Multiset Languages and Minimization Problem for Multiset Finite Automata

Pavel Martinek

Department of Mathematics
Faculty of Applied Informatics
Tomas Bata University in Zlín
Czech Republic

e-mail: pmartinek@utb.cz

LTA 2020, Brno
Outline of the talk

1. Introduction
2. Multiset grammars and multiset automata
3. Similarities and dissimilarities of multiset languages with
   ▶ string languages of Chomsky hierarchy
   ▶ languages accepted by jumping finite automata
4. Minimization problem for multiset finite automata
   ▶ in classical form
   ▶ in a generalized form
5. Conclusion
Introduction

The concept of multiset processing is present in various domains, e.g. in

- DNA computing
- membrane computing
- Petri nets
- chemical abstract machines
- etc.
Introduction

History of grammars which generate multisets starts with:

Solid fundamentals were put (on the basis of Formal languages methodology) in the beginning of 21st century, namely by:
- Manfred Kudlek and his collaborators (long-term project *Multiset languages* at University of Hamburg, 29 papers in 9 years),
- Csuha{j}j-Varj{u} E., Martín-Vide C., and Mitrana V., *Multiset automata*, in Multiset processing — mathematical, computer science, and molecular computing points of view, LNCS 2235, Springer, 2001
Multiset grammars and multiset automata

Grammars ... generate strings of elements
(whose order is strict)

Multiset grammars ... generate multisets of elements
(no order of the elements
in the multiset is given)
Multiset grammars and multiset automata

Grammars ... generate strings of elements
(whose order is strict)

Multiset grammars ... generate multisets of elements
(no order of the elements
in the multiset is given)

Automata ... accept strings of elements

Multiset automata ... accept multisets of elements
### Multisets

<table>
<thead>
<tr>
<th>Occurrence of its elements</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classical set</td>
<td>single</td>
</tr>
<tr>
<td>{a, c, d}</td>
<td></td>
</tr>
<tr>
<td>Multiset</td>
<td>single or multiplied</td>
</tr>
<tr>
<td>{a, a, c, d, d, d}</td>
<td></td>
</tr>
</tbody>
</table>
# Multisets

<table>
<thead>
<tr>
<th></th>
<th>Occurrence of its elements</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classical set</td>
<td>single</td>
<td>{a, c, d}</td>
</tr>
<tr>
<td>Multiset</td>
<td>single or multiplied</td>
<td>{a, a, c, d, d, d}</td>
</tr>
</tbody>
</table>

We will write the multiset \{a, a, c, d, d, d\} as

\[
\langle a\rangle^2 \oplus \langle c\rangle \oplus \langle d\rangle^3
\]

where

\[\langle a\rangle\] denotes singleton multiset (with the only element a),
\[\oplus\] denotes the operation of addition of multisets.
## Multisets

<table>
<thead>
<tr>
<th></th>
<th>Occurrence of its elements</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classical set</td>
<td>single</td>
<td>{a, c, d}</td>
</tr>
<tr>
<td>Multiset</td>
<td>single or multiplied</td>
<td>{a, a, c, d, d, d}</td>
</tr>
</tbody>
</table>

We will write the multiset \{a, a, c, d, d, d\} as

\[
\langle a \rangle^2 \oplus \langle c \rangle \oplus \langle d \rangle^3 \quad \text{or} \quad (2, 0, 1, 3) \quad \text{w.r.t.} \quad \Sigma = \{a, b, c, d\}
\]

where

\[\langle a \rangle\] denotes singleton multiset (with the only element \(a\)),

\[\oplus\] denotes the operation of addition of multisets.
Multisets

<table>
<thead>
<tr>
<th>Occurrence of its elements</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classical set</td>
<td>single</td>
</tr>
<tr>
<td></td>
<td>{a, c, d}</td>
</tr>
<tr>
<td>Multiset</td>
<td>single or multiplied</td>
</tr>
<tr>
<td></td>
<td>{a, a, c, d, d, d}</td>
</tr>
</tbody>
</table>

We will write the multiset \(\{a, a, c, d, d, d\}\) as
\[
\langle a \rangle^2 \oplus \langle c \rangle \oplus \langle d \rangle^3
\]
where
\[
\langle a \rangle \text{ denotes singleton multiset (with the only element } a), \\
\oplus \text{ denotes the operation of addition of multisets.}
\]

Further denotation:
\[
0_\Sigma \text{ ... the empty multiset,} \\
\Sigma \oplus \text{ ... the set of all multisets over alphabet } \Sigma, \\
\Sigma \langle \rangle \text{ ... the set of all singleton multisets over alphabet } \Sigma.
\]
Multiset grammars

**Multiset grammar:** $G = (N, \Sigma, P, S)$ where

- $N$ is an alphabet of nonterminals,
- $\Sigma$ is an alphabet of terminals ($N \cap \Sigma = \emptyset$),
- $P \subseteq [N^{\langle \rangle} \oplus (N \cup \Sigma)^{\oplus}] \times (N \cup \Sigma)^{\oplus}$ is a finite set of productions,
- $S \in N$ is the initial nonterminal.
Multiset grammars

Multiset grammar: \( G = (N, \Sigma, P, S) \) where
- \( N \) is an alphabet of nonterminals,
- \( \Sigma \) is an alphabet of terminals \((N \cap \Sigma = \emptyset)\),
- \( P \subseteq [N^\downarrow \oplus (N \cup \Sigma)^\oplus] \times (N \cup \Sigma)^\oplus \) is a finite set of productions,
- \( S \in N \) is the initial nonterminal.

For \( \mu_1, \mu_2 \in (N \cup \Sigma)^\oplus \), we define \( \mu_1 \Rightarrow \mu_2 \) if there are \((\alpha, \beta) \in P, \gamma \in (N \cup \Sigma)^\oplus \) such that \( \mu_1 = \gamma \oplus \alpha \) and \( \mu_2 = \gamma \oplus \beta \).

\( \Rightarrow^* \) ... reflexive and transitive closure of the relation \( \Rightarrow \)

\( M(G) = \{ \omega \in \Sigma^\oplus | \langle S \rangle \Rightarrow^* \omega \} \) ... the multiset language generated by \( G \)
1. Grammars $G$ as above are called *arbitrary* (or *unrestricted*).

2. Grammars $G$ with all productions $(\alpha, \beta) \in P$ restricted by the condition $|\alpha| \leq |\beta|$ (where $|\alpha|$ denotes cardinality of the multiset $\alpha$) are called *monotone*.

3. Grammars $G$ with all productions $(\alpha, \beta) \in P$ restricted by the condition $\alpha \in N^\langle \rangle$ are called *context-free*.

4. Grammars $G$ with all productions $(\alpha, \beta) \in P$ restricted by the conditions $\alpha \in N^\langle \rangle$ and $\beta \in [N^\langle \rangle \oplus \Sigma^\langle \rangle \cup \Sigma^\langle \rangle]$ are called *regular*. 
Example: Let $G = (\{S, A, B\}, \{a, b\}, P, S)$ where
$P = \{(\langle S \rangle, \langle S \rangle \oplus \langle S \rangle), (\langle S \rangle, \langle A \rangle \oplus \langle B \rangle), (\langle A \rangle, \langle a \rangle), (\langle B \rangle, \langle b \rangle)\}$.

Then:

$\langle S \rangle \Rightarrow \langle A \rangle \oplus \langle B \rangle \Rightarrow \langle a \rangle \oplus \langle B \rangle \Rightarrow \langle a \rangle \oplus \langle b \rangle$,

$\langle S \rangle \Rightarrow \langle S \rangle \oplus \langle S \rangle \Rightarrow^* \langle a \rangle \oplus \langle b \rangle \oplus \langle a \rangle \oplus \langle b \rangle = \langle a \rangle^2 \oplus \langle b \rangle^2$,

$\langle S \rangle \Rightarrow \langle S \rangle \oplus \langle S \rangle \Rightarrow \langle S \rangle \oplus \langle S \rangle \oplus \langle S \rangle \Rightarrow^* \langle a \rangle^3 \oplus \langle b \rangle^3$,

eetc.
Example: Let $G = (\{S, A, B\}, \{a, b\}, P, S)$ where
$P = \{(\langle S \rangle, \langle S \rangle \oplus \langle S \rangle), (\langle S \rangle, \langle A \rangle \oplus \langle B \rangle), (\langle A \rangle, \langle a \rangle), (\langle B \rangle, \langle b \rangle)\}$.

