RWTH Aachen

Department of Computer Science
Technical Report

Regular MSC Languages

Benedikt Bollig, Martin Leucker, and Thomas Noll

ISSN 0935–3232 · Aachener Informatik Berichte · AIB-2001-05

RWTH Aachen · Department of Computer Science · April 2001

Regular MSC Languages

Benedikt Bollig, Martin Leucker, and Thomas Noll

Lehrstuhl für Informatik II
Aachen University of Technology, Germany
{bollig, leucker, noll}@informatik.rwth-aachen.de

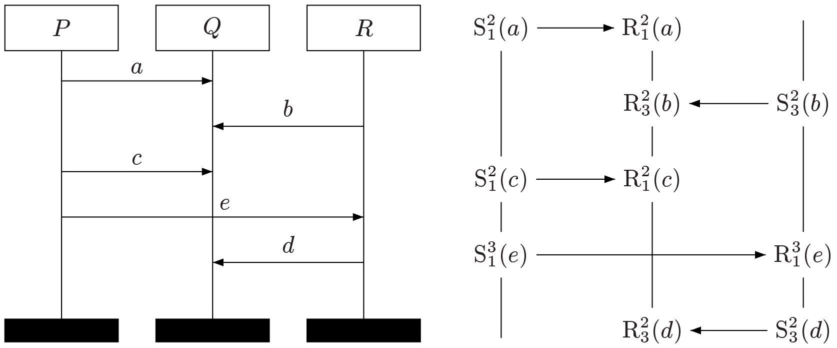
Abstract. In this paper we establish the concept of regularity for languages consisting of Message Sequence Charts (MSCs) under the aspect of realisability. To this aim we characterise their behaviour by string languages, and we give a natural definition of regularity in terms of appropriate right congruences. Moreover, we present a class of accepting automata, and, using this characterisation, establish several decidability and closure properties of MSC languages. We also provide a logical characterisation in terms of a monadic second order logic interpreted over MSCs. In contrast to existing work on regular MSC languages, our approach is not tailored to a fixed communication medium (as a FIFO channel, for example) but is applicable to a broad range of channel types like mixtures of stacks and FIFOs.

1 Introduction

Components of distributed systems usually communicate with each other via message passing. A sender process sends a message over a channel from which it is taken by the receiver process. A prominent formalism to model this kind of systems are *Message Sequence Charts* (MSCs) [9]. They are standardised, can be denoted textually as well as graphically, and are often employed in industry. Furthermore, they are quite similar to UML's sequence charts [3].

A message sequence chart defines a set of processes and a set of communication actions between these processes. In the visual presentation of an MSC, processes are drawn as vertical lines. A labelled arrow from one line to a second line corresponds to the communication event of sending the label value from the first process to the second. Since the vertical lines are interpreted as time axes, there is the general rule that arrows must not go "upwards" since this would describe a situation that a message is received before it was sent. Figure 1 (a) gives an example of an MSC.

When one considers the exact behaviour of an MSC, i.e., the sequences of actions which may be observed when the system is executed, one distinguishes between the so-called *visual-order semantics* and the *causal-order semantics* of an MSC. The visual order assumes that the events are ordered as shown in the MSC. That is, the events on a single process line are linearly ordered, and sending events precede their corresponding receiving events. For example, process Q in Figure 1 (a) reads an a symbol before it can read a b. In the causal order based semantics, a concrete communication medium between the processes is taken into account, for example, a first-in-first-out (FIFO) channel. Furthermore,



(a) (b)

Fig. 1. An MSC and its formalisation

read events on the same process line are not ordered unless they are "causality dependent". For example, read event b may occur before reading a: since process P might have sent a after R has sent b, and assuming a single FIFO incoming channel for Q, in practice Q will potentially read b before a. Note, that under the same communication assumption, reading c must occur before reading d. In the following, we focus on the visual order approach. But we should mention that—with minor modifications—our approach also works wrt. the causal order.

There has been spent a lot of effort for analysing, specifying, verifying, and synthesising MSCs [15, 13, 14, 2, 8, 7, 11, 1, 6, 12, 4]. However, most of the work was carried out assuming a fixed communication medium. In particular, [8] studies a notion of regular MSC languages which is based on a FIFO model. This restricts the applicability of this approach because there are numerous kinds of channels which, for example, can be designed in a FIFO manner, can alternatively behave as a *stack* or as a *multi-set*, and may have a fixed *capacity* or not. Just imagine nondeterministic message passing systems that make use of different channels. In this paper, we concentrate ourselves on the realisability of message passing systems in terms of regularity. We provide a characterisation of regular MSC languages by means of the Nerode right congruence, finite automata, asynchronous finite automata, and models of MSO formulas interpreted over MSCs. Thus, we provide the formal theory for further research.

2 Message Sequence Charts and Their Linearisations

In this section, we propose our formal model for MSCs, and we establish a string representation which describes their behaviour in a linear way.

2.1 Message Sequence Charts

Let $\mathcal{P}_N := \{1, \ldots, N\}$ be a nonempty set of *processes* and Λ a *message alphabet*. Let further $\Sigma_{\mathcal{S}} := \{\mathrm{S}_p^q(\lambda) \mid p, q \in \mathcal{P}_N,\ p \neq q,\ \lambda \in \Lambda\}$ and $\Sigma_{\mathcal{R}} := \{\mathrm{R}_p^q(\lambda) \mid p, q \in$

$\mathcal{P}_N,\ p \neq q,\ \lambda \in \Lambda\}$ denote the sets of *send* and *receive actions*, respectively, $\Sigma := \Sigma_{\mathcal{S}} \cup \Sigma_{\mathcal{R}}$ their union. An action $\mathrm{S}_p^q(\lambda)$ stands for sending a message λ from process p to process q, an action $\mathrm{R}_p^q(\lambda)$ for the corresponding receive action, which is then executed by process q. In this sense, $\mathit{Corr} := \{(\mathrm{S}_p^q(\lambda), \mathrm{R}_p^q(\lambda)) \mid p, q \in \mathcal{P}_N,\ p \neq q,\ \lambda \in \Lambda\}$ relates those actions which belong together. From now on, all premises and definitions are made wrt. a fixed set $\mathcal{P}_N$ of processes and a fixed message alphabet Λ where Σ denotes the corresponding set of actions.

A *message sequence chart* (MSC) is a tuple $M = (\{E_p\}_{p\in\mathcal{P}_N}, \{\preceq_p\}_{p\in\mathcal{P}_N}, f, L)$ where $\{E_p\}_{p\in\mathcal{P}_N}$ is a family of pairwise disjoint nonempty finite sets of so-called *events* each of which is totally ordered by a well-founded relation $\preceq_p \subseteq E_p \times E_p$. (For simplicity, we consider $\preceq_p$ as a relation over $E := \bigcup_{p\in\mathcal{P}_N} E_p$, the set of all events.) Let $P : E \cup \Sigma \to \mathcal{P}_N$ yield the process an event or an action belongs to, i.e., $P(e) = p$ iff $e \in E_p$, $P(\mathrm{S}_p^q(\lambda)) = p$, and $P(\mathrm{R}_p^q(\lambda)) = q$. M is required to induce a partition $E = S \cup R$ of the events into send (S) and receive events (R) such that f is a bijective mapping from S to R which associates each send event with a receive event satisfying the following:

- There is no sequence $e_1, \ldots, e_n$ of events such that $f(e_n) = e_1$ and for all $i \in \{1, \ldots, n-1\}$ either $f(e_i) = e_{i+1}$ or $e_i \preceq_{P(e_i)} e_{i+1}$.
- $L : E \to \Sigma$ provides information about the messages being interchanged by communicating events, whereby $e \in S$ implies both

$$L(e) = \mathrm{S}_{P(e)}^{P(f(e))}(\lambda) \text{ and } L(f(e)) = \mathrm{R}_{P(e)}^{P(f(e))}(\lambda)$$

  for some $\lambda \in \Lambda$.

For example, Figure 1 (b) presents a formal version of the MSC depicted in Figure 1 (a).

The *visual order* of an MSC is denoted by $\preceq \subseteq E \times E$ and is defined to be the reflexive and transitive closure of $\bigcup_{p\in\mathcal{P}_N} \preceq_p \cup \{(e, f(e)) \mid e \in S\}$. We easily verify

Proposition 1. *The visual order $\preceq$ is a partial order, i.e., it is reflexive, antisymmetric and transitive.*

Considering an MSC and all of its events, a partial execution (configuration) can be described by a downwards closed subset of events, containing those events which occurred so far. So let $M = (\{E_p\}_{p\in\mathcal{P}_N}, \{\preceq_p\}_{p\in\mathcal{P}_N}, f, L)$ be an MSC. A *configuration* of M is subset E' of E satisfying $E' = \downarrow E' := \{e \mid \exists e' \in E' : e \preceq e'\}$. $\mathit{Conf}(M)$ denotes the set of configurations of M. The execution of M may be described by a transition relation $\longrightarrow_M \subseteq \mathit{Conf}(M) \times \Sigma \times \mathit{Conf}(M)$ where $c \xrightarrow{\sigma}_M c'$ iff there exists $e \in E$ such that $L(e) = \sigma$ and $c' = c \cup \{e\}$.

2.2 MSC Words

In order to ascribe regularity of collections of MSCs to regularity of word languages, we will identify an MSC with a set of words that are called linearisations or MSC words. A linearisation represents a possible execution sequence of the events occurring in an MSC. In order to justify this view, it is necessary to guarantee that—up to isomorphism—from a set of linearisations a corresponding MSC can be unambiguously inferred and vice versa. We are then able to define an equivalence on MSC words whose equivalence classes on their own determine exactly one MSC and, as a whole, stand for the set of all MSCs.

