Acta Cybernetica 14 (1999) 117-133.

Syntactic Monoids of Codes*

H. Jürgensen[†]

Abstract

A general characterization theorem for syntactic monoids of codes that satisfy independence conditions of a special form is proved. This result provides insight in some known characterizations of classes of codes via syntactic monoids and provides a general mechanism for deriving new characterizations for other classes of languages.

1 Introduction

Many of the combinatorial properties of codes related to issues in information transmission, like synchronization delays or error resistance, can be expressed algebraically in terms of properties of the syntactic monoids of the codes themselves or of the syntactic monoids of the set of messages generated by the codes. Obtaining these algebraic characterizations is an important, but difficult problem.

Quite a few partial results have been obtained for various classes of codes, for example, for infix codes [15], outfix codes [9], and hypercodes [28], [30]. For a general overview, the reader should consult the books [2] and [25], the survey paper [12], and the other references listed at the end of this paper. A new characterization for infix codes and hypercodes has been obtained in [23]. In the present paper we extract and formulate a general characterization method from those specialized results which applies to all classes of codes that can be defined in a certain way. Thus, results analogous to those of [23] can now be obtained using this method for a large variety of classes of codes simply by proving that their definitions satisfy certain formal criteria.

While this method is applicable to any class of codes satisfying these criteria, its most elegant consequences seem to arise when it is applied to subclasses of the class of infix codes. We discuss the special cases of infix codes, infix codes which are also outfix codes, infix-shuffle codes of index n, hypercodes, solid codes, and reflective

*This research has been supported by the Natural Sciences and Engineering Research Council of Canada, Grant OGP0000243. An earlier version of this paper was presented at the Second International Colloquium on Words, Languages, and Combinatorics, Kyoto, 25–28 August, 1992 [11]. A summary of the results of the present paper is presented in Chapter 8 of [12].

[†]Dartment of Computer Science, The University of Western Ontario London, Ontario, Canada N6A 5B7 and Institut für Informatik Universität Potsdam Am Neuen Palais 10 D-14469 Potsdam, Germany, e-mail: helmut@uwo.ca.

codes. We also show why there is no such characterization for the intercodes of index 1, that is, the comma-free codes.

The paper is organized as follows: In Section 2, we introduce basic notions and notation. In Section 3, we investigate how conditions defining a class of codes are reflected in the syntactic monoids. In Sections 4 and 5, we then focus on classes of codes contained in the class of infix codes. Finally, Section 6 contains a few concluding remarks.

2 Basic Notions and Notation

In this section, we review the basic notions and introduce some notation. For further information regarding codes and their syntactic monoids the reader is referred to the books [2] and [25] and the survey paper [12].

An *alphabet* is a non-empty set.[1] Let X be an alphabet. Then X^* denotes the free monoid generated by X, that is, the set of all *words* over X, including the *empty word* 1, with concatenation as the multiplication.

Let M be a monoid. With every subset L of M one associates its *principal congruence* P_L given by

$$u \equiv v(P_L) \iff \big(\forall x, y \in M\ (xuy \in L \longleftrightarrow xvy \in L)\big).$$

The factor monoid M/P_L is the *syntactic monoid* of L and is denoted by $\operatorname{syn} L$. For $u \in M$, let $[u]_L$ denote the P_L-class of u. The canonical morphism of M onto $\operatorname{syn} L$, that is, the morphism $\sigma_L : u \mapsto [u]_L$, is called the *syntactic morphism*. The set L is said to be *disjunctive* if P_L is the equality relation. The *residue* of L is the set $W_L = \{u \mid u \in M \wedge MuM \cap L = \emptyset\}$.

A *language* over X is a subset of X^*. A language over a finite alphabet is said to be *regular* if it is accepted by a deterministic finite automaton. The syntactic monoid of a regular language is isomorphic with the transition monoid of the reduced complete finite automaton accepting the language.

A language L is said to be a *code* if the submonoid of X^*, which L generates, is freely generated by L. For the study of codes, the case of $|X| = 1$ is trivial and it is, therefore, common to assume without special mention that $|X| > 1$.

Let M be a monoid with zero element, $|M| \geq 2$. We denote the identity and the zero elements by 1 and 0, respectively. The intersection of all non-zero ideals, if it is different from $\{0\}$, is called the *core* of M, denoted by $\operatorname{core}(M)$. The set $\operatorname{annihil}(M) = \{c \mid \forall x \in M \setminus \{1\}\ xc = cx = 0\}$ is the set of *annihilators* of M.

A *pointed monoid*[2] is a pair (M, L) where M is a monoid and L is a subset of M. Let (M, L) and (M', L') be pointed monoids. A *pointed-monoid morphism* of (M, L) into (M', L') is a semigroup morphism φ of M into M' such that $\varphi^{-1}(L') = L$. Such a pointed-monoid morphism φ is *surjective, injective, bijective* if it is so as a

[1] In the literature on formal languages, alphabets are usually assumed to be finite. In this paper, the finiteness condition would not make an important difference. It is, therefore, omitted as in [23].

[2] Called *p-monoid* in [24].

semigroup morphism of M into M'; it is *non-erasing* if $\varphi^{-1}(1_{M'}) = \{1_M\}$. Let $\mathbb{P}$ denote the category of pointed monoids.

If (M, L) is a pointed monoid then σ_L is a surjective pointed-monoid morphism onto $(\operatorname{syn} L, \sigma_L(L))$. Moreover, if φ is a surjective pointed-monoid morphism of (M, L) onto (M', L') then there is a unique surjective pointed-monoid morphism ψ of (M', L') onto $(\operatorname{syn} L, \sigma_L(L))$ such that, for all $u \in M$, $\sigma_L(u) = \psi(\varphi(u))$.

A predicate P on $\mathbb{P}$ is said to be *invariant* if, for any pointed monoid (M, L), P satisfies the following conditions:

- For any surjective pointed-monoid morphism φ of (M, L), P is true on $(\varphi(M), \varphi(L))$ if P is true on (M, L).
- For any surjective non-erasing pointed-monoid morphism φ of (M, L), P is true on (M, L) if it is true on $(\varphi(M), \varphi(L))$.

The results of this paper are based on the following observation concerning predicates on the category of pointed monoids.

