RAIRO-Theor. Inf. Appl. 45 (2011) 77–97
DOI: 10.1051/ita/2011012

Available online at:
www.rairo-ita.org

CONSENSUAL LANGUAGES AND MATCHING FINITE-STATE COMPUTATIONS [*], [**]

Stefano Crespi Reghizzi[1] and Pierluigi San Pietro[1]

Abstract. An ever present, common sense idea in language modelling research is that, for a word to be a valid phrase, it should comply with multiple constraints at once. A new language definition model is studied, based on agreement or consensus between similar strings. Considering a regular set of strings over a bipartite alphabet made by pairs of unmarked/marked symbols, a match relation is introduced, in order to specify when such strings agree. Then a regular set over the bipartite alphabet can be interpreted as specifying another language over the unmarked alphabet, called the consensual language. A word is in the consensual language if a set of corresponding matching strings is in the original language. The family thus defined includes the regular languages and also interesting non-semilinear ones. The word problem can be solved in NLOGSPACE, hence in P time. The emptiness problem is undecidable. Closure properties are proved for intersection with regular sets and inverse alphabetical homomorphism. Several conditions for a consensual definition to yield a regular language are presented, and it is shown that the size of a consensual specification of regular languages can be in a logarithmic ratio with respect to a DFA. The family is incomparable with context-free and tree-adjoining grammar families.

Mathematics Subject Classification. 68Q45, 68Q42, 68Q19.

Keywords and phrases. Formal languages, finite automata, consensual languages, counter machines, polynomial time parsing, non-semilinear languages, Parikh mapping, descriptive complexity of regular languages, degree of grammaticality.

[*] *With partial support from PRIN 2005015419, FIRB "Applicazioni della Teoria degli Automi all'Analisi, Compilazione e Verifica di Software Critico e in Tempo Reale", and CNR-IEIIT.*

[**] *Preliminary, partial versions were presented at LATA 2008 [2] and ICTCS 2009 [3] conferences.*

[1] Dipartimento di Elettronica e Informazione, Politecnico di Milano, Piazza Leonardo da Vinci, 32, 20133 Milano, Italy; {crespi;sanpietro}@elet.polimi.it

Article published by EDP Sciences © EDP Sciences 2011

Introduction

An ever present, common sense idea in language modelling research is that, for a word to to be a valid phrase, it should comply with multiple constraints at once. Theories of grammar have taken various approaches for expressing the constraints by different mechanisms, such as by superimposing semantic constraints to syntactic ones, or by using intersections of, say, context-free languages. Of course, motivation for language definitions, based on agreement or reinforcement between separate processes, comes from the overwhelming complexity of monolithic definitions, and, in the case of natural language, is supported by the findings of neuro-linguistical research.

Here we propose a very simple novel mechanism, where the constraints are expressed by an elementary letter by letter agreement between strings belonging to a regular language. The alphabet is bipartite, made by pairs of unmarked/marked characters. The agreement is formalized by a k-ary relation, called match, that is satisfied by a set of k equally long strings if, in each position, exactly one word has an unmarked letter and the other strings have the same letter but marked. In our metaphor we view such strings as providing mutual consensus on the validity of the corresponding unmarked string. This justifies the name "consensual" proposed for the new family, which strictly includes the regular one.

Here some reader may prefer to jump to the definition (Defs. 1.1, 1.3 and 1.4) of consensual language, before reading the next discussion of the position of the new model from the perspective of language theory.

With respect to their storage, abstract language recognition devices can be classified as using tapes (Turing machines, push-down machines, nested push-down machines) or counters. The latter case includes various models of counter machines and also Petri Nets. Consensual languages are recognized by real-time non-deterministic multi counter machines with a linear bound on the counter values.

Considering the complexity of the word recognition problem, consensual languages belong to the polynomial time class.

With respect to generative capacity, the new family shares little ground with the families of context-free and mildly context-sensitive [8] languages. For instance, the Dyck language over two letters can be defined but not the language of palindromes. On the other hand interesting non-semilinear languages (in the Parikh sense [7]) can be easily defined.

Next we compare and contrast the computation performed by a consensual recognizer versus an alternating finite automaton [1]. Although both machines perform simultaneous computations for recognizing a given string, they apply entirely different acceptance criteria. All possible computations must be successful for a word to be recognized by an alternating machine when using universal non-determinism, and their number may be exponential with respect to the word length. On a consensual device, the computations performed on the finite automaton, which can be assumed to be deterministic, are not labelled by the input word (except in the trivial case when the language is regular) but by matching strings over the marked/unmarked alphabet. The number of computations is bounded by the input length.

Recalling that certain Petri net language families [4] include non-semilinear languages and that their recognizers use counters, a vague resemblance between the two models may be mentioned. In fact C.A. Petri introduced his nets as a formal model of synchronization between computations performed by finite automata and our model too specifies a matching rule between the labels of separate computations.

Notwithstanding the fact that the proposed approach has little to do with any classical formal language model we know, we hope its simplicity, expressivity and motivation may attract some attention.

The paper is organized as follows. Section 1 lists the basic definitions, and provides an example giving evidence of the strict inclusion of regular languages. Section 2 shows that the Parikh image may be not linear, and proves several closure properties. Section 3 defines a transition relation between multisets of states, corresponding to a multi-counter machine. Then it shows that the word recognition problem is in NLOGSPACE. Section 4 focuses on consensually defined regular languages, and gives sufficient conditions for a consensual language to be regular. It shows that consensual definitions can be exponentially more concise than definitions by deterministic finite automata. Section 5 proves the emptiness problem to be undecidable. Section 6 shows that the languages of palindromes and replicas exceed the power of consensual languages. The conclusion mentions directions for continuation.

1. First definitions

Let Σ be the *terminal* alphabet of the languages to be considered. The empty word is denoted by letter ϵ. Given a word x, its length is denoted by $|x|$ and the i-th letter is $x(i)$, $1 \leq i \leq |x|$. A deterministic finite automaton (DFA for short) is specified as $A = (\Delta, Q, \delta, q_0, F)$ where: Δ is a finite alphabet; Q is a finite set of *states*; $\delta : Q \times \Delta \rightarrow Q$ is the state-transition function, always assumed to be *total*; q_0 is the initial state, and $F \subseteq Q$ is the set of final states. The transition function δ can be extended as usual to $Q \times \Delta^* \rightarrow Q$, which is also total, *i.e.*, $\delta(q_0, y)$ is defined for every word y over Σ. A nondeterministic finite automaton N (NFA) is specified as $N = (\Delta, Q, \rightrightarrows_N, q_0, F)$ where the only difference with a DFA above is that the transition relation is $\rightrightarrows_N \subseteq Q \times \Delta \times Q$. Acceptance may be defined as usual for a DFA and for a NFA.

Let $\underline{\Sigma}$ be the disjoint alphabet obtained by marking each symbol $a \in \Sigma$ as $\underline{a}$, referred to as the *marked copy* of a.

The set $\Sigma \cup \underline{\Sigma}$ is denoted as $\widetilde{\Sigma}$ and qualified as *internal* alphabet, because its use is restricted to the technical device of consensual definitions.

The notion of agreement between strings over the internal alphabet is formalized by means of a function called match.

Definition 1.1. Match
The partial, symmetrical, and associative binary operator, called *match*

$$@ : \widetilde{\Sigma} \times \widetilde{\Sigma} \to \widetilde{\Sigma}$$

is defined as follows, for all $a \in \Sigma$:

$$\begin{cases} a@\underline{a} = \underline{a}@a = a; \\ \underline{a}@\underline{a} = \underline{a}; \\ \text{undefined}, & \text{in every other case.} \end{cases}$$

The operator can be naturally extended to strings of equal length, by assuming $\epsilon@\epsilon = \epsilon$. For all $w, w' \in \widetilde{\Sigma}^*$, with $|w| = |w'|$, and for all $a, b \in \widetilde{\Sigma}$

$$aw \; @ \; bw' = (a@b)(w@w')$$

where we assume that match yields precedence to concatenation.

Hence, the match is undefined on strings w, w' of unequal lengths, or else if there exists a position i such that $w(i)@w'(i)$ is undefined. The latter condition occurs in three cases: when both characters are in Σ, when both are in $\underline{\Sigma}$ and differ, and when either one is marked but is not the marked copy of the other.

Given $m > 0$ strings $w_1, \dots, w_m \in \widetilde{\Sigma}^*$, consider $w_1@w_2@\dots@w_m$ (which can be written without parentheses and in any order because the match operation is associative and commutative). If $w = w_1@w_2@\dots@w_m$ is defined then w is called the *match* of $w_1, w_2, \dots, w_m$. The match is *strong* if $w \in \Sigma^*$, *weak* otherwise. The cardinality m is called the *degree* of the match. Match w and every argument w_j have the same length $n = |w_j| = |w|$. Also, by Definition 1.1, if w is a strong match for each position $1 \leq i \leq n$, exactly one string, say w_k, is unmarked, *i.e.*, $w_k(i) \in \Sigma$ and $w_j(i) \in \underline{\Sigma}$ for all $j \neq k$. We say that word w_k *places* the letter into position i and the other strings *consent* to it.

