On Formal Descriptions of Code Properties

Stavros Konstantinidis (SK)
Saint Mary's University, Halifax, CAN

(with **Krystian Dudzinski**)

presented at **DCFS 2010** on Aug. 8, 2010

1 Introduction

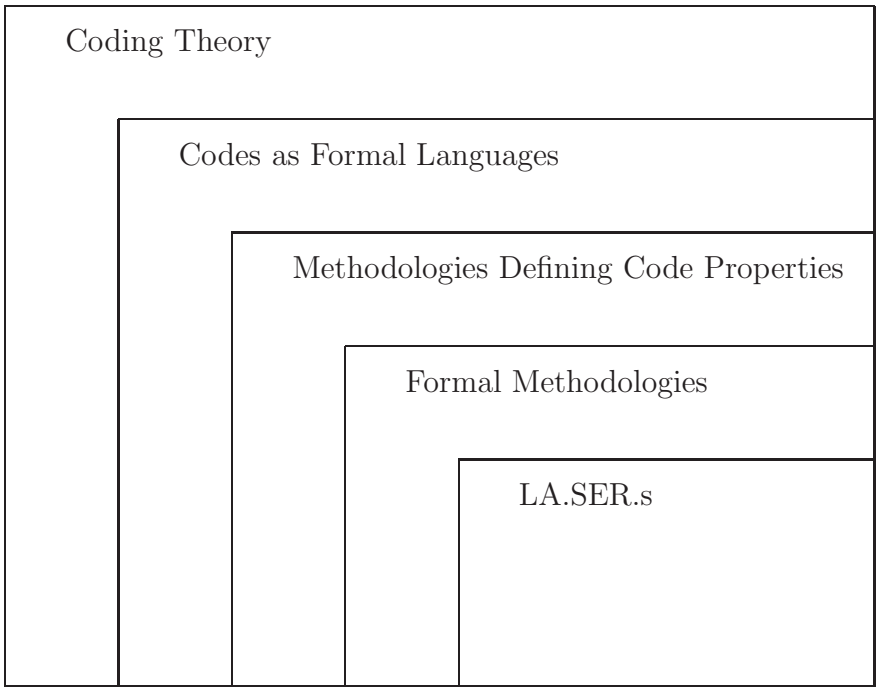

Figure 1: Some developments in Coding Theory.

2 Codes as Formal Languages

• **Code**. A language L that is uniquely decodable:

$v_1 \cdots v_n \ = \ x_1 \cdots x_m \ \rightarrow \ m = n$ and $x_i = v_i$ for all i

Decidable: *[Sardinas,Patterson'53], [Levenshtein'61], [Berstel,Perrin'84], [Head,Weber'93], [McCloskey'96], [Fernau, Reinhardt, Staiger'07]*

• **Code with decoding delay** d.

$$w \in vL^d\Sigma^* \cap xL^* \ \rightarrow \ x = v.$$

Decidable: *[Levenshtein'61], [Devolder,Latt.,Lit.,Staiger'94], [SK'02]*

• **Prefix code**. $v, vz \in L \ \rightarrow \ z = \lambda$ **or** $L \cap L\Sigma^+ = \emptyset$

• **Infix code**. $v, yvz \in L \ \rightarrow \ yz = \lambda$ **or**

$$L \cap (\Sigma^+ L\Sigma^* \cup \Sigma^* L\Sigma^+) = \emptyset$$

Decidable maximality: *[Lam'98], [Kari, SK'05]*

• **Thin language**. All words have different lengths

$$x \in L,\ v \in L,\ |v| = |x| \ \rightarrow \ x = v.$$

• **Error-detecting language (wrt channel γ)**.

$$x \in L,\ (x, w) \in \gamma\ , w \in L \ \rightarrow \ x = w.$$

Decidable: *[SK'02]*

E.g. $(x, w) \in \gamma$ iff $H(x, w) \leq 2$.

