DTDs mit einfachen regulären Ausdrücken

Mark Wiesemann

Seminar über „Automaten für XML“

15. November 2005

Übersicht

Übersicht

Einführung

Motivation

- XML-Dokumente sind über Dokumenttypdefinitionen (DTD) definiert
- DTDs können als erweiterte kontextfreie Grammatiken interpretiert werden
- auf den rechten Regelseiten können reguläre Ausdrücke vorkommen

Beispiel

$\texttt{library} \rightarrow \texttt{book}^*$

$\texttt{book} \rightarrow \texttt{title author}^+ \texttt{ date isbn}$

$\texttt{date} \rightarrow \texttt{month year}$

Einführung

Motivation

- XML-Dokumente sind über Dokumenttypdefinitionen (DTD) definiert
- DTDs können als erweiterte kontextfreie Grammatiken interpretiert werden
- auf den rechten Regelseiten können reguläre Ausdrücke vorkommen

Beispiel

$\texttt{library} \longrightarrow \texttt{book}^*$

$\texttt{book} \longrightarrow \texttt{title author}^+ \texttt{ date isbn}$

$\texttt{date} \longrightarrow \texttt{month year}$

Einführung

Motivation

- XML-Dokumente sind über Dokumenttypdefinitionen (DTD) definiert
- DTDs können als erweiterte kontextfreie Grammatiken interpretiert werden
- auf den rechten Regelseiten können reguläre Ausdrücke vorkommen

Beispiel

$\texttt{library} \longrightarrow \texttt{book}^*$

$\texttt{book} \longrightarrow \texttt{title author}^+ \texttt{ date isbn}$

$\texttt{date} \longrightarrow \texttt{month year}$

Einführung

Motivation

- XML-Dokumente sind über Dokumenttypdefinitionen (DTD) definiert
- DTDs können als erweiterte kontextfreie Grammatiken interpretiert werden
- auf den rechten Regelseiten können reguläre Ausdrücke vorkommen

Beispiel

$\texttt{library} \longrightarrow \texttt{book}^*$

$\texttt{book} \longrightarrow \texttt{title author}^+ \texttt{ date isbn}$

$\texttt{date} \longrightarrow \texttt{month year}$

Einführung

Motivation

- XML-Dokumente sind über Dokumenttypdefinitionen (DTD) definiert
- DTDs können als erweiterte kontextfreie Grammatiken interpretiert werden
- auf den rechten Regelseiten können reguläre Ausdrücke vorkommen
- in der Praxis: nur *einfache* reguläre Ausdrücke in DTDs
- Analyse von XML/DTDs erfordert Lösen von Entscheidungsproblemen (Inklusion, Äquivalenz, Schnitt)
- im Allgemeinen nicht effizient (PSPACE-vollständig)

Einführung

Motivation

- XML-Dokumente sind über Dokumenttypdefinitionen (DTD) definiert
- DTDs können als erweiterte kontextfreie Grammatiken interpretiert werden
- auf den rechten Regelseiten können reguläre Ausdrücke vorkommen
- in der Praxis: nur *einfache* reguläre Ausdrücke in DTDs
- Analyse von XML/DTDs erfordert Lösen von Entscheidungsproblemen (Inklusion, Äquivalenz, Schnitt)
- im Allgemeinen nicht effizient (PSPACE-vollständig)

Einführung

Motivation

- XML-Dokumente sind über Dokumenttypdefinitionen (DTD) definiert
- DTDs können als erweiterte kontextfreie Grammatiken interpretiert werden
- auf den rechten Regelseiten können reguläre Ausdrücke vorkommen
- in der Praxis: nur *einfache* reguläre Ausdrücke in DTDs
- Analyse von XML/DTDs erfordert Lösen von Entscheidungsproblemen (Inklusion, Äquivalenz, Schnitt)
- im Allgemeinen nicht effizient (PSPACE-vollständig)

reguläre Ausdrücke über Σ

Wiederholung

Induktive Definition:

- $\emptyset$, ϵ, a für $a \in \Sigma$ sind reguläre Ausdrücke
- sind r und s reguläre Ausdrücke, dann auch rs, $r + s$ und r^*

reguläre Ausdrücke über Σ

Wiederholung

Induktive Definition:

- $\varnothing$, ϵ, a für $a \in \Sigma$ sind reguläre Ausdrücke
- sind r und s reguläre Ausdrücke, dann auch rs, $r + s$ und r^*

reguläre Ausdrücke über Σ

Wiederholung

Induktive Definition:

- $\varnothing$, ϵ, a für $a \in \Sigma$ sind reguläre Ausdrücke
- sind r und s reguläre Ausdrücke, dann auch rs, $r + s$ und r^*

reguläre Ausdrücke über Σ

Wiederholung

Induktive Definition:

- $\varnothing$, ϵ, a für $a \in \Sigma$ sind reguläre Ausdrücke
- sind r und s reguläre Ausdrücke, dann auch rs, $r + s$ und r^*

Abkürzungen:

- $r?$ für $r + \epsilon$, r^+ für rr^*

reguläre Ausdrücke über Σ

Wiederholung

Induktive Definition:

- $\varnothing$, ϵ, a für $a \in \Sigma$ sind reguläre Ausdrücke
- sind r und s reguläre Ausdrücke, dann auch rs, $r + s$ und r^*

Abkürzungen:

- $r?$ für $r + \epsilon$, r^+ für rr^*

reguläre Ausdrücke über Σ

Wiederholung

Induktive Definition:

- $\varnothing$, ϵ, a für $a \in \Sigma$ sind reguläre Ausdrücke
- sind r und s reguläre Ausdrücke, dann auch rs, $r + s$ und r^*

Abkürzungen:

- $r?$ für $r + \epsilon$, r^+ für rr^*

Definierte Sprachen:

- $\mathcal{L}(\varnothing) = \varnothing$, $\mathcal{L}(\epsilon) = \{\epsilon\}$, $\mathcal{L}(a) = \{a\}$ für $a \in \Sigma$
- $\mathcal{L}(rs) = \{vw \mid v \in \mathcal{L}(r), w \in \mathcal{L}(s)\}$
- $\mathcal{L}(r + s) = \mathcal{L}(r) \cup \mathcal{L}(s)$
- $\mathcal{L}(r^*) = \{\epsilon\} \cup \bigcup_{i=1}^{\infty} \mathcal{L}(r)^i$

reguläre Ausdrücke über Σ

Wiederholung

Induktive Definition:

- $\varnothing$, ϵ, a für $a \in \Sigma$ sind reguläre Ausdrücke
- sind r und s reguläre Ausdrücke, dann auch rs, $r + s$ und r^*

Abkürzungen:

- $r?$ für $r + \epsilon$, r^+ für rr^*

Definierte Sprachen:

- $\mathcal{L}(\varnothing) = \varnothing$, $\mathcal{L}(\epsilon) = \{\epsilon\}$, $\mathcal{L}(a) = \{a\}$ für $a \in \Sigma$
- $\mathcal{L}(rs) = \{vw \mid v \in \mathcal{L}(r), w \in \mathcal{L}(s)\}$
- $\mathcal{L}(r + s) = \mathcal{L}(r) \cup \mathcal{L}(s)$
- $\mathcal{L}(r^*) = \{\epsilon\} \cup \bigcup_{i=1}^{\infty} \mathcal{L}(r)^i$

reguläre Ausdrücke über Σ

Wiederholung

Induktive Definition:

- $\varnothing$, ϵ, a für $a \in \Sigma$ sind reguläre Ausdrücke
- sind r und s reguläre Ausdrücke, dann auch rs, $r + s$ und r^*

Abkürzungen:

- $r?$ für $r + \epsilon$, r^+ für rr^*

Definierte Sprachen:

- $\mathcal{L}(\varnothing) = \varnothing$, $\mathcal{L}(\epsilon) = \{\epsilon\}$, $\mathcal{L}(a) = \{a\}$ für $a \in \Sigma$
- $\mathcal{L}(rs) = \{vw \mid v \in \mathcal{L}(r), w \in \mathcal{L}(s)\}$
- $\mathcal{L}(r + s) = \mathcal{L}(r) \cup \mathcal{L}(s)$
- $\mathcal{L}(r^*) = \{\epsilon\} \cup \bigcup_{i=1}^{\infty} \mathcal{L}(r)^i$

reguläre Ausdrücke über Σ

Wiederholung

Induktive Definition:

- $\varnothing$, ϵ, a für $a \in \Sigma$ sind reguläre Ausdrücke
- sind r und s reguläre Ausdrücke, dann auch rs, $r + s$ und r^*

Abkürzungen:

- $r?$ für $r + \epsilon$, r^+ für rr^*

Definierte Sprachen:

- $\mathcal{L}(\varnothing) = \varnothing$, $\mathcal{L}(\epsilon) = \{\epsilon\}$, $\mathcal{L}(a) = \{a\}$ für $a \in \Sigma$
- $\mathcal{L}(rs) = \{vw \mid v \in \mathcal{L}(r), w \in \mathcal{L}(s)\}$
- $\mathcal{L}(r + s) = \mathcal{L}(r) \cup \mathcal{L}(s)$
- $\mathcal{L}(r^*) = \{\epsilon\} \cup \bigcup_{i=1}^{\infty} \mathcal{L}(r)^i$

reguläre Ausdrücke über Σ

Wiederholung

Induktive Definition:

- $\varnothing$, ϵ, a für $a \in \Sigma$ sind reguläre Ausdrücke
- sind r und s reguläre Ausdrücke, dann auch rs, $r + s$ und r^*

Abkürzungen:

- $r?$ für $r + \epsilon$, r^+ für rr^*

Definierte Sprachen:

- $\mathcal{L}(\varnothing) = \varnothing$, $\mathcal{L}(\epsilon) = \{\epsilon\}$, $\mathcal{L}(a) = \{a\}$ für $a \in \Sigma$
- $\mathcal{L}(rs) = \{vw \mid v \in \mathcal{L}(r), w \in \mathcal{L}(s)\}$
- $\mathcal{L}(r + s) = \mathcal{L}(r) \cup \mathcal{L}(s)$
- $\mathcal{L}(r^*) = \{\epsilon\} \cup \bigcup_{i=1}^{\infty} \mathcal{L}(r)^i$

Klassen von einfachen regulären Ausdrücken

Definition

Sei Σ ein endliches Alphabet.

- $\mathrm{RE}(a, a^*) = \{e_1 \dots e_n \mid e_i = a \text{ oder } e_i = a^* \text{ für ein } a \in \Sigma\}$
- $\mathrm{RE}(a, a^+) = \{e_1 \dots e_n \mid e_i = a \text{ oder } e_i = a^+ \text{ für ein } a \in \Sigma\}$

Beispiele

Klassen von einfachen regulären Ausdrücken

Definition

Sei Σ ein endliches Alphabet.

- $\mathrm{RE}(a, a^*) = \{e_1 \dots e_n \mid e_i = a \text{ oder } e_i = a^* \text{ für ein } a \in \Sigma\}$
- $\mathrm{RE}(a, a^+) = \{e_1 \dots e_n \mid e_i = a \text{ oder } e_i = a^+ \text{ für ein } a \in \Sigma\}$

Beispiele

- in $\mathrm{RE}(a, a^*)$ liegen Ausdrücke wie
  - a^*
  - a^*b
  - ab^*c^*d
- in $\mathrm{RE}(a, a^+)$ liegen Ausdrücke wie
  - aa^+bc
  - ab
  - b

Klassen von einfachen regulären Ausdrücken

Definition

Sei Σ ein endliches Alphabet.

- $\mathrm{RE}(a, a^*) = \{e_1 \dots e_n \mid e_i = a \text{ oder } e_i = a^* \text{ für ein } a \in \Sigma\}$
- $\mathrm{RE}(a, a^+) = \{e_1 \dots e_n \mid e_i = a \text{ oder } e_i = a^+ \text{ für ein } a \in \Sigma\}$

Beispiele

- in $\mathrm{RE}(a, a^*)$ liegen Ausdrücke wie
  - a^*
  - $a^* b$
  - $ab^* c^* d$
- in $\mathrm{RE}(a, a^+)$ liegen Ausdrücke wie
  - $ab^+ cd$
  - ab
  - $b^+ c$

Klassen von einfachen regulären Ausdrücken

Definition

Sei Σ ein endliches Alphabet.

- $\mathrm{RE}(a, a^*) = \{e_1 \dots e_n \mid e_i = a \text{ oder } e_i = a^* \text{ für ein } a \in \Sigma\}$
- $\mathrm{RE}(a, a^+) = \{e_1 \dots e_n \mid e_i = a \text{ oder } e_i = a^+ \text{ für ein } a \in \Sigma\}$

Beispiele

- in $\mathrm{RE}(a, a^*)$ liegen Ausdrücke wie
  - a^*
  - $a^* b$
  - $a b^* c^* d$
- in $\mathrm{RE}(a, a^+)$ liegen Ausdrücke wie
  - $a b^+ c d$
  - $a b$
  - $b^+ c$

Klassen von einfachen regulären Ausdrücken

Definition

Sei Σ ein endliches Alphabet.

- $\mathrm{RE}(a, a^*) = \{e_1 \dots e_n \mid e_i = a \text{ oder } e_i = a^* \text{ für ein } a \in \Sigma\}$
- $\mathrm{RE}(a, a^+) = \{e_1 \dots e_n \mid e_i = a \text{ oder } e_i = a^+ \text{ für ein } a \in \Sigma\}$

Beispiele

- in $\mathrm{RE}(a, a^*)$ liegen Ausdrücke wie
  - a^*
  - $a^* b$
  - $a b^* c^* d$
- in $\mathrm{RE}(a, a^+)$ liegen Ausdrücke wie
  - $a b^+ c d$
  - $a b$
  - $b^+ c$

Dokumenttypdefinition

Definition

- Dokumenttypdefinition (DTD) ist Paar (d, s_d)
- d ist Funktion, die Σ-Symbole auf reguläre Ausdrücke über Σ abbildet; $s_d \in \Sigma$ ist Startsymbol

Beschreibung

Dokumenttypdefinition

Definition

- Dokumenttypdefinition (DTD) ist Paar (d, s_d)
- d ist Funktion, die Σ-Symbole auf reguläre Ausdrücke über Σ abbildet; $s_d \in \Sigma$ ist Startsymbol

Beschreibung

- beschreibt die Struktur eines XML-Dokuments
- kann als erweiterte kontextfreie Grammatik gesehen werden
- Grundaufbau:

```
<!DOCTYPE Name-einer-DTD [
  Liste der Elementdeklarationen
]>
```

- Elemente (Tags) sind definiert als

```
<!ELEMENT Elementname (Beschreibung des Elements)>
```

Dokumenttypdefinition

Definition

- Dokumenttypdefinition (DTD) ist Paar (d, s_d)
- d ist Funktion, die Σ-Symbole auf reguläre Ausdrücke über Σ abbildet; $s_d \in \Sigma$ ist Startsymbol

Beschreibung

- beschreibt die Struktur eines XML-Dokuments
- kann als erweiterte kontextfreie Grammatik gesehen werden
- Grundaufbau:
  - <!DOCTYPE *Name-der-DTD* [
  - *Liste der Elementdefinitionen*
  -]>
- Elemente (Tags) sind definiert als:
  - <!ELEMENT *Elementname (Unterelemente oder #PCDATA)*>

