \equiv \langle Q, \Sigma, \delta, q_i, Q_m \rangle$. Let us create a new unmarked absorbing state q_{n+1} called the dump state [8] and extend the transition function δ to $\delta_{ext} : (Q \cup \{q_{n+1}\}) \times (\Sigma \cup_j \Sigma_{uj}) \to (Q \cup \{q_{n+1}\})$ in the following manner:

$$\delta_{ext}(q_j, \sigma) = \begin{cases} \delta(q_j, \sigma) & \text{if} \quad q_j \in Q \quad \text{and} \quad \sigma \in \Sigma \\ q_{n+1} & \text{if} \quad q_j \in Q \quad \text{and} \quad \sigma \in \Sigma_{uj} \\ q_{n+1} & \text{if} \quad j = n+1 \quad \text{and} \quad \sigma \in \Sigma \cup \Sigma_{uj}. \end{cases}$$

Therefore the residue $\theta_j = 1 - \sum_k \tilde{\pi}_{jk}$ denotes the probability of the set of unmodelled events Σ_{uj} conditioned on the state j. The $\mathbf{\Pi}$ matrix can be similarly augmented to obtain a stochastic matrix $\mathbf{\Pi}_{aug}$ as follows:

$$\mathbf{\Pi}_{aug} = \begin{bmatrix} \pi_{11} & \pi_{12} & \dots & \pi_{1n} & \theta_1 \\ \pi_{21} & \pi_{22} & \dots & \pi_{2n} & \theta_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \pi_{n1} & \pi_{n2} & \dots & \pi_{nn} & \theta_n \\ 0 & 0 & \dots & 0 & 1 \end{bmatrix}.$$

Since the dump state q_{n+1} is not marked, its characteristic value $\chi(q_{n+1}) = 0$. The characteristic vector then augments to $\mathbf{X}_{aug} = [\mathbf{X}^T \; 0]^T$. With these extensions the language measure vector $\boldsymbol{\mu}_{aug} = [\mu_1 \; \mu_2 \; \cdots \; \mu_n \; \mu_{n+1}]^T = [\boldsymbol{\mu}^T \; \mu_{n+1}]^T$ of the augmented DFSA $G_{aug} \equiv \langle Q \cup \{q_{n+1}\}, \Sigma \cup_j \Sigma_{uj}, \delta_{ext}, q_i, Q_m \rangle$ can be expressed as:

$$\begin{pmatrix} \boldsymbol{\mu} \\ \mu_{n+1} \end{pmatrix} = \begin{pmatrix} \mathbf{\Pi}\boldsymbol{\mu} + \mu_{n+1} [\theta_1 \cdots \theta_n]^T \\ \mu_{n+1} \end{pmatrix} + \begin{pmatrix} \mathbf{X} \\ 0 \end{pmatrix}. \tag{5.1}$$

Since $\chi(q_{n+1}) = 0$ and all transitions from the absorbing state q_{n+1} lead to itself, $\mu_{n+1} = \mu(L_m(G_{n+1})) = 0$. Hence Equation 5.1 reduces to that for the original plant G_i. Thus, the event cost can be interpreted as conditional probability, where the residue $\theta_j = 1 - \sum_k \tilde{\pi}_{jk}$ accounts for the probability of all unmodelled events emanating from the state q_j. With this interpretation of event cost, $\tilde{\pi}[s, q_i]$ (Definition 2.5) denotes the probability of occurrence of the event string s in the plant model G_i starting at state q_i and terminating at state $\delta^*(s, q_i)$. Hence, $\nu(L(q, q_i))$ (Definition 2.8), which is a non-negative real number, is directly related (but not necessarily equal) to the total probability that state q_i would be reached as the plant operates. The language measure $\mu_i \equiv \mu(L(G_i)) = \sum_{q \in Q} \mu(L(q_i, q)) = \sum_{q \in Q} \nu(L(q_i, q))\chi(q)$ is then directly related (but not necessarily equal) to the expected value of the characteristic function. As mentioned earlier, the choice of the characteristic function (Definition 2.4) is solely based on the designer's perception of the importance assigned to the individual DFSA states. Therefore, in the setting of language measure, a supervisor's performance is superior if the supervised plant is more likely to terminate at a *good* marked state and/or less likely to terminate at a *bad* marked state.