Then:  
$\langle S \rangle \Rightarrow \langle A \rangle \oplus \langle B \rangle \Rightarrow \langle a \rangle \oplus \langle B \rangle \Rightarrow \langle a \rangle \oplus \langle b \rangle$, 
$\langle S \rangle \Rightarrow \langle S \rangle \oplus \langle S \rangle \Rightarrow \ast \langle a \rangle \oplus \langle b \rangle = \langle a \rangle^2 \oplus \langle b \rangle^2$, 
$\langle S \rangle \Rightarrow \langle S \rangle \oplus \langle S \rangle \Rightarrow \langle S \rangle \oplus \langle S \rangle \oplus \langle S \rangle \Rightarrow \ast \langle a \rangle^3 \oplus \langle b \rangle^3$, etc.

Hence  
$\langle S \rangle \Rightarrow \ast \langle a \rangle \oplus \langle b \rangle$, 
$\langle S \rangle \Rightarrow \ast \langle a \rangle^2 \oplus \langle b \rangle^2$, 
$\langle S \rangle \Rightarrow \ast \langle a \rangle^3 \oplus \langle b \rangle^3$, etc.
Example: Let $G = (\{S, A, B\}, \{a, b\}, P, S)$ where

\[ P = \{(\langle S \rangle, \langle S \rangle \oplus \langle S \rangle), (\langle S \rangle, \langle A \rangle \oplus \langle B \rangle), (\langle A \rangle, \langle a \rangle), (\langle B \rangle, \langle b \rangle)\}. \]

Then:

\[
\langle S \rangle \Rightarrow \langle A \rangle \oplus \langle B \rangle \Rightarrow \langle a \rangle \oplus \langle B \rangle \Rightarrow \langle a \rangle \oplus \langle b \rangle,\\
\langle S \rangle \Rightarrow \langle S \rangle \oplus \langle S \rangle \Rightarrow * \langle a \rangle \oplus \langle b \rangle \oplus \langle a \rangle \oplus \langle b \rangle = \langle a \rangle^2 \oplus \langle b \rangle^2,\\
\langle S \rangle \Rightarrow \langle S \rangle \oplus \langle S \rangle \Rightarrow \langle S \rangle \oplus \langle S \rangle \oplus \langle S \rangle \Rightarrow * \langle a \rangle^3 \oplus \langle b \rangle^3,\\
\text{etc.}
\]

Hence

\[
\langle S \rangle \Rightarrow * \langle a \rangle \oplus \langle b \rangle,\\
\langle S \rangle \Rightarrow * \langle a \rangle^2 \oplus \langle b \rangle^2,\\
\langle S \rangle \Rightarrow * \langle a \rangle^3 \oplus \langle b \rangle^3,\\
\text{etc.}
\]

Obviously:

\[
M(G) = \{\alpha \in \{a, b\}^\oplus | |\alpha|_a = |\alpha|_b > 0\}.
\]
Chomsky-like classification of multiset languages

**Definition**: Multiset languages generated by arbitrary, monotone, context-free and regular grammars are called *arbitrary, monotone, context-free* and *regular*, respectively.

**Assertion**: The family of multiset context-free languages is equal to the family of multiset regular languages. **Proof directly follows from Parikh's theorem.**
Definition: Multiset languages generated by arbitrary, monotone, context-free and regular grammars are called arbitrary, monotone, context-free and regular, respectively.

Assertion: The family of multiset context-free languages is equal to the family of multiset regular languages.

Proof directly follows from Parikh’s theorem.
A *multiset finite automaton* (mFA): $A = (Q, \Sigma, \delta, q_0, F)$ where

- $Q$ is a nonempty finite set of states,
- $\Sigma$ is an input alphabet,
- $\delta \subseteq Q \times \Sigma \times Q$ is a transition relation,
- $q_0$ is the initial state,
- $F \subseteq Q$ is a set of final states.
Multiset finite automata

A multiset finite automaton (mFA): $A = (Q, \Sigma, \delta, q_0, F)$ where

- $Q$ is a nonempty finite set of states,
- $\Sigma$ is an input alphabet,
- $\delta \subseteq Q \times \Sigma \times Q$ is a transition relation,
- $q_0$ is the initial state,
- $F \subseteq Q$ is a set of final states.

A configuration: $(q, \mu) \in Q \times \Sigma^\oplus$.

A computational step is a relation $\vdash \subseteq (Q \times \Sigma^\oplus) \times (Q \times \Sigma^\oplus)$ defined by $(q, \langle a \rangle \oplus \mu) \vdash (q', \mu)$ iff $(q, a, q') \in \delta$.

$\vdash^*$ denotes the reflexive and transitive closure of $\vdash$. 

Multiset finite automata

A *multiset finite automaton* (mFA): \( A = (Q, \Sigma, \delta, q_0, F) \) where

- \( Q \) is a nonempty finite set of states,
- \( \Sigma \) is an input alphabet,
- \( \delta \subseteq Q \times \Sigma \times Q \) is a transition relation,
- \( q_0 \) is the initial state,
- \( F \subseteq Q \) is a set of final states.

A *configuration*: \((q, \mu) \in Q \times \Sigma^{+}\).

A *computational step* is a relation \( \vdash \subseteq (Q \times \Sigma^{+}) \times (Q \times \Sigma^{+}) \) defined by \((q, \langle a \rangle \oplus \mu) \vdash (q', \mu)\) iff \((q, a, q') \in \delta\).

\( \vdash^{*} \) denotes the reflexive and transitive closure of \( \vdash \).

The *multiset language* \( M(A) \) accepted by \( A \) is defined by

\[
M(A) = \{ \omega \in \Sigma^{+} | (q_0, \omega) \vdash^{*} (q_f, 0_{\Sigma}) \text{ for some } q_f \in F \}.
\]
Multiset finite automata

A comparison:

Finite automaton $A$  
Multiset finite automaton $B$
Multiset finite automata

A comparison:

Finite automaton $A$  

Multiset finite automaton $B$

For example

- the finite automaton $A$ accepts the string $abab$ and does not accept the string $aabb$,
- the multiset finite automaton $B$ accepts the multiset $\{a, a, b, b\}$ (alternatively written as $\langle a \rangle^2 \oplus \langle b \rangle^2$).
Multiset finite automata

A comparison:

Finite automaton $A$

Multiset finite automaton $B$

Accepted languages

- $L(A) = \{(ab)^n \mid n \geq 0\}$,
- $M(B) = \{\langle a \rangle^n \oplus \langle b \rangle^n \mid n \geq 0\}$. 
## Chomsky hierarchy of string languages

<table>
<thead>
<tr>
<th>Languages</th>
<th>Grammars</th>
<th>Automata</th>
</tr>
</thead>
<tbody>
<tr>
<td>recursively enumerable</td>
<td>arbitrary</td>
<td>Turing machine</td>
</tr>
<tr>
<td>monotone</td>
<td>monotone</td>
<td>linear bounded automaton</td>
</tr>
<tr>
<td>context-free</td>
<td>context-free</td>
<td>pushdown automaton</td>
</tr>
<tr>
<td>regular</td>
<td>regular</td>
<td>finite automaton</td>
</tr>
<tr>
<td>Languages</td>
<td>Grammars</td>
<td>Automata</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>----------------</td>
<td>------------------------------------</td>
</tr>
<tr>
<td>recursively enumerable</td>
<td>arbitrary</td>
<td>Turing machine</td>
</tr>
<tr>
<td>monotone</td>
<td>monotone</td>
<td>linear bounded automaton</td>
</tr>
<tr>
<td>context-free</td>
<td>context-free</td>
<td>pushdown automaton</td>
</tr>
<tr>
<td>regular</td>
<td>regular</td>
<td>finite automaton</td>
</tr>
</tbody>
</table>