Dokumenttypdefinition

Definition

- Dokumenttypdefinition (DTD) ist Paar (d, s_d)
- d ist Funktion, die Σ-Symbole auf reguläre Ausdrücke über Σ abbildet; $s_d \in \Sigma$ ist Startsymbol

Beschreibung

- beschreibt die Struktur eines XML-Dokuments
- kann als erweiterte kontextfreie Grammatik gesehen werden
- Grundaufbau:

```
<!DOCTYPE Name-der-DTD [
   Liste der Elementdefinitionen
]>
```

- Elemente (Tags) sind definiert als:

```
<!ELEMENT Elementname (Beschreibung des Elements)>
```

Dokumenttypdefinition

Definition

- Dokumenttypdefinition (DTD) ist Paar (d, s_d)
- d ist Funktion, die Σ-Symbole auf reguläre Ausdrücke über Σ abbildet; $s_d \in \Sigma$ ist Startsymbol

Beschreibung

- beschreibt die Struktur eines XML-Dokuments
- kann als erweiterte kontextfreie Grammatik gesehen werden
- Grundaufbau:

```
<!DOCTYPE Name-der-DTD [
  Liste der Elementdefinitionen
]>
```

- Elemente (Tags) sind definiert als:

```
<!ELEMENT Elementname (Beschreibung des Elements)>
```

Dokumenttypdefinition

Definition

- Dokumenttypdefinition (DTD) ist Paar (d, s_d)
- d ist Funktion, die Σ-Symbole auf reguläre Ausdrücke über Σ abbildet; $s_d \in \Sigma$ ist Startsymbol

Beschreibung

- beschreibt die Struktur eines XML-Dokuments
- kann als erweiterte kontextfreie Grammatik gesehen werden
- Grundaufbau:

```
<!DOCTYPE Name-der-DTD [
  Liste der Elementdefinitionen
]>
```

- Elemente (Tags) sind definiert als:

```
<!ELEMENT Elementname (Beschreibung des Elements)>
```

Dokumenttypdefinition

Definition

- DOKUMENTTYPDEFINITION (DTD) ist Paar (d, s_d)
- d ist Funktion, die Σ-Symbole auf reguläre Ausdrücke über Σ abbildet; $s_d \in \Sigma$ ist Startsymbol

Beschreibung

- beschreibt die Struktur eines XML-Dokuments
- kann als erweiterte kontextfreie Grammatik gesehen werden
- Grundaufbau:

```
<!DOCTYPE Name-der-DTD [
  Liste der Elementdefinitionen
]>
```

- Elemente (Tags) sind definiert als:

```
<!ELEMENT Elementname (Beschreibung des Elements)>
```

Dokumenttypdefinition

Definition

- DOKUMENTTYPDEFINITION (DTD) ist Paar (d, s_d)
- d ist Funktion, die Σ-Symbole auf reguläre Ausdrücke über Σ abbildet; $s_d \in \Sigma$ ist Startsymbol

Beschreibung

- beschreibt die Struktur eines XML-Dokuments
- kann als erweiterte kontextfreie Grammatik gesehen werden
- Grundaufbau:

```
<!DOCTYPE Name-der-DTD [
  Liste der Elementdefinitionen
]>
```

- Elemente (Tags) sind definiert als:

```
<!ELEMENT Elementname (Beschreibung des Elements)>
```

Dokumenttypdefinition

Beispiel

$\texttt{library} \rightarrow \texttt{book}^*$
$\texttt{book} \rightarrow \texttt{title author}^+ \texttt{ date isbn}$
$\texttt{date} \rightarrow \texttt{month year}$

```
<!DOCTYPE library [
  <!ELEMENT library (book*)>
  <!ELEMENT book (title,
          author+, date, isbn)>
  <!ELEMENT title (#PCDATA)>
  <!ELEMENT author (#PCDATA)>
  <!ELEMENT date (month, year)>
  <!ELEMENT isbn (#PCDATA)>
  <!ELEMENT month (#PCDATA)>
  <!ELEMENT year (#PCDATA)>
]>
```

```
<library>
  <book>
    <title>XML in a Nutshell</title>
    <author>Harold</author>
    <author>Means</author>
    <date>
      <month>1</month>
      <year>2005</year>
    </date>
    <isbn>3-89721-339-7</isbn>
  </book>
  <book>
    ...
  </book>
</library>
```

Dokumenttypdefinition

Beispiel

$$\texttt{library} \longrightarrow \texttt{book}^*$$
$$\texttt{book} \longrightarrow \texttt{title author}^+ \texttt{ date isbn}$$
$$\texttt{date} \longrightarrow \texttt{month year}$$

```
<!DOCTYPE library [
  <!ELEMENT library (book*)>
  <!ELEMENT book (title,
          author+, date, isbn)>
  <!ELEMENT title (#PCDATA)>
  <!ELEMENT author (#PCDATA)>
  <!ELEMENT date (month, year)>
  <!ELEMENT isbn (#PCDATA)>
  <!ELEMENT month (#PCDATA)>
  <!ELEMENT year (#PCDATA)>
]>
```

```
<library>
  <book>
    <title>XML in a Nutshell</title>
    <author>Harold</author>
    <author>Means</author>
    <date>
      <month>1</month>
      <year>2005</year>
    </date>
    <isbn>3-89721-339-7</isbn>
  </book>
  <book>
    ...
  </book>
</library>
```

Dokumenttypdefinition

Beispiel

$\texttt{library} \rightarrow \texttt{book}^*$
$\texttt{book} \rightarrow \texttt{title author}^+ \texttt{ date isbn}$
$\texttt{date} \rightarrow \texttt{month year}$

```
<!DOCTYPE library [
  <!ELEMENT library (book*)>
  <!ELEMENT book (title,
          author+, date, isbn)>
  <!ELEMENT title (#PCDATA)>
  <!ELEMENT author (#PCDATA)>
  <!ELEMENT date (month, year)>
  <!ELEMENT isbn (#PCDATA)>
  <!ELEMENT month (#PCDATA)>
  <!ELEMENT year (#PCDATA)>
]>
```

```
<library>
  <book>
    <title>XML in a Nutshell</title>
    <author>Harold</author>
    <author>Means</author>
    <date>
      <month>1</month>
      <year>2005</year>
    </date>
    <isbn>3-89721-339-7</isbn>
  </book>
  <book>
    ...
  </book>
</library>
```

Entscheidungsprobleme

Definition

Sei $\mathcal{R}$ eine Klasse von regulären Ausdrücken.

- Das Inklusionsproblem für $\mathcal{R}$ ist, für $r, r' \in \mathcal{R}$ zu entscheiden, ob $L(r) \subseteq L(r')$ gilt.
- Das Äquivalenzproblem für $\mathcal{R}$ ist, für $r, r' \in \mathcal{R}$ zu entscheiden, ob $L(r) = L(r')$ gilt.
- Das Schnittproblem für $\mathcal{R}$ ist, für $r_1, \ldots, r_n \in \mathcal{R}$ (mit $n \in \mathbb{N}$) zu entscheiden, ob $\bigcap_{i=1}^{n} L(r_i) \neq \varnothing$ gilt.

Entscheidungsprobleme

Definition

Sei $\mathcal{R}$ eine Klasse von regulären Ausdrücken.