6. Summary, conclusions, and recommendations for future research

This paper reviews the concept, formulation and validation of a signed real measure for any regular language and its sublanguages based on the principles of measure theory and automata theory. While the domain of measure μ, i.e., $2^{L(G_i)}$ is partially ordered, its range, which is a subset of $\mathbf{R} \equiv (-\infty, \infty)$, becomes totaly ordered. As a result, the relative performance of different supervisors can be quantitatively evaluated in terms of the real signed measure of the controlled sublanguages. Positive weights are assigned to *good* marked states and negative weights to *bad* marked states so that a controllable supervisor is rewarded (penalized) for deleting strings terminating at *bad* (*good*) marked states. In order to evaluate and compare the performance of different supervisors a common quantitative tool is required. To this effect, the proposed procedure computes the measure of the controlled sublanguage generated by a supervisor using the event cost and characteristic function assigned for the unsupervised plant. Cost assignment to each event based on the state, where it is generated, has been shown similar to the conditional probability of the event. On the other hand, the characteristic function is chosen based on the designer's perception of the individual state's impact on the system performance. Two techniques are presented to compute the language measure for a DFSA. One of these two methods yields a closed form solution that is obtained as the unique solution of a set of linearly independent algebraic equations. The other method is based on a recursive procedure. The computational complexity of both language measure algorithms is polynomial in the number of the DFSA states.

6.1. Recommendations for future research. Further research is recommended for development of systematic procedures for assigning/identifying the event cost matrix and the characteristic vector. It is also worth investigating how to extend the field $GF(2)$, over which the vector space of languages has been defined, to much richer fields like the set of reals $\mathbf{R}$. Other areas of research include applications of the language measure in anomaly detection, model identification, model order reduction, and analysis and synthesis of robust and optimal control in the discrete-event setting. It would be challenging to extend the concept of (regular) language measure for languages higher up in the Chomsky Hierarchy [5] such as context free and context sensitive languages. This extension would lead to controller synthesis when the plant dynamics is modelled by non-regular languages such as the Petri Net.

References

[1] V. Drobot, *Formal Languages and Automata Theory*, Computer Science Press, 1989.

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

[3] J. Fu, A. Ray and C. M. Lagoa, *Unconstrained optimal control of regular languages*, IEEE Conference on Decision and Control, Las Vegas, Nevada, 2002, pp. 799–804.

[4] J. Fu, A. Ray and C. M. Lagoa, *Optimal control of regular languages with event disabling cost*, American Control Conference, Boulder, Colorado, 2003, pp. 1691–1695.

[5] J. C. Martin, *Introduction to Languages and the Theory of Computation*, 2nd ed., McGraw Hill, Boston, 1997.

[6] A. W. Naylor and G. R. Sell, *Linear Operator Theory in Engineering and Science*, Springer-Verlag, New York, 1982.

[7] R. J. Plemmons and A. Berman, *Nonnegative Matrices in the Mathematical Sciences*, Academic Press, 1979.

[8] P. J. Ramdage and W. M. Wonham, *Supervisory control of a class of discrete event processes*, SIAM J. Control Optim. 25 (1), 1987.

[9] A. Ray and S. Phoha, *Signed Real measure of regular languages for discrete-event automata*, Int. J. Control 76 (18), 1800–1808.

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

[11] R. Sengupta and S. Lafortune, *An optimal control theory for discrete event systems*, SIAM J. Control Optim. 36 (2) (1998), 488–541.

[12] X. Wang and A. Ray, *Signed Real Measure of Regular Languages*, American Control Conference, Anchorage, Alaska, 2002, pp. 3937–3942.

DEPARTMENT OF MECHANICAL ENGINEERING
THE PENNSYLVANIA STATE UNIVERSITY
UNIVERSITY PARK, PA 16802, USA
E-mail: azs110@psu.edu
axr2@psu.edu

Received February 10, 2003; revised version June 25, 2003.