Next we extend the match operator to two languages $L', L'' \subseteq \widetilde{\Sigma}^*$ over the internal alphabet:

$$L'@L'' = \{w'@w'' \mid w' \in L', w'' \in L''\}.$$

Clearly the operation may be applied to any number of languages.

If the arguments are regular languages, the match operator produces a regular language.

Proposition 1.2. *If $L', L'' \subseteq \widetilde{\Sigma}^*$ are regular languages then $L'@L''$ is also regular.*

Proof. Let $A' = (\widetilde{\Sigma}, Q', \delta', q_1', F')$ and $A'' = (\widetilde{\Sigma}, Q'', \delta'', q_1'', F'')$ be the DFAs' recognizing L', L'', respectively. Let $A'@A''$ be the (possibly nondeterministic) finite automaton $(\widetilde{\Sigma}, Q' \times Q'', \delta, (q_1', q_1''), F' \times F'')$, with $\delta : (Q' \times Q'') \times \widetilde{\Sigma} \to 2^{Q' \times Q''}$, such that for every $q', p' \in Q', q'', p'' \in Q''$, for every $a \in \Sigma$:

$$\begin{array}{lll} \langle p', p'' \rangle \in \delta(\langle q', q'' \rangle, a) & \text{if} & p' = \delta'(q', a), p'' = \delta''(q'', \underline{a}) \\ \langle p', p'' \rangle \in \delta(\langle q', q'' \rangle, a) & \text{if} & p' = \delta'(q', \underline{a}), p'' = \delta''(q'', a) \\ \langle p', p'' \rangle \in \delta(\langle q', q'' \rangle, \underline{a}) & \text{if} & p' = \delta'(q', \underline{a}), p'' = \delta''(q'', \underline{a}). \end{array}$$

The construction is similar to the usual Cartesian product of two DFAs'. But instead of the intersection, the product machine $A'@A''$ recognizes $L'@L''$, because the construction has been modified to match a with $\underline{a}$ and $\underline{a}$ with $\underline{a}$, but not to match a with a. □

The repeated application of the match operation to a language is formalized next. Let $L^{1@} = L$, $L^{i@} = L@L^{(i-1)@}$, $i \geq 2$. Notice that in general $L^{(i-1)@} \not\subseteq L^{i@}$.

Definition 1.3. Match closure
The *closure under match*, or @-closure, of a language $L \subseteq \widetilde{\Sigma}^*$ is:

$$L^@ = \bigcup_{i \geq 1} L^{i@}.$$

Focusing on languages over the terminal alphabet Σ, the main definition comes next.

Definition 1.4. Consensual language
Let B be in a language family $\mathcal{F}$. The *consensual language with base* $B \subseteq \widetilde{\Sigma}^+$ is the set

$$\mathcal{C}(B) = B^@ \cap \Sigma^*.$$

Language $\mathcal{C}(B)$ is also called a consensual language *based on* family $\mathcal{F}$, and the corresponding family is written $\mathcal{C}_{\mathcal{F}}$.

Therefore, a consensual language with base B includes all and only the strongly matches of the match closure. In this paper we study the family of consensual languages based on the family of regular languages, $\mathcal{C}_{\mathcal{REG}}$.

Example 1.5. Consider the regular language R defined by the regular expression

$$\underline{a}^* a \underline{a}^* \underline{b}^* b \underline{b}^*.$$

Then $R^@$ is the set of strings of the form:

$$\underline{a}^* a_1 \, \underline{a}^* a_2 \, \underline{a}^* \dots a_m \, \underline{b}^* b_1 \, \underline{b}^* b_2 \, \dots b_m \, \underline{b}^*$$

where $m \geq 1$, and each a_i is a and each b_i is b.

The consensual language with base R is $\mathcal{C}(R) = \{a^n b^n \mid n > 0\}$. Figure 1 shows that sentence $aaabbb$ can be obtained in different ways, matching together the strings of R in column v_i, or matching those in column w_i. Notice that every sentence w is obtained by means of a match of degree $|w|/2$.

i	v_i	w_i
1	$\underline{a}a\underline{a}b\underline{bb}$	$a\underline{aab}bb$
2	$a\underline{aabb}b$	$\underline{a}a\underline{a}bb\underline{b}$
3	$\underline{aa}a\underline{b}b\underline{b}$	$\underline{aa}ab\underline{bb}$
Match	$aaabbb$	$aaabbb$

Figure 1. In Example 1.5 word $aaabbb$ results from the strong matches in column v_i and w_i.

The example has shown that, although the base B is regular, languages $B^{@}$ and $\mathcal{C}(B)$, obtained by a match closure, may be non-regular.

However, from Proposition 1.2, for any finite i, $B^{i@}$ is regular if B is regular. This corresponds, in Definition 1.3, to the case where at most i strings $w_1, \ldots, w_i$ are matched.

2. First properties

We introduce further useful terminology and make intuitive comments about previous definitions and concepts.

First, we notice that we may remove or add to a base language B a subset of $\underline{\Sigma}^+$ without affecting the corresponding consensual language $\mathcal{C}(B)$. In fact, if $w@w'$ is defined and $w' \in \underline{\Sigma}^+$, then $w@w' = w$: strings purely made of marked characters are both useless and "harmless".

Proposition 2.1. *For every base language $B \subseteq \widetilde{\Sigma}^*$, and for every language $U \subseteq \underline{\Sigma}^+$,*

$$\mathcal{C}(B) = \mathcal{C}(B \cup U) = \mathcal{C}(B \setminus U).$$

As a consequence of the fact that the match of identical strings is undefined (if they contain at least one unmarked character) or useless (if the strings are completely marked), any phrase w of a consensual language can be obtained as the result of a strong match having degree not exceeding its length $|w|$.

Proposition 2.2.

$$\mathcal{C}(B) = \{w \in \Sigma^* \mid \exists k, 1 \leq k \leq |w|, w \in B^{k@}\}. \qquad (1)$$

A straightforward language family inclusion result comes next. Consider a deterministic finite automaton of the base language B. A word w is in the consensual language $\mathcal{C}(B)$, if, and only if, the automaton performs $1 \leq k \leq |w|$ successful computations, accepting a set of strings over $\widetilde{\Sigma}$ that strongly match to w. We also say that such computations strongly (or weakly) match.

The case $k = 1$ clearly corresponds to the usual recognition condition of a DFA. As the consensual language of Example 1.5 is not regular, we have:

Proposition 2.3. *The family $\mathcal{C}_{\mathcal{REG}}$ of consensual languages on a regular language base strictly includes the family of regular languages.*

The next example shows languages having a non-semilinear commutative image (in Parikh's sense [7]).

Example 2.4. Series of unary integers

(1) Series of identical unary integers.
Choose as base the language:

$$R_1 = (\underline{a}^* a \underline{a}^* \underline{b})^+ \cup (\underline{a}^+ b)^+.$$

Then the consensual language is:

$$L_1 = \mathcal{C}(R_1) = \{a^n b a^n b a^n b \dots a^n b \mid n > 0\}.$$

(2) Enumeration of unary integers.
The language $L_2 = \{baba^2b \dots ba^n b \mid n \geq 0\}$ is consensually defined by the regular base

$$R_2 = \left(\underline{ba}^+\right)^* b \left(\underline{a}^* a \underline{a}^* \underline{b}\right)^* .$$

For example, $babaab$ is the match of the following words in R_2: $ba\underline{b}a\underline{ab}$, $\underline{ba}b\underline{a}a\underline{b}$ and $\underline{babaa}b$.