— General decidability methods: *[Head,Weber'93], [Jürgensen, Salomaa, Yu'94], [Jürgensen'99], [Domaratzki'04], [Kari,SK'05]*

— General decidability of **maximality** methods: *[Domaratzki'04], [Kari,SK'05], [Van'06]*

3 Methodologies for Defining Code Properties

• **Partial word orders**. '$\prec_\alpha$' strict, length increasing, transitive.

$$L \in \mathcal{P}_\alpha \quad \text{iff} \quad \forall\, u, v \in L : \text{not } (u \prec_\alpha v)$$

[Shyr, Thierrin'77], [Van'06]

E.g. Prefix: $u \prec_{\text{p}} v$ iff $v = uz$

Theorem 1 $\mathcal{P}_{\text{code}}$ is not definable via any word order.

• **n-ary word relations**. *[Jürgenesen, Yu'91]*

'$\omega_\alpha \subseteq \Sigma \times \cdots \times \Sigma$' upward symmetric.
Can use a ternary relation for comma-free codes:

$$LL \,\cap\, \Sigma^+ L\, \Sigma^+ \,=\, \emptyset$$

Note that $\mathcal{P}_{\text{code}}$ is still not definable.

• **Dependence systems**. *[Jürgenesen, Yu'95]*

Let $n \in \mathbb{N} \cup \{\aleph_0\}$. We say that $\mathcal{P}_\alpha$ is an n-independence system if

$$L \in \mathcal{P}_\alpha \text{ iff } \forall\, L' \subseteq L \text{ with } |L'| < n :\ L' \in \mathcal{P}_\alpha$$

All previous properties are definable via independence systems. E.g., prefix codes are a 3-independence property and codes are $\aleph_0$-independence.

Theorem 2 Let $\mathcal{P}_\alpha$ be an independence property. Every $\mathcal{P}_\alpha$-language is included in a maximal $\mathcal{P}_\alpha$-language.

We use this methodology as a reference point.

4 Formal Methodologies

- To be compatible with and (when possible) more general than other existing methodologies.
- To be able to decide efficiently, given the **description** of a code property $\mathcal{P}$ and a regular language L, whether L satisfies $\mathcal{P}$.
- To be able to decide (efficiently?), given the description of a code property $\mathcal{P}$ and a regular language L, whether L is maximal with the property $\mathcal{P}$.
- To be able to build a **LA**nguage **SER**ver that allows a user to enter descriptions of code properties and produce answers to questions about languages with the desired code properties.

Theorem 3 *Let $n \geq 2$. The class of n-independence properties is uncountable. In fact, already the class of n-independence properties whose elements are prefix codes is uncountable.*

As any set of descriptions is countable, we cannot define/describe formally all possible independence properties.

• **Some existing (semi-)formal methodologies**.

Implicational conditions. *[Jürgensen'99]*

Example: suffix codes

$\varphi_{\mathrm{s}} \;=\; \text{“}\forall\, u, v, x : u \in L, v \in L, u = xv \rightarrow x = \lambda\text{”}.$

Trajectories. *[Domaratzki'04],[Domaratzki,Salomaa'06]*

Example: $\bar{e} = 1^*0^*$ defines suffix codes

$$(L \amalg_{\bar{e}} \Sigma^+) \cap L = \emptyset.$$

Language (in)equations. *[Kari,SK'05]*

Example: suffix codes

$$(L \leftarrow_{lq} \Sigma^+) \subseteq L^c \quad \text{iff} \quad (L \leftarrow_{lq} \Sigma^+) \cap L = \emptyset.$$

In fact better described by a transducer...

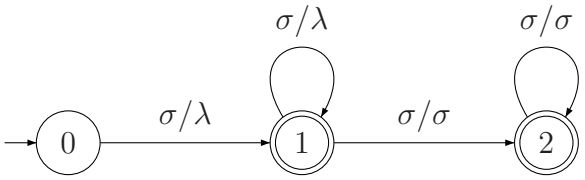

Figure 2: The suffix-code property.

