Regular sets of pomsets without autoconcurrency:

extending regular MSC languages

Jean Fanchon

LAAS_CNRS

FAC’04, Toulouse  9–10/03/2004

Notions of regularity and classes of pomsets

Pomsets without autoconcurrency
Word languages

Rational expressions
\{+, \cdot, \ast\}

Automata
\(L = L(A)\)

Monoid recognisability
\(\eta : \Sigma^* \rightarrow F\)
\(L = \eta^2(\eta(L))\)

Logic MSO
\(L = L(\varphi)\)
## Complementary Notions

<table>
<thead>
<tr>
<th>Expressions (rational)</th>
<th>Syntax, process algebras</th>
</tr>
</thead>
<tbody>
<tr>
<td>Automata, Petri Nets</td>
<td>Devices, recognition or implementation</td>
</tr>
<tr>
<td>Logical formulae</td>
<td>Behavioural properties, specification</td>
</tr>
<tr>
<td>Morphisms</td>
<td>Algebraic properties, closures</td>
</tr>
</tbody>
</table>

\[ p \models \varphi \iff p \not\in L(A) \]

**Logical specification** \( \iff \) **Automaton acceptance**
Classes of pomsets and regularity/recognisability notions

[1] Series-parallel


[4] Finitely generated

[5] Compatible languages Petri nets

[6] Pomsets without autoconcurrency

[7] Pomsets with autoconcurrency

References:

- [LW98, K02]
- [F99]
- [DGK00, K98, F03]
- [KM00]
- [FM02]
- [C, T]
### Recognisability notions and classes of pomsets

**Algebraic recognisability**

- **Monoids**:
  - Message Sequence Charts [Morin 02]
  - Pomsets without autoconcurrency [Fanchon 03]

- **SP–algebras** [Lodaya,Weil 98]  
  - Graph algebras [Courcelles 89...]

**Regularity by context equivalence**

- All pomsets [Fanchon,Morin 02]
  - Pomsets without autoconcurrency [Fanchon 03]

**Automata**

- Tiling systems : Graphs [Thomas 90]
- Step automata : Local traces [Kuske, Morin 00]
- ACA : Pomsets without autoconcurrency [Droste,Kuske,Gastin 98]

**MSO logic**

- Graphs [Courcelles 89...]
- Pomsets without autoconcurrency [Droste,Kuske,Gastin 98]
- Local traces [Kuske, Morin 00]  
  - Series–parallel [Kuske 02]
Pomsets: Labelled partial orders

\[ \Sigma \text{ a set of actions} \]

A pomset on \( \Sigma \):
\[ s = (E, \leq, \lambda) \]

\( \bullet \) \( E \) a set of events

\( \bullet \) \( \leq \subseteq E \times E \) a partial order relation
(causality, enabling)

\( \bullet \) \( \lambda : E \rightarrow \Sigma \) labelling

\( \bullet \) \( e \in E \) is an occurrence of the action \( \lambda(e) \)

\( \rightarrow \) direct precedence relation,
\( \leq \text{transitive closure of } \rightarrow \)
Traces, MSC, pomsets without autoconcurrency

{a}  a  a  a
{b,f,g}  b  f  g
{c}  c  c
{d}  d  d

1!2  a  b  c
1?3  a  b  c
2?1  a  b  c
3!1  a  b  c
4?2  a  b  c
Order, step, linear extensions

\[ a \rightarrow b \rightarrow a \rightarrow d \]

\[ a \leftarrow c \rightarrow d \]

\[ p \]

\[ q \in OE(p) \]

\[ r \in SE(p) \]

\[ s \in LE(p) \]
Local traces
Message Sequence Charts
Pomsets without autoconcurrency
Petri nets
Words
Mazurkiewicz Traces
Compatible languages
Petri nets
Tiles
Pomsets with autoconcurrency

Automata recognisability

[FU98]
Fork-join automaton

Series-parallel

Finitely generated

[ACA]
Step automaton

[KA00]
Local traces

[FM02]
Pomsets without autoconcurrency

Asynchronous cellular automaton

[FAG98]