Anyway, there remains the problem how to define an MSC word. Have a look at $w = \mathrm{S}_1^2(a)\mathrm{S}_1^2(a)\mathrm{R}_1^2(a)\mathrm{R}_1^2(a) \in \Sigma^*$. w might define the MSC M_1 given in Figure 4. But as w is also a correct linearisation of the MSC aside, relating the first and the fourth position of w we could likewise imagine that w represents M_2. Differing from [8] where an MSC like M_2 is not allowed, we therefore cannot unambiguously correlate a word contained in Σ^* with an MSC. In particular, if we make use of nondeterministic channels (which might allow MSCs to behave both in a FIFO manner and as a stack, for example), we need some further information about which positions belong together. For this purpose, each position of a word $w \in \Sigma^*$ is equipped with a natural number indicating the matching positions (namely those showing the same number). The words $\alpha_1, \alpha_2 \in (\Sigma \times \mathbb{N})^*$ from Figure 2 are such MSC words. Notice that α_1 will determine the MSC M_1, whereas M_2 will emerge from α_2.

We want to develop the respective theory step by step and first call a word $\alpha \in (\Sigma \times \mathbb{N})^*$

- *proper* iff for all $(\sigma, \tau) \in \mathit{Corr}$, $\pi \in \mathbb{N}$, and prefixes α' of α, $|\alpha'|_{(\tau,\pi)} \leq |\alpha'|_{(\sigma,\pi)} \leq |\alpha'|_{(\tau,\pi)} + 1$, and
- *complete* iff it is proper and for all $(\sigma, \tau) \in \mathit{Corr}$, and $\pi \in \mathbb{N}$, $|\alpha|_{(\sigma,\pi)} = |\alpha|_{(\tau,\pi)}$.

Thus, in a proper word, every receiving event (we sometimes refer to positions of MSC words as events) must be preceded by a sending counterpart, and, for each number π, at most one outstanding sending event is admitted.

Definition 1 (MSC Word). *A word* $\alpha = \frac{\sigma_1}{\pi_1} \ldots \frac{\sigma_\ell}{\pi_\ell} \in (\Sigma \times \mathbb{N})^*$ *is called an* MSC word *iff it is complete. The set of all MSC words is denoted by* MW, *the set of proper words by* PW.

To see some examples, look at the words $\alpha_1, \ldots, \alpha_4 \in (\Sigma \times \mathbb{N})^*$ as given in Figure 2. α_1 and α_2 are MSC words, whereas α_3 is certainly proper but not complete and α_4 is not even proper. We will refer to α_1 and α_2 as exemplary MSC words through the rest of the paper.

Given a proper word $\alpha = \frac{\sigma_1}{\pi_1} \ldots \frac{\sigma_\ell}{\pi_\ell} \in \mathrm{PW}$, we can determine which positions are *matching*. For $i, j \in \{1, \ldots, \ell\}$, we write $i \searrow_\alpha j$ iff

- $i < j$ and
- $(\sigma_i, \sigma_j) \in \mathit{Corr}$ and furthermore
- $j = \mathit{min}\{k \mid k > i \text{ and } \pi_k = \pi_i \text{ and } (\sigma_i, \sigma_k) \in \mathit{Corr}\}$.

Referring to the above example, the matching positions of α_1 and α_2 may be illustrated as shown in Figure 3, i.e., $1 \searrow_{\alpha_1} 3$ and $2 \searrow_{\alpha_1} 4$ as well as $1 \searrow_{\alpha_2} 4$ and $2 \searrow_{\alpha_2} 3$.

$$\alpha_1 = \begin{matrix} \mathrm{S}^2_1(a) & \mathrm{S}^2_1(a) & \mathrm{R}^2_1(a) & \mathrm{R}^2_1(a) \\ 1 & 3 & 1 & 3 \end{matrix} \qquad \alpha_2 = \begin{matrix} \mathrm{S}^2_1(a) & \mathrm{S}^2_1(a) & \mathrm{R}^2_1(a) & \mathrm{R}^2_1(a) \\ 1 & 2 & 2 & 1 \end{matrix}$$

$$\alpha_3 = \begin{matrix} \mathrm{S}^2_1(a) & \mathrm{S}^1_2(b) & \mathrm{R}^1_2(b) & \mathrm{S}^1_2(b) & \mathrm{R}^2_1(a) \\ 2 & 1 & 1 & 1 & 2 \end{matrix} \qquad \alpha_4 = \begin{matrix} \mathrm{S}^2_1(a) & \mathrm{S}^1_2(b) & \mathrm{R}^1_2(b) & \mathrm{S}^1_2(b) & \mathrm{S}^2_1(a) \\ 2 & 1 & 1 & 1 & 2 \end{matrix}$$

Fig. 2. Exemplary words

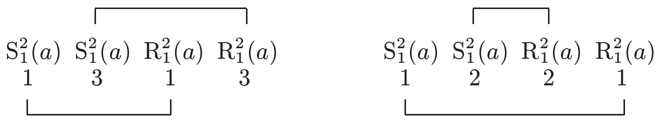


Fig. 3. Matching positions

Falling back on the matching relation, a word $\alpha = \genfrac{}{}{0pt}{}{\sigma_1}{\pi_1} \ldots \genfrac{}{}{0pt}{}{\sigma_\ell}{\pi_\ell} \in \mathrm{MW}$ generates an MSC $M(\alpha) := (\{E_p\}_{p \in \mathcal{P}_N}, \{\preceq_p\}_{p \in \mathcal{P}_N}, f, L)$ where

- $E_p = \{n \in \{1, \ldots, \ell\} \mid P(\sigma_n) = p\}$,
  $S = \{n \in \{1, \ldots, \ell\} \mid \sigma_n \in \Sigma_{\mathcal{S}}\}$,
  $R = \{n \in \{1, \ldots, \ell\} \mid \sigma_n \in \Sigma_{\mathcal{R}}\}$,
- $n \preceq_p m$ iff $n, m \in E_p$ and $n \leq m$,
- $f(n) = m$ iff $n \searrow_\alpha m$, and
- $L(n) = \sigma_n$.

For example, α_1 generates the MSC M_1 illustrated in Figure 4, whereas α_2 generates M_2.

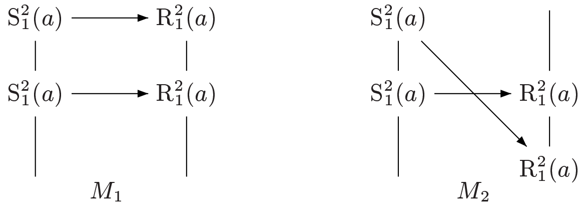


Fig. 4. MSCs generated by α_1 and α_2

Moreover, there is no problem in extending the above definition to proper words which then accordingly determine so-called “proper” MSCs.

Two different proper words may stand for—up to isomorphism—one and the same MSC or prefix of an MSC, respectively. In this sense, we define equivalence relations $\approx \subseteq \mathrm{PW} \times \mathrm{PW}$ and $\sim \subseteq \mathrm{MW} \times \mathrm{MW}$. The first identifies words with equivalent projections onto the second component, whereas the latter, as introduced further below, allows to permute the positions of an MSC word.

So, for $\alpha = \genfrac{}{}{0pt}{}{\sigma_1}{\pi_1} \cdots \genfrac{}{}{0pt}{}{\sigma_\ell}{\pi_\ell} \in \mathrm{PW}$ and $\beta = \genfrac{}{}{0pt}{}{\tau_1}{\rho_1} \cdots \genfrac{}{}{0pt}{}{\tau_m}{\rho_m} \in \mathrm{PW}$, let $\alpha \approx \beta$ iff

(1) $\ell = m$ and furthermore
(2) $\sigma_i = \tau_i$ for all $i \in \{1, \ldots, \ell\}$ and
(3) $i \searrow_\alpha j$ iff $i \searrow_\beta j$.

Remark 1. $\approx$ is an equivalence relation.

For instance, let α_1^n emerge from α_1 by replacing 3 in the natural-number component with some $n \in \mathbb{N}$. Then $\alpha_1^n \in \mathrm{MW}$ iff $n \neq 1$, and $\alpha_1^n \in \mathrm{MW}$ implies $\alpha_1 \approx \alpha_1^n$. But notice that $\alpha_1 \not\approx \alpha_2$ because condition (3) in the definition of $\approx$ is violated.

For a proper word $\alpha = \genfrac{}{}{0pt}{}{\sigma_1}{\pi_1} \cdots \genfrac{}{}{0pt}{}{\sigma_\ell}{\pi_\ell} \in \mathrm{PW}$, let $\mathit{open}(\alpha) \subseteq \Sigma_{\mathcal{S}} \times \mathbb{N}$ denote the set of those sending events which are not followed by a matching receive event, i.e., $\mathit{open}(\alpha) := \{(\sigma_i, \pi_i) \mid \sigma_i \in \Sigma_{\mathcal{S}}$ and there is no $j > i$ such that $i \searrow_\alpha j\}$. A word $\alpha \in \mathrm{PW}$ is called in *normal form* iff for all prefixes $\genfrac{}{}{0pt}{}{\sigma_1}{\pi_1} \cdots \genfrac{}{}{0pt}{}{\sigma_k}{\pi_k}$ of α, $\sigma_k \in \Sigma_{\mathcal{S}}$ implies $\pi_k = \mathit{min}\{\pi \in \mathbb{N} \mid (\sigma_k, \pi) \notin \mathit{open}(\genfrac{}{}{0pt}{}{\sigma_1}{\pi_1} \cdots \genfrac{}{}{0pt}{}{\sigma_{k-1}}{\pi_{k-1}})\}$. Thus, for every sending event, the lowest available number is chosen. Note that every equivalence class in $\mathrm{PW}/_\approx$ contains exactly one word in normal form. For $\alpha \in \mathrm{PW}$, let furthermore $\mathit{nf}(\alpha) = \beta$ iff $\alpha \approx \beta$ and β is in normal form. For instance, $\mathit{nf}(\alpha_1) = \alpha_1^2$, whereas α_2 is already in normal form so that $\mathit{nf}(\alpha_2) = \alpha_2$.

