COMPUTATIONAL SIMULATION FORMALIZED BY STRING-RELATION SYSTEMS

Ing. Martin ŠVEC, Doctoral Degree Programme (2)
Dept. of Information Systems, FIT, BUT
E-mail: svecm@fit.vutbr.cz

Supervised by: Dr. Alexander Meduna

ABSTRACT

This contribution introduces a more detailed approach to the rewriting processes of computational models. In contrast to the classical definition of equivalency of formal language models, which only requires that the yield sequences produce identical languages, this paper concentrate on the similarity of all steps of the yield sequences. More specifically, if there exists a suitable substitution mapping strings of every yield sequence in one model to a sequence of strings in such a way that this sequence forms a yield sequence in another model, we say that the first model is a simulation of the second one. The paper formalizes this concept and demonstrates it on examples.

1 INTRODUCTION

The formal language theory defines equivalent formal models as models that generate the same language. This definition of equivalency plays a crucial role in almost every transformation of formal language models, such as grammars or automata. However, taking a closer look at the transformations of equivalent models, we see that some transformations result in models that generate their languages in a more similar way than others.

Consider such a transformation converting one formal model to another equivalent model. Next, consider a yield sequence generated by the first model and a yield sequence of the second model. If there exists a substitution such that for every step of the first yield sequence, there is a corresponding subsequence of steps in the second yield sequence such that the substitution maps the first and the last string of the subsequence to the strings appearing in the given step of the first yield sequence, we say that the second yield sequence simulates the first one with respect to the given substitution. Furthermore, if the number of steps of the correspoding subsequence is limited by some finite natural number, such a simulation is said to be close. By a natural generalization of this simulation to all yield sequences of the models, we obtain a simulation-based relationship, reflecting the similarity of the yield sequences of these models. This paper provides a formal definition of the above described concept. Then, it demonstrates it on two detailed examples.

2 PRELIMINARIES

This contribution assumes that the reader is familiar with the language theory (see [2]).

Let V be an alphabet. V^* denotes the free monoid generated by V under the operation of concatenation. Let ε be the unit of V^* and $V^+ = V^* - \{\varepsilon\}$. Given a word, $w \in V^*$, $|w|$ represents the length of w and $\mathrm{alph}(w)$ denotes the set of all symbols occuring in w. Moreover, $\mathrm{sub}(w)$ denotes the set of all subwords of w. Let R be a binary relation on a set W. Instead of $u \in R(v)$, $u, v \in W$, we write vRu in this paper.

3 COMPUTATIONAL SIMULATION

Definition 1. A *string-relation system* is a quadruple $\Psi = (W, \Rightarrow, W_0, W_F)$, where W is a language, $\Rightarrow$ is a binary relation on W, $W_0 \subseteq W$ is a set of *start strings*, and $W_F \subseteq W$ is a set of *final strings*.

Every string, $w \in W$, represents a 0-step string-relation sequence in Ψ. For every $n \geq 1$, a sequence $w_0, w_1, \ldots w_n$, $w_i \in W$, $0 \leq i \leq n$, is an *n-step string-relation sequence*, symbolically written as $w_0 \Rightarrow w_1 \Rightarrow \ldots \Rightarrow w_n$, if for each $0 \leq i \leq n-1$, $w_i \Rightarrow w_{i+1}$.

If there is a string-relation sequence $w_0 \Rightarrow w_1 \Rightarrow \ldots \Rightarrow w_n$, where $n \geq 0$, we write $w_0 \Rightarrow^n w_n$. Furthermore, $w_0 \Rightarrow^* w_n$ means that $w_0 \Rightarrow^n w_n$ for some $n \geq 0$, and $w_0 \Rightarrow^+ w_n$ means that $w_0 \Rightarrow^n w_n$ for some $n \geq 1$. Obviously, from the mathematical point of view, $\Rightarrow^+$ and $\Rightarrow^*$ are the transitive closure of $\Rightarrow$ and the transitive and reflexive closure of $\Rightarrow$, respectively.