(3) Series of exponential unary numbers.
For $\Sigma = \{a, b, c\}$ let

$$R_3 = \underline{\Sigma}^* a \left(\underline{a} \cup \underline{c}\right)^* \underline{b} \left(\underline{a} \cup \underline{c}\right)^* c\underline{ac}\underline{\Sigma}^* \ \cup \ \underline{a}cb \left(\left(\underline{ac}\right)^+ b\right)^* \left(a\underline{c}\right)^+ b \ \cup \ acb.$$

The consensual language $\mathcal{C}(R_3)$ is

$$L_3 = \{ac\, b(ac)^2 b(ac)^4 b(ac)^8 b \dots (ac)^{2^m} b \mid m \geq 0\}.$$

Proof. Let us express the base language as the union of three clauses, numbered 1, 2, and 3:

$$R_3 = \underbrace{\underline{\Sigma}^* a(\underline{a} \cup \underline{c})^* \underline{b} \left(\underline{a} \cup \underline{c}\right)^* c\underline{ac}\underline{\Sigma}^*}_{1} \cup \underbrace{\underline{a}cb((\underline{ac})^+ b)^* (a\underline{c})^+ b}_{2} \cup \underbrace{acb}_{3}.$$

First we show that $L_3 \subseteq \mathcal{C}(R_3)$. Any word in $\mathcal{C}(R_3)$, apart from acb, must be obtained by matching a word in clause 2 with strings in clause 1, since neither regular expression can generate alone a word in Σ^+. We now show, by induction on the number $m \geq 1$ that if w_m is a string of the form

$$w_m = acb(ac)^2b(ac)^4b(ac)^8b\dots(ac)^{2^m}b$$

then $w_m \in \mathcal{C}(R_3)$ (and clearly L_3 is the set of all w_m above). The base step is $m = 0$, corresponding to the word acb, which is both in $\mathcal{C}(R_3)$ and in L_3. Assume now that the induction hypothesis holds for $m-1$. Hence, string

$$w_{m-1} = acb(ac)^2b(ac)^4b(ac)^8b\dots(ac)^{2^{m-1}}b \in \mathcal{C}(R_3).$$

But w_{m-1} must be obtained as a match of $h > 0$ strings $x_1, x_2, \dots x_h$ of clause 1, with one string

$$y_{m-1} = \underline{a}cb(\underline{ac})^2b\dots(\underline{ac})^{2^{m-2}}b(a\underline{c})^{2^{m-1}}b$$

of clause 2. But also

$$y_m = \underline{a}cb(\underline{ac})^2b\dots\underline{b}(\underline{ac})^{2^{m-1}}b(a\underline{c})^{2^m}b$$

is in clause 2, and if x_i is in clause 1 then also $x'_i = x_i(\underline{ac})^{2^m}\underline{b}$ is in clause 1, since clause 1 ends with $\underline{\Sigma}^*$. Therefore, by matching y_m with $x'_1, \dots, x'_m$, one obtains:

$$w'_m = acb(ac)^2b(ac)^4b(ac)^8b\dots b(ac)^{2^{m-2}}b(\underline{a}c)^{2^{m-1}}b(a\underline{c})^{2^m}b \in R_3^{@}.$$

Also, all strings

$$z_i = \underline{acb}(\underline{ac})^2\underline{b}\dots\underline{b}(\underline{ac})^{2^{m-2}}\underline{b}(\underline{ac})^i a\underline{c}(\underline{ac})^{2^{m-1}-i-1}\underline{b}(\underline{ac})^{2i}\underline{a}c\underline{a}c(\underline{ac})^{2^m-2i-2}\underline{b}$$

are in clause 1, for every i, $0 \leq i < 2^{m-1}$. Hence, for every a placed in group $m-1$ (the group with 2^{m-1} occurrences of ac), there must be two occurrences of c in group m. Hence, the number of c's (and therefore also of a's) in group m must be twice the number of c's in group $m-1$:

$$w_m = acb(ac)^2b(ac)^4b(ac)^8b\dots(ac)^{2^{m-1}}b(ac)^{2^m}b \in \mathcal{C}(R_3).$$

For the converse case, notice that, since any w in $\mathcal{C}(R_3)$ (except for acb) must match a word in clause 2, $\mathcal{C}(R_3) \subseteq acb\left((ac)^+b\right)^*$. An induction on the number $m \geq 0$ of groups $(ac)^+b$ in strings of $\mathcal{C}(R_3)$ completes the proof. The base case $m = 0$ corresponds to the word acb. By induction hypothesis, for all $0 \leq j \leq m$, the strings $acb\left((ac)^+b\right)^j$ are in L_3. Assume that the word x_m with m groups is not in L_3. Hence, there exists $i > 0$ such that the group in position i has a number of c which is not the double of the number of a of the group in position $i-1$. But $i = m$, otherwise one could define also a word which is not in L_3 while having less than m groups, contradicting the induction hypothesis. However, the only way to place a c in group m is by using strings in clause 1, which place two occurrences of c in group m for an occurrence of a in group m-1. Since no other strings can place a in group $m-1$, then the number of c's in group m must be exactly the double of the number of a in group $m-1$ (2^{m-1} by induction hypothesis), that is there are 2^m occurrences of c in group m. □

We state the basic closure properties of consensual languages in the next proposition. For two languages $L', L'' \subseteq \Sigma^*$ and a letter $s \notin \Sigma$, the *marked concatenation* [7] is the language $L'sL''$.

Proposition 2.5. *The family $\mathcal{C}_{\mathcal{REG}}$ is closed under:*

(1) *intersection with regular languages;*
(2) *inverse alphabetic homomorphism;*
(3) *reversal (or mirror reflection) operation;*
(4) *marked concatenation of consensual languages;*
(5) *union of consensual languages over disjoint alphabets.*

Proof. We separately argue for each statement.

(1) Let $R \subseteq \widetilde{\Sigma}^*, S \subseteq \Sigma^*$ be two regular languages, and let $h : \widetilde{\Sigma} \to \Sigma$ be the alphabetic homomorphism defined by $h(a) = h(\underline{a}) = a$ for every $a \in \Sigma$. We claim that
$$\mathcal{C}(R) \cap S = \mathcal{C}\left(R \cap h^{-1}(S)\right)$$
thus proving the statement.

Let $x \in \mathcal{C}(R) \cap S$. Therefore, $\exists k, 1 \le k \le |x|$, $\exists x_1, \dots, x_k \in R$ such that $x_1 @ x_2 \dots @ x_k = x$ and for every i, $1 \le i \le k$, $h(x_i) = x$. Hence, every $x_i \in h^{-1}(x) \subseteq h^{-1}(S)$ since $x \in S$. Hence, for every $i, 1 \le i \le k$, $x_i \in R \wedge x_i \in h^{-1}(S)$, and it follows that $x \in \mathcal{C}\left(R \cap h^{-1}(S)\right)$.

Assume now $x \in \mathcal{C}\left(R \cap h^{-1}(S)\right)$. Hence, $\exists k, 1 \le k \le |x|$, $\exists x_1, \dots, x_k$ such that $x_1 @ x_2 \dots @ x_k = x$ and for every i, $1 \le i \le k$, $x_i \in R \cap h^{-1}(S)$, with $h(x_i) = x$. Then $x \in \mathcal{C}(R)$ (since each $x_i \in R$). Also, $x \in h^{-1}(S)$ (since each $x_i \in h^{-1}(S)$). Therefore, $x \in S$, since $S = h^{-1}(S) \cap \Sigma^*$ and $x \in \Sigma^*$.

(2) Let $R \subseteq \widetilde{\Sigma}^*$ be a regular language, and let Δ be another finite alphabet. Let $h : \Delta \to \Sigma$ be a homomorphism. We need to prove that $h^{-1}(\mathcal{C}(R))$ is a consensual language with regular base.

Extend first h to the internal alphabet as follows: $\widehat{h} : \Delta \cup \underline{\Delta} \to \Sigma \cup \underline{\Sigma}$ is defined as $\widehat{h}(A) = h(A)$, $\widehat{h}(\underline{A}) = \underline{h(A)}$ for every $A \in \Delta$.

We notice that $\widehat{h}^{-1}(\underline{a} @ \underline{a}) = \widehat{h}^{-1}(\underline{a}) = \widehat{h}^{-1}(\underline{a}) @ \widehat{h}^{-1}(\underline{a})$, and that $\widehat{h}^{-1}$ $(\underline{a} @ a) = \widehat{h}^{-1}(a) = \widehat{h}^{-1}(\underline{a}) @ \widehat{h}^{-1}(a)$, and similarly for the case $a @ \underline{a}$. On the other hand both $\widehat{h}^{-1}(a) @ \widehat{h}^{-1}(a)$ and $\widehat{h}^{-1}(a @ a)$ are undefined.

Hence,

$$\widehat{h}^{-1}(X@Y) = \widehat{h}^{-1}(X)@\widehat{h}^{-1}(Y) \text{ for every } X, Y \in \widetilde{\Sigma}.$$

Therefore, if $u, u' \in (\Sigma \cup \underline{\Sigma})^*$ then $\widehat{h}^{-1}(u@u') = \widehat{h}^{-1}(u)@\widehat{h}^{-1}(u')$.

We now claim that $\widehat{h}^{-1}(R^@) = \left(\widehat{h}^{-1}(R)\right)^@$. From here the thesis follows, since $\widehat{h}^{-1}(R)$ is regular and $h^{-1}(\mathcal{C}(R)) = \widehat{h}^{-1}(R^@) \cap \Delta^* = \left(\widehat{h}^{-1}(R)\right)^@ \cap \Delta^*$. Let $x \in \widehat{h}^{-1}(R^@)$. Hence, there is $w \in R^@$ such that $x \in h^{-1}(w)$. By Definition 1.4, there exist $k > 0$ strings $w_1, \ldots, w_k \in R$, with $1 \le k \le |x|$, such that $w_1@\ldots@w_k = w$. Hence, $x \in \widehat{h}^{-1}(w) = \widehat{h}^{-1}(w_1)@\ldots@\widehat{h}^{-1}(w_k) \subseteq \left(\widehat{h}^{-1}(R)\right)^@$. Let $x \in \left(\widehat{h}^{-1}(R)\right)^@$. Hence, there exist $k > 0$ strings $x_1, \ldots, x_k \in \widehat{h}^{-1}(R)$ such that $x_1@\ldots@x_k = x$.