In the following, we will not distinguish between $\approx$-equivalent words.

Definition 2 (MSC Word Language). *A set* $\mathcal{L} \subseteq \mathrm{MW}$ *is called* MSC word language *iff* $\mathcal{L} = \mathcal{L}^\approx$ *where* $\mathcal{L}^\approx$ *denotes the* $\approx$*-closure of* $\mathcal{L}$.

Characterising regular languages within the scope of MSCs, a certain restriction of words and MSCs will prove to be important. Given a natural number B, $\alpha \in \mathrm{MW}$ is called *B-bounded* iff for all prefixes α' of α and actions $\sigma \in \Sigma_{\mathcal{S}}$, $|\mathit{open}(\alpha') \cap \{(\sigma, \pi) \mid \pi \in \mathbb{N}\}| \leq B$. Examples for 2-bounded MSC words are α_1 and α_2. Note that we could likewise call α B-bounded iff

- for all prefixes α' of α, $|\mathit{open}(\alpha')| \leq B$, or also iff
- for all prefixes α' of α and $p \in \mathcal{P}_N$, $|\mathit{open}(\alpha') \cap \{(\sigma, \pi) \mid \pi \in \mathbb{N}, P(\sigma) = p\}| \leq B$.

Our results also hold for these definitions.

In order to finally relate MSCs to the rich theories of languages and automata over words, the concept of linearisations of an MSC is essential. We call an MSC word $\alpha = \genfrac{}{}{0pt}{}{\sigma_1}{\pi_1} \cdots \genfrac{}{}{0pt}{}{\sigma_\ell}{\pi_\ell} \in (\Sigma \times \mathbb{N})^*$ a *linearisation* of an MSC $M = (\{E_i\}_{i \in \mathcal{P}_N}, \{\preceq_i\}_{i \in \mathcal{P}_N}, f, L)$ with a set of events $E = \{e_1, \ldots, e_\ell\}$ iff there are configurations $c_1, \ldots, c_\ell \in \mathit{Conf}(M)$ with $\emptyset \xrightarrow{\sigma_1}_M c_1 \xrightarrow{\sigma_2}_M \cdots \xrightarrow{\sigma_\ell}_M c_\ell$ and there is a bijective mapping $\chi : E \to \{1, \ldots, \ell\}$ such that

- for all $e \in E$, $L(e) = \sigma_{\chi(e)}$, and
- for all $e \in S$, $e' \in R$, $f(e) = e'$ implies $\chi(e) \searrow_\alpha \chi(e')$.

$Lin(M)$ denotes the set of linearisations of M. For a set $\mathcal{M}$ of MSCs, we canonically define $Lin(\mathcal{M}) := \bigcup\{Lin(M) \mid M \in \mathcal{M}\}$. For instance, the exemplary word α_1 is a linearisation of the MSC M_1 shown in Figure 4, and α_2 is a linearisation of M_2. When, above, we spoke of isomorphism of two MSCs, we actually meant "inducing the same set of linearisations" instead.

An MSC is called B-bounded iff all of its linearisations are B-bounded. A collection of MSCs (a collection of MSC words, respectively) is B-bounded iff all members are B-bounded. Furthermore, we speak of *boundedness* in general iff we deal with B-boundedness for any B.

We gradually turn towards $\sim \subseteq \mathrm{MW} \times \mathrm{MW}$, the equivalence relation that takes permutations of positions into account. So let $\alpha = \genfrac{}{}{0pt}{}{\sigma_1 \cdots \sigma_\ell}{\pi_1 \ldots \pi_\ell} \in \mathrm{MW}$ and $\preceq$ be the partial order belonging to $M(\alpha)$. Positions $i, j \in \{1, \ldots, \ell\}$ are called *interdependent* (written $iD_\alpha j$) iff $i \preceq j$ or $j \preceq i$. Interdependency generalises the definition of the context sensitive independence relation given in [8], there defined for the restricted case of FIFO channels. Note that D_α is an equivalence relation. Proposition 2 characterises the circumstances under which two positions in an MSC word are interdependent. We underly the same setting as supposed when we defined the relation D_α and state

Proposition 2. *For positions $i < j$, it holds $iD_\alpha j$ iff*

- *$P(\sigma_i) = P(\sigma_j)$, or*
- *$i \searrow_\alpha j$, or*
- *there is a position k, $i < k < j$, such that both $iD_\alpha k$ and $kD_\alpha j$.*

Proof. Let $i < j$ and $iD_\alpha j$. Then we have $i \preceq j$. According to the definition of $\preceq$ in the context of $M(\alpha)$, $i \preceq j$ implies one of the following three cases:

- $i \preceq_{P(\sigma_i)} j$. But then $P(\sigma_i) = P(\sigma_j)$.
- $i \in S$, $j \in R$, and $f(i) = j$. Hence $\pi_i = \pi_j$ and $i \searrow_\alpha j$. Note that $j \in S$ and $i \in R$ implies both $f(j) \neq i$ and furthermore that $i \not\preceq j$, if $P(\sigma_i) \neq P(\sigma_j)$.
- $i \preceq k$ and $k \preceq j$ hold for a suitable k and hence $iD_\alpha k$ and $kD_\alpha j$.

Conversely, let $P(\sigma_i) = P(\sigma_j)$. Then $i \preceq_{P(\sigma_i)} j$ and consequently $i \preceq j$. From $\pi_i = \pi_j$ and $i \searrow_\alpha j$, it follows both $i \in S$ and $f(i) = j$, and, taking transitivity into account, $iD_\alpha kD_\alpha j$ yields $i \preceq j$.

By means of interdependency we then define the relation $\sim$ to be the least equivalence relation satisfying the following: If $\alpha = \genfrac{}{}{0pt}{}{\sigma_1 \cdots \sigma_i \sigma_{i+1} \cdots \sigma_\ell}{\pi_1 \ldots \pi_i \pi_{i+1} \ldots \pi_\ell}$ and $\alpha' = \genfrac{}{}{0pt}{}{\sigma_1 \cdots \sigma_{i+1} \sigma_i \cdots \sigma_\ell}{\pi_1 \ldots \pi_{i+1} \pi_i \ldots \pi_\ell}$ and not $iD_\alpha(i+1)$ then $\alpha \sim \alpha'$.

This section results in the following important properties of sets of linearisations which are induced by MSCs. In particular, they establish the expected connections between linearisations and the equivalence relations $\approx$ and $\sim$.

Proposition 3. *For an MSC M and $\alpha \in Lin(M)$, $Lin(M) = Lin(M(\alpha))$.*

Proposition 4. *For $\alpha \in \mathrm{MW}$, $Lin(M(\alpha)) = [\alpha]_{(\approx \cup \sim)^*}$.*

In the following, as words $\alpha \approx \beta$ basically represent "one and the same" word, we want sets of MSC words to be $\approx$-closed.

3 Regular MSC Word Languages and Their Automata

3.1 Regular MSC Word Languages

We already mentioned that the regularity of collections of MSCs will be defined in terms of regular MSC word languages. But as MSC words are defined over the infinite alphabet $\Sigma \times \mathbb{N}$, we have to modify the usual notion of regularity. In [10], a definition of regular word languages over infinite alphabets is proposed by providing an extended automata model that employs a finite transition relation but generates a behaviour catering for the fact that we deal with an infinite alphabet. However, we first constitute an algebraic characterisation of regularity by means of a slightly adapted version of the Nerode right congruence and then establish its equivalence to an automata model which is similar to that described in [10], but canonically modified with regard to MSCs. We imagine such a model to be a finite automaton making use of a finite window whose positions occur in the labellings of the transitions—as well as elements of Σ—and indicate where to store a symbol of the infinite alphabet $\Sigma \times \mathbb{N}$ (concerning send actions) and where to take it from (concerning receive actions), respectively.

Given an MSC word language $\mathcal{L}$, recall the Nerode right congruence $\equiv_{\mathcal{L}} \subseteq \mathrm{PW} \times \mathrm{PW}$:

$$\begin{array}{l} \alpha \equiv_{\mathcal{L}} \beta \text{ iff } \forall \gamma \in (\Sigma \times \mathbb{N})^* : \\ \qquad\qquad\qquad \alpha\gamma \in \mathcal{L} \text{ iff } \beta\gamma \in \mathcal{L} \end{array}$$

Let $\mathcal{L}$ be an MSC word language. For our purposes, $\approxeq_{\mathcal{L}} \subseteq \mathrm{PW} \times \mathrm{PW}$ extends the Nerode right congruence and is defined by $\alpha \approxeq_{\mathcal{L}} \beta$ iff $\mathit{nf}(\alpha) \equiv_{\mathcal{L}} \mathit{nf}(\beta)$. Figure 5 illustrates the definition of $\approxeq_{\mathcal{L}}$.

$$\begin{array}{ccc} \alpha & \approxeq_{\mathcal{L}} & \beta \\ \approx & & \approx \\ \mathit{nf}(\alpha) & \equiv_{\mathcal{L}} & \mathit{nf}(\beta) \end{array}$$

Fig. 5. Extending the Nerode right congruence

Definition 3 (Regular MSC Word Language). *An MSC word language $\mathcal{L}$ is called* regular *iff $\approxeq_{\mathcal{L}}$ has finite index.*

The following characterisation of regular MSC word languages prepares for proving their correspondence with a certain class of finite automata which we introduce further below.