The term "type-0" might change in the full paper

5 Type-0 transducers

Rephrase definitions of prefix, suffix, infix code L:

$$P(L) \cap L = \emptyset \qquad S(L) \cap L = \emptyset \qquad I(L) \cap L = \emptyset$$

Generalize: property defined by type-0 transducer $\hat{t}$

$$\mathcal{P}_{0,\hat{t}} = \{L \subseteq \mathbf{M} \mid \hat{t}(L) \cap L = \emptyset\},$$

where

$$\forall\, w \in \mathbf{M},\ w \notin \hat{t}(w)$$

and $\mathbf{M}$ is our maximum set of words (e.g., $\mathbf{M} = \Sigma^*$).

Theorem 4 *Every trajectory property is a type-0 transducer property (effectively), which in turn is a 3-independence property. There is a time $O(|\hat{t}| \cdot |\hat{a}|^2)$ decision algorithm, and maximality also is decidable.*

Corollary: Decidable whether L is maximal thin.

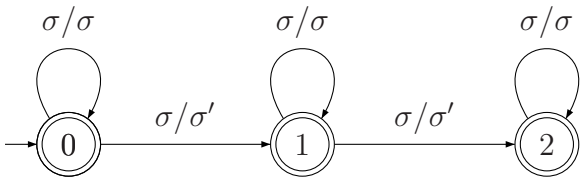

Figure 3: The 2-substitution-error-detection property.

same for "type-1"

6 Type-1 transducers

What about the properties of error-detection and -correction?

Property defined by type-1 transducer $\hat{t}$

$$\mathcal{P}_{1,\hat{t}} \;=\; \{L \subseteq \mathbf{M} \mid \;\; \forall\, x \in L,\;\; \hat{t}(x) \cap (L - x) = \emptyset\},$$

where

$$\forall\, w \in \mathbf{M},\; w \in \hat{t}(w).$$

Theorem 5 *Every type-0 transducer property is a type-1 property (effectively), which in turn is a 3-independence property. There is a polynomial time decision algorithm, and maximality also is decidable.*

Based on deciding transducer functionality *[Beal,Carton,Prieur,Sakarovitch'03]*

Corollary: Maximality of error-detection and -correction decidable.

• **Error-correction**.

$$x \in L, (x, w) \in \gamma, \text{ and } v \in L, (v, w) \in \gamma \rightarrow x = v.$$

A language is error-correcting for γ IFF it is error-detecting for $\gamma^{-1} \circ \gamma$.

7 Decidability of Maximality

- Let L be a language satisfying $\mathcal{P}_\alpha$. We have

$$\begin{array}{ll} & L \text{ is not maximal} \\ \underline{\text{iff}} & \exists\, w \in \mathbf{M} - L : (L + w) \in \mathcal{P}_\alpha \\ \underline{\text{iff}} & \exists\, w \in \mathbf{M} \cap L^c : w \notin R_\alpha(L) \qquad \leftarrow (?) \\ \underline{\text{iff}} & \mathbf{M} \cap L^c \cap R_\alpha(L)^c = \emptyset. \end{array}$$

- For both, $\mathcal{P}_{0,\hat{t}}$ and $\mathcal{P}_{1,\hat{t}}$, we have that

$$R_\alpha(L) = \hat{t}(L) + \hat{t}^{-1}(L).$$

When L is given via an NFA, there is effectively an NFA for L^c and $\hat{t}(L) + \hat{t}^{-1}(L)$.

- Unfortunately testing emptiness of

$$L^c \cap (\hat{t}(L) + \hat{t}^{-1}(L))^c$$

is PSPACE-complete, for given NFA and type-0 $\hat{t}$.

• **Some questions**.
– What if we fix $\hat{t}$, e.g., the suffix code property?
– What if L is given via a DFA?
– What's the state complexity of $\hat{t}(L)$, $\hat{t}^{-1}(L)$, $\hat{t}(L)+\hat{t}^{-1}(L)$?