- Das Inklusionsproblem für $\mathcal{R}$ ist, für $r, r' \in \mathcal{R}$ zu entscheiden, ob $L(r) \subseteq L(r')$ gilt.
- Das Äquivalenzproblem für $\mathcal{R}$ ist, für $r, r' \in \mathcal{R}$ zu entscheiden, ob $L(r) = L(r')$ gilt.
- Das Schnittproblem für $\mathcal{R}$ ist, für $r_1, \ldots, r_n \in \mathcal{R}$ (mit $n \in \mathbb{N}$) zu entscheiden, ob $\bigcap_{i=1}^{n} L(r_i) \neq \varnothing$ gilt.

Entscheidungsprobleme

Definition

Sei $\mathcal{R}$ eine Klasse von regulären Ausdrücken.

- Das Inklusionsproblem für $\mathcal{R}$ ist, für $r, r' \in \mathcal{R}$ zu entscheiden, ob $L(r) \subseteq L(r')$ gilt.
- Das Äquivalenzproblem für $\mathcal{R}$ ist, für $r, r' \in \mathcal{R}$ zu entscheiden, ob $L(r) = L(r')$ gilt.
- Das Schnittproblem für $\mathcal{R}$ ist, für $r_1, \ldots, r_n \in \mathcal{R}$ (mit $n \in \mathbb{N}$) zu entscheiden, ob $\bigcap_{i=1}^{n} L(r_i) \neq \varnothing$ gilt.

Entscheidungsprobleme

Definition

Sei $\mathcal{R}$ eine Klasse von regulären Ausdrücken.

- Das Inklusionsproblem für $\mathcal{R}$ ist, für $r, r' \in \mathcal{R}$ zu entscheiden, ob $L(r) \subseteq L(r')$ gilt.
- Das Äquivalenzproblem für $\mathcal{R}$ ist, für $r, r' \in \mathcal{R}$ zu entscheiden, ob $L(r) = L(r')$ gilt.
- Das Schnittproblem für $\mathcal{R}$ ist, für $r_1, \ldots, r_n \in \mathcal{R}$ (mit $n \in \mathbb{N}$) zu entscheiden, ob $\bigcap_{i=1}^{n} L(r_i) \neq \varnothing$ gilt.

Komplexitäten

Bemerkung

Die drei Entscheidungsprobleme sind für allgemeine reguläre Ausdrücke PSPACE-vollständig.

Ergebnisse von Martens, Neven und Schwentick

	Inklusion	Äquivalenz	Schnitt
a, a^+	PTIME	PTIME	PTIME
a, a^*	CONP-vollst.	PTIME	NP-vollst.
$a, a?$	CONP-vollst.	PTIME	NP-vollst.
$a, (+a)^*$	PSPACE-vollst.	PSPACE	NP-vollst.
S	PSPACE-vollst.	PSPACE	PSPACE
$\mathrm{RE}^{\leq k}$ $(k \geq 3)$	PTIME	PTIME	PSPACE-vollst.

Komplexitäten

Bemerkung

Die drei Entscheidungsprobleme sind für allgemeine reguläre Ausdrücke PSPACE-vollständig.

Ergebnisse von Martens, Neven und Schwentick

	Inklusion	Äquivalenz	Schnitt
a, a^+	PTIME	PTIME	PTIME
a, a^*	CONP-vollst.	PTIME	NP-vollst.
$a, a?$	CONP-vollst.	PTIME	NP-vollst.
$a, (+a)^*$	PSPACE-vollst.	PSPACE	NP-vollst.
S	PSPACE-vollst.	PSPACE	PSPACE
$\mathrm{RE}^{\leq k}$ $(k \geq 3)$	PTIME	PTIME	PSPACE-vollst.

Übersicht

Äquivalenzproblem für reguläre Ausdrücke

Theorem

Für RE(a, a^*) liegt das Äquivalenzproblem in PTIME.

Beweisidee

- definiere starke Sequenz-Normalform für reguläre Ausdrücke aus RE(a, a^*), um Äquivalenz zu bestimmen
- diese Normalform kann in PTIME berechnet werden
- zu zeigen:

  zwei reguläre Ausdrücke aus RE(a, a^*) sind äquivalent

  $\Leftrightarrow$

  sie haben die gleiche starke Sequenz-Normalform

Äquivalenzproblem für reguläre Ausdrücke

Theorem

Für RE(a, a^*) liegt das Äquivalenzproblem in PTIME.

Beweisidee

- definiere starke Sequenz-Normalform für reguläre Ausdrücke aus RE(a, a^*), um Äquivalenz zu bestimmen
- diese Normalform kann in PTIME berechnet werden
- zu zeigen:

zwei reguläre Ausdrücke aus RE(a, a^*) sind äquivalent

$\Leftrightarrow$

sie haben die gleiche starke Sequenz-Normalform

Äquivalenzproblem für reguläre Ausdrücke

Theorem

Für RE(a, a^*) liegt das Äquivalenzproblem in PTIME.

Beweisidee

- definiere starke Sequenz-Normalform für reguläre Ausdrücke aus RE(a, a^*), um Äquivalenz zu bestimmen
- diese Normalform kann in PTIME berechnet werden
- zu zeigen:

  zwei reguläre Ausdrücke aus RE(a, a^*) sind äquivalent

  $\Leftrightarrow$

  sie haben die gleiche starke Sequenz-Normalform

Äquivalenzproblem für reguläre Ausdrücke

Theorem

Für RE(a, a^*) liegt das Äquivalenzproblem in PTIME.

Beweisidee

- definiere starke Sequenz-Normalform für reguläre Ausdrücke aus RE(a, a^*), um Äquivalenz zu bestimmen
- diese Normalform kann in PTIME berechnet werden
- zu zeigen:

zwei reguläre Ausdrücke aus RE(a, a^*) sind äquivalent

$\Leftrightarrow$

sie haben die gleiche starke Sequenz-Normalform

Äquivalenzproblem für reguläre Ausdrücke

Theorem

Für RE(a, a^*) liegt das Äquivalenzproblem in PTIME.

Beweisidee

- definiere starke Sequenz-Normalform für reguläre Ausdrücke aus RE(a, a^*), um Äquivalenz zu bestimmen
- diese Normalform kann in PTIME berechnet werden
- zu zeigen:

zwei reguläre Ausdrücke aus RE(a, a^*) sind äquivalent

$\Leftrightarrow$

sie haben die gleiche starke Sequenz-Normalform

Sequenz-Normalform

Definition

Sei r ein regulärer Ausdruck aus RE(a, a^*) und sei $e \in \Sigma$.

- $e[i, j]$ gibt an, dass e mindestens i-mal und höchstens j-mal direkt nacheinander in r vorkommt
- $j = *$ gibt dabei an, dass die Anzahl der Wiederholungen von e unbeschränkt ist
- $e[1, 1]$ steht für e und $e[0, *]$ steht für e^*
- SEQUENZ-NORMALFORM entsteht durch Zusammenfassung von $e[i_1, j_1]$ und $e[i_2, j_2]$ zu $e[i_1 + i_2, j_1 + j_2]$, wann immer möglich

Beispiel

$aa^*aa^*b^*bb^*b^*a$ hat die Sequenz-Normalform $a[2, *]b[1, *]a[1, 1]$

Sequenz-Normalform

Definition

Sei r ein regulärer Ausdruck aus RE(a, a^*) und sei $e \in \Sigma$.

- $e[i, j]$ gibt an, dass e mindestens i-mal und höchstens j-mal direkt nacheinander in r vorkommt
- $j = *$ gibt dabei an, dass die Anzahl der Wiederholungen von e unbeschränkt ist
- $e[1, 1]$ steht für e und $e[0, *]$ steht für e^*
- SEQUENZ-NORMALFORM entsteht durch Zusammenfassung von $e[i_1, j_1]$ und $e[i_2, j_2]$ zu $e[i_1 + i_2, j_1 + j_2]$, wann immer möglich

Beispiel

$aa^*aa^*b^*bb^*b^*a$ hat die Sequenz-Normalform $a[2, *]b[1, *]a[1, 1]$

Sequenz-Normalform

Definition

Sei r ein regulärer Ausdruck aus $\mathrm{RE}(a, a^*)$ und sei $e \in \Sigma$.