It follows that there exist $k > 0$ strings $w_1, \ldots, w_k \in R$ such that $x_1 \in \widehat{h}^{-1}(w_1), \ldots, x_k \in \widehat{h}^{-1}(w_k)$, and therefore

$$x = x_1@\ldots@x_k \in \widehat{h}^{-1}(w_1)@\ldots@\widehat{h}^{-1}(w_k)$$

$$= \widehat{h}^{-1}(w_1@\ldots@w_k) \subseteq \widehat{h}^{-1}(R^@).$$

(3)–(5) The obvious proofs are based on simple transformations of the DFA recognizing the base language. □

3. Consensual Languages are in NLOGSPACE

In this section, we formalize the simultaneous computations in the consensual definition by means of multisets of states of a DFA accepting the base language; the multiplicity of a state in the multiset encodes the number of computations of the DFA that have reached that state. The consensual transition relation can be computed by a nondetermistic Turing machine. A multiset can be represented by multiplicity counters and only one counter for each state of the DFA is needed, whose value is linearly limited by the length of the input string. Using a binary encoding of each counter, word membership can be computed by a nondeterministic counter machine operating in logarithmic space.

3.1. Consensual transition relation

3.1.1. *Preliminaries on multisets*

Given a finite set Q, which in this paper is the set of states of a DFA, a *multiset* over Q is a total mapping $Z : Q \to \mathbb{N}$. The cardinality of multiset Z is $|Z| = \sum_{q \in Q} Z(q)$. For $q \in Q$, if $Z(q) > 0$ then we say that $q \in Z$ with *multiplicity* $Z(q)$. To illustrate, consider the multiset Z over $Q = \{p, q, r\}$ characterized by $Z(p) = 3, Z(q) = 0, Z(r) = 5$. We also use the alternative notations $\{p^3, r^5\}$ or $\{p, p, p, r, r, r, r, r\}$.

Given two multisets Z, Z' over Q, the *sum* $Z \uplus Z'$ and the *difference* $Z - Z'$ are the multisets specified by the following characteristic functions, for every $q \in Q$:

$$(Z \uplus Z')\,(q) = Z(q) + Z'(q), \quad (Z - Z')\,(q) = \max\left(0, Z\,(q) - Z'\,(q)\right).$$

If $f : Q \to \mathbb{N}^Q$ is a total mapping, associating each element $q \in Q$ with a multiset $f(q)$ and $Z : Q \to \mathbb{N}$ is a multiset $\{q_1, \ldots, q_m\}$, where $m = |Z|$ and the q_i's are not necessarily distinct, then let the *generalized sum* $\biguplus_{q \in Z} f(q)$ be $f(q_1) \uplus \cdots \uplus f(q_m)$.

Finally, define, for every multiset Z over Q, the *underlying set*

$$[\![Z]\!] = \{q \in Q \mid Z(q) > 0\}\,.$$

Clearly, $[\![Z \uplus Z']\!] = [\![Z]\!] \cup [\![Z']\!]$, $[\![\biguplus_{q \in Z} f(q)]\!] = \bigcup_{q \in [\![Z]\!]} [\![f(q)]\!]$.

3.1.2. *A consensual transition relation*

Let $A = (\widetilde{\Sigma}, Q, \delta, q_0, F)$ be a DFA and assume the transition function δ to be total. By the above notation, the function is naturally extended to a multiset Z over Q, positing $\delta(Z, a) = \biguplus_{q \in Z} \{\delta(q, a)\}$.

From this we define a transition relation on multisets of states.

Definition 3.1. The *consensual transition relation* of A, namely $\rightsquigarrow_A \subseteq \mathbb{N}^Q \times \Sigma \times \mathbb{N}^Q$, defined, for every $a \in \Sigma$ and for all multisets Z, Z' over Q as:

$$Z \overset{a}{\rightsquigarrow}_A Z' \text{ if } \exists q \in Z : \ Z' = \{\delta(q, a)\} \uplus \delta(Z - \{q\}, \underline{a})\,.$$

Relation $\overset{a}{\rightsquigarrow}_A$ can be extended as usual from a letter a to a word $w \in \Sigma^*$ *via* the inductive definition:

$$\begin{cases} Z \overset{\epsilon}{\rightsquigarrow}_A Z \\ Z \overset{wa}{\rightsquigarrow}_A Z'', \quad \text{if } \exists Z' \text{such that } Z \overset{w}{\rightsquigarrow}_A Z' \overset{a}{\rightsquigarrow}_A Z''. \end{cases}$$

It is evident that if $Z \overset{a}{\rightsquigarrow}_A Z'$ then $|Z| = |Z''|$, *i.e.*, the cardinality does not change.

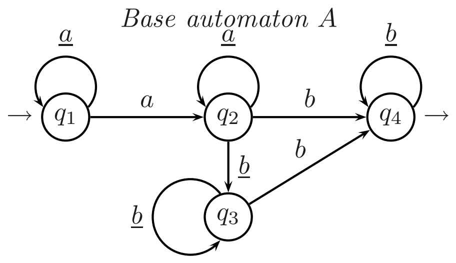
Two types of multisets have a special role: the *initial* multisets $\{(q_0)^k\}$, for every $k > 0$, and the *final* multisets Z such that $[\![Z]\!] \subseteq F$.

A crisp definition of consensual languages is obtained by means of the transition relation.

Proposition 3.2. *Let* $R \subseteq \widetilde{\Sigma}^*$ *and let* $A = (\widetilde{\Sigma}, Q, \delta, q_0, F)$ *be a DFA accepting* R. *Then*

$$\mathcal{C}(R) = \{w \mid \exists k > 0 \textit{ and a final multiset } Z \textit{ such that } \{(q_0)^k\} \overset{w}{\rightsquigarrow}_A Z\}.$$

The proposition follows immediately by combining the following Lemmata 3.4, 3.5. First, an example may help to clarify the construction.



Consensual transition relation accepting aaabbb:

$$\{q_1, q_1, q_1\} \overset{a}{\rightsquigarrow}_A \{q_1, q_1, q_2\} \overset{a}{\rightsquigarrow}_A \{q_1, q_2, q_2\} \overset{a}{\rightsquigarrow}_A \{q_2, q_2, q_2\}$$
$$\overset{b}{\rightsquigarrow}_A \{q_3, q_3, q_4\} \overset{b}{\rightsquigarrow}_A \{q_3, q_4, q_4\} \overset{b}{\rightsquigarrow}_A \{q_4, q_4, q_4\}$$

FIGURE 2. Recognizer of base language $R = \underline{a}^* a \underline{a}^* \underline{b}^* b \underline{b}^*$.

Example 3.3. The finite automaton accepting base language $R = \underline{a}^* a \underline{a}^* \underline{b}^* b \underline{b}^*$ (see Ex. 1.5) is shown in the top part of Figure 2, while the consensual transition relation accepting $aaabbb$ is shown in the bottom part.

Lemma 3.4. *If $\exists k > 0, \exists Z : Q \to \mathbb{N}$ such that $\{(q_0)^k\} \overset{w}{\rightsquigarrow}_A Z$ then there exist k words $w_1, \ldots, w_k \in \tilde{\Sigma}^*$ such that $w_1 @ w_2 @ \ldots @ w_k = w$ and $Z = \biguplus_{1 \le j \le k} \{\delta(q_0, w_j)\}$.*

Proof. The proof is by induction on $|w|$. If $|w| = 0$, then let $w_1 = \cdots = w_k = w = \epsilon$ and let $Z = \{(q_0)^k\}$. If $w > 0$ let $w = w'a$ for $w' \in \Sigma^*, a \in \Sigma$. Hence, if $\{(q_0)^k\} \overset{w}{\rightsquigarrow}_A Z$ then there exist $Z' : Q \to \mathbb{N}$ such that $\{(q_0)^k\} \overset{w'}{\rightsquigarrow}_A Z' \overset{a}{\rightsquigarrow}_A Z$. By induction hypothesis, there exist k words $w'_1, \ldots, w'_k \in \tilde{\Sigma}^*$ with $w'_1 @ \ldots @ w'_k = w'$, $Z' = \biguplus_{1 \le j \le k} \{\delta(q_0, w'_j)\}$.

