National School SIDRA 2017: Formal Methods for the Control of Large-scale Networked Nonlinear Systems with Logic Specifications

Lecture L4: Regular languages*

Abstract. In this lecture we will recall basic notions of regular languages. This lecture is based on [1] and [2].

1 Notation

The standard symbols $\cup$, $\cap$ and $\setminus$ denote the set union, set intersection and set difference. The symbols $\mathbb{N}$, $\mathbb{Z}$ and $\mathbb{R}$ denote the set of nonnegative integer, integer and real numbers, respectively. Given a vector $x \in \mathbb{R}^n$ we denote by $|x|$ the infinity norm of x.

2 Regular languages

We recall some notions on formal language theory. Let Y be a finite set representing the alphabet.

Definition 1. *A word w over Y is a finite sequence $y_1\, y_2 \dots y_l$ of symbols in Y.*

The length of a word $w = y_1\, y_2 \dots y_l$ is l.

Example 1. parola is a word over the Latin alphabet and its length is 6.

Definition 2. *The concatenation of two words $y_1\, y_2 \dots y_l$ and $y_{l+1}\, y_{l+2} \dots y_{l'}$ is the word*

$$y_1\, y_2 \cdots y_l y_{l+1}\, y_{l+2} \cdots y_{l'}.$$

The empty word is denoted by ε. The length of ε is zero. The empty word is the identity element of concatenation, i.e.

$$\varepsilon w = w \varepsilon = w.$$

Definition 3. *Given three words p, t and s and their concatenation pts, p is said a prefix of pts, t a substring of pts, and s a suffix of pts.*

It is easy to see that:

- Empty string and w are both prefixes of w, indeed, $w = \varepsilon w$ and $w = w\varepsilon$;
- Empty string and w are both substrings of w, indeed, $w = w\varepsilon\varepsilon$ and $w = \varepsilon w \varepsilon$;
- Empty string and w are both suffixes of w, indeed, $w = w\varepsilon$ and $w = \varepsilon w$.

We can now give the following

Definition 4. *The symbol Y^* denotes the Kleene closure of Y, that is the collection of all words over Y including the empty word.*

Remark 1. Unless $Y = \varnothing$ or $Y = \{\varepsilon\}$, the cardinality of Y^* is infinite.

Example 2. For $Y = \{a, b\}$ we get $Y^* = \{\varepsilon, a, b, aa, ab, ba, bb, \dots\}$.

Similarly,

Definition 5. *Given a word y over Y, the symbol $\{y\}^*$ denotes the Kleene closure of word y, that is the collection of all words, including the empty word, obtained by concatenating y with itself, an arbitrary but finite number of times.*

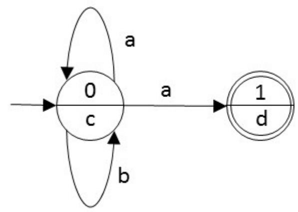

Fig. 1. Transition system T.

Example 3. For $y = ab$ we get $\{y\}^* = \{\varepsilon, ab, abab, ababab, ...\}$.

We can now give the following

Definition 6. *A language L over a finite set Y is a subset of Y^*.*

Example 4. The following sets are languages in the sense of the definition above:

- The Italian language L_1 over the Latin alphabet;
- The set L_2 of all words over $\{a, b\}$ ending with a;
- The set L_3 of all words over $\{a, b\}$ of the form $a^n b^n$, with n integer.

Remark 2. L_1 has finite cardinality while L_2 and L_3 have not.

We now extend the notion of concatenation to languages:

Definition 7. *The concatenation of two languages L_1 and L_2 is the language $L_1 L_2$ containing all and only the words $w_1 w_2$ obtained by the concatenation of a word $w_1 \in L_1$ and a word $w_2 \in L_2$.*

Definition 8. *The Kleene closure of a language L is defined as*

$$L^* = \{\varepsilon\} \cup L \cup LL \cup LLL...$$

We now introduce connections between languages and transition systems.