Theorem 1. *Let $\mathcal{L}$ be an MSC word language. $\mathcal{L}$ is regular iff $\{\mathit{nf}(\alpha) \mid \alpha \in \mathcal{L}\}$ is a regular word language over $\Sigma \times Q$ where Q is a finite subset of $\mathbb{N}$.*

Proof. ($\Longrightarrow$) Let $\mathcal{L}' = \{\mathit{nf}(\alpha) \mid \alpha \in \mathcal{L}\} \subseteq (\Sigma \times \mathbb{N})^*$ and $Q = \bigcup\{\{\pi_1, \ldots, \pi_\ell\} \mid \frac{\sigma_1}{\pi_1} \ldots \frac{\sigma_\ell}{\pi_\ell} \in \mathcal{L}'\}$. We show that both Q is finite and that $\equiv_{\mathcal{L}'}$ has finite index. Then

$\mathcal{L}'$ is a regular word language, and since $\mathcal{L}$ is an MSC word language and, as such, $\approx$-closed, ${\mathcal{L}'}^{\approx} = \mathcal{L}$. So let us prove the finiteness of Q. Suppose that Q is infinite. Then there is a family $\{\alpha_n\}_{n\in\mathbb{N}} \subseteq \mathcal{L}'$ such that (without loss of generality) $max_\pi(\alpha_n) = n$ and $\alpha_n = \genfrac{}{}{0pt}{}{\sigma_{n_1}}{\pi_{n_1}} \genfrac{}{}{0pt}{}{\cdots}{\cdots} \genfrac{}{}{0pt}{}{\sigma_{n_{k_n}}}{n} \genfrac{}{}{0pt}{}{\cdots}{\cdots} \genfrac{}{}{0pt}{}{\sigma_{n_\ell}}{\pi_{n_\ell}}$ where $\sigma_{n_{k_n}}$ is the last send action occurring together with n. Let $\alpha'_n = \genfrac{}{}{0pt}{}{\sigma_{n_1}}{\pi_{n_1}} \genfrac{}{}{0pt}{}{\cdots}{\cdots} \genfrac{}{}{0pt}{}{\sigma_{n_{k_n}}}{n}$ such that $nf(\alpha'_n) = \alpha'_n$. Since $\approxeq_\mathcal{L}$ has finite index, there are $i, j \in \mathbb{N}$, $i < j$, with $\alpha'_i \approxeq_\mathcal{L} \alpha'_j$. By definition, $nf(\alpha'_i) \equiv_\mathcal{L} nf(\alpha'_j)$ and consequently $\alpha'_i \equiv_\mathcal{L} \alpha'_j$. But for $\gamma = \genfrac{}{}{0pt}{}{\sigma_{i_{k_i+1}}}{\pi_{i_{k_i+1}}} \genfrac{}{}{0pt}{}{\cdots}{\cdots} \genfrac{}{}{0pt}{}{\sigma_{i_\ell}}{\pi_{i_\ell}}$, $\alpha_i\gamma \in \mathcal{L}$ and $\alpha_j\gamma \notin \mathcal{L}$, because j does not occur in γ but is necessary to make $\alpha_j\gamma$ an MSC word. So assuming that Q is infinite leads to a contradiction.

Let us then show that $\equiv_{\mathcal{L}'}$ has finite index, again by contradiction. Suppose that $\equiv_{\mathcal{L}'}$ has infinite index. Then there is a family $\{\alpha_n\}_{n\in\mathbb{N}} \subseteq \mathrm{PW}$ such that for all $n, m \in \mathbb{N}$ with $n \neq m$, $\alpha_n \not\equiv_{\mathcal{L}'} \alpha_m$, and, without loss of generality, α_n is in normal form. Since $\approxeq_\mathcal{L}$ has finite index, there are $i, j \in \mathbb{N}$, $i \neq j$, satisfying $\alpha_i \approxeq_\mathcal{L} \alpha_j$ and hence $nf(\alpha_i) \equiv_\mathcal{L} nf(\alpha_j)$. Let $\gamma \in (\Sigma \times \mathbb{N})^*$ such that $nf(\alpha_i)\gamma \in \mathcal{L}'$. Then $nf(\alpha_j)\gamma$ must also be an MSC word, furthermore in normal form and therefore contained in $\mathcal{L}'$. (γ has to close the same send events in $nf(\alpha_i)$ as in $nf(\alpha_j)$. The "rest" of γ forms a proper and complete word in normal form so that $nf(\alpha_j)\gamma$ is in normal form.) We conclude $nf(\alpha_i) \equiv_{\mathcal{L}'} nf(\alpha_j)$ and hence $\alpha_i \equiv_{\mathcal{L}'} \alpha_j$ leading to a contradiction.

($\Longleftarrow$) Let us first slightly extend the definition of $\approx$ by writing $\alpha \approx \beta$ for suffixes α and β of MSC words iff there is a word $\alpha' \in \mathrm{PW}$ such that $\alpha'\alpha \approx \alpha'\beta$.

Let $\mathcal{L}' = \{nf(\alpha) \mid \alpha \in \mathcal{L}\} \subseteq (\Sigma \times Q)^*$, $Q \subseteq \mathbb{N}$ finite. Suppose $\approxeq_\mathcal{L}$ has infinite index. There exists a family $\{\alpha_n\}_{n\in\mathbb{N}} \subseteq \mathrm{PW}$ with $\alpha_n \not\approxeq_\mathcal{L} \alpha_m$ and hence $nf(\alpha_n) \not\equiv_\mathcal{L} nf(\alpha_m)$ for all $n, m \in \mathbb{N}$ with $n \neq m$. But since $\equiv_{\mathcal{L}'}$ has finite index, we can find $i, j \in \mathbb{N}$, $i \neq j$, satisfying $nf(\alpha_i) \equiv_{\mathcal{L}'} nf(\alpha_j)$. Let $\gamma \in (\Sigma \times \mathbb{N})^*$ such that $nf(\alpha_i)\gamma \in \mathcal{L}$. Then there exists a word γ', $\gamma' \approx \gamma$, with $nf(\alpha_i)\gamma' \in \mathcal{L}'$ and consequently (since $nf(\alpha_i) \equiv_{\mathcal{L}'} nf(\alpha_j)$) $nf(\alpha_j)\gamma' \in \mathcal{L}'$. As $\mathcal{L}$ is the $\approx$-closure of $\mathcal{L}'$, we finally get $nf(\alpha_j)\gamma \in \mathcal{L}$ and altogether $nf(\alpha_i) \equiv_\mathcal{L} nf(\alpha_j)$ resulting in a contradiction.

Notice that, from this, it immediately follows

Corollary 1. *A regular MSC word language is bounded.*

3.2 MSC Finite-Memory Automata

We now present a class of automata which characterises MSC word languages. The definition resembles that of [10] but is modified to catch the properties of MSCs.

Definition 4 (MSC Finite-Memory Automaton). *An* MSC finite-memory automaton *(MFA) is a quintuple of the form $\mathcal{A} = (S, r, \Delta, q_0, F)$, where*

- *S is a nonempty finite set of* states*,*
- *$r \geq 1$ is a natural number called* window length*,*

- $\Delta \subseteq S \times (\Sigma \times \{1, \ldots, r\}) \times S$ *is the* transition relation*,*
- $q_0 \in S$ *is the* initial state*, and*
- $F \subseteq S$ *is the set of* final states*.*

Let $\mathcal{A}$ be an MFA as above, and let $\mathit{Conf}_{\mathcal{A}} := S \times ((\Sigma_{\mathcal{S}} \times \mathbb{N}) \cup \{\#\})^r$ denote the (infinite) set of so-called *configurations* of $\mathcal{A}$. Then $\mathcal{A}$ defines a transition relation $\Longrightarrow \subseteq \mathit{Conf}_{\mathcal{A}} \times (\Sigma \times \mathbb{N}) \times \mathit{Conf}_{\mathcal{A}}$ as follows:

- For $\sigma \in \Sigma_{\mathcal{S}}$, $(s, \mathbf{w}) \overset{(\sigma,\pi)}{\Longrightarrow} (t, \mathbf{v})$ iff (σ, π) does not occur in $\mathbf{w}$ and there is a transition $(s, (\sigma, k), t) \in \Delta$ such that
  - $\mathbf{w}[k] = \#$,
  - $\mathbf{v}[k] = (\sigma, \pi)$, and
  - for each $l \neq k$, $\mathbf{w}[l] = \mathbf{v}[l]$.
- For $\sigma \in \Sigma_{\mathcal{R}}$, $(s, \mathbf{w}) \overset{(\sigma,\pi)}{\Longrightarrow} (t, \mathbf{v})$ iff there is a transition $(s, (\sigma, k), t) \in \Delta$ such that
  - $\mathbf{w}[k] = (\tau, \pi)$ where $(\tau, \sigma) \in \mathit{Corr}$, and
  - $\mathbf{v}[k] = \#$, and
  - for each $l \neq k$, $\mathbf{w}[l] = \mathbf{v}[l]$.