Since $Z' \overset{a}{\rightsquigarrow}_A Z$, $\exists q \in Z'$ such that $Z = \{\delta(q, a)\} \uplus \delta(Z' - \{q\}, \underline{a})$. Since $q \in Z'$, $\exists h$, $1 \le h \le k$, such that $\delta(q_0, w'_h) = q$. Hence, $\delta(q, a) = \delta(q_0, w'_h a)$. Let $w_h = w'_h a$ and, for every $j \ne h$, $1 \le j \le k$, let $w_j = w'_j \underline{a}$. Hence, $w_1 @ \ldots @ w_k = w$. Also:

$$\begin{aligned} Z = \; & \{\delta(q, a)\} \uplus \delta(Z' - \{q\}, \underline{a}) = \\ & \{\delta(q_0, w'_h a)\} \uplus \delta\left(\biguplus_{1 \le j \le k, j \ne h} \{\delta(q_0, w'_j)\}, \underline{a}\right) = \\ & \{\delta(\delta(q_0, w'_h), a)\} \uplus \biguplus_{1 \le j \le k, j \ne h} \{\delta(\delta(q_0, w'_j), \underline{a})\} = \\ & \{\delta(q_0, w'_h a)\} \uplus \biguplus_{1 \le j \le k, j \ne h} \{\delta(q_0, w'_j \underline{a})\} = \biguplus_{1 \le j \le k} \{\delta(q_0, w_j)\}. \end{aligned}$$ □

Lemma 3.5. *For every $w \in \Sigma^*$, for every $k > 0$, let $w_1, \ldots, w_k \in \tilde{\Sigma}^*$ be such that $w_1 @ w_2 @ \ldots @ w_k = w$, and let $Z : Q \to \mathbb{N}$ be the multiset $Z = \biguplus_{1 \le j \le k} \{\delta(q_0, w_j\})$. Then, $\{(q_0)^k\} \overset{w}{\rightsquigarrow}_A Z$.*

Proof. The proof is by induction on $|w|$. If $|w| = 0$, then let $w_1 = \cdots = w_k = w = \epsilon$ and let $Z = \{(q_0)^k\}$: by definition, $\{(q_0)^k\} \overset{\epsilon}{\rightsquigarrow}_A Z$. If $w > 0$ let $w = w'a$

for $w' \in \Sigma^*, a \in \Sigma$. Hence, since $w_1 @ w_2 @ \ldots @ w_k = w$, there exists h, $1 \leq h \leq k$, and there exist $w'_1, \ldots, w'_k \in \tilde{\Sigma}^*$ such that $w_h = w'_h a$, $w_j = w'_j \underline{a}$ for $j \neq h$. Let $Z' = \biguplus_{1 \leq j \leq k} \{\delta(q_0, w'_j)\}$. Since $w_1 @ w_2 @ \ldots @ w_k = w$, by induction hypothesis $\{(q_0)^k\} \overset{w'}{\rightsquigarrow}_A Z'$. Let $Z = \biguplus_{1 \leq j \leq k} \{\delta(q_0, w_j)\}$ Hence, $Z = \{\delta(q_0, w'_h a)\} \uplus \biguplus_{1 \leq j \leq k, j \neq h} \{\delta(q_0, w'_j \underline{a})\} = \{\delta(\delta(q_0, w'_h), a)\} \uplus \delta\left(\biguplus_{1 \leq j \leq k, j \neq h} \{\delta(q_0, w'_j)\}, \underline{a}\right)$. Let $q = \delta(q_0, w'_h) \in Z'$. Then $Z = \{\delta(q, a)\} \uplus \delta(Z' - \{q\}, \underline{a})$. By Definition 3.1, $Z' \overset{a}{\rightsquigarrow}_A Z$, hence $\{(q_0)^k\} \overset{w}{\rightsquigarrow}_A Z$. □

Given Proposition 3.2, the NLOGSPACE complexity of consensual languages follows almost immediately:

Theorem 3.6. *The word membership problem for the family $\mathcal{C}_{\mathcal{REG}}$ is in the complexity class NLOGSPACE (hence in P).*

Proof. Clearly, the transition relation $\rightsquigarrow_A$ can be computed by a nondeterministic Turing machine that, operating on an input word $w \in \Sigma^*$ of length $n \geq 0$, guesses k and first stores $\{(q_0)^k\}$ and then makes a nonderministic move computing $\overset{a}{\rightsquigarrow}_A$ when reading each input symbol $a \in \Sigma$.

Notice that one can always assume that $k \leq n$, since a match with at most n words is enough to determine w. In a computation with input length n, the multiset to be stored at each step has cardinality exactly $k \leq n$. Hence, the space for storing a multiset Z over Q is logarithmic in n: a multiset can be represented by its characteristic function, which in this case requires a counter for each state in Q, and hence, it is enough to use $|Q| \lceil log_2 n \rceil$ bits, with $|Q|$ a constant.

Also, a move of the machine to simulate $\overset{a}{\rightsquigarrow}_A$ only requires modifying the counter values. The possible operations on a counter to implement the multiset operations are: add or substract 1, reset to zero and store the result of adding two or more counters. These operations may require additional constant or, at most, logarithmic space. Hence, for each regular base language R, membership in $\mathcal{C}(R)$ is in NLOGSPACE. □

4. Regular consensual languages

We have seen that consensual languages on a regular base may or may not be regular. In this section we provide conditions ensuring regularity and we investigate the descriptive complexity of consensual specifications of regular languages.

4.1. Conditions for regularity

Bounded degree. The first result is obvious: when there is a bound i such that $L^{i@} = L^{(i+1)@}$, *i.e.*, $L^@$ has *bounded degree*, then $L^@$ is regular. However, in general the converse is not true: $L^{i@}$ may be not equal to $L^{(i+1)@}$ even when $L^@$ is regular: the degree of a regular consensual language can be unbounded. For instance, consider the regular expression $R = \underline{a}^* a \underline{a}^*$; then $\mathcal{C}(R) = a^+$ is regular and $R^@$ is $(\underline{a}^* a \underline{a}^*)^+$ while $R^{i@}$ is $(\underline{a}^* a \underline{a}^*)^i$. Hence $R^{i@} \not\subseteq R^{(i+1)@}$.

Maximal degree. We say that a base language R has *maximal degree* if for every word $w \in \mathcal{C}(R)$ there exist $|w|$ distinct strings $w_1, \ldots, w_{|w|} \in R - \underline{\Sigma}^*$ such that $w = w_1 @ \ldots @ w_{|w|}$.

We are going to prove that a consensual language on a regular base, having maximal degree, is regular. The result is non-obvious because in this case the multisets describing the simultaneous matching computations have unbounded cardinality. To simplify the proof, first notice that if R has maximal degree, then we have $\mathcal{C}(R) = \mathcal{C}(R \cap \underline{\Sigma}^* \Sigma \underline{\Sigma}^*)$, since, if $w = w_1 @ \ldots @ w_{|w|}$, then each w_i is in $\underline{\Sigma}^* \Sigma \underline{\Sigma}^*$. Hence, it is enough to show that consensual languages with $R \subseteq \underline{\Sigma}^* \Sigma \underline{\Sigma}^*$ are regular.

Theorem 4.1. *Let $L = \mathcal{C}(R)$, with R regular and of maximal degree. Then L is regular.*

The proof of Theorem 4.1 requires a few additional considerations and definitions.

In the following, let $A = (\tilde{\Sigma}, Q, \delta, q_0, F)$ be a DFA, with δ being total, and such that $L(A) = R \cup \underline{\Sigma}^+$. Since the words in $\underline{\Sigma}^+$ do not give any contribution to the strong match, we have $\mathcal{C}\left(L(A)\right) = \mathcal{C}(R)$.

Therefore, set Q of states can be partitioned into two disjoint sets S, T, such that $Q = S \cup T$, $q_0 \in S$, and, for every $x, y \in \underline{\Sigma}^*$ and $a \in \Sigma$, we have $\delta(q_0, x) \in S$ and $\delta(q_0, xay) \in T$.

The following lemma is immediate.

Lemma 4.2. *For every $w \in \Sigma^*$, $k > |w|$, and for every multiset Z over Q, if $\{(q_0)^k\} \overset{w}{\rightsquigarrow}_A Z$ then:*

(1) $|[\![Z]\!] \cap S| = 1$ and

(2) $\Sigma_{q \in S} Z(q) = k - |w|$;

that is, exactly one of the states in S occurs in Z and its multiplicity is $k - |w|$.

We construct from A a nondeterministic finite automaton (NFA) that recognizes $\mathcal{C}(R)$.

Definition 4.3. For a DFA A as above, let N be the NFA $(\Sigma, 2^Q, \rightrightarrows_N, \{q_0\}, 2^F)$ where the states are all subsets of Q, $\{q_0\}$ is the initial state, the final states are all subsets of F, and $\rightrightarrows_N \subseteq 2^Q \times \Sigma \times 2^Q$ is the transition relation, defined as follows, for every $a \in \Sigma$, and for all sets $Q', Q'' \subseteq Q$:

$$Q' \overset{a}{\rightrightarrows}_N Q'' \text{ if, and only if } \exists q \in Q' : Q'' = \{\delta(q, a)\} \cup \bigcup_{p \in Q'} \{\delta(p, \underline{a})\}.$$

Relation $\rightrightarrows_N$ can be extended as usual to $2^Q \times \Sigma^* \times 2^Q$.