Definition 9. *The input language (resp. output language) of a transition system $T = (X, X_0, U, \longrightarrow, X_m, Y, H)$, denoted $\mathcal{L}^u(T)$ (resp. $\mathcal{L}^y(T)$), is the collection of all its input runs (resp. output runs).*

Definition 10. *The marked input language (resp. marked output language) of a transition system $T = (X, X_0, U, \longrightarrow, X_m, Y, H)$, denoted as $\mathcal{L}^u_m(T)$ (resp. $\mathcal{L}^y_m(T)$), is the collection of all its input runs*

$$r_U : u_0 \, u_1 \, ... \, u_{l-1},$$

(resp. output runs

$$r_Y : H(x_0) \, H(x_1) \, ... \, H(x_l))$$

such that the corresponding sequence of transitions

$$x_0 \xrightarrow{u_0} x_1 \xrightarrow{u_1} \ldots \xrightarrow{u_{l-1}} x_l$$

with $x_0 \in X_0$, is with ending state $x_l \in X_m$.

Remark 3. Note that $\mathcal{L}_m^u(T) \subseteq \mathcal{L}^u(T)$ and $\mathcal{L}_m^y(T) \subseteq \mathcal{L}^y(T)$.

Example 5. For the transition system T in Fig. 1, we have:

$$\begin{array}{l} \mathcal{L}^u(T) = \{\varepsilon, a, b, aa, ab, ba, bb, ...\} = \{a, b\}^*, \\ \mathcal{L}^y(T) = \{\varepsilon, c, cc, cd, ccc, ccd, ...\}, \\ \mathcal{L}_m^u(T) = \{a, aa, ba, ...\}, \\ \mathcal{L}_m^y(T) = \{cd, ccd, ...\}. \end{array}$$

We can now give the following

Definition 11. *A language L over a finite set U is said regular if there exists a symbolic transition system T with input set U such that*

$$L = \mathcal{L}_m^u(T).$$

Example 6. Consider the language L over $\{a, b\}$ composed of all words ending with a. This language coincides with $\mathcal{L}_m^u(T)$ where transition system T is defined in Example 5. Hence, L is regular.

Example 7. A classical example of a language that is not regular is the language L defined over $\{a, b\}$ and composed of all words of the form $a^n b^n$ with n integer. By having a look at Fig. 2, it is clear that a transition system that marks L needs an infinite number of states, because it needs to count how many a's have been processed to be able to replicate the same number of b's.

Here below we recall some important properties of regular languages.

Theorem 1. *If L_1 and L_2 are regular languages, then the following languages are regular:*

- *The prefix closure of L_i, denoted $\overline{L_i}$, i.e. the set of all words that are prefixes of all words in L_i;*
- L_i^*;
- $Y^* \backslash L_i$;
- $L_1 \cup L_2$;
- $L_1 L_2$;
- $L_1 \cap L_2$.

The proof of the standard result above is constructive, in the sense that given symbolic transition systems marking L_1 and L_2, symbolic transition systems marking $\overline{L_i}$, L_i^*, $Y^* \backslash L_i$, $L_1 \cup L_2$, $L_1 L_2$, $L_1 \cap L_2$ are constructed. An example is reported below.

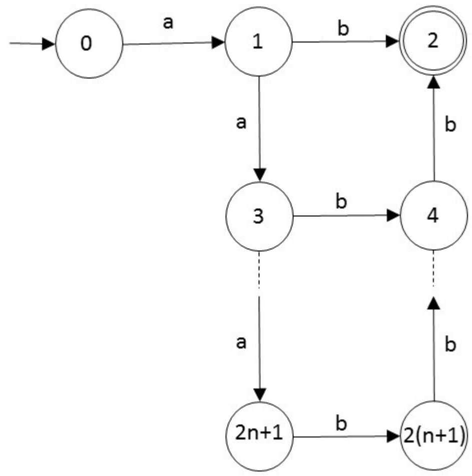

Fig. 2. Transition system marking language L defined over $\{a, b\}$ and composed of all words of the form $a^n b^n$ with n integer. Outputs are not represented meaning that the output function is the identity.