Theorem 1 *Let P be an invariant predicate on $\mathbb{P}$ and let $\mathcal{L}_P$ be the class of languages L over X for which P is true on (X^*, L). The following statements are true:*

(1) If σ_L is non-erasing then $L \in \mathcal{L}_P$ if and only if P is true on the pointed monoid $(\operatorname{syn} L, \sigma_L(L))$.

(2) If P is decidable on finite pointed monoids, L is (constructively) regular, and σ_L is non-erasing, then it is decidable whether $L \in \mathcal{L}_P$.

Proof: The first claim is just a restatement of the definition of invariance, applied to σ_L. For the second claim, if L is constructively given as a regular language then one can compute the syntactic monoid $\operatorname{syn} L$ and the set $\sigma_L(L)$. Note that σ_L is a pointed-monoid morphism. The fact that L is regular implies that $\operatorname{syn} L$ is finite. Therefore, P is decidable on $(\operatorname{syn} L, \sigma_L(L))$. The invariance of P implies that it is decidable whether $L \in \mathcal{L}_P$. □

To apply Theorem 1 to a given predicate P, one has to establish that P is invariant and decidable on finite pointed monoids. In this paper we focus on predicates on $\mathbb{P}$ which can be expressed in a special form called implicational independence condition.

3 Codes and Independence Conditions

Let X be an alphabet with $|X| > 1$. Many natural classes of codes over X are characterized by *independence conditions* on the free monoid X^*. A systematic study of this characterization method is presented in [17] and [18]; see also [12].

The independence conditions can be presented in various forms. The most general approach uses abstract dependence systems in the sense of universal algebra[3] with possible restrictions to finitely based dependence systems [12]. Incomparability with respect to certain partial orders, like the *prefix order*

$$u \underset{\mathrm{p}}{\leq} v \iff v \in uX^*,$$

as studied in [4], [14], and [25] is a quite special case of this approach. In this paper we only consider independence conditions that can be expressed in the form of implications involving equations over X^*.

Recall, for example, that the prefix order $\leq_{\mathrm{p}}$ defines the class $\mathcal{L}_{\mathrm{p}}$ of all prefix codes over X by

$$L \in \mathcal{L}_{\mathrm{p}} \iff L \subseteq X^+ \wedge \forall u,v \in L \left(u \underset{\mathrm{p}}{\leq} v \to u = v \right).$$

This condition could also be expressed as

$$L \subseteq X^+ \wedge \forall u,x \in X^* \ \left((u \in L \wedge ux \in L) \to x = 1\right).$$

We now turn to defining this latter form of independence condition in more abstract terms.

Let M be an arbitrary monoid, let $\mathcal{M}$ be a finite set of subsets[4] of M, and let V be a set of variable symbols, such that $M \cap V = \emptyset$. Moreover, let Λ denote a set variable ranging over all the subsets[5] of M.

In the sequel we need to consider words built from elements of M and V, that is, words in $(M \cup V)^*$. An *equation* over M takes the form $u = v$ and an *inclusion* takes the form $u \in \Lambda$, where u and v are words over $(M \cup V)$. A *basic implicational independence condition* has the form

$$\langle\texttt{quantifier prefix}\rangle\big(\langle\texttt{formula}\rangle \longrightarrow \langle\texttt{formula}\rangle\big)$$

where the quantifier prefix and the formulæ satisfy the following conditions:

(I1) The quantifier prefix specifies, for each variable symbol in V, its range explicitly, that is, as one of the sets in $\mathcal{M}$. Moreover, the quantifier prefix may include quantification over the number of variables used. It involves only universal quantifiers.

(I2) The formulæ are disjunctions of conjunctions of equations and inclusions over M.

The first formula will be referred to as the *premiss,* the second one as the *conclusion* of the basic implicational independence condition.

[3] See [3], [5], and [6].

[4] In all our examples below we have $\mathcal{M} = \{M\}$.

[5] More precisely, the range of Λ is the set of all subsets of whichever monoid is being considered.

These still rather informal definitions can obviously be expressed rigorously.[6] An *implicational independence condition* is a conjunction of basic implicational independence conditions.

If I is an implicational independence condition on the monoid M and $L \subseteq M$, then *I is satisfied on* the pointed monoid (M, L) if I is true on M for $\Lambda = L$.

We list a few natural examples of implicational independence conditions for the case of $M = X^*$ where X is an alphabet with at least two elements and $\mathcal{M} = \{X^*\}$.

Example 1 *Let X be an alphabet with $|X| > 1$. In Table 1 we exhibit a list of implicational independence conditions defining classes of languages over X studied in the context of codes. Each implicational independence condition is identified by its property name which is also used to identify the corresponding class of languages. The prefix-shuffle codes, suffix-shuffle codes, infix-shuffle codes, and outfix-shuffle codes of index $n = 1$ are the prefix codes $\mathcal{L}_{\mathrm{p}}$, suffix codes $\mathcal{L}_{\mathrm{s}}$, infix codes $\mathcal{L}_{\mathrm{i}}$, and outfix codes $\mathcal{L}_{\mathrm{o}}$, respectively, in the usual sense.[7] The infix-shuffle codes of index n are called n-shuffle codes in [17] and elsewhere; in [19], [20] they are called n-infix codes; see also [22], [21]. The shuffle codes of index n as well as some of the other types of codes listed in Table 1 may look like mathematical artifacts; they do, however, capture important aspects of error detection; for details, see [12]. The relation between the classes of codes listed in Table 1 is shown in Figures 1 and 2. The n-codes shown there, for $n \geq 2$, are languages such that every subset of up to n elements is a code [7], [10]; the n-ps-codes are languages, such that every subset of up to n elements is a prefix code or a suffix code [8]. For details on infix and outfix codes and the classes $\mathcal{L}_{\mathrm{pi}}$ and $\mathcal{L}_{\mathrm{si}}$ see [9].*

Many natural classes of codes or code-related languages can be defined using implicational independence conditions. There are, however, some natural classes of codes characterized by finitely based dependence systems for which it seems to be impossible to express the independence condition in terms of an implicational independence condition; the class of uniform codes or block codes, that is, of codes all elements of which have the same length, seems to be an example of this kind.