Proof of Theorem 4.1. The proof shows that when R has maximal degree, then $\mathcal{C}(R) = L(N)$.

First, we show that $\mathcal{C}(R) \subseteq L(N)$.

Assume that for all $w \in \Sigma^*$, for all $k > |w|$, there exists $Z : Q \to \mathbb{N}$ such that; $\{(q_0)^k\} \overset{w}{\rightsquigarrow}_A Z$. We show by induction on $|w|$ that in this case $\{q_0\} \overset{w}{\rightrightarrows}_N [\![Z]\!]$.

The thesis then follows considering the case when $[\![Z]\!] \subseteq F$, also noticing that the assumption $k > |w|$ does not change the language (since $\underline{\Sigma}^* \subseteq R$).

The base step $w = \epsilon$ is immediate. For the inductive step, let $w = za$, for $z \in \Sigma^*, a \in \Sigma$. Since $\{(q_0)^k\} \overset{w}{\rightsquigarrow}_A Z'$, there exists $Z'' : Q \to \mathbb{N}$ such that $\{(q_0)^k\} \overset{z}{\rightsquigarrow}_A Z'' \overset{a}{\rightsquigarrow}_A Z'$. By induction hypothesis, $\{q_0\} \overset{z}{\rightrightarrows}_N [\![Z'']\!]$. Since $Z'' \overset{a}{\rightsquigarrow}_A Z'$, there exists $q \in Z''$ such that $Z' = \{\delta(q,a)\} \uplus \delta(Z'' - \{q\}, \underline{a})$. Since $q \in [\![Z'']\!]$, then define $Q' = \{\delta(q,a)\} \cup \delta([\![Z'']\!], \underline{a})$. Clearly, $[\![Z'']\!] \overset{a}{\rightrightarrows}_N Q'$. However, in general it might be that $[\![Z]\!] \subsetneq Q'$, since Q' also includes $\delta(q, \underline{a})$. However, $q \in S$, hence $Z''(q) = k - |z| = k - |w| + 1$, which is greater than 1 since $k > |w|$ by hypothesis. Hence, $Z'' - \{q\}$ has still at least one occurrence of q. Therefore, $\delta(q, \underline{a}\}$ is in Z'. Hence, $[\![Z']\!] = Q'$.

Conversely, we show that $L(N) \subseteq \mathcal{C}(\mathcal{R})$.

Assume $\{q_0\} \overset{w}{\rightrightarrows}_N Q'$. We prove by induction on $|w|$, that for all $k > |w|$ there exists a multiset Z' over Q such that $[\![Z']\!] = Q'$ and $\{(q_0)^k\} \overset{w}{\rightsquigarrow}_A Z'$. The thesis follows immediately by considering the case $Q' \subseteq F$. The base step is trivial. For the inductive step, let $w = za$, for $z \in \Sigma^*, a \in \Sigma$. Then $\exists Q'' \subseteq Q$ such that $\{q_0\} \overset{z}{\rightrightarrows}_N Q'' \overset{a}{\rightrightarrows}_N Q'$. By induction hypothesis, $\exists Z'' : Q \to \mathbb{N}$ such that $[\![Z'']\!] = Q''$ and $\{(q_0)^k\} \overset{z}{\rightsquigarrow}_A Z''$. Since $Q'' \overset{a}{\rightrightarrows}_N Q'$, there exists $q \in Q''$ such that $Q' = \{\delta(q,a)\} \cup \delta(Q'', \underline{a})$. Let $Z' = \{\delta(q,a)\} \uplus \delta(Z'' - \{q\}, \underline{a})$, which by definition means that $Z'' \overset{w}{\rightsquigarrow}_A Z'$. We are left to prove that $[\![Z']\!] = Q'$. The proof is now identical to the converse case above. Q' also includes $\delta(q, \underline{a})$, while Z' might not include $\delta(q, \underline{a})$. However, $q \in Q_0$, hence $Z''(q) = k - |z| = k - |w| + 1$, which is greater than 1 since $k > |w|$ by hypothesis. Hence, $Z'' - \{q\}$ has still at least one occurrence of q. Therefore, $\delta(q, \underline{a}\}$ is in Z'. Hence, $[\![Z']\!] = Q'$. □

We observe that the above result can be generalized to the case of R being included in $\underline{\Sigma}^*\Sigma^h\underline{\Sigma}^*$, for some finite $h \geq 1$; in other words, each matching word may place up to h letters within a window of width h.

4.2. Descriptive complexity of regular consensual languages

We show that some regular languages can be described much more succinctly by using consensual languages. In what follows, the *size* $\|R\|$ of a regular language R is defined to be the number of states of the smallest nondeterministic automata accepting R.

Theorem 4.4. *There exists a family of regular languages $\{L_m \mid m \geq 2\}$ over Σ and another family of regular languages $\{R_m \mid m \geq 2\}$ over $\widetilde{\Sigma}$ such that $L_m = \mathcal{C}(R_m)$ and the size of each R_m is asymptotically at least exponentially smaller than the size of L_m.*

Proof. Let $\Sigma = \{a\}$ and denote with p_i, $i \geq 1$, the i-th prime number (*e.g.*, $p_1 = 2, \ldots, p_5 = 11, \ldots$). For every $m > 0$, the product of the first m primes, sometimes called the *primorial* of p_m, is

$$p_m\# = \prod_{1\le i\le m} p_i.$$

For every $m \ge 1$, let

$$L_m = \{a^{m+h\cdot p_m\#} \mid h \ge 1\}).$$

and let

$$R_m = \underline{a}^m a^+ \cup \bigcup_{1\le i\le m} \underline{a}^{i-1} a \underline{a}^{m-i} \left(\underline{a}^{p_i}\right)^+ .$$

We claim that $\mathcal{C}(R_m) = L_m$. If $u \in \mathcal{C}(R_m)$ then u is the match of $m+1$ words $u_0@u_1@\dots@u_m$, with $u_0 \in \underline{a}^m a^+$ and each u_i in $\underline{a}^{i-1}a\underline{a}^{m-i}\left(\underline{a}^{p_i}\right)^+$, since u has a prefix $\underline{a}^m$ to be strongly matched, and each u_i places one a as the i-th character. Since the suffix of u_i after a^m must be of a length multiple of each p_i, then there exist $k_1,\dots,k_m \ge 1$ such that $|u_i| = m + k_i p_i$. Hence, $k_1p_1 = k_2p_2 = \dots = k_mp_m$. Since the p_i's are distinct primes, there exists $h \ge 1$ such that $k_1p_1 = hp_1p_2\dots p_m$. Therefore, $|u| = m + hp_1p_2\dots p_m$ and $u \in L_m$.

Conversely, if $u = a^{m+h\ p_m\#} \in L_m$ then u is the match of $\underline{a}^m a^{hp_m\#}$ with m words $u_1,\dots,u_m \in R_m$, with each $u_i = \underline{a}^{i-1}a\underline{a}^{m-i}\underline{a}^{hp_1\cdots p_m}$, because obviously $\underline{a}^{hp_1\cdots p_m}$ is in $(\underline{a}^{p_i})^+$. Hence, $u \in \mathcal{C}(R_m)$.

Since the length of the shortest word in L_m is $m+p_m\#$, every (deterministic or nondeterministic) finite automaton needs at least $m+p_m\#$ states to accept L_M: $\|L_m\| \ge m+p_m\#$. Since $p_i > i$, for every $i \ge 1$, it is obvious that $p_i\# > i!$, *i.e.*, primorials are greater than factorials. Hence, it follows that:

$$\|L_m\| \ge m + p_m\# \ge p_m\# = p_m p_{m-1}\# \ge p_m(m-1)!$$

On the other hand, the number of states of a minimal DFA for R_m is

$$\frac{(m+1)(m+2)}{2} + 1 + \sum_{1\le i\le m} p_i.$$

Hence,

$$\|R_m\| \le 3m + 1 + \sum_{1\le i\le m} p_i \le 3m + 1 + mp_m.$$

Therefore, $\frac{\|L_m\|}{\|R_m\|}$ is $\Omega(\frac{(m-1)!p_m}{mp_m})$ which is $\Omega((m-2)!)$, which is also $\Omega(2^m)$. □

5. Undecidability of emptiness

In this section, the undecidability of emptiness checking is proved, by means of a reduction from the halting problem of a 2-counter Minsky machine [6][1].

[1] The idea of the proof has been suggested by an anonymous referee.