Example 8. Let us show that if L is regular then also $\overline{L}$ is regular. If L is regular there exists a symbolic transition system T such that

$$L = \mathcal{L}_m^u(T).$$

Starting from T, first construct the coaccessible part $\mathrm{Coac}(T)$ of T and then mark all the states of $\mathrm{Coac}(T)$. Call the obtained transition system T'. It is easy to see that

$$\overline{L} = \mathcal{L}_m^u(T').$$

Moreover, if T is symbolic, so is $\mathrm{Coac}(T)$ and also, so is T'. Hence, $\overline{L}$ is regular.

We now introduce the notion of regular expressions:

Definition 12. *A regular expression over a finite set U is defined recursively as follows:*

1) *The following are regular expressions:*
   - $\varnothing$*, denoting the empty set;*
   - ε*, denoting the set* $\{\varepsilon\}$*;*
   - u*, denoting the set* $\{u\}$*, for all* $u \in U$*.*
2) *If r and s are regular expressions, then*
   - rs*, denoting the concatenation of r and s, is a regular expression;*
   - $(r + s)$*, denoting the union of* $\{r\}$ *and* $\{s\}$*, is a regular expression;*
   - r^* *and* s^**, denoting the Kleene closure of r and s, respectively, are regular expressions.*

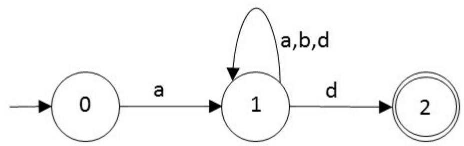

Fig. 3. Transition system T of Example 9. Outputs are not represented meaning that the output function is the identity.

3) There are no regular expressions other than those constructed by applying rules 1) and 2) above a finite number of times.

We can now recall the following important result:

Theorem 2. *(Kleene's Theorem) Any language that can be denoted by a regular expression is a regular language; conversely, any regular language can be denoted by a regular expression.*

There are several automatic tools to translate a regular expression in a symbolic transition system marking the corresponding regular language and vice versa. We conclude this section with two examples.

Example 9. Let $U = \{a, b, c, d\}$. The regular expression corresponding to all words starting with a, ending with d and with no symbols c is given by

$$a(a + b + d)^* d$$

The regular expression above corresponds to the regular language:

$$L = \{a\}\{a, b, d\}^*\{d\}.$$

A symbolic transition system T marking L is depicted in Fig. 3.

Example 10. In Fig. 4, we illustrate how to find a regular expression corresponding to the regular language marked by symbolic transition system T. Starting from T one can construct T' where transition from state $1'$ to $0'$ is labelled by the regular expression $b+ca$. By this construction we did not alter words marked by T. Starting from T' we construct T'' which again does not alter the language marked by T' and hence, by T. Note that T' and T'' are not transition systems because some of their labels are regular expressions; they are only instrumental in deriving regular expressions. It is easy to see that the regular expression associated to T'', and hence to T, is

$$(a(b + ca))^* a.$$

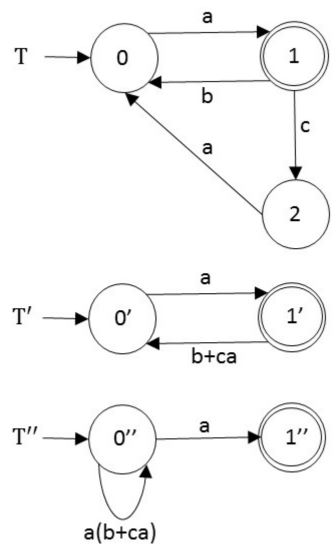

Fig. 4. Transition system T of Example 10. Outputs are not represented meaning that the output function is the identity.

3 Modeling logic specifications via regular languages

We have seen in lecture L3 that transition systems are general enough to describe the physical and the cyber parts of CPS. We now show how transition systems are useful also to model logic specifications.
Consider a finite collection of points Y_Q of $\mathbb{R}^n$. This set can be viewed as an alphabet on the basis of which a regular language

$$L_Q \subseteq Y_Q^*$$

can be constructed. This regular language models the specifications we are interested in when controlling a CPS. In the sequel we present some examples of logic specifications that are of interest when controlling CPS.