RE  \[\uparrow\uparrow\]
MON  \[\uparrow\uparrow\]
CF   \[\uparrow\uparrow\]
REG  \[\uparrow\uparrow\]  \[\uparrow\uparrow\] ... proper inclusion
Chomsky-like hierarchy of multiset languages

<table>
<thead>
<tr>
<th>Multiset languages</th>
<th>Multiset grammars</th>
<th>Automata</th>
</tr>
</thead>
<tbody>
<tr>
<td>arbitrary</td>
<td>arbitrary</td>
<td>multiset Turing machine</td>
</tr>
<tr>
<td>monotone</td>
<td>monotone</td>
<td>multiset linear bounded automaton</td>
</tr>
<tr>
<td>regular</td>
<td>context-free &amp; regular</td>
<td>multiset finite automaton</td>
</tr>
</tbody>
</table>
## Chomsky-like hierarchy of multiset languages

<table>
<thead>
<tr>
<th>Multiset languages</th>
<th>Multiset grammars</th>
<th>Automata</th>
</tr>
</thead>
<tbody>
<tr>
<td>arbitrary</td>
<td>arbitrary</td>
<td>multiset Turing machine</td>
</tr>
<tr>
<td>monotone</td>
<td>monotone</td>
<td>multiset linear bounded automaton</td>
</tr>
<tr>
<td>regular</td>
<td>context-free &amp; regular</td>
<td>multiset finite automaton</td>
</tr>
</tbody>
</table>

- **mARB**
  - ↑
- **mMON**
  - ↑
- **mREG**
  - ↑↑

... inclusion with unclear properness

... proper inclusion
Similarities and dissimilarities of multiset languages with string languages

- Similarities – some (we use techniques and concepts invented for exploration of string languages, for example generating and accepting devices).
- Dissimilarities – usual due to work with multisets (their elements are not ordered).
Similarities and dissimilarities of m-languages with string languages

\[ \text{RE} = \mathcal{L}(\text{TM}) \]

\[ \uparrow \uparrow \]

\[ \text{MON} \]

\[ \uparrow \uparrow \]

\[ \text{CF} \]

\[ \uparrow \uparrow \]

\[ \text{REG} = \mathcal{L}(\text{FA}) \]
Matrix grammar with appearance checking:  \( G = (N, \Sigma, M, S, F) \) where

- \( N, \Sigma \) and \( S \in N \) are as in a context-free grammar
- \( M = \{m_1, m_2, \ldots, m_n\} \) is a finite set of finite sequences of context-free productions (incl. erasing productions) using symbols from \( N \cup \Sigma \cup \{\varepsilon\} \).
- \( F \) is a subset of productions contained in \( M \).

\[ x \Rightarrow y \text{ with } x, y \in (N \cup \Sigma)^* \text{ ... a direct derivation which uses productions of a sequence } m_i \in M \]

1. either all of them one by one
2. or productions contained in \( F \) can be omitted if they cannot be applied; the other productions must be used (respecting their order in the sequence)

\( \Rightarrow^* \text{ ... reflexive and transitive closure of the relation } \Rightarrow \)

\( L(G) = \{w \in \Sigma^* | S \Rightarrow^* w\} \)
Example: Let $G = (\{S, X, Y\}, \{a, b\}, \{m_1, m_2, m_3, m_4\}, S, \emptyset)$ where

$m_1 = (S \to XX)$,

$m_2 = (X \to aY, X \to aX, Y \to X)$,

$m_3 = (X \to bY, X \to bX, Y \to X)$,

$m_4 = (X \to \varepsilon, X \to \varepsilon)$. 
Example: Let $G = (\{S, X, Y\}, \{a, b\}, \{m_1, m_2, m_3, m_4\}, S, \emptyset)$ where

$m_1 = (S \rightarrow XX),$

$m_2 = (X \rightarrow aY, X \rightarrow aX, Y \rightarrow X),$

$m_3 = (X \rightarrow bY, X \rightarrow bX, Y \rightarrow X),$

$m_4 = (X \rightarrow \varepsilon, X \rightarrow \varepsilon).$

We have for all $w \in \{a, b\}^*$:

$(m_2) \quad wXwX \quad \cdots \quad waYwX \quad \cdots \quad waYwaX \quad \cdots \quad waXwaX,$

$(m_3) \quad wXwX \quad \cdots \quad wbYwX \quad \cdots \quad wbYwbX \quad \cdots \quad wbXwbX,$

$(m_4) \quad wXwX \quad \cdots \quad wwX \quad \cdots \quad ww.$
Example: Let $G = (\{S, X, Y\}, \{a, b\}, \{m_1, m_2, m_3, m_4\}, S, \emptyset)$ where

$m_1 = (S \rightarrow XX),$

$m_2 = (X \rightarrow aY, X \rightarrow aX, Y \rightarrow X),$

$m_3 = (X \rightarrow bY, X \rightarrow bX, Y \rightarrow X),$

$m_4 = (X \rightarrow \varepsilon, X \rightarrow \varepsilon).$

We have for all $w \in \{a, b\}^*$:

$(m_2) \quad wXwX \quad \Rightarrow \quad waYwX \quad \Rightarrow \quad waYwaX \quad \Rightarrow \quad waXwaX,$

$(m_3) \quad wXwX \quad \Rightarrow \quad wbYwX \quad \Rightarrow \quad wbYwbX \quad \Rightarrow \quad wbXwbX,$

$(m_4) \quad wXwX \quad \Rightarrow \quad wwX \quad \Rightarrow \quad ww.$

Hence $S \Rightarrow_{m_1} XX \Rightarrow_{m_2, m_3}^* wXwX \Rightarrow_{m_4} ww.$

So, $L(G) = \{ww \mid w \in \{a, b\}^* \}.$
Similarities and dissimilarities of m-languages with string languages

\[ \text{RE} = \mathcal{L}(\text{TM}) = \text{MAT}_{ac} \]
\[ \uparrow\uparrow \]
\[ \text{MON} \]
\[ \uparrow\uparrow \]
\[ \text{CF} \]
\[ \uparrow\uparrow \]
\[ \text{REG} = \mathcal{L}(\text{FA}) \]
Similarities and dissimilarities of m-languages with string languages

\[
\begin{align*}
\text{RE} &= \mathcal{L}(\text{TM}) = \text{MAT}_{ac} \\
\text{MON} &= \text{REG} = \mathcal{L}(\text{FA}) \\
\text{CF} &= \text{mMON} \\
\text{mARB} &= \mathcal{L}(\text{mTM}) \\
\text{mCF} &= \text{mREG} = \mathcal{L}(\text{mFA})
\end{align*}
\]
Similarities and dissimilarities of m-languages with string languages

\[ \text{RE} = \mathcal{L}(\text{TM}) = \text{MAT}_{ac} \]
\[ \uparrow \uparrow \]
\[ \text{MON} \]
\[ \uparrow \uparrow \]
\[ \text{CF} \]
\[ \uparrow \uparrow \]
\[ \text{REG} = \mathcal{L}(\text{FA}) \]

\[ \text{mMAT}_{ac} \]
\[ \uparrow \uparrow \]
\[ \text{mARB} = \mathcal{L}(\text{mTM}) \]
\[ \uparrow \]
\[ \text{mMON} \]
\[ \uparrow \uparrow \]
\[ \text{mCF} = \text{mREG} = \mathcal{L}(\text{mFA}) \]
Similarities and dissimilarities of m-languages with string languages

\[
\begin{align*}
\text{RE} &= \mathcal{L}(\text{TM}) = \text{MAT}_{ac} \\
\uparrow\uparrow \\
\text{MON} \\
\uparrow\uparrow \\
\text{CF} \\
\uparrow\uparrow \\
\text{REG} &= \mathcal{L}(\text{FA}) \\
\uparrow\uparrow \\
\text{mMAT}_{ac} &= \mathcal{L}(\text{mTM}_{ac}) \\
\uparrow\uparrow \\
\text{mARB} &= \mathcal{L}(\text{mTM}) \\
\uparrow \\
\text{mMON} \\
\uparrow\uparrow \\
\text{mCF} &= \text{mREG} = \mathcal{L}(\text{mFA})
\end{align*}
\]
Similarities and dissimilarities of m-languages with string languages

\[ \text{RE} = \mathcal{L}(\text{TM}) = \text{MAT}_{ac} \]
\[ \uparrow \uparrow \]
\[ \text{MON} \]
\[ \uparrow \uparrow \]
\[ \text{CF} \]
\[ \uparrow \uparrow \]
\[ \text{REG} = \mathcal{L}(\text{FA}) = \mathcal{L}(\text{dFA}) \]

\[ \text{mMAT}_{ac} = \mathcal{L}(\text{mTM}_{ac}) \]
\[ \uparrow \uparrow \]
\[ \text{mARB} = \mathcal{L}(\text{mTM}) \]
\[ \uparrow \]
\[ \text{mMON} \]
\[ \uparrow \uparrow \]
\[ \text{mCF} = \text{mREG} = \mathcal{L}(\text{mFA}) \]
Similarities and dissimilarities of m-languages with string languages