Thus, the meaning of a transition $(s, (\mathrm{S}_p^q(\lambda), k), t)$ is the following: if $\mathcal{A}$ is in state s, it is able to read an input symbol $(\mathrm{S}_p^q(\lambda), \pi)$, $\pi \in \mathbb{N}$, iff the kth position of its window is currently free and furthermore $(\mathrm{S}_p^q(\lambda), \pi)$ does not occur elsewhere in the window, i.e., there is no further open $(\mathrm{S}_p^q(\lambda), \pi)$-labelled send event. Taking the transition, the automaton stores $(\mathrm{S}_p^q(\lambda), \pi)$ in the kth position and enters state t. If, in contrast, the automaton reads an input symbol $(\mathrm{R}_p^q(\lambda), \pi)$, there has to be a transition $(s, (\mathrm{R}_p^q(\lambda), k), t)$ such that the kth position of the window currently shows the corresponding send symbol $(\mathrm{S}_p^q(\lambda), \pi)$. Replacing this symbol with $\#$, the automaton enters state t.

A *run* of $\mathcal{A}$ on a word $\alpha = \frac{\sigma_1}{\pi_1} \cdots \frac{\sigma_\ell}{\pi_\ell} \in (\Sigma \times \mathbb{N})^*$ is a corresponding sequence $(s_0, \mathbf{w}_0)(s_1, \mathbf{w}_1) \ldots (s_\ell, \mathbf{w}_\ell)$ of configurations such that

- $s_0 = q_0$,
- $\mathbf{w}_0 = \#^r$, and
- for each $i \in \{1, \ldots, \ell\}$, $(s_{i-1}, \mathbf{w}_{i-1}) \overset{(\sigma_i,\pi_i)}{\Longrightarrow} (s_i, \mathbf{w}_i)$.

The run is *accepting* iff in addition $s_n \in F$ and $\mathbf{w}_\ell = \#^r$. $\mathcal{L}(\mathcal{A}) := \{\alpha \mid$ there is an accepting run of $\mathcal{A}$ on $\alpha\}$ forms the language defined by $\mathcal{A}$. We conclude that matching events in an accepted word use one and the same position of the window for their "agreement".

Due to the conditions we laid down for making a transition and accepting a word, the automaton will accept MSC words only. Recall that a run is accepting as soon as it ends in a final configuration featuring an empty window. That way, completeness of accepted words is ensured. Furthermore, a receive symbol has to be preceded by a corresponding send symbol which, on its part, has to wait for the corresponding receive symbol before reading the identical send symbol again.

Thus, we make sure that an accepted word is proper. Moreover, the recognised language is $\approx$-closed because matching symbols can be read with—up to the MSC word condition—arbitrary natural numbers. Notice that a regular MSC word language is not necessarily $\sim$-closed. We sum up these considerations stating the following proposition.

Proposition 5. *Given an MFA $\mathcal{A}$, $\mathcal{L}(\mathcal{A})$ is an MSC word language.*

For example, let us consider the MFAs $\mathcal{A}_1$ and $\mathcal{A}_2$ illustrated by Figure 6 and behaving in a FIFO manner and as a stack, respectively. For the sake of clarity, let S stand for $\mathrm{S}_1^2(a)$ and R for $\mathrm{R}_1^2(a)$. Note that our MFAs permit only process 1 to send and only process 2 to receive a message a.

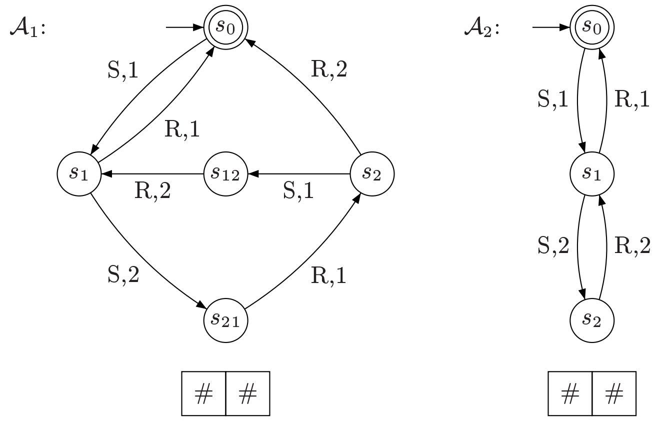


Fig. 6. Two MFAs

Recall our exemplary words α_1 and α_2. In fact, $\alpha_1 \in \mathcal{L}(\mathcal{A}_1)$ and $\alpha_2 \in \mathcal{L}(\mathcal{A}_2)$, but $\alpha_1 \notin \mathcal{L}(\mathcal{A}_2)$ and $\alpha_2 \notin \mathcal{L}(\mathcal{A}_1)$. An accepting run of $\mathcal{A}_1$ on α_1, for instance, first writes $(\mathrm{S}_1^2(a), 1)$ into the first position of the window and then $(\mathrm{S}_1^2(a), 3)$ into the second, whereupon the window is cleared in the same order, reading first $(\mathrm{R}_1^2(a), 1)$ and then $(\mathrm{R}_1^2(a), 3)$.

Theorem 2. *An MSC word language $\mathcal{L}$ is regular iff there is an MFA $\mathcal{A}$ such that*

$$\mathcal{L} = \mathcal{L}(\mathcal{A}).$$

Proof. According to Theorem 1, we specify respective automata.

($\Longrightarrow$) Let $\mathcal{A} = (S, \longrightarrow, q_0, F)$ be a finite automaton with $\longrightarrow \subseteq S \times (\Sigma \times Q) \times S$ for a finite set $Q \subseteq \mathbb{N}$. A corresponding MFA $\mathcal{A}' = (S', r, \Delta', q_0', F')$ satisfying $\mathcal{L}(\mathcal{A}') = \mathcal{L}(\mathcal{A})^{\approx}$ is given by

- $S' = S$,
- $r = |Q|$,

- $(s,(\sigma,k),t) \in \Delta$ iff $s \xrightarrow{(\sigma,k)} t$,
- $q'_0 = q_0$, and
- $F' = F$.

($\Longleftarrow$) Given an MFA $\mathcal{A} = (S, r, \Delta, q_0, F)$, let $Q = \{1, \ldots, r\}$ and $\mathcal{A}' = (S', \longrightarrow, q'_0, F')$ be the corresponding finite automaton satisfying $\mathcal{L}(\mathcal{A}') = \{nf(\alpha) \mid \alpha \in \mathcal{L}(\mathcal{A})\}$ as follows:

- $S' = S \times ((\Sigma_{\mathcal{S}} \times Q) \cup \{\#\})^r$,
- $\longrightarrow \subseteq S' \times (\Sigma \times Q) \times S'$ where $(s, \mathbf{w}) \xrightarrow{(\sigma,\pi)} (t, \mathbf{v})$ iff $(s, \mathbf{w}) \overset{(\sigma,\pi)}{\Longrightarrow} (t, \mathbf{v})$ and $\pi = min\{\pi' \in \mathbb{N} \mid (\sigma, \pi') \text{ does not occur in } \mathbf{w}\}$,
- $q'_0 = (q_0, \#^r)$,
- $F' = F \times \{\#^r\}$.

We recall some basic properties of regular word languages.

Theorem 3. *It is decidable whether a regular MSC word language is empty.*

Proof. Theorem 3 is a direct consequence of Theorem 1 and the proof of Theorem 2. Given an MFA $\mathcal{A}$, construct the corresponding finite automaton $\mathcal{A}'$, which is then checked for emptiness by means of standard techniques. And clearly, $\mathcal{L}(\mathcal{A}) = \emptyset$ iff $\mathcal{L}(\mathcal{A}') = \emptyset$.

Theorem 4. *Regular MSC word languages are closed under union, intersection, concatenation, and Kleene star.*

Proof. Let $\mathcal{A}_1 = (S_1, r_1, \Delta_1, q_{01}, F_1)$ and $\mathcal{A}_2 = (S_2, r_2, \Delta_2, q_{02}, F_2)$ be MFAs with $S_1 \cap S_2 = \emptyset$.

Closure under union. We build an MFA $\mathcal{A}^{\cup} = (S, r, \Delta, q_0, F)$, $q_0 \notin S_1 \cup S_2$, satisfying $\mathcal{L}(\mathcal{A}^{\cup}) = \mathcal{L}(\mathcal{A}_1) \cup \mathcal{L}(\mathcal{A}_2)$ as follows:

- $S = S_1 \cup S_2 \cup \{q_0\}$,
- $r = r_1 + r_2$,
- $(s,(\sigma,k),t) \in \Delta$ iff
  - $s = q_0$ and $(q_{01},(\sigma,k),t) \in \Delta_1$, or
  - $s = q_0$ and $(q_{02},(\sigma,k-r_1),t) \in \Delta_2$, or
  - $(s,(\sigma,k),t) \in \Delta_1$, or
  - $(s,(\sigma,k-r_1),t) \in \Delta_2$,
- $F = F_1 \cup F_2$.

The main idea is to assign to each of the automata an exclusive part of the window. A run firstly decides to enter either $\mathcal{A}_1$ or $\mathcal{A}_2$ and is henceforth limited to the respective part of the window.