8 LA.SER. (in progress)

• **Current capability**. Web server accepting names of two files containing the automaton (language) and transducer (property) in **Grail** format, and returns whether the language satisfies the type-0 property.

`http://laser.cs.smu.ca/transducer/`

• **Next capability**. Add translation from trajectory property to type-0 property.

Return two witness words if property not satisfied -- suggested by Prof. Dassow

• **Then**. Add type-1 transducer properties.

• **Then**. Extend to computing languages with desired properties. *[Lam], [Van'06]*

• **LA.SER. Architecture**.

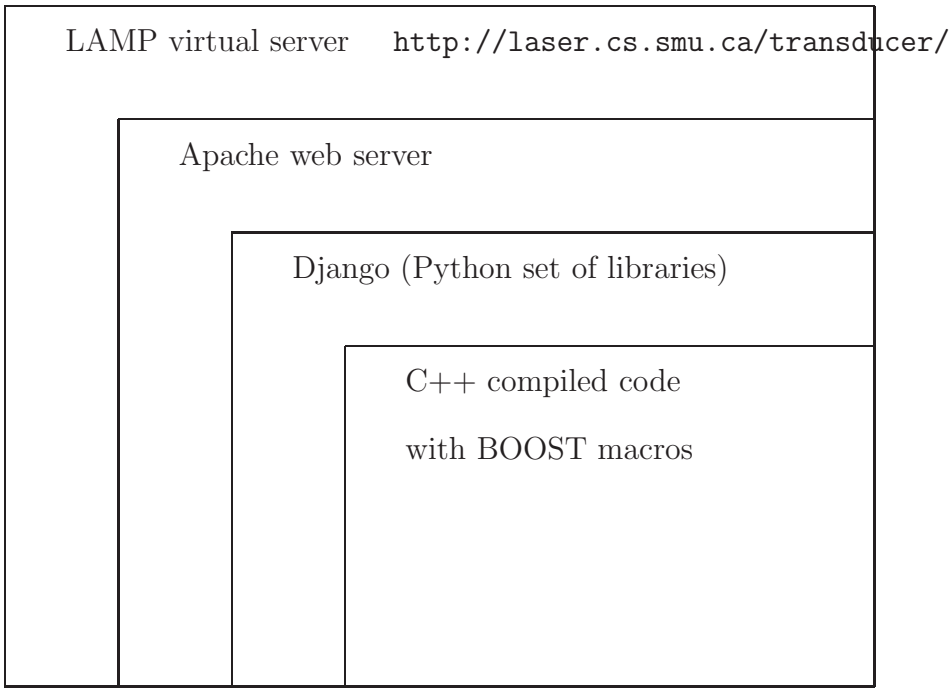

Figure 4: LA.SER. architecture.

`http://www.djangoproject.com/`
`http://www.boost.org/`

- Sample Grail file (automaton accepting ab^*):

```
(START) |- 1
1 a 2
2 b 2
2 -| (FINAL)
```

9 More theory in progress

What about decidability for properties like unique, or finite delay, decodability?

• **Look at this**. L is a code IFF

$$\forall\, x \in L,\ (\hat{t}_L^{-1} \circ \hat{t}_L)(x) \cap (L - x) = \emptyset,$$

where $\hat{t}_L(w) = wL^*$.

Compare with $\mathcal{P}_{1,\hat{t}}$

$$\forall\, x \in L,\ \ \hat{t}(x) \cap (L - x) = \emptyset.$$

• A similar $\hat{t}_L$ works also for finite delay decodability. Unfortunately, $\hat{t}_L$ depends on L.

NOTE: More references should be added...

References

[1] M.P. Béal, O. Carton, C. Prieur and J. Sakarovitch. Squaring transducers: an efficient procedure for deciding functionality and sequentiality. *Theoretical Computer Science* **292**:1 (2003), 45-63.