<table>
<thead>
<tr>
<th>RE</th>
<th>$\mathcal{L}(TM) = \text{MAT}_{ac}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>MON</td>
<td></td>
</tr>
<tr>
<td>CF</td>
<td></td>
</tr>
<tr>
<td>REG</td>
<td>$\mathcal{L}(FA) = \mathcal{L}(dFA)$</td>
</tr>
</tbody>
</table>

$mMAT_{ac} = \mathcal{L}(mTM_{ac})$

$mARB = \mathcal{L}(mTM)$

$mMON$

$mCF = mREG = \mathcal{L}(mFA)$

$\mathcal{L}(dmFA)$
Definition: An mFA $A = (Q, \Sigma, \delta, q_0, F)$ is said to be \textit{deterministic} (we write $dmFA$) if the following condition is satisfied:

For all $q, r, r' \in Q$, $a, a' \in \Sigma$, if $(q, a, r) \in \delta$ and $(q, a', r') \in \delta$, then $a = a'$ and $r = r'$. 
Similarities and dissimilarities of m-languages with string languages

**Definition:** An mFA $A = (Q, \Sigma, \delta, q_0, F)$ is said to be *deterministic* (we write $\text{dmFA}$) if the following condition is satisfied:

For all $q, r, r' \in Q$, $a, a' \in \Sigma$, if $(q, a, r) \in \delta$ and $(q, a', r') \in \delta$, then $a = a'$ and $r = r'$.

**Example:** dmFA $A$: nondeterministic mFA $B$:
Similarities and dissimilarities of multiset languages with string languages accepted by jumping finite automata

- Similarities – wide (the concepts of multiset and jumping finite automata have a lot in common).
- Dissimilarities – rare (despite the difference between strings and multisets).
String languages accepted by jumping finite automata

Finite automaton:

```
  a a b b a b ...
```

... input

control
String languages accepted by jumping finite automata

Finite automaton: a b b a b

... input

control
String languages accepted by jumping finite automata

Finite automaton:

```
 Finite automaton:  
  a
  a
  b
  b
  a
  b
  ... input
  control
```
String languages accepted by jumping finite automata

Finite automaton:

Jumping finite automaton:
String languages accepted by jumping finite automata

Finite automaton:

Jumping finite automaton:
String languages accepted by jumping finite automata

Finite automaton:

Jumping finite automaton:
String languages accepted by jumping finite automata

Finite automaton:

Jumping finite automaton:
String languages accepted by jumping finite automata

Finite automaton:

Jumping finite automaton:
String languages accepted by jumping finite automata

A jumping finite automaton (JFA): $A = (Q, \Sigma, \delta, q_0, F)$ where

- $Q$ is a nonempty finite set of states,
- $\Sigma$ is an input alphabet, $\Sigma \cap Q = \emptyset$,
- $\delta \subseteq Q \times \Sigma \times Q$ is a transition relation,
- $q_0$ is the initial state,
- $F \subseteq Q$ is a set of final states.
A jumping finite automaton (JFA): \( A = (Q, \Sigma, \delta, q_0, F) \) where

- \( Q \) is a nonempty finite set of states,
- \( \Sigma \) is an input alphabet, \( \Sigma \cap Q = \emptyset \),
- \( \delta \subseteq Q \times \Sigma \times Q \) is a transition relation,
- \( q_0 \) is the initial state,
- \( F \subseteq Q \) is a set of final states.

A configuration: \( uqv \in \Sigma^* Q \Sigma^* \) (\( uv \) is the not yet processed content of the input string)

A jumping relation is a relation \( \bowtie \subseteq \Sigma^* Q \Sigma^* \times \Sigma^* Q \Sigma^* \) defined by \( (uqav, u'rv') \in \bowtie \) iff \( (q, a, r) \in \delta \) and \( uv = u'v' \).

\( \bowtie^* \) denotes the reflexive and transitive closure of \( \bowtie \).
A jumping finite automaton (JFA): $A = (Q, \Sigma, \delta, q_0, F)$ where

- $Q$ is a nonempty finite set of states,
- $\Sigma$ is an input alphabet, $\Sigma \cap Q = \emptyset$,
- $\delta \subseteq Q \times \Sigma \times Q$ is a transition relation,
- $q_0$ is the initial state,
- $F \subseteq Q$ is a set of final states.

A configuration: $uqv \in \Sigma^* Q \Sigma^*$ ($uv$ is the not yet processed content of the input string)

A jumping relation is a relation $\sim \subseteq \Sigma^* Q \Sigma^* \times \Sigma^* Q \Sigma^*$ defined by $(uqav, u'rv') \in \sim$ iff $(q, a, r) \in \delta$ and $uv = u'v'$.

$\sim^*$ denotes the reflexive and transitive closure of $\sim$.

The language $L(A)$ accepted by $A$ is defined by

$$L(A) = \{uv \mid u, v \in \Sigma^*, (uq_0v, q_f) \in \sim^* \text{ for some } q_f \in F\}.$$
String languages accepted by jumping finite automata

A comparison:

Finite automaton $A$  Jumping finite automaton $B$
String languages accepted by jumping finite automata

A comparison:

Finite automaton $A$

Jumping finite automaton $B$

For example

- the finite automaton $A$ accepts the string $abab$ and does not accept the string $aabb$,
- the jumping finite automaton $B$ accepts both the string $abab$ and the string $aabb$. 
String languages accepted by jumping finite automata

A comparison:

Finite automaton $A$

Jumping finite automaton $B$

Accepted languages

- $L(A) = \{(ab)^n \mid n \geq 0\}$,
- $L(B) = \{w \in \{a, b\}^* \mid |w|_a = |w|_b\}$. 
String languages accepted by jumping finite automata

Another comparison:

Multiset finite automaton $A$  
Jumping finite automaton $B$

Accepted languages

$\mathit{M}(A) = \{ \langle a \rangle^n \oplus \langle b \rangle^n \mid n \geq 0 \}$,

$\mathit{L}(B) = \{ w \in \{ a, b \}^* \mid |w|_a = |w|_b \}$.
Similarities and dissimilarities of multiset languages with languages accepted by jumping finite automata

Theorem: If a language $L \subseteq \Sigma^*$ is accepted by some jumping finite automaton then for every $w \in L$, any permutation of symbols in $w$ is in $L$. 
Similarities and dissimilarities of multiset languages with languages accepted by jumping finite automata

**Theorem**: If a language $L \subseteq \Sigma^*$ is accepted by some jumping finite automaton then for every $w \in L$, any permutation of symbols in $w$ is in $L$.

**Definition**: Let $\Sigma = \{a_1, a_2, \ldots, a_n\}$ be an alphabet. The mapping $\Psi : \Sigma^* \rightarrow \mathbb{N}^n$ such that

$$\Psi(w) = (|w|_{a_1}, |w|_{a_2}, \ldots, |w|_{a_n})$$

for any $w \in \Sigma^*$ is called *Parikh mapping* (over $\Sigma$). Here, $|w|_{a_i}$ denotes the number of occurrences of $a_i$ in $w$ and $\mathbb{N}$ is the set of non-negative integers.
Similarities and dissimilarities of multiset languages with languages accepted by jumping finite automata

**Theorem:** If a language \( L \subseteq \Sigma^* \) is accepted by some jumping finite automaton then for every \( w \in L \), any permutation of symbols in \( w \) is in \( L \).