Closure under intersection. Recall that

$$\varphi : \begin{cases} \{1, \ldots, r_1\} \times \{1, \ldots, r_2\} \to \{1, \ldots, r_1 \cdot r_2\} \\ (k, l) \mapsto (k-1) \cdot r_2 + l \end{cases}$$

is a bijective mapping. We build an MFA $\mathcal{A}^{\cap} = (S, r, \Delta, q_0, F)$ satisfying $\mathcal{L}(\mathcal{A}^{\cap}) = \mathcal{L}(\mathcal{A}_1) \cap \mathcal{L}(\mathcal{A}_2)$ according to

- $S = S_1 \times S_2$,
- $r = r_1 \cdot r_2$,
- $((s_1, s_2), (\sigma, k), (t_1, t_2)) \in \Delta$ iff
  - $(s_1, (\sigma, \mathrm{proj}_1(\varphi^{-1}(k)), t_1) \in \Delta_1$ and
  - $(s_2, (\sigma, \mathrm{proj}_2(\varphi^{-1}(k)), t_2) \in \Delta_2$,
- $q_0 = (q_{0_1}, q_{0_2})$,
- $F = F_1 \times F_2$.

The automata are synchronously working on a product window. For example, $\mathcal{A}_1$ and $\mathcal{A}_2$, reading a send symbol simultaneously, are eventually forced to fetch it back from the same position of the window.

Closure under concatenation. We build an MFA $\mathcal{A}^C = (S, r, \Delta, q_0, F)$ satisfying $\mathcal{L}(\mathcal{A}^C) = \mathcal{L}(\mathcal{A}_1) \cdot \mathcal{L}(\mathcal{A}_2)$. $\mathcal{A}^C$ is given by

- $S = S_1 \cup S_2$,
- $r = r_1 + r_2$,
- $(s, (\sigma, k), t) \in \Delta$ iff
  - $(s, (\sigma, k), t) \in \Delta_1$, or
  - $s \in F_1$ and $(q_{0_2}, (\sigma, k - r_1), t) \in \Delta_2$, or
  - $(s, (\sigma, k - r_1), t) \in \Delta_2$,
- $q_0 = q_{0_1}$, and
- $F = F_2$.

The construction follows closely that one used for the union of automata. The automata control separate parts of the window, respectively, but work successively.

Closure under Kleene star. We are out for an MFA $\mathcal{A}^* = (S, r, \Delta, q_0, F)$, $q_0 \notin S_1$ satisfying $\mathcal{L}(\mathcal{A}^*) = \mathcal{L}(\mathcal{A}_1)^*$. For this purpose, let $\mathcal{A}' = (S', \longrightarrow, q_0', F')$ be the finite automaton emerging from $\mathcal{A}_1$ according to the proof of Theorem 2. We specify $\mathcal{A}^*$ as follows:

- $S = S' \cup \{q_0\}$,
- $r = r_1$,
- $\Delta = \longrightarrow \cup \{(q_0, (\sigma, k), t) \mid \text{there is a } (q_0', (\sigma, k), t) \in \longrightarrow\} \cup \{(s, (\sigma, k), t) \mid s \in F'$ and there is a $(q_0', (\sigma, k), t) \in \longrightarrow\}$,
- $F = F' \cup \{q_0\}$.

As we have to ensure that the window is cleared from time to time, we make use of the finite automaton according to the proof of Theorem 2. Being in a state of F', we can emanate from an empty window when continuing reading.

We easily see that regular MSC word languages are not closed under complementation because the complement of a regular MSC word language is always unbounded (see also [10]).

In contrast to the general case of regular languages over infinite alphabets (see [10]), the inclusion problem is likewise decidable. This fact is of great importance with regard to the development of model checking algorithms.

Theorem 5. *Given MFAs $\mathcal{A}_1$ and $\mathcal{A}_2$, it is decidable whether $\mathcal{L}(\mathcal{A}_1) \subseteq \mathcal{L}(\mathcal{A}_2)$.*

Proof. Constructing finite automata according to the ones we already employed in the proof of Theorem 4, we are able to ascribe decidability to the case of regular languages over finite alphabets.

4 Regular MSC Languages and Their Automata

4.1 Regular MSC Languages

We now extend our theory of regular MSC word languages to collections of MSCs. Regularity of such a collection is ascribed to the regularity of the set of corresponding linearisations.

Definition 5 (Regular MSC Language). *A collection $\mathcal{M}$ of MSCs is called a* regular MSC language *iff $Lin(\mathcal{M})$ is a regular MSC word language.*

According to this definition, regular MSC languages are necessarily $\sim$-closed. Therefore they are not characterised by MSC finite-memory automata, which were presented in the previous section, and hence we have to develop a new automata model. Moreover it should be noted that a regular MSC language is bounded.

4.2 Asynchronous MSC Finite-Memory Automata

The following automata model employs different automata components each of which executes the actions of one single process. They communicate with each other over a window roughly as featured by an MFA. The crucial point is that this window is not anymore bounded by a natural number r. In order to preserve the $\sim$-closedness of the recognised languages, the components rather have to restrict themselves, whereas the window is communication medium only. For example, we could imagine an automata component that has to keep a send action waiting until it executes a certain receive action which, in turn, has to be preceded by a corresponding send action executed by another component. In fact, our view generalises the model proposed in [8], which has its origins in [16].

Definition 6. *An* Asynchronous MSC Finite-Memory Automaton *(AMFA) is a family $\mathcal{A} = (\{\mathcal{A}_p\}_{p \in \mathcal{P}_N}, r, Q_{in})$ of so-called* local automata *together with a natural number $r \geq 1$ and a set of* global initial states *Q_{in}. A local automaton is of the form $\mathcal{A}_p = (S_p, \Delta_p, F_p)$ where*

- *S_p is a nonempty finite set of* local states*,*
- *$\Delta_p \subseteq S_p \times (\Sigma_p \times \{1, \ldots, r\}) \times S_p$ is a set of* local transitions *(Σ_p contains the actions belonging to process p), and*
- *$F_p \subseteq S_p$ is a set of* local final states*.*

The set of initial states, Q_{in}, is a subset of $S_{\mathcal{A}}$ where $S_{\mathcal{A}} := \times_{p \in \mathcal{P}_N} S_p$ denotes the set of global states *of $\mathcal{A}$.*

Let $\mathcal{A}$ be an AMFA as above and $\mathit{Conf}_{\mathcal{A}} := S_{\mathcal{A}} \times \mathcal{P}((\Sigma_{\mathcal{S}} \times \mathbb{N}) \times \{1, \ldots, r\})$ denote the (infinite) set of its configurations. Let furthermore $\overline{s}[p]$ denote the pth component of a global state $\overline{s} \in S_{\mathcal{A}}$. We define a transition relation $\Longrightarrow \subseteq \mathit{Conf}_{\mathcal{A}} \times (\Sigma \times \mathbb{N}) \times \mathit{Conf}_{\mathcal{A}}$ as follows:

- For $\sigma \in \Sigma_{\mathcal{S}}$ with $P(\sigma) = p$, $(\overline{s}, \mathtt{W}) \overset{(\sigma,\pi)}{\Longrightarrow} (\overline{t}, \mathtt{V})$ iff, for all $l \in \{1, \ldots, r\}$, $((\sigma, \pi), l) \notin \mathtt{W}$, and there is a transition $(\overline{s}[p], (\sigma, k), \overline{t}[p]) \in \Delta_p$ such that
  - $\mathtt{V} = \mathtt{W} \cup \{((\sigma, \pi), k)\}$ and
  - for all $l \in \mathcal{P}_N - \{p\}$, $\overline{s}[l] = \overline{t}[l]$.
- For $\sigma \in \Sigma_{\mathcal{R}}$ with $P(\sigma) = p$ and $(\tau, \sigma) \in \mathit{Corr}$, $(\overline{s}, \mathtt{W}) \overset{(\sigma,\pi)}{\Longrightarrow} (\overline{t}, \mathtt{V})$ iff there is a transition $(\overline{s}[p], (\sigma, k), \overline{t}[p]) \in \Delta_p$ such that
  - $((\tau, \pi), k) \in \mathtt{W}$,
  - $\mathtt{V} = \mathtt{W} - \{((\tau, \pi), k)\}$, and
  - for all $l \in \mathcal{P}_N - \{p\}$, $\overline{s}[l] = \overline{t}[l]$.

A *run* of $\mathcal{A}$ on a word $\alpha = \frac{\sigma_1}{\pi_1} \ldots \frac{\sigma_\ell}{\pi_\ell} \in (\Sigma \times \mathbb{N})^*$ is defined in analogy to the MFA case. That is, we are dealing with a sequence $(\overline{s}_0, \mathtt{W}_0)(\overline{s}_1, \mathtt{W}_1) \ldots (\overline{s}_\ell, \mathtt{W}_\ell)$ of configurations such that

- $\overline{s}_0 \in Q_{in}$,
- $\mathtt{W}_0 = \emptyset$, and
- for each $i \in \{1, \ldots, \ell\}$, $(\overline{s}_{i-1}, \mathtt{W}_{i-1}) \overset{(\sigma_i,\pi_i)}{\Longrightarrow} (\overline{s}_i, \mathtt{W}_i)$.

The run is *accepting* iff $\mathtt{W}_\ell = \emptyset$ and for all $p \in \mathcal{P}_N$, $\overline{s}_\ell[p] \in F$. Finally, $\mathcal{L}(\mathcal{A}) := \{\alpha \mid$ there is an accepting run of $\mathcal{A}$ on $\alpha\}$ denotes the language defined by $\mathcal{A}$.

Let $\mathit{Reach}(\mathcal{A})$ denote the set of configurations reachable within a run of $\mathcal{A}$. Given $B \in \mathbb{N}$, we call $\mathcal{A}$ *B-bounded* iff for all $(\overline{s}, \mathtt{W}) \in \mathit{Reach}(\mathcal{A})$ and actions $\sigma \in \Sigma_{\mathcal{S}}$, $|\mathtt{W} \cap \{((\sigma, \pi), k) \mid \pi \in \mathbb{N},\ k \in \{1, \ldots, r\}\}| \leq B$, and we call it *bounded* iff it is B-bounded for some $B \in \mathbb{N}$.