Safety specifications

Given a subset of good states G of $\mathbb{R}^n$, my specification requires to stay all the time inside G.
How to model this specification as a regular expression?
Suppose that G has interior and is given as the union of a finite collection of hyperrectangles. Consider the collection of points g_j in

$$G_\eta = G \cap (\eta\mathbb{Z}^n),$$

where $\eta \in \mathbb{R}^+$ represents the accuracy of the specification approximation. Under the assumptions placed on G, there exists $\hat{\eta} \in \mathbb{R}^+$ such that $G_\eta \neq \varnothing$ for any $\eta \leq \hat{\eta}$, see [3]. The regular expression modeling the safety specification corresponds to all and only words with symbols g_j, that is

$$\left(\sum_{g_j \in G_\eta} g_j\right)\left(\sum_{g_j \in G_\eta} g_j\right)^* .$$

Reachability specifications

Starting from a set of initial states $I \subseteq \mathbb{R}^n$, my specification requires to reach in finite time a target set $T \subseteq \mathbb{R}^n$.
How to model this specification as a regular expression?
Suppose that I and T have interior and are given as the unions of finite collections of hyperrectangles. Let $D \subseteq \mathbb{R}^n$ be a set representing the domain of interest and assume it has interior, is given as the union of a finite collection of hyperrectangles, and contains sets I and T. Consider the set I_η of points i_j in the lattice $\eta\mathbb{Z}^n$ that are far away from I no more than η, where $\eta \in \mathbb{R}^+$ represents the accuracy of the specification approximation, i.e. for any $i_j \in I_\eta$ there exists $x_j \in I$ such that

$$|i_j - x_j| \leq \eta.$$

Note that $I_\eta \neq \varnothing$ for any $\eta \in \mathbb{R}^+$. Consider the collection of points t_j in the set

$$T_\eta = T \cap (\eta\mathbb{Z}^n).$$

Consider the collection of points d_j in the set

$$D_\eta = D \cap (\eta\mathbb{Z}^n).$$

Under the assumptions placed on T and D, there exists $\hat{\eta} \in \mathbb{R}^+$ such that $T_\eta \neq \varnothing$ and $D_\eta \neq \varnothing$ for any $\eta \leq \hat{\eta}$. The regular expression modeling the reachability specification corresponds to all and only words starting with symbols in I_η and with last symbols in T_η, i.e.

$$\left(\sum_{i_j \in I_\eta} i_j\right)\left(\sum_{d_j \in D_\eta} d_j\right)^*\left(\sum_{t_j \in T_\eta} t_j\right).$$

Reach and stay with obstacle avoindance specifications

Starting from a set of initial states $I \subseteq \mathbb{R}^n$, my specification requires to reach in finite time a target set $T \subseteq \mathbb{R}^n$, while avoiding the set of obstacles $O \subseteq \mathbb{R}^n$ and then remain definitely in the target set T.
We suppose that

$$I \cap O \cap T = \varnothing.$$

How to model this specification as a regular expression?
Suppose that I, O and T have interior and are given as the unions of finite collections of hyperrectangles. Let $D \subseteq \mathbb{R}^n$ be a set representing the domain of interest and assume it has interior, is given as the union of a finite collection of hyperrectangles, and contains sets I, O and T. Consider the set I_η of points i_j in the lattice $\eta\mathbb{Z}^n$ that are far away from I no more than η, where $\eta \in \mathbb{R}^+$ represents the accuracy of the specification approximation, i.e. for any $i_j \in I_\eta$ there exists $x_j \in I$ such that

$$|i_j - x_j| \leq \eta.$$

Note that $I_\eta \neq \varnothing$ for any $\eta \in \mathbb{R}^+$. Consider the set O_η of points o_j in the lattice $\eta\mathbb{Z}^n$ that are far away from O no more than η, i.e. for any $o_j \in O_\eta$ there exists $x_j \in O$ such that

$$|o_j - x_j| \leq \eta.$$

Note that $O_\eta \neq \varnothing$ for any $\eta \in \mathbb{R}^+$. Consider the collection of points t_j in the set