Let $\Psi = (W, \Rightarrow, W_0, W_F)$ be a string-relation system. A string-relation sequence in Ψ, $u \Rightarrow^* v$, where $u, v \in W$, is called a *yield sequence*, if $u \in W_0$. If $u \Rightarrow^* v$ is a yield sequence and $v \in W_F$, $u \Rightarrow^* v$ is *successful*.

Let $D(\Psi)$ and $SD(\Psi)$ denote the set of all yield sequences and all successful yield sequences in Ψ, respectively.

Example 1. To illustrate the way we use string-relation systems in this contribution, consider a context-free grammar $G = (V, T, P, S)$, where V, T, P, and S are the total alphabet, the terminal alphabet, the set of productions, and the start symbol, respectively. In the standard way (see [2]), define the direct derivation $\Rightarrow$ on V^*, the set of G's sentential forms $F(G)$, and the language of G, $L(G)$. Then, introduce a string-relation system $\Psi = (V^*, \Rightarrow, \{S\}, T^*)$. Observe that $w_0 \Rightarrow w_1 \Rightarrow \ldots \Rightarrow w_n$ is a yield sequence in Ψ if and only if $w_n \in F(G)$. Furthermore, $w_0 \Rightarrow w_1 \Rightarrow \ldots \Rightarrow w_n$ is a successful yield sequence if and only if $w_n \in L(G)$.

Example 2. To give another example of a string-relation system, consider a finite automaton, $M = (Q, \Sigma, R, s, F)$, where Q, Σ, R, s, and F are the set of states, the alphabet of input symbols, the set of computational states, the start state, and the set of final states in M, respectively. Define the move relation, $\vdash$, on $Q\Sigma^*$, and the language of M, $L(M)$ (see [2] for the formal definition). Then, create a string-relation system, $\Psi = (Q\Sigma^*, \vdash, \{s\}\Sigma^*, F)$. Examine string-relation sequences in Ψ to see that $q_0w_0 \vdash q_1w_1 \vdash \ldots \vdash q_nw_n$ is a yield sequence if and only if M makes a sequence of moves $q_0w_0 \vdash q_1w_1 \vdash \ldots \vdash q_nw_n$, where $s = q_0$, $q_i \in Q$, $w_i \in \Sigma^*$, $0 \leq i \leq n$, and n is a non-negative integer. Notice that if $q_0w_0 \vdash$

$q_1w_1 \vdash \ldots \vdash q_nw_n$ is a successful yield sequence in Ψ, $q_n \in F$ and $w_n = \varepsilon$. Finally, observe that $q_0w_0 \vdash q_1w_1 \vdash \ldots \vdash q_nw_n$ is a successful yield sequence if and only if $w_0 \in L(M)$.

Definition 2. Let $\Psi = (W, \Rightarrow_\Psi, W_0, W_F)$ and $\Omega = (W', \Rightarrow_\Omega, W'_0, W'_F)$ be two string-relation systems, and let σ be a substitution from W' to W. Furthermore, let d be a yield sequence in Ψ of the form $w_0 \Rightarrow_\Psi w_1 \Rightarrow_\Psi \ldots \Rightarrow_\Psi w_{n-1} \Rightarrow_\Psi w_n$, where $w_i \in W$, $0 \leq i \leq n$, for some $n \geq 0$. A yield sequence, h, in Ω *simulates d with respect to* σ, symbolically written as $h \rhd_\sigma d$, if h is of the form $y_0 \Rightarrow_\Omega^{m_1} y_1 \Rightarrow_\Omega^{m_2} \ldots \Rightarrow_\Omega^{m_{n-1}} y_{n-1} \Rightarrow_\Omega^{m_n} y_n$, where $y_j \in W'$, $0 \leq j \leq n$, $m_k \geq 1$, $1 \leq k \leq n$, and $w_i \in \sigma(y_i)$ for all $0 \leq i \leq n$. If, in addition, there exists $m \geq 1$ such that $m_k \leq m$ for each $1 \leq k \leq n$, then h *m-closely simulates d with respect to* σ, symbolically written as $h \rhd_\sigma^m d$.