Theorem 5.1. *Let $R \subseteq \widetilde{\Sigma}^*$. It is undecidable whether $\mathcal{C}(R) = \emptyset$.*

Proof. Define a Minsky machine $C = (S, M, i, f)$, where S is a finite set of states, $i \neq f \in S$ are the initial and final states, respectively, and $M \subseteq (S - \{f\}) \times S \times \{true, test\}^2 \times \{INC, DEC, STAY\}^2$ is a finite set of moves. Without loss of generality, assume that for every $p \in S - \{f\}, q \in S$, there exists at most one tuple (t_1, t_2, i_1, i_2), with $t_1, t_2 \in \{true, test\}$ and $i_1, i_2 \in \{INC, DEC, STAY\}$ such that $(p, q, t_1, t_2, i_1, i_2) \in M$.

A configuration of a counter machine is an element of $S \times \mathbb{N} \times \mathbb{N}$. The initial configuration is $(i, 0, 0)$, and a final configuration is (f, i, j), for every $i, j \in \mathbb{N}$. A transition relation $\longrightarrow_C \subseteq S \times \mathbb{N} \times \mathbb{N} \times S \times \mathbb{N} \times \mathbb{N}$ can easily be defined as usual. For instance, if $(p, q, true, test, INC, STAY) \in M$ then *e.g.*, $(p, 3, 0) \longrightarrow_C (q, 4, 0)$: the first counter is not tested ("true"), the second counter is tested for 0, and then the first counter is incremented and the second counter stays.

Let $\Sigma = S \cup \{X, Y, 1, 2\}$. For every $p \in S - \{f\}, q \in S$, define the following regular languages, on the alphabet $\Sigma \cup \underline{\Sigma}$.

$$\begin{array}{lcl}
INIT & = & \underline{1}\underline{2}\underline{1}\underline{2}i\underline{\Sigma}^* \\
NEXT(p,q) & = & \underline{\Sigma}^*\underline{1212}(\underline{XY})^*\underline{p}(\underline{XY})^*\underline{1212}(\underline{XY})^*q\underline{\Sigma}^* \\
COPY_1(p,q) & = & \underline{\Sigma}^*X\underline{Y}(\underline{XY})^*\underline{p}(\underline{XY})^*\underline{1212}(\underline{XY})^*\underline{X}Y(\underline{XY})^*\underline{q}\underline{\Sigma}^* \\
COPY_2(p,q) & = & \underline{\Sigma}^*\underline{p}(\underline{XY})^*X\underline{Y}(\underline{XY})^*\underline{1212}(\underline{XY})^*\underline{q}(\underline{XY})^*\underline{X}Y\underline{\Sigma}^* \\
test_1(p,q) & = & \underline{\Sigma}^*1\underline{212}\underline{p}(\underline{XY})^*\underline{1212}(\underline{XY})^*\underline{q}\underline{\Sigma}^* \\
true_1(p,q) & = & \underline{\Sigma}^*1\underline{212}(\underline{XY})^*\underline{p}(\underline{XY})^*\underline{1212}(\underline{XY})^*\underline{q}\underline{\Sigma}^* \\
test_2(p,q) & = & \underline{\Sigma}^*\underline{1}2\underline{12}(\underline{XY})^*\underline{p}\underline{1212}(\underline{XY})^*\underline{q}\underline{\Sigma}^* \\
true_2(p,q) & = & \underline{\Sigma}^*\underline{1}2\underline{12}(\underline{XY})^*\underline{p}(\underline{XY})^*\underline{1212}(\underline{XY})^*\underline{q}\underline{\Sigma}^* \\
INC_1(p,q) & = & \underline{\Sigma}^*\underline{12}1\underline{2}(\underline{XY})^*\underline{p}(\underline{XY})^*\underline{1212X}Y(\underline{XY})^*\underline{q}\underline{\Sigma}^* \\
DEC_1(p,q) & = & \underline{\Sigma}^*\underline{12}1\underline{2}X\underline{Y}(\underline{XY})^*\underline{p}(\underline{XY})^*\underline{1212}(\underline{XY})^*\underline{q}\underline{\Sigma}^* \\
STAY_1(p,q) & = & \underline{\Sigma}^*\underline{12}1\underline{2}(\underline{XY})^*\underline{p}(\underline{XY})^*\underline{1212}(\underline{XY})^*\underline{q}\underline{\Sigma}^* \\
INC_2(p,q) & = & \underline{\Sigma}^*\underline{121}2(\underline{XY})^*\underline{p}(\underline{XY})^*\underline{1212}(\underline{XY})^*\underline{q}\underline{X}Y\underline{\Sigma}^* \\
DEC_2(p,q) & = & \underline{\Sigma}^*\underline{121}2(\underline{XY})^*\underline{p}X\underline{Y}(\underline{XY})^*\underline{1212}(\underline{XY})^*\underline{q}\underline{\Sigma}^* \\
STAY_2(p,q) & = & \underline{\Sigma}^*\underline{121}2(\underline{XY})^*\underline{p}(\underline{XY})^*\underline{1212}(\underline{XY})^*\underline{q}\underline{\Sigma}^* \\
HALT & = & \underline{\Sigma}^*1212(X\underline{Y})^*\underline{f}(X\underline{Y})^*.
\end{array}$$

Then, associate a regular language $[m]$ with every move m of a 2-counter machine $C = (S, M, i, f)$. For every $m = (p, q, t_1, t_2, i_1, i_2)$, with $t_1, t_2 \in \{test, true\}$, $i_1, i_2 \in \{INC, DEC, STAY\}$ let:

$$[m] = NEXT(p,q) \cup \bigcup_{j=0,1} \left(\begin{array}{ll}
(\textbf{if } t_j \text{ is } test & \textbf{then } test_j(p,q) \textbf{ else } true_j(p,q)\textbf{fi}) \\
\cup & \\
(\textbf{if } i_j \text{ is } INC & \textbf{then } INC_j(p,q) \\
\textbf{elsif } i_j \text{ is } DEC & \textbf{then } DEC_j(p,q) \\
\textbf{else } STAY_j(p,q) & \\
\textbf{fi}) &
\end{array} \right).$$

For instance, if m is $(p, q, test, test, DEC, INC)$ then $[m]$ is:

$$NEXT(p,q) \cup COPY_1(p,q) \cup COPY_2(p,q) \cup$$
$$\cup\ test_1(p,q) \cup\ test_2(p,q) \cup DEC_1(p,q) \cup INC_2(p,q).$$

Define R as the union of $HALT \cup INIT \cup \bigcup_{m \in M} [m]$. Then $\mathcal{C}(R)$ is non-empty if, and only if, C has a halting computation. The main idea is that $\mathcal{C}(R)$ represents the set of all halting runs of C. Every word of $\mathcal{C}(R)$ has the form

$$1212i(1212(XY)^*S(XY)^*)^*1212(XY)^*f(XY)^*$$

where the subwords 1212 separate the representation of two configurations of C. For instance, the prefix $1212i1212$ represents the initial configuration $(i,0,0)$ of C, while a subword $1212(XY)^n p(XY)^m 1212$ represents the configuration (p,n,m), and a suffix $1212(XY)^n f(XY)^m$ represents a final configuration (f,n,m). Also, the definition of R is such that if there is a subword of the form

$$1212(XY)^n p(XY)^m 1212(XY)^{n'} q(XY)^{m'} 1212$$

then $(p,n,m) \longrightarrow_C (q,n',m')$.

Let $F = 1212(XY)^*S(XY)^*$. We show that if $(i,0,0) \longrightarrow_C^+ (f,n,m)$ then there is a word $w \in 1212iF^*1212(XY)^n f(XY)^m$ such that $w \in \mathcal{C}(R)$, *i.e.*, every halting run of C corresponds to a word of $\mathcal{C}(R)$.

By induction on $h \geq 1$, we first prove the following claim (1):

(1) if $(i,0,0) \longrightarrow_C^{h-1} (p,n,m) \longrightarrow_C (q,n',m')$ then there is a word in $R^@$ of the form:

$$1212iF^{h-1}1212(XY)^n p(XY)^m \underline{1212}(\underline{X}Y)^{n'} q(\underline{X}Y)^{m'} \underline{F}^* \,.$$

The base case is:

if $(i,0,0) \longrightarrow_C (q,n',m')$ with $0 \leq n', m' \leq 1$, then

$$1212i\underline{1212}(\underline{X}Y)^{n'} q(\underline{X}Y)^{m'} \underline{F}^* \in R^@.$$

By the set $INIT$,

$$W_0 = \underline{1212}i\underline{1212}(\underline{XY})^{n'} \underline{q}(\underline{XY})^{m'} \underline{F}^* \subseteq R^@ \,.$$

The move $(i, q, test, test, i_1, i_2)$ is in C, with $i_1, i_2 \in \{INC, STAY\}$(and *e.g.*, $i_1 = INC$ if $n' = 1$, $i_1 = STAY$ if $n' = 0$, etc.). Hence, $NEXT(i,q)$ is in $(i, q, test, test, i_1, i_2)$, and $W_1 = \underline{1212}i\underline{1212}(\underline{XY})^{n'} q(\underline{XY})^{m'} \underline{F}^* \subseteq R^@$. Notice that $COPY_1(i,q), COPY_2(i,q)$ cannot match with W_1. Both $test_1(i,q)$ and $test_2(i,q)$ may match with W_1 and hence $W_2 = 12\underline{12}i\underline{1212}(\underline{XY})^{n'} q(\underline{XY})^{m'} \underline{F}^* \subseteq R^@$. Assume that $n' = 1, m' = 0$, hence, $i_1 = INC, i_2 = STAY$. The other cases may be dealt with analogously. Hence, both $INC_1(i,q)$ and $STAY_2(i,q)$ may match with W_2: $W_3 = 1212i\underline{1212X}Yq\underline{F}^* \subseteq R^@$.