**Definition:** Let \( \Sigma = \{a_1, a_2, \ldots, a_n\} \) be an alphabet. The mapping \( \Psi : \Sigma^* \rightarrow \mathbb{N}^n \) such that
\[
\Psi(w) = (|w|_{a_1}, |w|_{a_2}, \ldots, |w|_{a_n})
\]
for any \( w \in \Sigma^* \) is called *Parikh mapping* (over \( \Sigma \)). Here, \( |w|_{a_i} \) denotes the number of occurrences of \( a_i \) in \( w \) and \( \mathbb{N} \) is the set of non-negative integers.

**Example:** \( \Sigma = \{a, b, c\}, \ v = abaa, \ w = aaab \),
\[
\Psi(v) = (3, 1, 0) = \Psi(w)
\]
Similarities and dissimilarities of multiset languages with languages accepted by jumping finite automata

**Theorem:** If a language \( L \subseteq \Sigma^* \) is accepted by some jumping finite automaton then for every \( w \in L \), any permutation of symbols in \( w \) is in \( L \).

**Definition:** Let \( \Sigma = \{a_1, a_2, \ldots, a_n\} \) be an alphabet. The mapping \( \Psi : \Sigma^* \rightarrow \mathbb{N}^n \) such that
\[
\Psi(w) = (|w|_{a_1}, |w|_{a_2}, \ldots, |w|_{a_n})
\]
for any \( w \in \Sigma^* \)
is called *Parikh mapping* (over \( \Sigma \)). Here, \( |w|_{a_i} \) denotes the number of occurrences of \( a_i \) in \( w \) and \( \mathbb{N} \) is the set of non-negative integers.

For any language \( L \subseteq \Sigma^* \), we define \( \Psi(L) = \{\Psi(w) \mid w \in L\} \).

**Example:** If \( \Sigma = \{a, b\} \) and \( L = \{w \in \{a, b\}^* \mid |w|_a = |w|_b\} \), then \( \Psi(L) = \{(n, n) \mid n \in \mathbb{N}\} \).
Let $\Sigma = \{a_1, a_2, \ldots, a_n\}$.

- Image of any $w \in \Sigma^*$ by Parikh mapping is the n-tuple $(|w|a_1, |w|a_2, \ldots, |w|a_n)$.

- A multiset $\alpha \in \Sigma^\oplus$ can be represented as the n-tuple $(|\alpha|a_1, |\alpha|a_2, \ldots, |\alpha|a_n)$.

So, Parikh mapping of a language $L \subseteq \Sigma^*$ represents a multiset language.
Theorem: If a language $L \subseteq \Sigma^*$ is accepted by some jumping finite automaton then $\Psi(L)$ is accepted by some multiset finite automaton.

Proof: Straightforward. (Both automata have identical state diagrams.)

Theorem: If a multiset language $M \subseteq \Sigma^\oplus$ is accepted by some multiset finite automaton then there is exactly one language $L \subseteq \Sigma^*$ such that $M = \Psi(L)$ and $L$ is accepted by some jumping finite automaton.

Proof: Straightforward. (Both automata have identical state diagrams. The uniqueness follows from the fact that every language accepted by jumping finite automaton contains with every word also any permutation of its symbols.)
Similarities and dissimilarities of multiset languages with languages accepted by jumping finite automata

Consequence: \( \mathcal{L}(\text{JFA}) \) corresponds to \( \mathcal{L}(\text{mFA}) \) and vice versa.

Example:

Jumping finite automaton \( A \)

- \( q_0 \rightarrow q_1 \) on \( a \)
- \( q_0 \rightarrow q_1 \) on \( b \)

Multiset finite automaton \( B \)

- \( q_0 \rightarrow q_1 \) on \( a \)
- \( q_0 \rightarrow q_1 \) on \( b \)

Accepted languages

- \( L(A) = \{ w \in \{ a, b \}^* \mid \lvert w \rvert_a = \lvert w \rvert_b \} \),
- \( \psi(L(A)) = \{ (n, n) \mid n \in \mathbb{N} \} = M(B) \).
Similarities and dissimilarities of multiset languages with languages accepted by jumping finite automata

**Consequence:** \( \mathcal{L}(\text{JFA}) \) corresponds to \( \mathcal{L}(\text{mFA}) \) and vice versa.

**Note:** \( \mathcal{L}(\text{mFA}) \) is equal to the set of all semilinear sets.

**Definition:** A *semilinear set* \( M \) over \( \mathbb{N}^n \) is defined by

\[
M = \bigcup_{i=1}^{k} \left\{ u_{i,0} + l_{i,1}u_{i,1} + \ldots + l_{i,m_i}u_{i,m_i} \mid l_{i,1}, \ldots, l_{i,m_i} \in \mathbb{N} \right\}
\]

where

\[
k, m_1, \ldots, m_k \in \mathbb{N}, \ u_{1,0}, \ldots, u_{1,m_1}, \ldots, u_{k,0}, \ldots, u_{k,m_k} \in \mathbb{N}^n.
\]
## Some closure properties

<table>
<thead>
<tr>
<th>Operation</th>
<th>$\mathcal{L}$ (JFA)</th>
<th>$\mathcal{L}$ (mFA)</th>
</tr>
</thead>
<tbody>
<tr>
<td>union</td>
<td>$+$</td>
<td>$+$</td>
</tr>
<tr>
<td>intersection</td>
<td>$+$</td>
<td>$+$</td>
</tr>
<tr>
<td>complement</td>
<td>$+$</td>
<td>$+$</td>
</tr>
<tr>
<td>concatenation</td>
<td>$-$</td>
<td>$xxx$</td>
</tr>
<tr>
<td>multiset addition</td>
<td>$xxx$</td>
<td>$+$</td>
</tr>
<tr>
<td>homomorphism</td>
<td>$-$</td>
<td>$+$</td>
</tr>
<tr>
<td>substitution</td>
<td>$-$</td>
<td>$+$</td>
</tr>
</tbody>
</table>
Similarities and dissimilarities of multiset languages with languages accepted by jumping finite automata

Definition: A *homomorphism on strings* is defined as a mapping $h : \Sigma^* \rightarrow \Delta^*$ such that

$$h(\varepsilon) = \varepsilon \text{ and } h(u \cdot v) = h(u) \cdot h(v) \text{ for all } u, v \in \Sigma^*.$$

Note that homomorphism “respects” concatenation.

Example: $h : \{0, 1\}^* \rightarrow \{a, b\}^*$ where $h(0) = ab$ and $h(1) = ba$. Then $h(011) = abbaba$.

Definition: A *homomorphism on multisets* is defined as a mapping $h : \Sigma^\oplus \rightarrow \Delta^\oplus$ such that

$$h(0_\Sigma) = 0_\Delta \text{ and } h(\alpha \oplus \beta) = h(\alpha) \oplus h(\beta) \text{ for all } \alpha, \beta \in \Sigma^\oplus.$$
Similarities and dissimilarities of multiset languages with languages accepted by jumping finite automata

**Definition:** A *substitution on strings* is defined as a mapping $s : \Sigma^* \rightarrow 2^{\Delta^*}$ such that

$s(\varepsilon) = \{\varepsilon\}$ and $s(u \cdot v) = s(u) \cdot s(v)$ for all $u, v \in \Sigma^*$.

Note that substitution “respects” concatenation.

**Definition:** A *substitution on multisets* is defined as a mapping $s : \Sigma^{\oplus} \rightarrow 2^{\Delta^{\oplus}}$ such that

$s(0_{\Sigma}) = \{0_{\Delta}\}$ and $s(\alpha \oplus \beta) = s(\alpha) \oplus s(\beta)$ for all $\alpha, \beta \in \Sigma^{\oplus}$. 