Suppose that φ is a morphism of M onto a monoid M', and that I is an implicational independence condition on M. Then φ induces an implicational independence condition on M', denoted by $\varphi(I)$, as follows:

(I3) If I is a conjunction of basic implicational independence conditions then let $\varphi(I)$ be the conjunction of the images, under φ, of these basic implicational independence conditions. Let $\mathcal{M}'$ be the set of the images of the sets in $\mathcal{M}$. In the quantifier prefixes of I, replace any range in $\mathcal{M}$ by its image in $\mathcal{M}'$.

(I4) For a word u over $(M \cup V)$, let $\varphi(u)$ be the word over $(M' \cup V)$ obtained by mapping the elements of M into M' according to φ and leaving all variables

[6] In particular, quantification over the number of variables would need to be expanded.

[7] In some cases, as in that of the prefix codes, one would have to require explicitly that $L \subseteq X^+$ to rule out the trivial case of $L = \{1\}$ in which L is not a code. On the other hand, including this degenerate case allows for a simpler statement of the results.

Family	Name	Implicational condition
$\mathcal{L}_{\mathrm{code}}$, codes	I_{code}	$\forall m \forall n \forall x_1, \ldots, x_m, y_1, \ldots, y_n \in X^*$ $((x_1 \in \Lambda \wedge \ldots \wedge x_m \in \Lambda \wedge y_1 \in \Lambda \wedge \ldots$ $\wedge\, y_n \in \Lambda \wedge x_1 \cdots x_m = y_1 \cdots y_n)$ $\rightarrow x_1 = y_1, \ldots, x_n = y_n)$
$\mathcal{L}_{2\text{-code}}$, 2-codes	$I_{2\text{-code}}$	$\forall u, v((u \in \Lambda \wedge v \in \Lambda \wedge uv = vu)$ $\rightarrow u = v)$
$\mathcal{L}_{2\text{-ps}}$, 2-ps-codes	$I_{2\text{-ps}}$	$\forall u, x, y\, ((u \in \Lambda \wedge ux \in \Lambda \wedge yu \in \Lambda$ $\wedge\, ux = yu) \rightarrow x = y = 1)$
$\mathcal{L}_{\mathrm{p}_n}$, prefix-shuffle codes of index n	I_{p_n}	$\forall x_1, \ldots, x_n, y_1, \ldots, y_n$ $((x_1 \cdots x_n \in \Lambda$ $\wedge\, x_1 y_1 x_2 y_2 \cdots x_n y_n \in \Lambda)$ $\rightarrow y_1 = \cdots = y_n = 1)$
$\mathcal{L}_{\mathrm{i}_n}, \mathcal{L}_{\mathrm{o}_n}, \mathcal{L}_{\mathrm{s}_n}$, infix-, outfix-, suffix-shuffle codes of index n	$I_{\mathrm{i}_n}, I_{\mathrm{o}_n}, I_{\mathrm{s}_n}$	analogous to $\mathcal{L}_{\mathrm{p}_n}$
$\mathcal{L}_{\mathrm{p}_n} \cap \mathcal{L}_{\mathrm{s}_n}$	$I_{\mathrm{p}_n, \mathrm{s}_n}$	$I_{\mathrm{p}_n} \wedge I_{\mathrm{s}_n}$
$\mathcal{L}_{\mathrm{b}}$, bifix codes	I_{b}	see $\mathcal{L}_{\mathrm{p}_n} \cap \mathcal{L}_{\mathrm{s}_n}$ for $n = 1$
$\mathcal{L}_{\mathrm{i}_n} \cap \mathcal{L}_{\mathrm{o}_n}$	$I_{\mathrm{i}_n, \mathrm{o}_n}$	$I_{\mathrm{i}_n} \wedge I_{\mathrm{o}_n}$
$\mathcal{L}_{\mathrm{h}}$, hypercodes	I_{h}	$\forall n \forall x_0, \ldots, x_n, y_1, \ldots, y_n \in X^*$ $((x_0 \cdots x_n \in \Lambda \wedge x_0 y_1 x_1 y_2 \cdots y_n x_n \in \Lambda)$ $\rightarrow y_1 = \cdots = y_n = 1)$
$\mathcal{L}_{\mathrm{refl}}$, reflective languages	I_{refl}	$\forall x, y \in X^*\, (xy \in \Lambda \rightarrow yx \in \Lambda)$
$\mathcal{L}_{\mathrm{pi}}$, p-infix codes	I_{pi}	$\forall u, x, y((u \in \Lambda \wedge xuy \in \Lambda) \rightarrow y = 1)$
$\mathcal{L}_{\mathrm{si}}$, s-infix codes	I_{si}	$\forall u, x, y((u \in \Lambda \wedge xuy \in \Lambda) \rightarrow x = 1)$
$\mathcal{L}_{\mathrm{inter}_n}$, intercodes of index n	I_{inter_n}	$\forall u_1, \ldots, u_{n+1}, v_1, \ldots, v_n, x, y$ $((u_1 \in \Lambda \wedge \cdots \wedge u_{n+1} \in \Lambda$ $\wedge\, v_1 \in \Lambda \wedge \cdots \wedge v_n \in \Lambda$ $\wedge\, u_1 \cdots u_{n+1} = x v_1 \cdots v_n y)$ $\rightarrow ((x = 1 \wedge y = u_{n+1})$ $\vee\, (x = u_1 \wedge y = 1)))$
$\mathcal{L}_{\mathrm{ol\text{-}free}}$, overlap-free languages	$I_{\mathrm{ol\text{-}free}}$	$\forall x, y, z\, ((xy \in \Lambda \wedge yz \in \Lambda)$ $\rightarrow (x = 1 \vee z = 1 \vee y = 1))$
$\mathcal{L}_{\mathrm{solid}}$, solid codes	I_{solid}	$I_{\mathrm{i}} \wedge I_{\mathrm{ol\text{-}free}}$

Table 1: Implicational conditions for some of the language classes.