$$T_\eta = T \cap (\eta\mathbb{Z}^n).$$

Consider the collection of points d_j in the set

$$D_\eta = D \cap (\eta\mathbb{Z}^n).$$

Under the assumptions placed on T and D, there exists $\hat{\eta} \in \mathbb{R}^+$ such that $T_\eta \neq \varnothing$ and $D_\eta \neq \varnothing$ for any $\eta \leq \hat{\eta}$. The regular expression modeling the specification under consideration corresponds to all and only words starting with symbols in I_η, with no symbols in O_η and ending with symbols in T_η, i.e.

$$\left(\sum_{i_j \in I_\eta} i_j\right)\left(\sum_{d_j \in D_\eta \setminus O_\eta} d_j\right)^*\left(\sum_{t_j \in T_\eta} t_j\right)\left(\sum_{t_j \in T_\eta} t_j\right)^*.$$

Synchronization specifications

Starting from a set of initial states $I \subseteq \mathbb{R}^n$ my specification requires to reach a set $R \subseteq \mathbb{R}^n$ in no more than 2s, stay there for at most $4s$ and then reach a target set $T \subseteq \mathbb{R}^n$ in no less than $3s$ but in finite time. We suppose that

$$I \cap R \cap T = \varnothing.$$

This specification is relevant in multi–agent environments where a resource is shared by different agents.

How to model this specification as a regular expression?
Suppose that I, R and T have interior and are given as the unions of finite collections of hyperrectangles. Let $D \subseteq \mathbb{R}^n$ be a set representing the domain of interest and assume it has interior, is given as the union of a finite collection of hyperrectangles, and contains sets I, R and T. Consider the set I_η of points i_j in the lattice $\eta\mathbb{Z}^n$ that are far away from I no more than η, where $\eta \in \mathbb{R}^+$ represents the accuracy of the specification approximation, i.e. for any $i_j \in I_\eta$ there exists $x_j \in I$ such that

$$|i_j - x_j| \leq \eta.$$

Note that $I_\eta \neq \varnothing$ for any $\eta \in \mathbb{R}^+$. Consider the collection of points r_j in the set

$$R_\eta = R \cap (\eta\mathbb{Z}^n).$$

Consider the collection of points t_j in the set

$$T_\eta = T \cap (\eta\mathbb{Z}^n).$$

Consider the collection of points d_j in the set

$$D_\eta = D \cap (\eta\mathbb{Z}^n).$$

Under the assumptions placed on R, T and D, there exists $\hat{\eta} \in \mathbb{R}^+$ such that $R_\eta \neq \varnothing$, $T_\eta \neq \varnothing$ and $D_\eta \neq \varnothing$ for any $\eta \leq \hat{\eta}$. Suppose to consider an internal clock of our digital controller $\tau = 1s$. By setting the following regular expressions

$$\begin{array}{l} I' = \sum_{i_j \in I_\eta} i_j; \\ R' = \sum_{r_j \in R_\eta} r_j; \\ T' = \sum_{t_j \in T_\eta} t_j; \\ D' = \sum_{d_j \in D_\eta \setminus R_\eta} d_j; \\ D'' = \sum_{d_j \in D_\eta \setminus T_\eta} d_j. \end{array}$$

the required regular espression is given by:

$$I'(\varepsilon + D')(R' + R'R' + R'R'R' + R'R'R'R')(D''D''(D'')^*)T'.$$

References

1. C.G. Cassandras and S. Lafortune. *Introduction to Discrete Event Systems*. Kluwer Academic Publishers, 1999.
2. G. Pola, P. Pepe, and M. D. Di Benedetto. Decentralized approximate supervisory control of networks of nonlinear control systems. *IEEE Transactions on Automatic Control*, 2017. Submitted for publication. Available online at arxiv.org/abs/1606.04647 [math.OC].
3. M. Zamani, M. Mazo, G. Pola, and P. Tabuada. Symbolic models for nonlinear control systems without stability assumptions. *IEEE Transactions of Automatic Control*, 57(7):1804–1809, July 2012.