Definition 3. Let $\Psi = (W, \Rightarrow_\Psi, W_0, W_F)$ and $\Omega = (W', \Rightarrow_\Omega, W'_0, W'_F)$ be two string-relation systems, and let σ be a substitution from W' to W. Let $X \subseteq D(\Psi)$ and $Y \subseteq D(\Omega)$. *Y simulates X with respect to* σ, written as $Y \rhd_\sigma X$, if the following two conditions hold:

1. for every $d \in X$, there is $h \in Y$ such that $h \rhd_\sigma d$;
2. for every $h \in Y$, there is $d \in X$ such that $h \rhd_\sigma d$.

Let m be a positive integer. *Y m-closely simulates X with respect to* σ, $Y \rhd_\sigma^m X$, provided that:

1. for every $d \in X$, there is $h \in Y$ such that $h \rhd_\sigma^m d$;
2. for every $h \in Y$, there is $d \in X$ such that $h \rhd_\sigma^m d$.

Definition 4. Let $\Psi = (W, \Rightarrow_\Psi, W_0, W_F)$ and $\Omega = (W', \Rightarrow_\Omega, W'_0, W'_F)$ be two string-relation systems. If there exists a substitution σ from W' to W such that $D(\Omega) \rhd_\sigma D(\Psi)$ and $SD(\Omega) \rhd_\sigma SD(\Psi)$, then Ω is said to be Ψ's *computational simulator* and *successful-computational simulator*, respectively. Furthermore, if there is an integer, $m \geq 1$, such that $D(\Omega) \rhd_\sigma^m D(\Psi)$ and $SD(\Omega) \rhd_\sigma^m SD(\Psi)$, Ω is called an *m-close computational simulator* and *m-close successful-computational simulator* of Ψ, respectively. If there exists a homomorphism ρ from W' to W such that $D(\Omega) \rhd_\rho D(\Psi)$, $SD(\Omega) \rhd_\rho SD(\Psi)$, $D(\Omega) \rhd_\rho^m D(\Psi)$, and $SD(\Omega) \rhd_\rho^m SD(\Psi)$, then Ω is Ψ's *homomorphic computational simulator*, *homomorphic successful-computational simulator*, *m-close homomorphic computational simulator* and *m-close homomorphic successful-computational simulator*, respectively.

Example 3. Let us demonstrate the idea of computational simulations on grammars generating the language $L = \{a^nb^n : n \geq 1\}$. Consider

$$\begin{array}{rcl} G_1 & = & (V_1, \{a,b\}, P_1, S), \text{ where} \\ V_1 & = & \{S,a,b\}, \\ P_1 & = & \{S \rightarrow ab,\ S \rightarrow aSb\}. \end{array}$$

Clearly, every derivation in G_1 has the form

$$S \Rightarrow_{G_1} aSb \Rightarrow_{G_1} aaSbb \Rightarrow_{G_1} \ldots \Rightarrow_{G_1} a^{n-1}Sb^{n-1} \Rightarrow_{G_1} a^nb^n$$

for some $n \geq 1$. The language of G_1 is L. Next, consider

$$\begin{array}{rcl} G_2 & = & (V_2, \{a,b\}, P_2, S), \text{ where} \\ V_2 & = & \{S, A, B, a, b\}, \\ P_2 & = & \{S \to aB,\ B \to Ab,\ A \to aB,\ B \to b\}. \end{array}$$

G_2 makes every derivation in this way

$$S \Rightarrow_{G_2} aB \Rightarrow_{G_2} aAb \Rightarrow_{G_2} aaBb \Rightarrow_{G_2} aaAbb \Rightarrow_{G_2} \ldots \Rightarrow_{G_2} a^nBb^{n-1} \Rightarrow_{G_2} a^nAb^n,$$

where $n \geq 1$. Furthermore, every sentential form a^nBb^{n-1} can be rewritten to a^nb^n. Obviously, $L(G_2) = L(G_1) = L$.