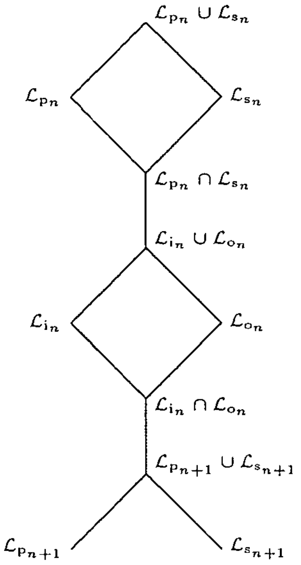
$\mathcal{L}_{p_n} \cup \mathcal{L}_{s_n}$
$\mathcal{L}_{p_n}$
$\mathcal{L}_{s_n}$
$\mathcal{L}_{p_n} \cap \mathcal{L}_{s_n}$
$\mathcal{L}_{i_n} \cup \mathcal{L}_{o_n}$
$\mathcal{L}_{i_n}$
$\mathcal{L}_{o_n}$
$\mathcal{L}_{i_n} \cap \mathcal{L}_{o_n}$
$\mathcal{L}_{p_{n+1}} \cup \mathcal{L}_{s_{n+1}}$
$\mathcal{L}_{p_{n+1}}$
$\mathcal{L}_{s_{n+1}}$

Figure 1: The relation between the shuffle codes.

unchanged.[8] For an equation $u = v$ over M let $\varphi(u = v)$ be the equation $\varphi(u) = \varphi(v)$ over M'. For an inclusion $u \in \Lambda$, let $\varphi(u \in \Lambda)$ be the inclusion $\varphi(u) \in \Lambda$. A formula over M is mapped by φ onto the corresponding formula of the images of the equations and inclusions.

This mechanism of induced implicational independence conditions is to be used in subsequent sections of this paper, to carry implicational independence conditions on X^* defining certain classes of languages into the syntactic monoids of these languages. Hence we consider the following two properties of an implicational independence condition I on a pointed monoid (M, L).

(I5) For any surjective pointed-monoid morphism φ of (M, L), if a premiss of I is false for some assignment of values to the variables then the image of that premiss is also false in $(\varphi(M), \varphi(L))$ for the corresponding value assignment.

(I6) For any surjective non-erasing pointed-monoid morphism φ of (M, L), if a conclusion of I is false for some assignment of values to the variables then the image of that conclusion is also false in $(\varphi(M), \varphi(L))$ for the corresponding value assignment.

[8] We assume that $M' \cap V = \emptyset$.

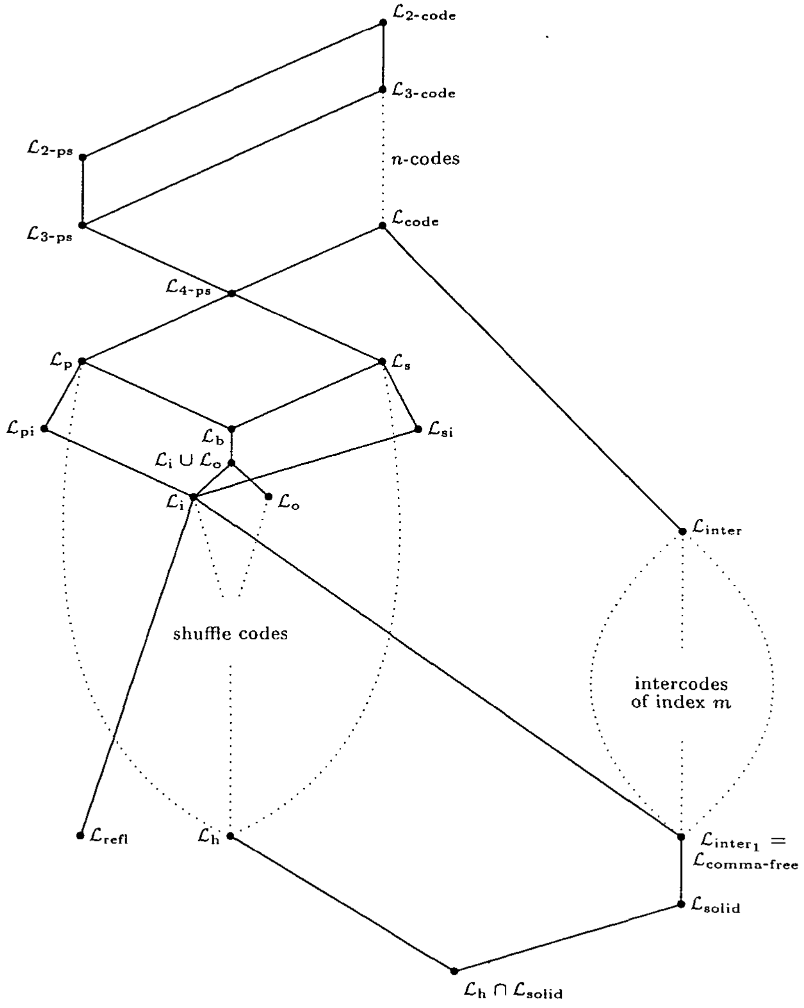
$\mathcal{L}_{2\text{-code}}$
$\mathcal{L}_{3\text{-code}}$
n-codes
$\mathcal{L}_{\text{code}}$
$\mathcal{L}_{2\text{-ps}}$
$\mathcal{L}_{3\text{-ps}}$
$\mathcal{L}_{4\text{-ps}}$
$\mathcal{L}_{\text{p}}$
$\mathcal{L}_{\text{s}}$
$\mathcal{L}_{\text{pi}}$
$\mathcal{L}_{\text{b}}$
$\mathcal{L}_{\text{si}}$
$\mathcal{L}_{\text{i}} \cup \mathcal{L}_{\text{o}}$
$\mathcal{L}_{\text{i}}$
$\mathcal{L}_{\text{o}}$
$\mathcal{L}_{\text{inter}}$
shuffle codes
intercodes of index m
$\mathcal{L}_{\text{refl}}$
$\mathcal{L}_{\text{h}}$
$\mathcal{L}_{\text{inter}_1} =$
$\mathcal{L}_{\text{comma-free}}$
$\mathcal{L}_{\text{solid}}$
$\mathcal{L}_{\text{h}} \cap \mathcal{L}_{\text{solid}}$

Figure 2: The relation between the classes of languages introduced in Table 1. Lines indicate (known) proper inclusions. Dotted lines indicate hierarchies. Intersections and unions are not, in general, indicated.