- $e[i, j]$ gibt an, dass e mindestens i-mal und höchstens j-mal direkt nacheinander in r vorkommt
- $j = *$ gibt dabei an, dass die Anzahl der Wiederholungen von e unbeschränkt ist
- $e[1, 1]$ steht für e und $e[0, *]$ steht für e^*
- SEQUENZ-NORMALFORM entsteht durch Zusammenfassung von $e[i_1, j_1]$ und $e[i_2, j_2]$ zu $e[i_1 + i_2, j_1 + j_2]$, wann immer möglich

Beispiel

$aa^*aa^*b^*bb^*b^*a$ hat die Sequenz-Normalform $a[2, *]b[1, *]a[1, 1]$

Sequenz-Normalform

Definition

Sei r ein regulärer Ausdruck aus $\mathrm{RE}(a, a^*)$ und sei $e \in \Sigma$.

- $e[i, j]$ gibt an, dass e mindestens i-mal und höchstens j-mal direkt nacheinander in r vorkommt
- $j = *$ gibt dabei an, dass die Anzahl der Wiederholungen von e unbeschränkt ist
- $e[1, 1]$ steht für e und $e[0, *]$ steht für e^*
- SEQUENZ-NORMALFORM entsteht durch Zusammenfassung von $e[i_1, j_1]$ und $e[i_2, j_2]$ zu $e[i_1 + i_2, j_1 + j_2]$, wann immer möglich

Beispiel

$aa^*aa^*b^*bb^*b^*a$ hat die Sequenz-Normalform $a[2, *]b[1, *]a[1, 1]$

Sequenz-Normalform

Definition

Sei r ein regulärer Ausdruck aus RE(a, a^*) und sei $e \in \Sigma$.

- $e[i, j]$ gibt an, dass e mindestens i-mal und höchstens j-mal direkt nacheinander in r vorkommt
- $j = *$ gibt dabei an, dass die Anzahl der Wiederholungen von e unbeschränkt ist
- $e[1, 1]$ steht für e und $e[0, *]$ steht für e^*
- SEQUENZ-NORMALFORM entsteht durch Zusammenfassung von $e[i_1, j_1]$ und $e[i_2, j_2]$ zu $e[i_1 + i_2, j_1 + j_2]$, wann immer möglich

Beispiel

$aa^*aa^*b^*bb^*b^*a$ hat die Sequenz-Normalform $a[2, *]b[1, *]a[1, 1]$

Sequenz-Normalform

Definition

Sei r ein regulärer Ausdruck aus $\mathrm{RE}(a, a^*)$ und sei $e \in \Sigma$.

- $e[i, j]$ gibt an, dass e mindestens i-mal und höchstens j-mal direkt nacheinander in r vorkommt
- $j = *$ gibt dabei an, dass die Anzahl der Wiederholungen von e unbeschränkt ist
- $e[1, 1]$ steht für e und $e[0, *]$ steht für e^*
- Sequenz-Normalform entsteht durch Zusammenfassung von $e[i_1, j_1]$ und $e[i_2, j_2]$ zu $e[i_1 + i_2, j_1 + j_2]$, wann immer möglich

Beispiel

$aa^*aa^*b^*bb^*b^*a$ hat die Sequenz-Normalform $a[2, *]b[1, *]a[1, 1]$

Sequenz-Normalform

Problem

In einigen Fällen ist Sequenz-Normalform nicht ausreichend, um Äquivalenz zu bestimmen.

Beispiel

$r_1(a, b, i, l) = a[i, *]b[0, *]a[0, *]b[1, *]a[l, *]$
$r_2(a, b, i, l) = a[i, *]b[1, *]a[0, *]b[0, *]a[l, *]$

Beobachtungen

- r_1 und r_2 sind äquivalent
- die Sequenz-Normalformen sind aber nicht gleich

Sequenz-Normalform

Problem

In einigen Fällen ist Sequenz-Normalform nicht ausreichend, um Äquivalenz zu bestimmen.

Beispiel

$r_1(a, b, i, l) = a[i, *]b[0, *]a[0, *]b[1, *]a[l, *]$
$r_2(a, b, i, l) = a[i, *]b[1, *]a[0, *]b[0, *]a[l, *]$

Beobachtungen

- r_1 und r_2 sind äquivalent
- die Sequenz-Normalformen sind aber nicht gleich

Sequenz-Normalform

Problem

In einigen Fällen ist Sequenz-Normalform nicht ausreichend, um Äquivalenz zu bestimmen.

Beispiel

$r_1(a, b, i, l) = a[i, *]b[0, *]a[0, *]b[1, *]a[l, *]$
$r_2(a, b, i, l) = a[i, *]b[1, *]a[0, *]b[0, *]a[l, *]$

Beobachtungen

- r_1 und r_2 sind äquivalent
- die Sequenz-Normalformen sind aber nicht gleich

Sequenz-Normalform

Problem

In einigen Fällen ist Sequenz-Normalform nicht ausreichend, um Äquivalenz zu bestimmen.

Beispiel

$r_1(a, b, i, l) = a[i, *]b[0, *]a[0, *]b[1, *]a[l, *]$
$r_2(a, b, i, l) = a[i, *]b[1, *]a[0, *]b[0, *]a[l, *]$

Beobachtungen

- r_1 und r_2 sind äquivalent
- die Sequenz-Normalformen sind aber nicht gleich

starke Sequenz-Normalform

Definition (starke Sequenz-Normalform)

Sei r ein regulärer Ausdruck aus RE(a, a^*).

- r' entsteht durch Ersetzen von $r_1(a, b, i, l)$ durch $r_2(a, b, i, l)$ in r (mit $a, b \in \Sigma, i, l \in \mathbb{N}$)
  $r_1(a, b, i, l) = a[i, *]b[0, *]a[0, *]b[1, *]a[l, *]$
  $r_2(a, b, i, l) = a[i, *]b[1, *]a[0, *]b[0, *]a[l, *]$
- STARKE SEQUENZ-NORMALFORM (sSNF) entsteht, wenn diese Ersetzung so oft wie möglich durchgeführt wird

Bemerkung

- das Verfahren terminiert

starke Sequenz-Normalform

Definition (starke Sequenz-Normalform)

Sei r ein regulärer Ausdruck aus $\mathrm{RE}(a, a^*)$.

- r' entsteht durch Ersetzen von $r_1(a, b, i, l)$ durch $r_2(a, b, i, l)$ in r (mit $a, b \in \Sigma, i, l \in \mathbb{N}$)
  $r_1(a, b, i, l) = a[i, *]b[0, *]a[0, *]b[1, *]a[l, *]$
  $r_2(a, b, i, l) = a[i, *]b[1, *]a[0, *]b[0, *]a[l, *]$
- STARKE SEQUENZ-NORMALFORM (sSNF) entsteht, wenn diese Ersetzung so oft wie möglich durchgeführt wird

Bemerkung

- das Verfahren terminiert
- die entstehende Normalform ist unabhängig von der Reihenfolge der Ersetzungen und daher eindeutig

starke Sequenz-Normalform

Definition (starke Sequenz-Normalform)

Sei r ein regulärer Ausdruck aus RE(a, a^*).