Investigate the derivations in G_1 and G_2 in terms of computational simulations. To do so, introduce the corresponding string-relation systems $\Psi_1 = (V_1^*, \Rightarrow_{G_1}, \{S\}, \{a,b\}^*)$ and $\Psi_2 = (V_2^*, \Rightarrow_{G_2}, \{S\}, \{a,b\}^*)$ by analogy with Example 1. Notice that Ψ_1 and Ψ_2 are defined so that their yield sequences correspond to the above derivations in G_1 and G_2. Then, introduce a homomorphism σ_2 from V_2^* to V_1^* as $\sigma_2(S) = \sigma_2(A) = S$, $\sigma_2(B) = \sigma_2(b) = b$, $\sigma_2(a) = a$. Let us show that Ψ_2 is a 2-close homomorphic computational simulator of Ψ_1 with respect to σ_2. First, inspect all steps of yield sequences in Ψ_1:

1. for $S \Rightarrow_{G_1} ab$, there is $S \Rightarrow_{G_2} aB \Rightarrow_{G_2} ab$;
2. for $S \Rightarrow_{G_1} aSb$, Ψ_2 makes $S \Rightarrow_{G_2} aB \Rightarrow_{G_2} aAb$, where $\sigma_2(aAb) = aSb$;
3. for $a^{n-1}Sb^{n-1} \Rightarrow_{G_1} a^nSb^n$, $n \geq 2$, there is $a^{n-1}Ab^{n-1} \Rightarrow_{G_2} a^nBb^{n-1} \Rightarrow_{G_2} a^nAb^n$, where $\sigma_2(a^{n-1}Ab^{n-1}) = a^{n-1}Sb^{n-1}$, $\sigma_2(a^nAb^n) = a^nSb^n$;
4. for $a^{n-1}Sb^{n-1} \Rightarrow_{G_1} a^nb^n$, $n \geq 2$, there exists $a^{n-1}Ab^{n-1} \Rightarrow_{G_2} a^nBb^{n-1} \Rightarrow_{G_2} a^nb^n$ with $\sigma_2(a^{n-1}Ab^{n-1}) = a^{n-1}Sb^{n-1}$ and $\sigma_2(a^nb^n) = a^nb^n$.

That is, every step in any yield sequence from Ψ_1 can be simulated by two steps in Ψ_2. Hence, by induction on the length of yield sequences in Ψ_1, prove that every $d \in D(\Psi_1)$ is 2-close-simulatable by some $h \in D(\Psi_2)$ with respect to σ_2; in symbols, $h \rhd^2_{\sigma_2} d$. Next, observe that every $h \in D(\Psi_2)$ is a 2-close homomorphic simulation of some $d \in D(\Psi_1)$. Indeed, $S \Rightarrow^*_{G_2} a^nAb^n$ and $S \Rightarrow^*_{G_2} a^nb^n$, $n \geq 1$, are 2-close simulations of yield sequences from Ψ_1. The other forms of yield sequences in Ψ_2 are of the form $S \Rightarrow_{G_2} aB$ and $S \Rightarrow^+_{G_2} a^nAb^n \Rightarrow_{G_2} a^{n+1}Bb^n$, $n \geq 1$. Because $\sigma_2(B) = b$, the first sequence is a 1-close simulation of $S \Rightarrow_{G_1} ab$ and the second sequence is a 2-close simulation of $S \Rightarrow^+_{G_1} a^nSb^n \Rightarrow_{G_2} a^{n+1}b^{n+1}$. Hence, for every $h \in D(\Psi_2)$, there exists $d \in D(\Psi_1)$ such that $h \rhd^2_{\sigma_2} d$. As a result, $D(\Psi_2) \rhd^2_{\sigma_2} D(\Psi_1)$; that is, Ψ_2 is a 2-close homomorphic computational simulator of Ψ_1.

Return to the grammars G_1 and G_2. Quite intuitively, the 2-closeness of their derivations means that the grammars generate their sentential forms in a very similar way. Indeed, while G_1 inserts new occurences of symbols a and b in one derivation step, G_2 does the same in two steps.