Theorem 2 *Let (M, L) be a pointed monoid and let I be an implicational independence condition on (M, L). Let φ be a surjective pointed-monoid morphism of (M, L). If I has property I5 and is satisfied on (M, L) then $\varphi(I)$ is satisfied on $(\varphi(M), \varphi(L))$. Conversely, when φ is also non-erasing, if I has property I6 and $\varphi(I)$ is satisfied on $(\varphi(M), \varphi(L))$ then I is satisfied on (M, L).*

Proof: Let (M, L) be a pointed monoid and φ be a surjective pointed-monoid morphism of (M, L) onto the pointed monoid (M', L'). As I is a conjunction of basic implicational independence conditions, I is satisfied on (M, L) if and only if each of its basic components is satisfied on (M, L). Therefore, we only need to prove the claim for I a basic implicational independence condition.

First note that if f is a formula appearing in I and f is true for some assignment of values to the variable symbols then $\varphi(f)$ is also true for the corresponding assignment of values. Indeed, if u and v are words over $(M \cup V)$ and $u = v$ for some value assignment then $\varphi(u) = \varphi(v)$ for the corresponding value assignment; this follows from the fact that φ is a semigroup morphism. Similarly, if $u \in L$ then $\varphi(u) \in \varphi(L) = L'$ as φ is surjective and $\varphi^{-1}(L') = L$, hence $\varphi(L) = L'$. Thus, as f is a disjunction of conjunctions of equations and inclusions, if f is true on (M, L) for some value assignment then also $\varphi(f)$ is true on (M', L') for the corresponding value assignment. Moreover, as φ is surjective, every value assignment in $(M'L')$ corresponds to – that is, is the image of – a value assignment in (M, L).

Let p and c be the premiss and conclusion of I, respectively, and suppose that I has property I5 and is satisfied on (M, L). Consider an assignment of values in M to the variable symbols occurring in p and c. If the premiss p is false under this assignment then also $\varphi(p)$ is false under the corresponding assignment in (M', L') by I5. On the other hand, if the premiss p is true under this assignment, then also the conclusion c must be true under this assignment. But then also $\varphi(p)$ and $\varphi(c)$ are true under the corresponding assignment. Hence, as the implication $p \to c$ is true under any value assignment also the implication $\varphi(p) \to \varphi(c)$ is true under any value assignment. Thus $\varphi(I)$ is satisfied on (M', L').

Conversely, assume that φ is also non-erasing and that I has property I6, and suppose $\varphi(I)$ is satisfied on (M', L'). Hence, for a value assignment either $\varphi(p)$ is false or both $\varphi(p)$ and $\varphi(c)$ are true. If $\varphi(p)$ is false then, for any value assignment in (M, L) that is mapped onto the given one by φ, also p is false. Suppose now that both $\varphi(p)$ and $\varphi(c)$ are true. Consider a value assignment in (M, L) which is mapped onto the given one by φ. If p is false under this assignment then $p \to c$ is true as needed. If p is true under this assignment then, by I6, c cannot be false as $\varphi(c)$ is true. This shows that I is satisfied on (M, L). □

An implicational independence condition I on a monoid M is said to be *free* if the only element of M occurring in I is 1 and if $\mathcal{M} = \{M\}$. A free implicational independence condition can be interpreted over any pointed monoid (M', L'). One treats M as a variable symbol for monoids, M having the value M' in this case, and 1 as denoting the identity element of the monoid under consideration. For a free implicational independence condition I let $\mathcal{L}_I$ be the class if pointed monoids (X^*, L), with X an alphabet, on which I is satisfied.

Theorem 3 *Let P be a predicate on $\mathbb{P}$, defined by a free implicational independence condition I. If I satisfies I5 and I6 for all pointed monoids then P is invariant.*

Proof: Consider a pointed monoid (M,L) and a surjective pointed-monoid morphism φ of (M,L) onto (M',L'). Suppose P is true on (M,L). Then I is satisfied on (M,L) as P is defined by I and I is free. By I5, using Theorem 2, $\varphi(I)$ is true on (M',L'). Thus P is true on (M',L').

For the converse, assume that φ is also non-erasing and that P is true on (M',L'). Then $\varphi(I)$ is satisfied on (M',L') and, by I6 and Theorem 2, I is satisfied on (M,L), hence P is true on (M,L). □

When I is a free implicational independence condition, instead of saying that $\varphi(I)$ is satisfied on $\varphi(M,L)$, it is more convenient to say that I is satisfied on $\varphi(M,L)$. Since this is unambiguous, we make this simplification in the sequel.[9]

Lemma 1 *Let (M,L) be a pointed monoid and let I be an implicational independence condition on M. The following statements hold true:*

(1) Suppose that, if a premiss p of I is false for some value assignment, then also $\sigma_L(p)$ is false for the corresponding value assignment. Then I satisfies I5.

(2) Suppose that σ_L is non-erasing and that, if a conclusion c of I is false for some value assignment, then also $\sigma_L(c)$ is false for the corresponding value assignment. Then I satisfies I6.

Proof: To prove (1), consider a surjective pointed-monoid morphism φ of (M,L) onto (M',L'). Then there is a surjective pointed-monoid morphism ψ of (M',L') onto $(\operatorname{syn} L, \sigma_L(L))$ such that $\sigma_L(u) = \psi(\varphi(u))$ for all $u \in M$. Suppose a premiss p of I is false; hence $\sigma_L(p)$ is false by assumption; if, however, $\varphi(p)$ were true then also $\psi(\varphi(p))$ would have to be true, a contradiction. Hence, $\varphi(p)$ is false. Thus I satisfies I5.

The proof of (2) is analogous; one only notes that σ_L being non-erasing implies that ψ has to be non-erasing. □

By Lemmma 1, it is sufficient to check I5 and I6 for syntactic morphisms. Invariance in general can be established by proving it for syntactic morphisms.

Lemma 2 *Let I be a free implicational independence condition. The following statements hold true:*

(1) If the premisses of I contain only inclusions then I satisfies I5 on any pointed monoid.

(2) If the conclusions of I contain only inclusions or equations of the form $u = 1$ then I satisfies I6 on any pointed monoid.

[9] For a completely rigorous treatment, one should build the category of pointed monoids from the category of monoids in such a way that the identity element is treated as a nullary operation symbol. In this way, a free implicational independence condition would not refer to any specific pointed monoid any more.

Proof: Let (M, L) and (M', L') be pointed monoids and let φ be a surjective pointed-monoid morphism of (M, L) onto (M', L').

First consider an inclusion $u \in \Lambda$. If this inclusion is true on (M', L') for some value assignment then, because of $\varphi^{-1}(L') = L$, this inclusion must be true on (M, L) for any pre-image, under φ, of this value assignment.

