ON MORPHISMS OF TRACE MONOIDS

by
Edward OCHMANSKI
Institute of Computer Science
Polish Academy of Sciences
P.O. Box 22, PKiN
00 - 901 WARSZAWA

Morphisms of trace monoids are studied in the paper. Their relations to morphisms of free monoids are described.Injective trace morphisms are investigated and decidability of injectivity is proved for morphisms of abelian trace monoids. An effective general characterization of trace morphisms preserving regularity (recognizability) is given.

INTRODUCTION

Trace monoids were used by Cartier/Foata [3] for studying of rearrangements of words. Mazurkiewicz [10] has used them as a mathematical model of concurrent processes. The eighties years have brought several meaningful results of trace theory, as well in describing behaviour of concurrent systems as in the field of algebraic properties of trace monoids (see [1,4,11] for a general picture).

It seems however that morphisms of trace monoids are still unexplored. Some results about them are presented in this paper. Relations of trace morphisms to morphisms of free monoids are investigated (Chapters 2 and 6). Some results about injective trace morphisms (Chapters 3, 4 and 6) and an effective characterization of trace morphisms preserving regularity (Chapter 5) are given.

Many of previous results about trace monoids (from [6,12,13] among others) are utilized in the paper. But the Duboc's [8] characterization of solutions of the equation $a\beta = \beta a$ in trace monoids seems to be essential for studying of trace morphisms. This characterization was also obtained (although without an explicit formulation) by Cori/Métilvier [9].

Although the paper is purely mathematical, it is strictly related to concurrent systems. Namely, any morphism can be considered as an interpretation of actions of such systems.

A knowledge of basic notions of formal language - and monoid-theories ([9] for instance) is assumed. Some familiarity with trace monoids is needed, but formally not necessary.
1 - BASIC NOTIONS

A concurrent alphabet \((A, I)\) is a finite set \(A\) with a symmetric and irreflexive relation \(I\), called independence. The complement relation \(D = A \times A \setminus I\) is called dependence. A concurrent alphabet \((A, I)\) defines a trace monoid \(T = M(A, I)\) as a monoid given by a presentation \((A^*, \{ab = ba | (a, b) \in I\})\). Members of a trace monoid are called traces and they are denoted by \([w]\) for \(w \in A^*\). Any trace is a finite set of equivalent words. Subsets of a trace monoid are called trace languages and are denoted by \([L] = \{[w] | w \in L\}\) for \(L \subseteq A^*\). A closure (with respect to \(I\)) operation \(C_1: 2^A \rightarrow 2^A\) is defined as \(C_1(L) = \bigcup \{[w] | w \in L\}\).

For a trace \(\alpha = [w]\) we denote \(\text{alph}(\alpha) = \text{alph}(w)\) as the set of all letters in \(w\). For traces \(\alpha\) and \(\beta\) we write \(\alpha \beta\) iff \(\text{alph}(\alpha) \times \text{alph}(\beta) \subseteq I\). Otherwise we write \(\alpha \not\beta\). We say that a trace \(\alpha\) is connected iff \(\text{alph}(\alpha) \not\emptyset\) is connected with respect to the dependency relation \(D = A \times A \setminus I\). We say that a trace language \(K\) is connected iff all traces in \(K\) are connected. A trace \(\gamma[e]\) is a component of a trace \(\alpha[e]\) iff \(\gamma\) is connected and \(\alpha = \beta\gamma\) for some trace \(\beta\) independent of \(\gamma\) (i.e. \(\beta I\gamma\)). The decomposition \(\alpha/\gamma\) of a trace \(\alpha\) is the set of all its components if \(\alpha[e]\) and \(\{[\epsilon]\}\) if \(\alpha = [\epsilon]\).

The set \(\text{REX}\) of rational expressions over \(A\) is the least set of words over \(A = \{0, e, (, ), U, ., *\}\) such that \(\{0, e, a | a \in A\} \in \text{REX}\) and if \(R_1, R_2 \in \text{REX}\) then \((R_1 U R_2), (R_1 . R_2), R_1^* \in \text{REX}\). Rational expressions define trace languages: \(s(0) = \emptyset, s(e) = \{[\epsilon]\}, s(a) = \{[a]\}\) for \(a \in A\), \(s(R_1 U R_2) = s(R_1) U s(R_2), s(R_1 . R_2) = s(R_1) . s(R_2), s(R_1^*) = (s(R_1))^*\).

A trace language \(K\) is called:

- rational iff \(s(R) = K\) for some rational expression \(R\),
- regular iff \(UK\) is a regular word language.

Remark -

1°) Rational expressions can also be considered as a tool for characterizing regular trace languages (see [13]).

2°) Regular trace languages are also called recognizable. It seems that it is better to use the term "recognizable" in the meaning "recognizable by a device" (Zielonka's [15] asynchronous automata, for instance).

3°) Rational and regular trace languages are called in [1] existentially regular and consistently regular, respectively.

A mapping \(f\) from a monoid \(M_1\) into a monoid \(M_2\) is a morphism iff \(f(xy) = f(x)f(y)\) for any \(x, y \in M_1\). It is a trace morphism if \(M_1, M_2\) are trace monoids and a word morphism if \(M_1, M_2\) are free.

2 - TRACE MORPHISMS AND THEIR RELATIONS TO WORD MORPHISMS

Let \(T_1 = M(A_1, I_1)\) and \(T_2 = M(A_2, I_2)\) be trace monoids.