Minimization problem for multiset finite automata

Definition: A and B are equivalent iff $M(A) = M(B)$. 

Example:

\[
\begin{align*}
A &: \quad \begin{array}{c|cc}
q_0 & a & b \\
q_1 & & \\
q_2 & & \\
q_3 & & \\
\end{array} \\
B &: \quad \begin{array}{c|c}
q_0 & a, b \\
q_1 & \\
\end{array}
\end{align*}
\]

The automaton $B$ is minimal ($M(B) = \{\langle a \rangle^n \cup \langle b \rangle^n | n \geq 0 \}$).
Minimization problem for multiset finite automata

**Definition:** $A$ and $B$ are equivalent iff $M(A) = M(B)$.

**Definition:** An mFA $A$ is called *minimal* if there is no equivalent mFA $B$ with smaller number of states.
Minimization problem for multiset finite automata

**Definition:** $A$ and $B$ are equivalent iff $M(A) = M(B)$.

**Definition:** An mFA $A$ is called *minimal* if there is no equivalent mFA $B$ with smaller number of states.

**Example:**

$A$: 

- $q_0 \xrightarrow{a} q_0$
- $q_0 \xrightarrow{b} q_1$
- $q_1 \xrightarrow{b} q_2$
- $q_2 \xrightarrow{a} q_3$
- $q_3 \xrightarrow{a} q_2$

$B$: 

- $q_0 \xrightarrow{b} q_1$
- $q_0 \xrightarrow{a} q_1$

The automaton $B$ is minimal ($M(B) = \{ \langle a \rangle^n \oplus \langle b \rangle^n \mid n \geq 0 \}$).
Minimization problem for multiset finite automata

**Problem of minimization:** If an automaton of certain type is given, then we look for an equivalent minimal automaton of the same type.

Optimally, the minimal automaton is unique (up to isomorphism).

Note: the following parts are based on

Minimization problem for multiset finite automata

An example of nonisomorphic minimal nondeterministic multiset finite automata:

\[ M(A) = M(B) = \{ \langle a \rangle, \langle a \rangle \oplus \langle b \rangle \} \]
Minimization problem for multiset finite automata

An example of nonisomorphic and isomorphic minimal deterministic multiset finite automata:

\[ C : \quad q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \]

\[ D : \quad q_0 \xrightarrow{b} q_1 \xrightarrow{a} q_2 \]

\[ M(C) = M(D) = \{ \langle a \rangle \oplus \langle b \rangle \} \]

C and D are not isomorphic.
Minimization problem for multiset finite automata

An example of nonisomorphic and isomorphic minimal deterministic multiset finite automata:

\[ C : \quad q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \]

\[ D : \quad q_0 \xrightarrow{b} q_1 \xrightarrow{a} q_2 \]

\[ M(C) = M(D) = \{ \langle a \rangle \oplus \langle b \rangle \} \]

\[ C \text{ and } D \text{ are not isomorphic.} \]

\[ D' : \quad q_0 \xrightarrow{a} q_2 \xrightarrow{b} q_1 \]

\[ M(C) = M(D') = \{ \langle a \rangle \oplus \langle b \rangle \} \]

\[ C \text{ and } D' \text{ are isomorphic.} \]
Minimization process for deterministic multiset finite automata

1. Removing unreachable states.
Minimization process for deterministic multiset finite automata

1. Removing unreachable states.

Example:

```
q_3 \rightarrow a q_0 \rightarrow b q_1 \rightarrow a q_2
```

1. Removing unreachable states.

Example:
Minimization process for deterministic multiset finite automata

1. Removing unreachable states.
2. Removing nonterminating states.

Example:
Minimization process for deterministic multiset finite automata

1. Removing unreachable states.
2. Removing nonterminating states.

Example:

```
q_3 \rightarrow q_0 \rightarrow q_1 \rightarrow q_2
\text{a} \rightarrow \text{b} \rightarrow \text{a}
```

\[ q_3 \rightarrow q_0 \rightarrow q_1 \rightarrow q_2 \]

\[ \text{a} \rightarrow \text{b} \rightarrow \text{a} \]
Minimization process for deterministic multiset finite automata

3. Lexicographic reordering of transitions:

Input alphabet $\Sigma = \{a, b, c, d\}$
3. Lexicographic reordering of transitions:

Input alphabet $\Sigma = \{a, b, c, d\}$
Minimization process for deterministic multiset finite automata

Key states for lexicographically ordered sequences of transitions at deterministic multiset finite automata:

- the initial state,
Minimization process for deterministic multiset finite automata

Key states for lexicographically ordered sequences of transitions at deterministic multiset finite automata:
- the initial state,
Minimization process for deterministic multiset finite automata

Key states for lexicographically ordered sequences of transitions at deterministic multiset finite automata:
- the initial state,
Minimization process for deterministic multiset finite automata

Key states for lexicographically ordered sequences of transitions at deterministic multiset finite automata:

- the initial state,
- final states,
Minimization process for deterministic multiset finite automata

Key states for lexicographically ordered sequences of transitions at deterministic multiset finite automata:

- the initial state,
- final states,
Minimization process for deterministic multiset finite automata

Key states for lexicographically ordered sequences of transitions at deterministic multiset finite automata:

- the initial state,
- final states,
- the state $q_c$ for which $(q, a, q_c) \in \delta$ and $(r, b, q_c) \in \delta$ with $q \neq r$. 

![Diagram of a deterministic multiset finite automaton with key states highlighted: $q_0$, $q_1$, $q_c$, $q_2$, $q_3$, $q_4$, $q_5$, $q_6$.]
Minimization process for deterministic multiset finite automata

Key states for lexicographically ordered sequences of transitions at deterministic multiset finite automata:

- the initial state,
- final states,
- the state $q_c$ for which $(q, a, q_c) \in \delta$ and $(r, b, q_c) \in \delta$ with $q \neq r$.

![Diagram of automaton states]
Minimization process for deterministic multiset finite automata

**Definition**: A dmFA $A = (Q, \Sigma, \delta, q_0, F)$ is said to be a *deterministic multiset finite automaton with lexicographically ordered transitions* if for any sequence of transitions $(q_i, a_i, q_{i+1})_{i=1}^n$ from $\delta$ with $n \geq 1$, the sequence $(a_i)_{i=1}^n$ is lexicographically ordered whenever $q_1, q_{n+1} \in F \cup \{q_0, q_c\}$ and $q_2, \ldots, q_n \notin F \cup \{q_0, q_c\}$.
**Definition:** A dmFA $A = (Q, \Sigma, \delta, q_0, F)$ is said to be a *deterministic multiset finite automaton with lexicographically ordered transitions* if for any sequence of transitions $(q_i, a_i, q_{i+1})_{i=1}^{n}$ from $\delta$ with $n \geq 1$, the sequence $(a_i)_{i=1}^{n}$ is lexicographically ordered whenever $q_1, q_{n+1} \in F \cup \{q_0, q_c\}$ and $q_2, \ldots, q_n \not\in F \cup \{q_0, q_c\}$.
Minimization process for deterministic multiset finite automata


**Definition:** States \( p, q \in Q \) of a dmFA \( A = (Q, \Sigma, \delta, q_0, F) \) are called *distinguishable* iff there exists \( \alpha \in \Sigma^\oplus \) satisfying either

a) \((p, \alpha) \vdash^* (p', 0_\Sigma)\) with \( p' \in F \) and \((q, \alpha) \vdash^* (q', 0_\Sigma)\) with \( q' \notin F \)

or

b) \((p, \alpha) \vdash^* (p', 0_\Sigma)\) with \( p' \notin F \) and \((q, \alpha) \vdash^* (q', 0_\Sigma)\) with \( q' \in F \).

States, which are not distinguishable, are called *indistinguishable*. 

**Definition:** States \( p, q \in Q \) of a dmFA \( A = (Q, \Sigma, \delta, q_0, F) \) are called *distinguishable* iff there exists \( \alpha \in \Sigma^+ \) satisfying either

a) \((p, \alpha) \vdash^* (p', 0_{\Sigma}) \) with \( p' \in F \) and 
   \((q, \alpha) \vdash^* (q', 0_{\Sigma}) \) with \( q' \not\in F \)

or

b) \((p, \alpha) \vdash^* (p', 0_{\Sigma}) \) with \( p' \not\in F \) and 
   \((q, \alpha) \vdash^* (q', 0_{\Sigma}) \) with \( q' \in F \).