Suppose now that a premiss p of I consists solely of inclusions. If p is false for some value assignment on (M, L) then, by the preceding argument, $\varphi(p)$ must be false on (M', L').

Finally assume that φ is also non-erasing. Then $u = 1$ can only be true in (M', L') if it is true in (M, L). Thus, if a conclusion c of I consists only of inclusions and equations of the form $u = 1$ then, if it is false on (M, L), it is also false on (M', L'). □

We now examine some of the implicational independence conditions of Example 1 to determine which of these satisfy I5 or I5 and I6.

Theorem 4 *Let X be an alphabet with $|X| > 1$. Consider the implicational independence conditions listed in Example 1. The following statements hold true.*

(a) I_{code} and I_{inter_1} do not satisfy I5.

(b) I_{p_n}, I_{s_n}, I_{i_n}, I_{o_n}, $I_{\mathrm{i}_n,\mathrm{o}_n}$, $I_{\mathrm{p}_n,\mathrm{s}_n}$, I_{h}, I_{solid}, I_{refl} satisfy I5 and I6.

Proof: By Lemma 1 it is sufficient to consider syntactic morphisms.

We first prove statement (a): Let L be an infix code with $|L| > 1$. Clearly, L satisfies I_{code}. By [15], $\sigma_L(L)$ is a single element c in $\operatorname{syn} L$, and c is an annihilator different from 0. Let $u, v \in L$, $u \neq v$. Then $u^2 \neq v^2$. However, $\sigma_L(u^2) = c^2 = 0 = \sigma_L(v^2)$. Therefore, I_{code} does not satisfy I5.

Now consider I_{inter_1}. Every intercode of index 1 is an infix code [27]. Let L be an intercode of index 1. Therefore, the implicational independence condition $\sigma_L(I_{\mathrm{inter}_1})$ reduces to

$$\forall x, y \ \left(c^2 = xcy \rightarrow x = 1 \vee y = 1\right)$$

where x and y range over the syntactic monoid of L. As c is an annihilator, $c^2 = xcy = 0$ for all choices of x and y except $x = y = 1$. On the other hand, this premiss need not be true in X^*. For instance, let $X = \{a, b\}$ and $L = \{ab\}$; by [16], L is a solid code, hence an intercode of index 1. Let $x = y = a$. The only choice for u, v, w is $u = v = w = ab$. Hence $uv = abab \neq aaba = xwy$, but $\sigma_L(uv) = 0 = \sigma_L(xwy)$.

Statement (b) follows by Lemma 2. □

Note that the argument used to prove statement (b) does not apply in the cases of $\mathcal{L}_{2\text{-code}}$, $\mathcal{L}_{2\text{-ps}}$, or $\mathcal{L}_{\mathrm{inter}_n}$ as, in all these cases, the implicational independence condition provided in Example 1 contains an equation in the premiss.

Theorem 5 *Let M be a monoid such that $M \setminus \{1\}$ is a subsemigroup of M. Let I be a free implicational independence condition satisfying I5 and I6 on any pointed* *monoid. Then M is isomorphic with the syntactic monoid of a language L over an alphabet X such that I is satisfied on (X^*, L) if and only if M has a disjunctive subset L' such that I is satisfied on (M, L').*

Proof: Suppose $L \subseteq X^*$ is such that I is satisfied on (X^*, L) and that $M \simeq \operatorname{syn} L$. Then $\sigma_L(L)$ is disjunctive in $\operatorname{syn} L$ and I is satisfied on $(\operatorname{syn} L, \sigma_L(L))$ by I5.

Conversely, let L' be a disjunctive subset of M such that I is satisfied on (M, L'). Let $X \subseteq M \setminus \{1\}$ be a set of generators of M. The embedding of X in M induces a morphism φ of X^* onto M. Let $L = \varphi^{-1}(L')$. Then $\varphi = \sigma_L$, φ is non-erasing, and I is satisfied on (X^*, L) by I6. □

Thus, many classes of languages defined by independence conditions have a characterization by syntactic monoids in the following sense: M is the syntactic monoid of such a language if and only if M contains a disjunctive subset which satisfies the conditions that characterize the respective class of languages. In the next sections of this paper, we apply this property to subclasses of the class of infix codes.

We conclude the present section with an interesting consequence of Theorem 5 regarding the decidability of language properties.

Theorem 6 *Let I be a free implicational independence condition satisfying I5 and I6 on all pointed monoids. Let X be an alphabet and let L be a regular language over X. If L is given effectively and σ_L is non-erasing then it is decidable whether I is satisfied on (X^*, L).*

Proof: Let L be a regular language, given in some effective way. Construct the reduced complete deterministic finite automaton A accepting L. Then $\operatorname{syn} L$ is isomorphic with the transition monoid of A. Moreover, one can compute $\sigma_L(L)$. Since $\operatorname{syn} L$ is finite one can check whether I is satisfied on $(\operatorname{syn} L, \sigma_L(L))$. If so I is satisfied on (X^*, L); otherwise it is not. □

With Theorem 6, we have a general proof of the decidability of certain code properties which, so far, has only been obtained for special cases with a special proof for each case. Another quite different general technique for proving such decidability results is provided in [13].

4 Infix Codes

In this section, we consider classes of codes "low in the hierarchy," that is, classes of codes contained in the class $\mathcal{L}_{\mathrm{i}}$, the class of infix codes. The syntactic monoids of infix codes have some special properties which render it particularly easy to express implicational independence conditions in them.

The following characterization of monoids which are isomorphic with syntactic monoids of infix codes is given in [23]. Note that some of the conditions imply that the monoid be subdirectly irreducible (see also [15]). In stating this result, we refer to the following list of properties of a monoid M.

(M_1) $M \setminus \{1\}$ is a subsemigroup of M.

(M_2) M has a zero.

(M_3) M has a disjunctive element c distinct from 1 and 0 such that $c = xcy$ implies $x = y = 1$.

(M_4) M has a disjunctive zero.

(M_5) There is an element $c \in \operatorname{annihil}(M)$ distinct from 0 such that $\operatorname{core}(M) = \{c, 0\}$.

(M_6) There is an element c distinct from 0 such that $c \in \operatorname{core}(M) \cap \operatorname{annihil}(M)$.