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

[3] J. Devolder, M. Latteux, I. Litovsky and L. Staiger. Codes and infinite words. *Acta Cybernetica* **11** (1994), 241–256.

[4] M. Domaratzki. Trajectory-based codes. *Acta Informatica* **40**:6-7 (2004), 491–527.

[5] M. Domaratzki and K. Salomaa. Codes defined by multiple sets of trajectories. *Theoretical Computer Science* **366**:3 (2006), 182–193.

[6] T. Head and A. Weber. Deciding code related properties by means of finite transducers. In R. Capocelli, A. de Santis and U. Vaccaro (eds). *Sequences II, Methods in Communication, Security,*

and Computer Science, 260–272. Springer Berlin, 1993.

[7] H. Jürgensen. Syntactic monoids of codes. *Acta Cybernetica* **14** (1999), 117–133.

[8] H. Jürgensen and S. Konstantinidis. The hierarchy of codes. In Z. Ésik (ed). *Fundamentals of Computation Theory, FCT'93. Lecture Notes in Computer Science* ***710***, 50–68. Springer-Verlag Berlin, 1993.

[9] H. Jürgensen and S. Konstantinidis. Codes. In [20], pp 511–607.

[10] H. Jürgensen, K. Salomaa and S. Yu. Transducers and the decidability of independence in free monoids. *Theoretical Computer Science* **134** (1994), 107–117.

[11] H. Jürgensen and S.S. Yu. Relations on free monoids, their independent sets, and codes. *Intern. J. Computer Mathematics* **40** (1991), 17–46.

[12] H. Jürgensen and S.S. Yu. *Dependence systems and hierarchies of families of languages*. Unpublished manuscript, 1995

[13] L. Kari and S. Konstantinidis. Language equations, maximality and error-detection. *J. Computer and System Sciences* **70** (2005), 157–178.

[14] L. Kari, S. Konstantinidis and P. Sosík. On properties of bond-free DNA languages. *Theoretical Computer Science* **334**:1-3 (2005), 131–159.

[15] S. Konstantinidis. Transducers and the properties of error-detection, error-correction, and finite-delay decodability. *J. Universal Computer Science* **8**:2 (2002), 278–291.

[16] N.H. Lam. *Finite maximal infix codes.* Technical report 98/A2, Institute of Mathematics, Vietnam National Centre for Natural Science and Technology, 1998.

[17] N.H. Lam. Finite maximal solid codes. *Theoretical Computer Science* **262**:1 (2001), 333–347.

[18] V. Levenshtein. Certain properties of code systems. *Soviet Physics Dokl.* **6** (1962), 858–860.

[19] A.A. Markov. Nonrecurrent Coding. *Problemy Kibernet.* **8** (1962), 169–180 (in Russian).

[20] G. Rozenberg and A. Salomaa (eds). *Handbook of Formal Languages, Vol. 1.* Springer-Verlag, Berlin, 1997.

[21] A.A. Sardinas and G.W. Patterson. A necessary and suffcient condition for the unique decomposi-

tion of coded messages. *IRE Intern. Convention Record* **8** (1953), 104–108.

[22] H.J. Shyr. *Free Monoids and Languages*. Hon Min Book Company, Taichung, Taiwan, 1991.

[23] H.J. Shyr and G. Thierrin. Codes and binary relations. *Séminaire d'Algèbre Paul Dubreil, Paris 1975–1976 (29ème Année), Lecture Notes in Mathematics* **586** (1977), 180–188.

[24] S. Yu. Regular Languages. In [20], pp 41–110.

[25] S.S. Yu. *Languages and Codes*. Tsang Hai Publishing Co., Taichung, Taiwan, 2005.

[26] D.L. Van, K.V. Hung and P.T. Huy. Codes and length-increasing transitive binary relations. *Theoretical Aspects of Computing – ICTAC 2005, Lecture Notes in Computer Science* **3722** (2005), 29–48.