Example 4. Consider G_1 from Example 3. Let us demonstrate that the following grammar, G_3, homomorphically simulates G_1, but the closeness of this simulation is not limited by any number.

$$\begin{array}{rcl} G_3 & = & (V_3, \{a,b\}, P_3, S), \text{ where} \\ V_3 & = & \{S, M, A, B, X, Z, a, b\}, \\ P_3 & = & \{S \to ZXMXZ,\ ZA \to ZXa,\ BZ \to bXZ, \\ & & Xa \to aX,\ bX \to Xb,\ XMX \to AMB,\ XMX \to AB, \\ & & aA \to Aa,\ Bb \to bB,\ ZA \to a,\ BZ \to b\}; \end{array}$$

Introduce a string-relation system $\Psi_3 = (V_3^*, \Rightarrow_{G_3}, \{S\}, \{a,b\}^*)$ and a homomorphism σ_3 from V_3 to V_1 as $\sigma_3(S) = \sigma_3(M) = S$, $\sigma_3(A) = \sigma_3(a) = a$, $\sigma_3(B) = \sigma_3(b) = b$, $\sigma_3(X) = \sigma_3(Z) = \varepsilon$. Inspect the definition of P_3 to see that for every derivation step $a^{n-1}Sb^{n-1} \Rightarrow_{G_1} a^nSa^n$, $n \geq 1$, G_3 makes a derivation $ZXa^{n-1}Mb^{n-1}XZ \Rightarrow_{G_3}^{2n-2} Za^{n-1}XMXb^{n-1}Z \Rightarrow_{G_3} Za^{n-1}AMBb^{n-1}Z \Rightarrow_{G_3}^{2n-2} ZAa^{n-1}Mb^{n-1}BZ \Rightarrow_{G_3}^{2} ZXa^nMb^nXZ$. Analogously, for every $a^{n-1}Sb^{n-1} \Rightarrow_{G_1} a^nb^n$, $n > 0$, there is $ZXa^{n-1}Mb^{n-1}XZ \Rightarrow_{G_3}^{2n-2} Za^{n-1}XMXb^{n-1}Z \Rightarrow_{G_3} Za^{n-1}ABb^{n-1}Z \Rightarrow_{G_3}^{2n-2} ZAa^{n-1}Mb^{n-1}BZ \Rightarrow_{G_3}^{2} a^nb^n$ in G_3. Informally, while G_1 inserts new occurences of symbols a and b in the middle of a sentential form, G_3 adds as and bs to the ends of the corresponding sentential form. It is rather easy to prove that if $d \in D(\Psi_1)$, there exists $h \in D(\Psi_3)$ such that $h \triangleright_{\sigma_3} d$. Furthermore, it can also be demonstrated that for every $h \in D(\Psi_3)$, there is some $d \in D(\Psi_1)$ such that $h \triangleright_{\sigma_3} d$. However, observe that G_3 simulates every derivation step of G_1 by a sequence of steps whose number depends on the length of the rewritten sentential form. Therefore, $D(\Psi_3) \triangleright_{\sigma_3} D(\Psi_1)$, but there exists no m satisfying $D(\Psi_3) \triangleright_{\sigma_3}^{m} D(\Psi_1)$.

ACKNOWLEDGEMENTS

The paper has been prepared as a part of the solution of GAČR project No. 201/04/0441 and with the support of the research plan J22/98:262200012.

REFERENCES

[1] Hamburger, H., Richards, D.: Logic and Language Models for Computer Science, Prentice Hall, London, 2002.

[2] Meduna, A.: Automata and Languages: Theory and Applications, Springer, London, 2000.

[3] Meduna, A., Švec, M.: Forbidding ET0L grammars, Theoretical Computer Science 306 (2003), 449–469.

[4] Meduna, A., Švec, M.: Computational Simulation Formalized by String-Relation Systems: General Concept and its Illustration in Terms of Lindenmayer Grammars, submitted.

[5] Parkes, A.: Introduction to Languages, Machines, and Logic: Computable Languages, Abstract Machines, and Formal Logic, Springer, London, 2002.