Theorem 7 [23] *The following conditions on a monoid M are equivalent.*

(1) M is isomorphic with the syntactic monoid of an infix code.

(2) M has the properties M_1, M_2, *and* M_3.

(3) M has the properties M_1, M_4, *and* M_5.

(4) M has the properties M_1, M_4, *and* M_6.

If L is an infix code and $c \in \operatorname{syn} L$ is the element of condition M_3, M_5, or M_6 then $c = \sigma_L(L)$. Hence, condition M_4 states in particular that c satisfies the implicational independence condition I_{i}. This observation allows the following generalization of Theorem 7. For an arbitrary implicational independence condition, let $I(c)$ be the implicational independence condition obtained by substituting the symbols '$= c$' for every occurence of the symbols '$\in \Lambda$'.

Theorem 8 *Let I be an implicational independence condition satisfying I5 and I6 on all pointed monoids. If $\mathcal{L}_I \subseteq \mathcal{L}_{\mathrm{i}}$ then the following conditions on a monoid M are equivalent.*

(1) M is isomorphic with the syntactic monoid $\operatorname{syn} L$ *of some L with $(X^*, L) \in \mathcal{L}_I$.*

(2) M has the properties M_1, M_2, M_3, *and $I(c)$.*

(3) M has the properties M_1, M_4, M_5, *and $I(c)$.*

(4) M has the properties M_1, M_4, M_6, *and $I(c)$.*

Proof: If M is isomorphic with the syntactic monoid $\operatorname{syn} L$ of some language L with $(X^*, L) \in \mathcal{L}_I$ then L is an infix code, and M has the properties M_1, M_2, and M_3. As $c = \sigma_L(L)$, also $I(c)$ holds true.

For infix codes, statements (2), (3), and (4) are equivalent by Theorem 7. Moreover, the proof shows that in each of M_3, M_5, and M_6, the element c is actually the same element of M. Hence, these statements are also equivalent for the class $\mathcal{L}_I$.

By Theorem 7, statement (4) implies that M is isomorphic with the syntactic monoid of an infix code L. Moreover, from the proof it follows that $\sigma_L(L) = c$. As $\varphi(I)$, for $\Lambda = L$, is equivalent to $I(c)$, it follows that I is satisfied on (X^*, L). □

Thus, Theorem 8 provides for a general method of characterizing the syntactic monoids of those classes of codes which are contained in the class of infix codes and are given by an implicational independence condition satisfying I5 and I6. In particular, this includes all the cases listed in Theorem 4(b) except the prefix and the suffix codes.

5 Infix and Outfix Codes

In [23], Corollary 1, a characterization of the syntactic monoids of hypercodes is derived which in part forms a special case of Theorem 8. In this section we show that also the remaining parts of that result can be obtained as a special case from a quite general theorem. For the result of [23] on hypercodes, the following observation is crucial: For a hypercode L, every P_L-class different from W_L is a hypercode. A similar statement can be made about other classes of codes contained in the class of outfix codes. See Figure 1 for the hierarchy of shuffle codes. In essence, we consider the classes contained in $\mathcal{L}_{\mathrm{i}} \cap \mathcal{L}_{\mathrm{o}} = \mathcal{L}_{\mathrm{i}_1} \cap \mathcal{L}_{\mathrm{o}_1}$. This class is characterized by the free implicational independence condition

$$\begin{aligned}\forall u, x, y \in X^* \ \big(&((u \in \Lambda \wedge xuy \in \Lambda) \to x = y = 1)\\ &\wedge ((xy \in \Lambda \wedge xuy \in \Lambda) \to u = 1)\big)\end{aligned}$$

which satisfies I5 and I6 on all pointed monoids. Thus, also the syntactic monoids of codes in $\mathcal{L}_{\mathrm{i}} \cap \mathcal{L}_{\mathrm{o}}$ can be characterized using Theorem 8.

Theorem 9 *The following statements hold true.*

(a) If L is an outfix code then every P_L-class different from the residue of L is an outfix code.

(b) If L is an infix-shuffle code of index n with $n \geq 3$ then every P_L-class different from the residue of L is an infix-shuffle code of index $n-2$.

(c) If L is a hypercode then every P_L-class different from the residue of L is a hypercode.

Proof: The proof of (a) is given in [9].

Let L be an infix-shuffle code of index n with $n \geq 3$, and consider P_L-equivalent words u, v such that u is not in the residue of L. Hence, there are s and t such that $sut \in L$ and, therefore, also $svt \in L$. Suppose that

$$u = u_1 u_2 \cdots u_{n-2} \quad \text{and} \quad v = v_1 u_1 v_2 u_2 \cdots v_{n-2} u_{n-2} v_{n-1}.$$

Letting $s = v_0$, $t = u_{n-1}$ and $v_n = 1$, one obtains $sut = svt$ from the fact that L is an infix-shuffle code of index n. This implies $u = v$. Thus, the class of u is an infix-shuffle code of index $n-2$.

The statement concerning hypercodes is proved in [23]. It is, of course, also an immediate consequence of (b) as a language is a hypercode if and only if it is an infix-shuffle code of index n for every n. □

By Theorem 9(b) and Figure 1, if $L \in \mathcal{L}_{\mathrm{i}} \cap \mathcal{L}_{\mathrm{o}}$, $L \in \mathcal{L}_{\mathrm{p}_n}$, $L \in \mathcal{L}_{\mathrm{s}_n}$, $L \in \mathcal{L}_{\mathrm{i}_n}$, $L \in \mathcal{L}_{\mathrm{o}_n}$, or $L \in \mathcal{L}_{\mathrm{h}}$ for $n \geq 2$, then every P_L-class different from the residue is an outfix code.

In view of Theorem 9, the result of [23] on hypercodes can be generalized significantly.

Theorem 10 *Let I be an implicational independence condition satisfying I5 and I6 on all pointed monoids and such that $\mathcal{L}_I \subseteq \mathcal{L}_{\mathrm{i}}$.*

Let I' be an implicational independence condition satisfying I5 and I6 on all elements of $\mathcal{L}_I$ and such that I implies I' and I' implies I_{i}.

Suppose that, if $(X^*, L) \in \mathcal{L}_I$, for every P_L-class L' different from W_L, I' is satisfied on (X^*, L'). Then the following conditions on a monoid M are equivalent.

(1) M is isomorphic with the syntactic monoid of a code in $\mathcal{L}_I$.