The inductive case is dealt with analogously to the base case: by induction hypothesis, there exists a word w_0 in $1212iF^{h-1}\underline{1212}(\underline{X}Y)^n p(\underline{X}Y)^m \underline{F}^* \cap R^@$. Hence, there is $w_1 \in R^@$ of the form

$$1212iF^{h-1}\underline{1212}(\underline{X}Y)^n p(\underline{X}Y)^m \underline{1212}(\underline{XY})^{n'} q(\underline{XY})^{m'} \underline{F}^* \,.$$

Assume that $x = (p, q, true, true, DEC, INC) \in M$ (by the unicity assumption on M, there is no other move from p to q). Hence, $n' = n-1, m' = m+1$. Then, $COPY_1(p,q), COPY_2(p,q) \in [x]$: there is $w_2 \in R^@$ of the form

$$1212iF^{h-1}\underline{1212}(XY)^{n-1}\underline{X}Yp(XY)^m\underline{1212}(\underline{X}Y)^{n-1}q(\underline{X}Y)^m\underline{XYF}^* \,.$$

Since also $true_1(p,q), true_2(p,q) \in [x]$, there is $w_3 \in R^@$ of the form

$$1212iF^{h-1}12\underline{12}(XY)^{n-1}\underline{X}Yp(XY)^m\underline{1212}(\underline{X}Y)^{n-1}q(\underline{X}Y)^m\underline{XYF}^* \,.$$

Since also $DEC_1(p,q), INC_1(p,q) \in [x]$, there is $w_4 \in R^@$ of the form

$$1212iF^{h-1}1212(XY)^{n-1}XYp(XY)^m\underline{1212}(\underline{X}Y)^{n-1}q(\underline{X}Y)^m\underline{X}Y\underline{F}^* \,.$$

thus ending the induction for the case of a move x as above. The other cases of moves are analogous and are omitted for brevity.

Finally, by the claim (1), when $q = f$ if $(i, 0, 0) \longrightarrow (f, n', m')$ then there exists $w \in R^@$ of the form $1212iF^*\underline{1212}(\underline{X}Y)^{n'}q(\underline{X}Y)^{m'}$. By a match with $HALT$, the result is a word $w' \in R^@$ of the form $1212iF^*1212(XY)^{n'}q(XY)^{m'}$, which is in $\mathcal{C}(R)$ since it has no marking.

The proof of the converse claim that every word of $\mathcal{C}(R)$ is a halting run of C is similar and can be safely omitted. □

6. Comparisons with other families

In order to compare consensual languages with some classical language families, we show that certain languages exceed the capacity of consensual languages based on regular sets.

Proposition 6.1. *The languages* $\{ucu^R \mid u \in \{a,b\}^*\}$ *and* $\{ucu \mid u \in \{a,b\}^*\}$ *are not in the family* $\mathcal{C}_{\mathcal{REG}}$.

Proof. Let $L = \{ucu \mid u \in \{a,b\}^*\}$. The proof for $\{ucu^R \mid u \in \{a,b\}^*\}$ is completely analogous. Assume by contradiction there is a DFA $A = (\{a, b, \underline{a}, \underline{b}\}, Q, \delta, q_0, F)$ such that $\mathcal{C}(L(A)) = L$. For a given input word of length $n > 0$, the nondeterministic Turing machine simulating the consensual transition relation $\rightsquigarrow^*$ only stores a multiset over Q, of cardinality $m \leq n$, hence, there are at most $(n+1)^{|Q|}$ different configurations. On the other hand, there are 2^n different strings in $\{a,b\}^n$, and for n large enough, the number of possible strings is much larger than the number $(n+1)^{|Q|}$ of different multisets: there exist $u, w \in \{a,b\}^n$, $u \neq w$, such that there exist $m \leq n$, a multiset Z over Q and two multisets Z', Z'' over F, such that $\{(q_0)^m\} \overset{u}{\rightsquigarrow}_A Z \overset{cu}{\rightsquigarrow}_A Z'$, $\{(q_0)^m\} \overset{w}{\rightsquigarrow}_A Z \overset{cw}{\rightsquigarrow}_A Z''$. But then also $\{(q_0)^m\} \overset{u}{\rightsquigarrow}_A Z \overset{cw}{\rightsquigarrow}_A Z''$, a contradiction since $ucw \notin L$. □

From Proposition 6.1, and from Example 2.4 it follows:

Corollary 6.2. *The family $\mathcal{C}_{\mathcal{R}EG}$ is not comparable with the families of context-free languages and of tree adjoining languages* [5].

Among the typical context-free languages, the Dyck sets with two or more pairs of parentheses trespass the family $\mathcal{C}_{\mathcal{R}EG}$. To see it it suffices to observe that the proof of Proposition 6.1 also applies to the case of language

$$L = \left\{ u\ h(u^R) \mid u \in \{a,b\}^* \right\}$$

where h is the morphism $h(a) = a', h(b) = b'$.

Let D_2 be the Dyck language with opening parentheses a, b and closing parentheses a', b' respectively. Let R be the regular language composed of all strings on $\{a, b, a', b'\}$ where there is no occurrence of the factors $a'a, b'b, a'b, b'a$. Hence, $D_2 \cap R = L$, and, if D_2 were in $\mathcal{C}_{\mathcal{R}EG}$, then by closure of $\mathcal{C}_{\mathcal{R}EG}$ under intersection with regular languages, also L would be in $\mathcal{C}_{\mathcal{R}EG}$. □

7. Conclusion

The simple notion of consensus between simultaneous computations, formulated by means of strong matching, though surely not the only one possible and sensible, yet permits rather remarkable selectivity in language definition. As we see it, the interest of the family of consensual languages, based on matching finite-state computations, comes from a combination of properties; it includes the regular family, and actually offers a very concise representation of some regular sets; it includes non-semilinear languages; and it has a time-polynomial word problem. Altogether this research proposes a new way of looking at finite-state devices as language recognizers.

Since the model is new and research is in its early stages, many questions are open for investigation, for instance concerning minimality, decidability of equivalence, determinism of the counter (or multi-set) machine, as well as the study of closure properties beyond the basic ones considered.

Concerning variations on the theme of consensual computations, we mention some possibilities. Two variations would be to allow (or to oblige) a finite number $k > 1$ of component words to place each letter in each position of the match. Such devices would then model systems where stronger consensus between independent computations is possible (or is requested), in order for a word to be accepted. We believe our definitions, though possibly the simplest, already capture a rather rich range of language paradigms. But of course actual experimentation would be needed.

Moreover, we hope that the consensual approach could be fruitfully investigated for other families of base languages, both weaker and stronger than the regular ones.

At last, we think the idea of consensual computation could provide some formal support to the linguistic requirement of assigning different *degrees of grammaticality* to sentences that satisfy some, but not all, semantic constraints.

Acknowledgements. The first author is indebted to Valentino Braitenberg for inspiring discussions on brain theory and formal language models.

References

[1] A.K. Chandra, D. Kozen and L.J. Stockmeyer, Alternation. *J. ACM* **28** (1981) 114–133.

[2] S. Crespi Reghizzi and P. San Pietro, Consensual definition of languages by regular sets, in *LATA. Lecture Notes in Computer Science* **5196** (2008) 196–208.

[3] S. Crespi Reghizzi and P. San Pietro, Languages defined by consensual computations. in *ICTCS09* (2009).

[4] M. Jantzen, On the hierarchy of Petri net languages. *ITA* **13** (1979).

[5] A. Joshi and Y. Schabes, Tree-adjoining grammars, in *Handbook of Formal Languages*, Vol. 3, G. Rozenberg and A. Salomaa, Eds. Springer, Berlin, New York (1997), 69–124.

[6] M. Minsky, *Computation: Finite and Infinite Machines.* Prentice-Hall, Englewood Cliffs (1976).

[7] A. Salomaa, *Theory of Automata*. Pergamon Press, Oxford (1969).

[8] K. Vijay-Shanker and D.J. Weir, The equivalence of four extensions of context-free grammars. *Math. Syst. Theor.* **27** (1994) 511–546.

Communicated by A. Cherubini.
Received December 24, 2009. Accepted November 18, 2010.