- r' entsteht durch Ersetzen von $r_1(a, b, i, l)$ durch $r_2(a, b, i, l)$ in r (mit $a, b \in \Sigma, i, l \in \mathbb{N}$)
  $r_1(a, b, i, l) = a[i, *]b[0, *]a[0, *]b[1, *]a[l, *]$
  $r_2(a, b, i, l) = a[i, *]b[1, *]a[0, *]b[0, *]a[l, *]$
- STARKE SEQUENZ-NORMALFORM (sSNF) entsteht, wenn diese Ersetzung so oft wie möglich durchgeführt wird

Bemerkung

- das Verfahren terminiert
- die entstehende Normalform ist unabhängig von der Reihenfolge der Ersetzungen und daher eindeutig

starke Sequenz-Normalform

Definition (starke Sequenz-Normalform)

Sei r ein regulärer Ausdruck aus RE(a, a^*).

- r' entsteht durch Ersetzen von $r_1(a, b, i, l)$ durch $r_2(a, b, i, l)$ in r (mit $a, b \in \Sigma, i, l \in \mathbb{N}$)
  $r_1(a, b, i, l) = a[i, *]b[0, *]a[0, *]b[1, *]a[l, *]$
  $r_2(a, b, i, l) = a[i, *]b[1, *]a[0, *]b[0, *]a[l, *]$
- STARKE SEQUENZ-NORMALFORM (sSNF) entsteht, wenn diese Ersetzung so oft wie möglich durchgeführt wird

Bemerkung

- das Verfahren terminiert
- die entstehende Normalform ist unabhängig von der Reihenfolge der Ersetzungen und daher eindeutig

starke Sequenz-Normalform

Definition (starke Sequenz-Normalform)

Sei r ein regulärer Ausdruck aus $\mathrm{RE}(a, a^*)$.

- r' entsteht durch Ersetzen von $r_1(a, b, i, l)$ durch $r_2(a, b, i, l)$ in r (mit $a, b \in \Sigma, i, l \in \mathbb{N}$)
  $r_1(a, b, i, l) = a[i, *]b[0, *]a[0, *]b[1, *]a[l, *]$
  $r_2(a, b, i, l) = a[i, *]b[1, *]a[0, *]b[0, *]a[l, *]$
- STARKE SEQUENZ-NORMALFORM (sSNF) entsteht, wenn diese Ersetzung so oft wie möglich durchgeführt wird

Bemerkung

- das Verfahren terminiert
- die entstehende Normalform ist unabhängig von der Reihenfolge der Ersetzungen und daher eindeutig

Notationen

Notationen

- Wenn f ein Ausdruck $e[i,j]$ ist, schreibe $u(f)$ für die obere Grenze j und $l(f)$ für die untere Grenze i.
- Sei $r = r_1 \dots r_n$ ein regulärer Ausdruck in Sequenz-Normalform. Dann ist $\max(r) = \max\{u(r_i) \mid u(r_i) \neq *\}$.
- Sei a ein beliebiges Zeichen und w eine Zeichenkette. Dann heißt eine maximale Teil-Zeichenkette v von w, die die Form a^k hat, ein $\textsc{Block}$ von w.

Notationen

Notationen

- Wenn f ein Ausdruck $e[i,j]$ ist, schreibe $u(f)$ für die obere Grenze j und $l(f)$ für die untere Grenze i.
- Sei $r = r_1 \dots r_n$ ein regulärer Ausdruck in Sequenz-Normalform. Dann ist $\max(r) = \max\{u(r_i) \mid u(r_i) \neq *\}$.
- Sei a ein beliebiges Zeichen und w eine Zeichenkette. Dann heißt eine maximale Teil-Zeichenkette v von w, die die Form a^k hat, ein $\textsc{Block}$ von w.

Notationen

Notationen

- Wenn f ein Ausdruck $e[i,j]$ ist, schreibe $u(f)$ für die obere Grenze j und $l(f)$ für die untere Grenze i.
- Sei $r = r_1 \dots r_n$ ein regulärer Ausdruck in Sequenz-Normalform. Dann ist $\max(r) = \max\{u(r_i) \mid u(r_i) \neq *\}$.
- Sei a ein beliebiges Zeichen und w eine Zeichenkette. Dann heißt eine maximale Teil-Zeichenkette v von w, die die Form a^k hat, ein BLOCK von w.

Notationen

Notationen

- Wenn f ein Ausdruck $e[i, j]$ ist, schreibe $u(f)$ für die obere Grenze j und $l(f)$ für die untere Grenze i.
- Sei $r = r_1 \dots r_n$ ein regulärer Ausdruck in Sequenz-Normalform. Dann ist $\max(r) = \max\{u(r_i) \mid u(r_i) \neq *\}$.
- Sei a ein beliebiges Zeichen und w eine Zeichenkette. Dann heißt eine maximale Teil-Zeichenkette v von w, die die Form a^k hat, ein $\textsc{Block}$ von w.

Beweis

zu zeigen:

zwei reguläre Ausdrücke aus RE(a, a^*) sind äquivalent

$\Leftrightarrow$

sie haben die gleiche starke Sequenz-Normalform

Beweis für $\Leftarrow$

Seien r und s zwei reguläre Ausdrücke aus RE(a, a^*) mit gleicher starker Sequenz-Normalform.

- nach Konstruktion:
  $\mathcal{L}(r) = \mathcal{L}(\text{sSNF}(r))$ für jedes r in RE(a, a^*)
- $\Rightarrow \mathcal{L}(r) = \mathcal{L}(\text{sSNF}(r)) = \mathcal{L}(\text{sSNF}(s)) = \mathcal{L}(s)$
- $\Rightarrow r$ und s sind äquivalent

Beweis

zu zeigen:

zwei reguläre Ausdrücke aus RE(a, a^*) sind äquivalent

$\Leftrightarrow$

sie haben die gleiche starke Sequenz-Normalform

Beweis für $\Leftarrow$

Seien r und s zwei reguläre Ausdrücke aus RE(a, a^*) mit gleicher starker Sequenz-Normalform.

- nach Konstruktion:
  $\mathcal{L}(r) = \mathcal{L}(\text{sSNF}(r))$ für jedes r in RE(a, a^*)
- $\Rightarrow \mathcal{L}(r) = \mathcal{L}(\text{sSNF}(r)) = \mathcal{L}(\text{sSNF}(s)) = \mathcal{L}(s)$
- $\Rightarrow r$ und s sind äquivalent

Beweis

zu zeigen:

zwei reguläre Ausdrücke aus RE(a, a^*) sind äquivalent

$\Leftrightarrow$

sie haben die gleiche starke Sequenz-Normalform

Beweis für $\Leftarrow$

Seien r und s zwei reguläre Ausdrücke aus RE(a, a^*) mit gleicher starker Sequenz-Normalform.

- nach Konstruktion:
  $\mathcal{L}(r) = \mathcal{L}(\text{sSNF}(r))$ für jedes r in RE(a, a^*)
- $\Rightarrow \mathcal{L}(r) = \mathcal{L}(\text{sSNF}(r)) = \mathcal{L}(\text{sSNF}(s)) = \mathcal{L}(s)$
- $\Rightarrow r$ und s sind äquivalent

Beweis

zu zeigen:

zwei reguläre Ausdrücke aus RE(a, a^*) sind äquivalent

$\Leftrightarrow$

sie haben die gleiche starke Sequenz-Normalform

Beweis für $\Leftarrow$

Seien r und s zwei reguläre Ausdrücke aus RE(a, a^*) mit gleicher starker Sequenz-Normalform.