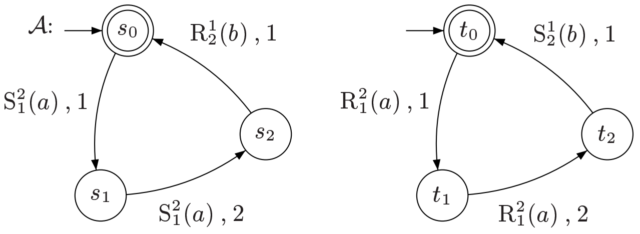


Fig. 7. A 2-bounded AMFA

Theorem 6. *Let $\mathcal{L} \subseteq$ MW be an MSC word language. The following statements are equivalent.*

(1) There is a regular MSC language $\mathcal{M}$ with $Lin(\mathcal{M}) = \mathcal{L}$.
(2) $\mathcal{L}$ is a $\sim$-closed regular MSC word language.
(3) There is a bounded AMFA $\mathcal{A}$ such that $\mathcal{L}(\mathcal{A}) = \mathcal{L}$.

Proof. The equivalence of (1) and (2) immediately follows from the definitions. To prove the equivalence between (2) and (3) is more involved and uses, similar to [8], Zielonka's theorem [16]. Given a bounded AMFA, it is an easy task to define an equivalent MFA. Conversely and given a $\sim$-closed regular MSC word language $\mathcal{L}$, we are able, due to the strong relation between MSCs and Mazurkiewicz traces (see [8]) and according to [16], to build a so-called asynchronous Büchi automaton recognising $\mathcal{L}'$ with $\mathcal{L}'^{\approx} = \mathcal{L}$. But this automaton recognises exactly $\mathcal{L}$ if considered as AMFA.

Thus, bounded AMFAs characterise exactly those regular MSC word languages that are $\sim$-closed and therewith exactly the regular MSC languages. For example, the 2-bounded AMFA $\mathcal{A}$ given in Figure 7 recognises the $(\sim \cup \approx)^*$-closure of

$$\left\{ \begin{pmatrix} \mathrm{S}_1^2(a) & \mathrm{S}_1^2(a) & \mathrm{R}_1^2(a) & \mathrm{R}_1^2(a) & \mathrm{S}_2^1(b) & \mathrm{R}_2^1(b) \\ 1 & 2 & 1 & 2 & 1 & 1 \end{pmatrix}^n \;\middle|\; n \geq 0 \right\}.$$

5 A Logical Characterisation

We formulate a monadic second-order logic which characterises exactly the class of regular MSC languages applying Büchi's famous result [5]. Given a supply $\mathrm{Var} = \{x, y, \ldots\}$ of individual variables and a supply $\mathrm{VAR} = \{X, Y, \ldots\}$ of set variables, the syntax of $\mathrm{MSO}(\mathcal{P}_N, \Lambda)$ is defined according to

$$\mathrm{MSO}(\mathcal{P}_N, \Lambda) ::= L_\sigma(x) \mid x \in X \mid x \preceq y \mid \neg\varphi \mid \varphi_1 \vee \varphi_2 \mid \exists x \varphi \mid \exists X \varphi$$

where $x \in \mathrm{Var}$ and $X \in \mathrm{VAR}$. Moreover we allow the usual abbreviations. Let M be an MSC with set of events E and labelling function L. Given an interpretation function $\mathcal{I}$ which assigns to an individual variable x an event $\mathcal{I}(x) \in E$ and to a set variable X a set $\mathcal{I}(X) \subseteq E$ of events, the satisfaction relation $M \models_{\mathcal{I}} \varphi$ for a formula $\varphi \in \mathrm{MSO}(\mathcal{P}_N, \Lambda)$ is inductively defined as follows:

- $M \models_{\mathcal{I}} L_\sigma(x)$ iff $L(\mathcal{I}(x)) = \sigma$,
- $M \models_{\mathcal{I}} x \in X$ iff $\mathcal{I}(x) \in \mathcal{I}(X)$,
- $M \models_{\mathcal{I}} x \preceq y$ iff $\mathcal{I}(x) \preceq \mathcal{I}(y)$,
- $M \models_{\mathcal{I}} \neg\varphi$ iff $M \not\models_{\mathcal{I}} \varphi$,
- $M \models_{\mathcal{I}} \varphi \vee \psi$ iff $M \models_{\mathcal{I}} \varphi$ or $M \models_{\mathcal{I}} \psi$,
- $M \models_{\mathcal{I}} \exists x \varphi$ iff $\exists e \in E: \; M \models_{\mathcal{I}[x/e]} \varphi$,
- $M \models_{\mathcal{I}} \exists X \varphi$ iff $\exists E' \subseteq E: \; M \models_{\mathcal{I}[X/E']} \varphi$.

From now on, we only consider formulas without free variables and accordingly write $M \models \varphi$ instead of $M \models_{\mathcal{I}} \varphi$. For $B \in \mathbb{N}$, let $\mathcal{M}_\varphi^B := \{M \mid M$ is B-bounded and $M \models \varphi\}$.

Theorem 7. *Given a collection $\mathcal{M}$ of MSCs, $\mathcal{M}$ is regular MSC language iff there are a formula $\varphi \in \mathrm{MSO}(\mathcal{P}_N, \Lambda)$ and $B \in \mathbb{N}$ such that*

$$Lin(\mathcal{M}) = Lin(\mathcal{M}_\varphi^B).$$

Proof. The proof is similar to the one in [8].

($\Longrightarrow$) Let $\mathcal{M}$ be a regular MSC language. We can find $B \in \mathbb{N}$ such that $Lin(\mathcal{M})$ is a B-bounded regular MSC word language. Wrt. Büchi's Theorem [5] and Theorem 1, there is a formula $\varphi \in \mathrm{MSO}(\Sigma \times \{1, \ldots, B\})$ over words such that $\mathcal{L}_\varphi = \{nf(\alpha) \mid \alpha \in Lin(\mathcal{M})\}$ where $\mathcal{L}_\varphi = \{\alpha \mid \alpha \models \varphi\}$. Accordingly, our aim is to define a formula $\overline{\varphi} \in \mathrm{MSO}(\mathcal{P}_N, \Lambda)$ with $Lin(\mathcal{M}_{\overline{\varphi}}^B) = \mathcal{L}_\varphi^{\approx}$. So let $\overline{\varphi} = \|\varphi\|$ where

- $\|L_{(\sigma,\pi)}(x)\| := L_\sigma(x)$,
- $\|x \in X\| := x \in X$,
- $\|\neg\psi\| := \neg\|\psi\|$,
- $\|\psi_1 \vee \psi_2\| := \|\psi_1\| \vee \|\psi_2\|$,
- $\|\exists x \psi\| := \exists x \|\psi\|$,
- $\|\exists X \psi\| := \exists X \|\psi\|$,
- $\|x \leq y\| := x = y \vee \neg(y \preceq x)$.

The last item weakens the relation $\leq$, since (informally spoken) $x \leq y$ on the part of words does not imply $x \preceq y$ on the part of MSCs. Conversely, $x = y \vee \neg(y \preceq x)$ guarantees that it is possible to permute the word under consideration such that then $x \leq y$.

($\Longleftarrow$) We transform φ into a sentence $\overline{\varphi} \in \mathrm{MSO}(\Sigma \times \{1, \ldots, B\})$ over words such that $\mathcal{L}_{\overline{\varphi}} = \{nf(\alpha) \mid \alpha \in Lin(\mathcal{M}_\varphi^B)\}$. $\overline{\varphi}$ is of the form $Comp \wedge Normal \wedge \|\varphi\|$. *Comp* ensures that only MSC words are defined, *Normal* guarantees that we finally deal with MSC words in normal form, and furthermore $\|\varphi\|$ helps to restrict the set of such MSC words to the ones contained in $Lin(\mathcal{M}_\varphi^B)$. Anyway, due to the underlying finite alphabet, $\|\varphi\|$ specifies B-bounded words. So let

$$Comp = \bigwedge_{\substack{(\sigma,\tau) \in Corr \\ \pi \in \{1,\ldots,B\}}} \Phi_1(\sigma,\tau,\pi) \wedge \Phi_2(\sigma,\tau,\pi) \wedge \Phi_3(\sigma,\tau,\pi) \qquad \text{where}$$

- $\Phi_1(\sigma,\tau,\pi) = \forall x \forall y [x < y \wedge L_{(\sigma,\pi)}(x) \wedge L_{(\sigma,\pi)}(y) \longrightarrow \exists z (x < z < y \wedge L_{(\tau,\pi)}(z))]$ ensures that (σ,π) and (τ,π) only occur in turns,
- $\Phi_2(\sigma,\tau,\pi) = \forall x [L_{(\tau,\pi)}(x) \longrightarrow \exists y (y < x \wedge L_{(\sigma,\pi)}(y))]$ guarantees that (τ,π) is preceded by its send event (σ,π), and furthermore
- $\Phi_3(\sigma,\tau,\pi) = \forall x [L_{(\sigma,\pi)}(x) \longrightarrow \exists y (x < y \wedge L_{(\tau,\pi)}(y))]$ makes sure that a send event (σ,π) is finally followed by its receive event (τ,π).