States, which are not distinguishable, are called *indistinguishable*.

**Example:**

Diagram of a deterministic multiset finite automaton.

**Definition:** States $p, q \in Q$ of a dmFA $A = (Q, \Sigma, \delta, q_0, F)$ are called *distinguishable* iff there exists $\alpha \in \Sigma^+$ satisfying either

a) $(p, \alpha) \vdash^* (p', 0_\Sigma)$ with $p' \in F$ and $(q, \alpha) \vdash^* (q', 0_\Sigma)$ with $q' \notin F$

or

b) $(p, \alpha) \vdash^* (p', 0_\Sigma)$ with $p' \notin F$ and $(q, \alpha) \vdash^* (q', 0_\Sigma)$ with $q' \in F$.

States, which are not distinguishable, are called *indistinguishable*.

**Example:**

```
q_5  <--- c  ---> q_4
     |     |     |
     d     a   d  \\
q_0  <----- a  -------> q_1
     |     |
     c     d

q_2  <----- c  -------> q_3
     |     |     |
     d     a   d  \\
```


Definition: States \( p, q \in Q \) of a dmFA \( A = (Q, \Sigma, \delta, q_0, F) \) are called *distinguishable* iff there exists \( \alpha \in \Sigma^\oplus \) satisfying either

a) \((p, \alpha) \vdash^* (p', 0_\Sigma)\) with \(p' \in F\) and \((q, \alpha) \vdash^* (q', 0_\Sigma)\) with \(q' \notin F\)

or

b) \((p, \alpha) \vdash^* (p', 0_\Sigma)\) with \(p' \notin F\) and \((q, \alpha) \vdash^* (q', 0_\Sigma)\) with \(q' \in F\).

States, which are not distinguishable, are called *indistinguishable*.

Example:
Minimization process for deterministic multiset finite automata


Definition: States $p, q \in Q$ of a dmFA $A = (Q, \Sigma, \delta, q_0, F)$ are called *distinguishable* iff there exists $\alpha \in \Sigma^+$ satisfying either

a) $(p, \alpha) \vdash^* (p', \emptyset_{\Sigma})$ with $p' \in F$ and $(q, \alpha) \vdash^* (q', \emptyset_{\Sigma})$ with $q' \notin F$

or

b) $(p, \alpha) \vdash^* (p', \emptyset_{\Sigma})$ with $p' \notin F$ and $(q, \alpha) \vdash^* (q', \emptyset_{\Sigma})$ with $q' \in F$.

States, which are not distinguishable, are called *indistinguishable*.

Example:

Definition: States \( p, q \in Q \) of a dmFA \( A = (Q, \Sigma, \delta, q_0, F) \) are called *distinguishable* iff there exists \( \alpha \in \Sigma^\oplus \) satisfying either

a) \((p, \alpha) \vdash^* (p', 0_{\Sigma})\) with \( p' \in F \) and \((q, \alpha) \vdash^* (q', 0_{\Sigma})\) with \( q' \notin F \)

or

b) \((p, \alpha) \vdash^* (p', 0_{\Sigma})\) with \( p' \notin F \) and \((q, \alpha) \vdash^* (q', 0_{\Sigma})\) with \( q' \in F \).

States, which are not distinguishable, are called *indistinguishable*.

Example:

**Definition:** States $p, q \in Q$ of a dmFA $A = (Q, \Sigma, \delta, q_0, F)$ are called *distinguishable* iff there exists $\alpha \in \Sigma^+$ satisfying either

a) $(p, \alpha) \vdash^* (p', 0_{\Sigma})$ with $p' \in F$ and $(q, \alpha) \vdash^* (q', 0_{\Sigma})$ with $q' \not\in F$

or

b) $(p, \alpha) \vdash^* (p', 0_{\Sigma})$ with $p' \not\in F$ and $(q, \alpha) \vdash^* (q', 0_{\Sigma})$ with $q' \in F$.

States, which are not distinguishable, are called *indistinguishable*.

**Example:**
5. Solving situation around state $q_c$:
   a) shuffling transitions,
   b) merging indistinguishable states,
   c) final lexicographic reordering of transitions.
Minimization process for deterministic multiset finite automata

Example:

```
Example:

<table>
<thead>
<tr>
<th>q_0</th>
<th>a</th>
<th>q_c</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>a</td>
<td>q_a</td>
</tr>
<tr>
<td>q_c</td>
<td>a</td>
<td>q_1</td>
</tr>
<tr>
<td>q_1</td>
<td>a</td>
<td>q_2</td>
</tr>
<tr>
<td>q_2</td>
<td>a</td>
<td>q_3</td>
</tr>
<tr>
<td>q_3</td>
<td>a</td>
<td>q_4</td>
</tr>
<tr>
<td>q_4</td>
<td>a</td>
<td>q_5</td>
</tr>
<tr>
<td>q_5</td>
<td>a</td>
<td>q_0</td>
</tr>
</tbody>
</table>
```

Transitions:
- q_0 -> q_c on input a
- q_c -> q_1 on input a
- q_1 -> q_2 on input a
- q_2 -> q_3 on input a
- q_3 -> q_4 on input a
- q_4 -> q_5 on input a
- q_5 -> q_0 on input a
Minimization process for deterministic multiset finite automata

Example:

\[ q_0 \rightarrow a \rightarrow q_c \rightarrow a \rightarrow q_1 \rightarrow a \rightarrow q_2 \rightarrow a \rightarrow q_3 \rightarrow c \rightarrow q_4 \rightarrow a \rightarrow q_5 \rightarrow a \rightarrow q_0 \]
Minimization process for deterministic multiset finite automata

Example:

\[
\begin{align*}
q_0 & \xrightarrow{a} q_c \xrightarrow{a} q_1 & \xrightarrow{a} q_2 \\
q_0 & \xrightarrow{c} q_4 \xrightarrow{c} q_3 & \xrightarrow{a} q_0
\end{align*}
\]
Minimization process for deterministic multiset finite automata

Example:

```
q0 → a → q_c → a → q_1 → a → q_2
|      |   |      |   |      |   |
|      |   |      |   |      |   |
|      |   |      |   |      |   |
|      |   |      |   |      |   |
|      |   |      |   |      |   |

q_0 → a → q_c → a → q_1 → a → q_2
|   |   |      |   |      |   |
|   |   |      |   |      |   |
|   |   |      |   |      |   |
|   |   |      |   |      |   |
|   |   |      |   |      |   |

q_5 → a → q_4 → a → q_3
|   |   |      |   |      |   |
|   |   |      |   |      |   |
|   |   |      |   |      |   |
|   |   |      |   |      |   |
|   |   |      |   |      |   |

q_0 → a → q_c → a → q_1 → a → q_2
|   |   |      |   |      |   |
|   |   |      |   |      |   |
|   |   |      |   |      |   |
|   |   |      |   |      |   |
|   |   |      |   |      |   |
```
Minimization process for deterministic multiset finite automata

Example:

\[
\begin{align*}
q_0 & \quad \rightarrow \quad a \rightarrow q_c \\
q_c & \quad \rightarrow \quad a \rightarrow q_1 \\
q_1 & \quad \rightarrow \quad a \rightarrow q_2 \\
q_2 & \quad \rightarrow \quad a \rightarrow q_3 \\
q_3 & \quad \rightarrow \quad a \rightarrow q_4 \\
q_4 & \quad \rightarrow \quad a \rightarrow q_5 \\
q_5 & \quad \rightarrow \quad a \rightarrow q_0
\end{align*}
\]
Minimization process for deterministic multiset finite automata

Example:

- \( q_0 \) to \( q_c \) on \( a \)
- \( q_c \) to \( q_1 \) on \( a \)
- \( q_1 \) to \( q_2 \) on \( a \)
- \( q_2 \) to \( q_3 \) on \( a \)
- \( q_3 \) to \( q_4 \) on \( a \)
- \( q_4 \) to \( q_5 \) on \( a \)
- \( q_5 \) to \( q_3 \) on \( a \)
- \( q_0 \) to \( q_c \) on \( c \)
- \( q_1 \) to \( q_2 \) on \( c \)
- \( q_2 \) to \( q_3 \) on \( c \)
- \( q_3 \) to \( q_4 \) on \( c \)
- \( q_4 \) to \( q_5 \) on \( c \)
- \( q_5 \) to \( q_3 \) on \( c \)
Minimization process for deterministic multiset finite automata

Example:
Minimization process for deterministic multiset finite automata

Example:
Minimization process for deterministic multiset finite automata

Example:
Minimization process for deterministic multiset finite automata

Example:

```
q_0 \rightarrow a \rightarrow q_c \rightarrow a \rightarrow q_1 \rightarrow a \rightarrow q_2

q_5 \rightarrow c \rightarrow q_4 \rightarrow a \rightarrow q_3

q_0 \rightarrow a \rightarrow q_c

q_5 \rightarrow a \rightarrow q_4
```

Graphical representation of the minimization process.
Minimization process for deterministic multiset finite automata

Example:

- Transition from $q_0$ to $q_c$ on input $a$
- Transition from $q_c$ to $q_1$ on input $a$
- Transition from $q_1$ to $q_2$ on input $a$
- Transition from $q_2$ to $q_3$ on input $a$
- Transition from $q_3$ to $q_4$ on input $a$
- Transition from $q_4$ to $q_5$ on input $a$
- Transition from $q_5$ to $q_c$ on input $a$
- Transition from $q_0$ to $q_c$ on input $c$
- Transition from $q_c$ to $q_1$ on input $c$
- Transition from $q_1$ to $q_2$ on input $c$
- Transition from $q_2$ to $q_3$ on input $c$
- Transition from $q_3$ to $q_4$ on input $c$
- Transition from $q_4$ to $q_5$ on input $c$
- Transition from $q_5$ to $q_c$ on input $c$

- Transition from $q_0$ to $q_c$ on input $a$
- Transition from $q_c$ to $q_1$ on input $a$
- Transition from $q_1$ to $q_2$ on input $a$
- Transition from $q_2$ to $q_3$ on input $a$
- Transition from $q_3$ to $q_4$ on input $a$
- Transition from $q_4$ to $q_5$ on input $a$
- Transition from $q_5$ to $q_c$ on input $a$

- Transition from $q_0$ to $q_c$ on input $c$
- Transition from $q_c$ to $q_1$ on input $c$
- Transition from $q_1$ to $q_2$ on input $c$
- Transition from $q_2$ to $q_3$ on input $c$
- Transition from $q_3$ to $q_4$ on input $c$
- Transition from $q_4$ to $q_5$ on input $c$
- Transition from $q_5$ to $q_c$ on input $c$
Minimization problem for multiset finite automata

Theorem: For every dmFA $A$, there is an equivalent minimal deterministic multiset finite automaton with lexicographically ordered transitions which is unique up to isomorphism.
Minimization problem for multiset finite automata

**Theorem:** For every dmFA $A$, there is an equivalent minimal deterministic multiset finite automaton with lexicographically ordered transitions which is unique up to isomorphism.

**Remark:** For some nondeterministic multiset finite automata, no unique equivalent minimal multiset finite automata exist.
Minimization problem for multiset finite automata in a generalized form

An unusual concept of a generalized multiset finite automaton with suppressed nonfinal states allows to grasp the minimization somewhat differently.

A generalized multiset finite automaton with suppressed nonfinal states: $A = (Q, \Sigma, \delta, q_0, F)$ where

- $Q, \Sigma$ and $q_0$ are as in mFA,
- $F \subseteq Q$ is a set of final states such that $F \cup \{q_0\} = Q$,
- $\delta \subseteq Q \times \Sigma^\oplus \times Q$ is a finite transition relation satisfying the following condition.

If there are $q_1, \ldots, q_k \in Q$ such that $(q_{i-1}, 0, q_i) \in \delta$ for all $i \in \{1, \ldots, k\}$ and $q_k \in F$, then $q_0 \in F$. 
Generalized multiset finite automata with suppressed nonfinal states

Examples:

A:

\[
\begin{align*}
q_0 & \xrightarrow{a^2} q_1 \\
q_1 & \xrightarrow{a^3} q_2 \\
q_2 & \xrightarrow{a \oplus b^2}
\end{align*}
\]

B:

\[
\begin{align*}
q_0 & \xrightarrow{a} q_1 \\
q_1 & \xrightarrow{b \oplus c} q_2 \\
q_2 & \xrightarrow{a \oplus b}
\end{align*}
\]
Idea of a transformation to a generalized multiset automaton with suppressed nonfinal states
Idea of a transformation to a generalized multiset automaton with suppressed nonfinal states
Idea of a transformation to a generalized multiset automaton with suppressed nonfinal states

\[ B : \begin{array}{c}
q_0 \quad a \quad q_1 \\
\quad \downarrow \quad d \\
q_0 \quad a \oplus b \quad q_2 \\
\quad \downarrow \\
q_0 \quad a \oplus b \oplus \langle c \rangle \quad q_2 \\
\quad \downarrow \\
q_0 \quad a \oplus b \oplus \langle c \rangle \oplus \langle a \rangle \oplus \langle b \rangle \\
\end{array} \]

\[ B' : \begin{array}{c}
q_0 \quad \langle a \rangle \oplus \langle b \rangle \\
\quad \downarrow \\
q_0 \quad \langle a \rangle \oplus \langle b \rangle \quad q_2 \\
\quad \downarrow \\
q_0 \quad \langle a \rangle \oplus \langle b \rangle \\
\end{array} \]

\[ B'' : \begin{array}{c}
q_0 \quad \langle a \rangle \oplus \langle b \rangle \\
\quad \downarrow \\
q_0 \quad \langle a \rangle \oplus \langle b \rangle \quad q_2 \\
\quad \downarrow \\
q_0 \quad \langle c \rangle \oplus \langle a \rangle \oplus \langle b \rangle \\
\end{array} \]
Idea of a transformation to a generalized multiset automaton with suppressed nonfinal states

\[
B : \quad q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2
\]

\[
B' : \quad q_0 \xrightarrow{\langle a \rangle \oplus \langle b \rangle} q_2
\]

\[
B'' : \quad q_0 \xrightarrow{\langle a \rangle \oplus \langle b \rangle} q_2
\]

\[
M(B) = M(B'')
\]
Generalized multiset finite automata with suppressed nonfinal states and their minimization

**Theorem:** Generalized multiset finite automata with suppressed nonfinal states accept the family of multiset languages accepted by multiset finite automata.
Generalized multiset finite automata with suppressed nonfinal states and their minimization

**Theorem**: Generalized multiset finite automata with suppressed nonfinal states accept the family of multiset languages accepted by multiset finite automata.

**Theorem**: For every deterministic multiset finite automaton, there is an equivalent minimal deterministic generalized multiset finite automaton with suppressed nonfinal states which is unique up to isomorphism.
Generalized multiset finite automata with suppressed nonfinal states and their minimization

**Theorem**: Generalized multiset finite automata with suppressed nonfinal states accept the family of multiset languages accepted by multiset finite automata.

**Theorem**: For every deterministic multiset finite automaton, there is an equivalent minimal deterministic generalized multiset finite automaton with suppressed nonfinal states which is unique up to isomorphism.

**Note**: Generally, the previous theorem does not hold true for nondeterministic generalized multiset finite automata with suppressed nonfinal states.
An example of nonisomorphic minimal nondeterministic generalized multiset finite automata with suppressed nonfinal states

\[ M(A) = M(B) = \{ \langle a \rangle \oplus \langle b \rangle^m \mid m \geq 0 \} \cup \{ \langle a \rangle^n \mid n \geq 1 \} \]
Conclusion

Multiset languages theory:

▶ Large field for further research
▶ Many unsolved problems
Thank you for your attention