- nach Konstruktion:
  $\mathcal{L}(r) = \mathcal{L}(\text{sSNF}(r))$ für jedes r in RE(a, a^*)
- $\Rightarrow \mathcal{L}(r) = \mathcal{L}(\text{sSNF}(r)) = \mathcal{L}(\text{sSNF}(s)) = \mathcal{L}(s)$
- $\Rightarrow r$ und s sind äquivalent

Beweis

zu zeigen:

zwei reguläre Ausdrücke aus RE(a, a^*) sind äquivalent

$\Leftrightarrow$

sie haben die gleiche starke Sequenz-Normalform

Beweis für $\Leftarrow$

Seien r und s zwei reguläre Ausdrücke aus RE(a, a^*) mit gleicher starker Sequenz-Normalform.

- nach Konstruktion:
  $\mathcal{L}(r) = \mathcal{L}(\text{sSNF}(r))$ für jedes r in RE(a, a^*)
- $\Rightarrow \mathcal{L}(r) = \mathcal{L}(\text{sSNF}(r)) = \mathcal{L}(\text{sSNF}(s)) = \mathcal{L}(s)$
- $\Rightarrow r$ und s sind äquivalent

Beweis

zu zeigen:

zwei reguläre Ausdrücke aus RE(a, a^*) sind äquivalent

$\Leftrightarrow$

sie haben die gleiche starke Sequenz-Normalform

Voraussetzungen für $\Rightarrow$

- Seien $r = r_1 \dots r_n$ und $s = s_1 \dots s_m$ aus RE(a, a^*) zwei äquivalente reguläre Ausdrücke.
- O.B.d.A. seien r und s in starker Sequenz-Normalform (kann in PTIME berechnet werden).
- Sei k um eins größer als jede obere Grenze (z.B. $k = 1 + \max(\max(A), \max(B))$).

Beweis

zu zeigen:

zwei reguläre Ausdrücke aus RE(a, a^*) sind äquivalent

$\Leftrightarrow$

sie haben die gleiche starke Sequenz-Normalform

Voraussetzungen für $\Rightarrow$

- Seien $r = r_1 \dots r_n$ und $s = s_1 \dots s_m$ aus RE(a, a^*) zwei äquivalente reguläre Ausdrücke.
- O.B.d.A. seien r und s in starker Sequenz-Normalform (kann in PTIME berechnet werden).
- Sei k um eins größer als jede obere Grenze $\neq *$ $(k = 1 + \max\{\max(r), \max(s)\})$.

Beweis

zu zeigen:

zwei reguläre Ausdrücke aus RE(a, a^*) sind äquivalent

$\Leftrightarrow$

sie haben die gleiche starke Sequenz-Normalform

Voraussetzungen für $\Rightarrow$

- Seien $r = r_1 \dots r_n$ und $s = s_1 \dots s_m$ aus RE(a, a^*) zwei äquivalente reguläre Ausdrücke.
- O.B.d.A. seien r und s in starker Sequenz-Normalform (kann in PTIME berechnet werden).
- Sei k um eins größer als jede obere Grenze $\neq *$ $(k = 1 + \max\{\max(r), \max(s)\})$.

Beweis

zu zeigen:

zwei reguläre Ausdrücke aus RE(a, a^*) sind äquivalent

$\Leftrightarrow$

sie haben die gleiche starke Sequenz-Normalform

Voraussetzungen für $\Rightarrow$

- Seien $r = r_1 \dots r_n$ und $s = s_1 \dots s_m$ aus RE(a, a^*) zwei äquivalente reguläre Ausdrücke.
- O.B.d.A. seien r und s in starker Sequenz-Normalform (kann in PTIME berechnet werden).
- Sei k um eins größer als jede obere Grenze $\neq *$ $(k = 1 + \max\{\max(r), \max(s)\})$.

Beweis

zu zeigen:

zwei reguläre Ausdrücke aus RE(a, a^*) sind äquivalent

$\Leftrightarrow$

sie haben die gleiche starke Sequenz-Normalform

Voraussetzungen für $\Rightarrow$

- Seien $r = r_1 \dots r_n$ und $s = s_1 \dots s_m$ aus RE(a, a^*) zwei äquivalente reguläre Ausdrücke.
- O.B.d.A. seien r und s in starker Sequenz-Normalform (kann in PTIME berechnet werden).
- Sei k um eins größer als jede obere Grenze $\neq *$ $(k = 1 + \max\{\max(r), \max(s)\})$.

Beweis

Beweis für $e(r_i) = e(s_i)$

- bilde $v = v_1 \dots v_n$ und $w = w_1 \dots w_m$
- $v_i = \begin{cases} e(r_i)^k & , u(r_i) = * \\ e(r_i)^{u(r_i)} & , \text{sonst} \end{cases}$, $w_i = \begin{cases} e(s_i)^k & , u(s_i) = * \\ e(s_i)^{u(s_i)} & , \text{sonst} \end{cases}$
- $v \in r \Rightarrow$ (wegen Äquivalenz-Annahme:) $v \in s$
- v besteht aus n Blöcken $\Rightarrow s$ muss mind. n Faktoren haben
- analog für w
- $\Rightarrow m = n$
- $\Rightarrow e(r_i) = e(s_i)$

Beweis

Beweis für $e(r_i) = e(s_i)$

- bilde $v = v_1 \dots v_n$ und $w = w_1 \dots w_m$
- $v_i = \begin{cases} e(r_i)^k & , u(r_i) = * \\ e(r_i)^{u(r_i)} & , \text{sonst} \end{cases}$, $w_i = \begin{cases} e(s_i)^k & , u(s_i) = * \\ e(s_i)^{u(s_i)} & , \text{sonst} \end{cases}$
- $v \in r \Rightarrow$ (wegen Äquivalenz-Annahme:) $v \in s$
- v besteht aus n Blöcken $\Rightarrow$ s muss mind. n Faktoren haben
- analog für w
- $\Rightarrow m = n$
- $\Rightarrow e(r_i) = e(s_i)$

Beweis

Beweis für $e(r_i) = e(s_i)$

- bilde $v = v_1 \dots v_n$ und $w = w_1 \dots w_m$
- $v_i = \begin{cases} e(r_i)^k & , u(r_i) = * \\ e(r_i)^{u(r_i)} & , \text{sonst} \end{cases}$, $w_i = \begin{cases} e(s_i)^k & , u(s_i) = * \\ e(s_i)^{u(s_i)} & , \text{sonst} \end{cases}$
- $v \in r \Rightarrow$ (wegen Äquivalenz-Annahme:) $v \in s$
- v besteht aus n Blöcken $\Rightarrow$ s muss mind. n Faktoren haben
- analog für w
- $\Rightarrow m = n$
- $\Rightarrow e(r_i) = e(s_i)$

Beweis

Beweis für $e(r_i) = e(s_i)$

- bilde $v = v_1 \dots v_n$ und $w = w_1 \dots w_m$
- $v_i = \begin{cases} e(r_i)^k & , u(r_i) = * \\ e(r_i)^{u(r_i)} & , \text{sonst} \end{cases}$, $w_i = \begin{cases} e(s_i)^k & , u(s_i) = * \\ e(s_i)^{u(s_i)} & , \text{sonst} \end{cases}$
- $v \in r \Rightarrow$ (wegen Äquivalenz-Annahme:) $v \in s$
- v besteht aus n Blöcken $\Rightarrow$ s muss mind. n Faktoren haben
- analog für w
- $\Rightarrow m = n$
- $\Rightarrow e(r_i) = e(s_i)$

Beweis

Beweis für $e(r_i) = e(s_i)$

- bilde $v = v_1 \dots v_n$ und $w = w_1 \dots w_m$
- $v_i = \begin{cases} e(r_i)^k & , u(r_i) = * \\ e(r_i)^{u(r_i)} & , \text{sonst} \end{cases}$, $w_i = \begin{cases} e(s_i)^k & , u(s_i) = * \\ e(s_i)^{u(s_i)} & , \text{sonst} \end{cases}$
- $v \in r \Rightarrow$ (wegen Äquivalenz-Annahme:) $v \in s$
- v besteht aus n Blöcken $\Rightarrow$ s muss mind. n Faktoren haben
- analog für w
- $\Rightarrow m = n$
- $\Rightarrow e(r_i) = e(s_i)$