Notice that, in the context of *Comp*, Φ_1 and Φ_2 lead to properness, whereas completeness is only guaranteed with the help of Φ_3. Let furthermore

$$\mathit{Normal} = \bigwedge_{\substack{(\sigma,\tau)\in \mathit{Corr}\\ \pi\in\{2,\dots,B\}}} \forall x[L_{(\sigma,\pi)}(x) \longrightarrow \exists y(y<x \wedge L_{(\sigma,\pi-1)}(y)\wedge \forall z(y<z<x \longrightarrow \neg L_{(\tau,\pi-1)}(z)))]$$

Finally, we inductively obtain $\|\varphi\|$ as above except for

- $\|L_\sigma(x)\| := \bigvee_{\pi\in\{1,\dots,B\}} L_{(\sigma,\pi)}(x)$ and
- $\|x \preceq y\| := \trianglelefteq(x,y)$. $\trianglelefteq$ reflects what we stated in Proposition 2 according to

$$\trianglelefteq(x,y) := \exists X[\,x\in X \wedge y \in X \wedge \forall z(z\in X \wedge z \neq y \longrightarrow \exists z'(\,z'\in X \wedge z<z' \wedge \mathit{Proc}(z) = \mathit{Proc}(z') \vee \searrow(z,z')))]$$

where *Proc* is used and defined in the obvious manner and the matching predicate is given as follows:

$$\searrow(x,y) := \bigvee_{\substack{(\sigma,\tau)\in \mathit{Corr}\\ \pi\in\{1,\dots,B\}}} [x<y \wedge L_{(\sigma,\pi)}(x) \wedge L_{(\tau,\pi)}(y) \wedge \nexists z(x<z<y \wedge L_{(\tau,\pi)}(z))]$$

In fact, one can easily see that the formula $\overline{\varphi}$ has the desired properties. From Büchi's Theorem [5], it immediately follows the one direction of Theorem 7.

References

1. R. Alur, K. Etessami, and M. Yannakakis. Inference of message sequence charts. In *22nd International Conference on Software Engineering*, pages 304–313, 2000.
2. R. Alur and M. Yannakakis. Model checking of message sequence charts. In *Proc. 10th Intl. Conf. on Concurrency Theory*, volume 1664 of *Lecture Notes in Computer Science*, pages 114–129. Springer Verlag, 1999.
3. João Araújo. Formalizing sequence diagrams. In Luis Andrade, Ana Moreira, Akash Deshpande, and Stuart Kent, editors, *Proceedings of the OOPSLA'98 Workshop on Formalizing UML. Why? How?*, volume 33, 10 of *ACM SIGPLAN Notices*, New York, 1998. ACM Press.
4. Benedikt Bollig and Martin Leucker. Modelling, Specifying, and Verifying Message Passing Systems. In *Proceedings of the Symposium on Temporal Representation and Reasoning (TIME01)*. IEEE Computer Society Press, 2001. to be published.
5. J. Büchi. Weak second order logic and finite automata. *Z. Math. Logik, Grundlag. Math.*, 5:66–62, 1960.
6. E. Gunter, A. Muscholl, and D. Peled. Compositional Message Sequence Charts. In *Proceedings of the 7th International Conference on Tools and algorithms for the construction and analysis of systems (TACAS '01)*, Lecture Notes in Computer Science. Springer-Verlag Inc., 2001.
7. J. G. Henriksen, M. Mukund, K. Narayan Kumar, and P. S. Thiagarajan. On message sequence graphs and finitely generated regular msc languages. In *Proceedings of 27th International Colloquium on Automata, Languages and Programming (ICALP'2000)*, volume 1853 of *Lecture Notes in Computer Science*, pages 675–686. Springer-Verlag, 2000.

8. J. G. Henriksen, M. Mukund, K. Narayan Kumar, and P. S. Thiagarajan. Regular collections of message sequence charts. In *Proceedings of 25th International Symposium on Mathematical Foundations of Computer Science (MFCS'2000)*, volume 1893 of *Lecture Notes in Computer Science*, pages 405–414. Springer-Verlag, 2000.
9. ITU-TS. ITU-TS Recommendation Z.120: Message Sequence Chart 1996 (MSC96). Technical report, ITU-TS, Geneva, 1996.
10. Michael Kaminski and Nissim Francez. Finite-memory automata. *Theoretical Computer Science*, 134(2):329–363, November 1994.
11. B. Meenakshi and R. Ramanujam. Reasoning about message passing in finite state environments. In *Proceedings of 27th International Colloquium on Automata, Languages and Programming (ICALP'2000)*, volume 1853 of *Lecture Notes in Computer Science*. Springer-Verlag, 2000.
12. Rémi Morin. On Regular Message Sequence Chart Languages and Relationships to Mazurkiewicz Trace Theory. In F. Honsell and M. Miculan, editors, *Foundations of Software Science and Computation Structures (FoSSaCS'01)*, volume 2030 of *LNCS*, Genova, Italy, 2001. Springer Verlag.
13. A. Muscholl. Matching specifications for message sequence charts. In *Foundations of Software Science and Computation Structures (FoSSaCS'98)*, pages 273–287. Springer Verlag, 1998.
14. Anca Muscholl and Doron Peled. Message sequence graphs and decision problems on mazurkiewicz traces. In Tomasz Wierzbicki Miroslaw Kutylowski, Leszek Pacholski, editor, *Mathematical Foundations of Computer Science 1999, 24th International Symposium, MFCS'99, Szklarska Poreba, Poland, September 6-10, 1999, Proceedings.*, volume 1672 of *Lecture Notes in Computer Science*. Springer, 1999.
15. Anca Muscholl, Zhendong Su, and Doron Peled. Deciding properties for message sequence charts. In *Foundations of Software Science and Computation Structures (FoSSaCS'98)*, volume 1578 of *LNCS*, Lisbon, Portugal, 1998. Springer Verlag.
16. Wiesław Zielonka. Notes on finite asynchronous automata. *R.A.I.R.O. — Informatique Théorique et Applications*, 21:99–135, 1987.

Aachener Informatik-Berichte

This is a list of recent technical reports. To obtain copies of technical reports please consult ftp://ftp.informatik.rwth-aachen.de/pub/reports/index.html or send your request to: Informatik-Bibliothek, RWTH Aachen, Ahornstr. 55, 52056 Aachen, Email: `biblio@informatik.rwth-aachen.de`

98-01 *	Jahresbericht 1997
98-02 *	S. Gruner/ M. Nagel / A. Schürr: Fine-grained and Structure-oriented Integration Tools are Needed for Product Development Processes
98-03 *	S. Gruner: Einige Anmerkungen zur graphgrammatischen Spezifikation von Integrationswerkzeugen nach Westfechtel, Janning, Lefering und Schürr
98-04	O. Kubitz: Mobile Robots in Dynamic Environments
98-05 *	M. Leucker / St. Tobies: Truth — A Verification Platform for Distributed Systems
98-07 *	M. Arnold / M. Erdmann / M. Glinz / P. Haumer / R. Knoll / B. Paech / K. Pohl / J. Ryser / R. Studer / K. Weidenhaupt: Survey on the Scenario Use in Twelve Selected Industrial Projects
98-08	H. Aust: Sprachverstehen und Dialogmodellierung in natürlichsprachlichen Informationssystemen
98-09	Th. Lehmann: Geometrische Ausrichtung medizinischer Bilder am Beispiel intraoraler Radiographien
98-10	M. Nicola / M. Jarke: Performance Modeling of Distributed and Replicated Databases
98-11	A. Schleicher / B. Westfechtel / D. Jäger: Modeling Dynamic Software Processes in UML
98-12	W. Appelt / M. Jarke: Interoperable Tools for Cooperation Support using the World Wide Web
98-13 *	K. Indermark: Semantik rekursiver Funktionsdefinitionen mit Striktheitsinformation
99-01	Jahresbericht 1998
99-02	F. Huch: Verifcation of Erlang Programs using Abstract Interpretation and Model Checking — Extended Version
99-03	R. Gallersdörfer / M. Jarke / M. Nicola: The ADR Replication Manager
99-04 *	M. Alpuente / M. Hanus / S. Lucas / G. Vidal: Specialization of Functional Logic Programs Based on Needed Narrowing
99-07 *	Th. Wilke: CTL+ is exponentially more succinct than CTL
99-08 *	O. Matz: Dot-Depth and Monadic Quantifier Alternation over Pictures
2000-01	Jahresbericht 1999

2000-02 * **Jens Vöge / Marcin Jurdziński: A Discrete Strategy Improvement Algorithm for Solving Parity Games**
2000-04 * **Andreas Becks, Stefan Sklorz, Matthias Jarke: Exploring the Semantic Structure of Technical Document Collections: A Cooperative Systems Approach**
2000-05 **Mareike Schoop: Cooperative Document Management**
2000-06 **Mareike Schoop (Ed.): Proceedings of the Fifth International Workshop on the Language-Action Perspective on Communication Modelling**
2000-07 **Markus Mohnen / Pieter Koopman (Eds.): Proceedings of the 12th International Workshop of Functional Languages**
2000-08 * **Thomas Arts / Thomas Noll: Verifying Generic Erlang Client-Server Implementations**
2001-01 **Jahresbericht 2000**
2001-02 * **Benedikt Bollig / Martin Leucker: Deciding LTL over Mazurkiewicz Traces**
2001-03 * **Thierry Cachat: The power of one-letter rational languages Abstract or Paper**
2001-04 * **Benedikt Bollig / Martin Leucker / Michael Weber: Local Parallel Model Checking for the Alternation Free μ-calculus**

* These reports are also available electronically as PostScript-files
via anonymous FTP on `ftp.informatik.rwth-aachen.de` (IP address 137.226.225.3)
or via WWW at URL `ftp://ftp.informatik.rwth-aachen.de/pub/reports/index.html`