Tiles

Pomsets with autoconcurrency

Algebraic structures and recognisability

SP algebra

Series-parallel

Finitely generated

Pomset generators

[LE(L)]

Monoids

Words
Mazurkiewicz Traces
Message Sequence Charts

Local traces

Compatible languages

Petri nets

Pomsets with autoconcurrency

CCS, TCSP, ACP etc.
Mazurkiewicz trace languages

Co–rational expressions
\{+, \cdot, \circ \rightarrow \}

Trace Automata
\[ LE(L) = L(A) \]

Monoid recognisability
\[ \eta : M(\Sigma, \|) \rightarrow F \]
\[ L = \eta^{-1}(\eta[L]) \]

Logic MSO
\[ L = L(\varphi) \]

Asynchronous cellular automata
\[ L = L(ACA) \]
Languages of series–parallel pomsets

Series–rational expressions
\{+, \cdot, *, \parallel\}

Fork–join Automata
\[ L = L(A) \]

Logic MSO
\[ L = L(\varphi) \]

SP recognisability
\[ \eta : SP(\Sigma) \rightarrow F \]
\[ L = \eta^{-1}(\eta(L)) \]
Languages of Message sequence charts

- Co-rational
  - \{+, \cdot, \text{co-star}\}

- Regular
  - \(LE(L) = L(A)\)

- Recognisable

- Logic MSO
  - \(L = L(\varphi)\)

- Realisable
  - \(L = L(\Pi_{i=1}^{n} A_i)\)

- Finitely generated

- Bounded
From MSC to pomsets without autoconcurrency

Notion of regularity not based on linear extentions  (Fanchon–Morin 02)

Notion of boundedness  (Kuske 98)

Recognisability → Monoid

Automata → Asynchronous cellular automata

Properties independant from MSC labeling

Proofs do not rely on MSC labeling

Regularity notions

Regularité of LE(L) → Traces

Regularity of SE(L) → MSC

Regularity of OE(L) → Local Traces

Regularity of L → Petri Nets

[FM02] → Compatibles
Prefixes, left factors and residues.

Prefix
Left Factor
Residue
(Monoidal) Residue
### Recognisable vs Regular (1)

If $L_1$, $L_2$, and $L$ are regular then

**Closures**

- $L_1 \cdot L_2$ is regular
- $L_1 \cup L_2$ is regular
- $L^*$ is regular if $L$ is connected and $\overline{L}$ is bounded

**Property**

$L^*$ is bounded iff every iterative connected component is strongly connected

Language $L$ is *-connected iff it is defined by rational expression where every iterative component is connected

**Theorems**

- $L$ recognisable $\iff$ $L$ *-connected
- $L$ recognisable and bounded $\implies$ $L$ regular
Regular vs Recognisable (2)

For MSC, regular => bounded and recognisable

both are false for pomsets without autoconcurrency

\{a, b, a \rightarrow b\}^* is regular not bounded

\(S\) is a free set: \(\hat{S} / S \triangleleft S^+ = \emptyset\)

Theorem

If \(L\) is finitely generated by a free set, then

\(L\) regular => \(L\) recognisable

Theorem

\(S\) finite and free

\(S^*\) regular \(<=\) \(\hat{S}\) bounded
Sets of pomsets without autoconcurrency

- Co-rationality
  - Finitely generated sets
  - Finitely generated bounded sets
  - Freely finitely generated sets

- Monoid recognisability
  - Finitely generated sets
  - All sets

- Regularity
  - Finitely generated bounded sets

- ACA recognisability
  - Freely finitely generated sets

- MSO definability
  - Bounded sets

References:
- [DGK]
- [FM]
- [F03,F04]
Conclusions and perspectives

Results  MSC ----> pomsets without autoconcurrency
         Many extended properties

Significant notion of regularity of [FM02]

Extension of MSC to multicast, distributed extended scenarios

Graphs of scenarios (rational expressions)

Links with MSO logic and ACA

Multicast communication alphabets  i:J  ?J  J < I

Realisability of languages of scenarios