(2) M has the properties M_1, M_2, M_3, *and $I(c)$ and every element $x \in M \setminus \{0\}$ satisfies $I'(x)$.*

(3) M has the properties M_1, M_2, M_3, *and $I(c)$.*

Proof: The assumption about the P_L-classes different from W_L implies that $I'(x)$ holds true for every $x \in M \setminus \{0\}$. Thus, (1) implies (2). Obviously, statement (3) follows from (2). The remaining implication is already stated in Theorem 8. □

The case of hypercodes [23] is a special case of this result as are the cases of infix-shuffle codes of index n and of those infix codes which are also outfix codes.

6 Concluding Remarks

The main result of this paper is a general method for characterizing the syntactic monoids of codes when the class of codes is defined in a special formal way. The main application is to classes of codes, low in the hierarchy, that is, below the classes of infix codes and outfix codes.

The properties of infix codes and outfix codes that lead to particularly simple characterizations are the following: The syntactic monoid of an infix code has a disjunctive element. Every syntactic class of an outfix code is an outfix code. The obvious next step seems to be to abstract these properties and extend the results to higher regions of the hierarchy in possibly some restricted form.

References

[1] *Abstracts, Second International Colloquium on Words, Languages, and Combinatorics, Kyoto, 25–28 August, 1992.* Kyoto, 1992.

[2] J. Berstel, D. Perrin: *Theory of Codes.* Academic Press, Orlando, 1985.

[3] P. M. Cohn: *Universal Algebra*. D. Reidel Publishing Co., Dordrecht, revised ed., 1981.

[4] P. H. Day, H. J. Shyr: Languages defined by some partial orders. *Soochow J. Math.* **9** (1983), 53–62.

[5] F. Gécseg, H. Jürgensen: Algebras with dimension. *Algebra Universalis* **30** (1993), 422–446.

[6] F. Gécseg, H. Jürgensen: Dependence in algebras. *Fund. Inform.* **25** (1996), 247–256.

[7] M. Ito, H. Jürgensen, H. J. Shyr, G. Thierrin: Anti-commutative languages and n-codes. *Discrete Appl. Math.* **24** (1989), 187–196.

[8] M. Ito, H. Jürgensen, H. J. Shyr, G. Thierrin: n-Prefix-suffix languages. *Internat. J. Comput. Math.* **30** (1989), 37–56.

[9] M. Ito, H. Jürgensen, H. J. Shyr, G. Thierrin: Outfix and infix codes and related classes of languages. *J. Comput. System Sci.* **43** (1991), 484–508.

[10] M. Ito, H. Jürgensen, H. J. Shyr, G. Thierrin: Languages whose n-element subsets are codes. *Theoret. Comput. Sci.* **96** (1992), 325–344.

[11] H. Jürgensen: Syntactic monoids of codes. In [1], 108–112.

[12] H. Jürgensen, S. Konstantinidis: Codes. In G. Rozenberg, A. Salomaa (editors): *Handbook of Formal Language Theory*. Springer-Verlag, Berlin, 1997, vol. 1, 511–607.

[13] H. Jürgensen, K. Salomaa, S. Yu: Transducers and the decidability of independence in free monoids. *Theoret. Comput. Sci.* **134** (1994), 107–117.

[14] H. Jürgensen, H. J. Shyr, G. Thierrin: Codes and compatible partial orders on free monoids. In S. Wolfenstein (editor): *Algebra and Order, Proceedings of the 1st International Symposium on Ordered Algebraic Structures, Luminy-Marseilles, 1984*. 323–333, Heldermann-Verlag, Berlin, 1986.

[15] H. Jürgensen, G. Thierrin: Infix codes. In M. Arató, I. Kátai, L. Varga (editors): *Topics in the Theoretical Bases and Applications of Computer Science, Proceedings of the 4th Hungarian Computer Science Conference, Györ, 1985*. 25–29, Akadémiai Kiadó, Budapest, 1986.

[16] H. Jürgensen, S. S. Yu: Solid codes. *J. Inform. Process. Cybernet., EIK* **26** (1990), 563–574.

[17] H. Jürgensen, S. S. Yu: Relations on free monoids, their independent sets, and codes. *Internat. J. Comput. Math.* **40** (1991), 17–46.

[18] H. Jürgensen, S. S. Yu: Dependence systems and hierarchies of families of languages. Manuscript, 1996. In preparation.

[19] D. Y. Long: k-Outfix codes. *Chinese Ann. Math. Ser. A* **10** (1989), 94–99, in Chinese.

[20] D. Y. Long: k-Prefix codes and k-infix codes. *Acta Math. Sinica* **33** (1990), 414–421, in Chinese.

[21] D. Y. Long: n-Infix-outfix codes. In [1], 50–51.

[22] D. Y. Long: k-Bifix codes. *Riv. Mat. Pura Appl.* **15** (1994), 33–55.

[23] M. Petrich, G. Thierrin: The syntactic monoid of an infix code. *Proc. Amer. Math. Soc.* **109** (1990), 865–873.

[24] J. Sakarovitch: Un cadre algébrique pour l'étude des monoïdes syntactiques. In *Séminaire P. Dubreil (Algèbre), 28e année.* 14. Paris, 1974/75.

[25] H. J. Shyr: *Free Monoids and Languages.* Hon Min Book Company, Taichung, second ed., 1991.

[26] H. J. Shyr, G. Thierrin: Codes and binary relations. In M. P. Malliavin (editor): *Séminaire d'algèbre Paul Dubreil, Paris 1975–1976, (29ème Année). Lecture Notes in Computer Science* **586**, 180–188, Springer-Verlag, Berlin, 1977.

[27] H. J. Shyr, S. S. Yu: Intercodes and some related properties. *Soochow J. Math.* **16** (1990), 95–107.

[28] G. Thierrin: The syntactic monoid of a hypercode. *Semigroup Forum* **6** (1973), 227–231.

[29] G. Thierrin, S. S. Yu: Shuffle relations and codes. *J. Inform. and Optim. Sci.* **12** (1991), 441–449.

[30] E. Valkema: Syntaktische Monoide und Hypercodes. *Semigroup Forum* **13** (1976/77), 119–126.

[31] S. S. Yu: A characterization of intercodes. *Internat. J. Comput. Math.* **36** (1990), 39–45.