Beweis

Beweis für $e(r_i) = e(s_i)$

- bilde $v = v_1 \dots v_n$ und $w = w_1 \dots w_m$
- $v_i = \begin{cases} e(r_i)^k & , u(r_i) = * \\ e(r_i)^{u(r_i)} & , \text{sonst} \end{cases}$, $w_i = \begin{cases} e(s_i)^k & , u(s_i) = * \\ e(s_i)^{u(s_i)} & , \text{sonst} \end{cases}$
- $v \in r \Rightarrow$ (wegen Äquivalenz-Annahme:) $v \in s$
- v besteht aus n Blöcken $\Rightarrow$ s muss mind. n Faktoren haben
- analog für w
- $\Rightarrow m = n$
- $\Rightarrow e(r_i) = e(s_i)$

Beweis

Beweis für $e(r_i) = e(s_i)$

- bilde $v = v_1 \dots v_n$ und $w = w_1 \dots w_m$
- $v_i = \begin{cases} e(r_i)^k & , u(r_i) = * \\ e(r_i)^{u(r_i)} & , \text{sonst} \end{cases}$, $w_i = \begin{cases} e(s_i)^k & , u(s_i) = * \\ e(s_i)^{u(s_i)} & , \text{sonst} \end{cases}$
- $v \in r \Rightarrow$ (wegen Äquivalenz-Annahme:) $v \in s$
- v besteht aus n Blöcken $\Rightarrow$ s muss mind. n Faktoren haben
- analog für w
- $\Rightarrow m = n$
- $\Rightarrow e(r_i) = e(s_i)$

Beweis

Beweis für $e(r_i) = e(s_i)$

- bilde $v = v_1 \dots v_n$ und $w = w_1 \dots w_m$
- $v_i = \begin{cases} e(r_i)^k & , u(r_i) = * \\ e(r_i)^{u(r_i)} & , \text{sonst} \end{cases}$, $w_i = \begin{cases} e(s_i)^k & , u(s_i) = * \\ e(s_i)^{u(s_i)} & , \text{sonst} \end{cases}$
- $v \in r \Rightarrow$ (wegen Äquivalenz-Annahme:) $v \in s$
- v besteht aus n Blöcken $\Rightarrow$ s muss mind. n Faktoren haben
- analog für w
- $\Rightarrow m = n$
- $\Rightarrow e(r_i) = e(s_i)$

Beweis

Beweis für $u(r_i) = u(s_i)$

- wenn $u(r_i) = *$, dann muss v_i von einem Faktor mit oberer Grenze $*$ in s gematcht werden
- $\Rightarrow u(s_i) = *$
- analog für $u(s_i) = * \quad \Rightarrow u(r_i) = *$
- $u(r_i) = * \Leftrightarrow u(s_i) = *$
- analog für Grenzen $< *$
- $\Rightarrow u(r_i) = u(s_i)$

Beweis für $l(r_i) = l(s_i)$

- betrachte $v_i = w_1 \ldots w_n$ und $r_i = w'_1 \ldots w'_n$, mit $w'_j = (\ldots)^{?}$ und $w'_j = a(\ldots)$
- $\Rightarrow l(r_i) = l(s_i)$ (Induktionsbeweis)

Beweis

Beweis für $u(r_i) = u(s_i)$

- wenn $u(r_i) = *$, dann muss v_i von einem Faktor mit oberer Grenze $*$ in s gematcht werden
- $\Rightarrow u(s_i) = *$
- analog für $u(s_i) = * \quad \Rightarrow u(r_i) = *$
- $u(r_i) = * \Leftrightarrow u(s_i) = *$
- analog für Grenzen $< *$
- $\Rightarrow u(r_i) = u(s_i)$

Beweis für $l(r_i) = l(s_i)$

- betrachte $v_i = w_1 \ldots w_n$ und $r_i = w_1 \ldots w_n$ bzw. $s_i = w_1 \ldots w_n$ und $w_i = \varepsilon | s_j$
- $\Rightarrow l(r_i) = l(s_i)$ (Induktion über n)

Beweis

Beweis für $u(r_i) = u(s_i)$

- wenn $u(r_i) = *$, dann muss v_i von einem Faktor mit oberer Grenze $*$ in s gematcht werden
- $\Rightarrow u(s_i) = *$
- analog für $u(s_i) = *$ $\quad \Rightarrow u(r_i) = *$
- $u(r_i) = * \Leftrightarrow u(s_i) = *$
- analog für Grenzen $< *$
- $\Rightarrow u(r_i) = u(s_i)$

Beweis für $l(r_i) = l(s_i)$

Beweis

Beweis für $u(r_i) = u(s_i)$

- wenn $u(r_i) = *$, dann muss v_i von einem Faktor mit oberer Grenze $*$ in s gematcht werden
- $\Rightarrow u(s_i) = *$
- analog für $u(s_i) = * \quad \Rightarrow u(r_i) = *$
- $u(r_i) = * \Leftrightarrow u(s_i) = *$
- analog für Grenzen $< *$
- $\Rightarrow u(r_i) = u(s_i)$

Beweis für $l(r_i) = l(s_i)$

- betrachte $v_i = w_1 \ldots w_n$ und $r_i = w_1 \ldots w_n \ldots w_{n+k}$ und $w_i = \varepsilon$ für $i > n$
- $\Rightarrow l(r_i) = l(s_i)$ (Induktion)

Beweis

Beweis für $u(r_i) = u(s_i)$

- wenn $u(r_i) = *$, dann muss v_i von einem Faktor mit oberer Grenze $*$ in s gematcht werden
- $\Rightarrow u(s_i) = *$
- analog für $u(s_i) = * \quad \Rightarrow u(r_i) = *$
- $u(r_i) = * \Leftrightarrow u(s_i) = *$
- analog für Grenzen $< *$
- $\Rightarrow u(r_i) = u(s_i)$

Beweis für $l(r_i) = l(s_i)$

Beweis

Beweis für $u(r_i) = u(s_i)$

- wenn $u(r_i) = *$, dann muss v_i von einem Faktor mit oberer Grenze $*$ in s gematcht werden
- $\Rightarrow u(s_i) = *$
- analog für $u(s_i) = * \quad \Rightarrow u(r_i) = *$
- $u(r_i) = * \Leftrightarrow u(s_i) = *$
- analog für Grenzen $< *$
- $\Rightarrow u(r_i) = u(s_i)$

Beweis für $l(r_i) = l(s_i)$

- betrachte $v_i = w_1 \ldots w_j$ und $s = w'_1 \ldots w'_k$, mit $w'_1 = \text{letter}^{l(s_i)}$ und $w'_j = \text{elsy}^{l(s_i)}$
- $\Rightarrow l(r_i) = l(s_i)$ (für untere Grenzen $= 0$)

Beweis

Beweis für $u(r_i) = u(s_i)$

- wenn $u(r_i) = *$, dann muss v_i von einem Faktor mit oberer Grenze $*$ in s gematcht werden
- $\Rightarrow u(s_i) = *$
- analog für $u(s_i) = * \quad \Rightarrow u(r_i) = *$
- $u(r_i) = * \Leftrightarrow u(s_i) = *$
- analog für Grenzen $< *$
- $\Rightarrow u(r_i) = u(s_i)$

Beweis für $l(r_i) = l(s_i)$

- betrachte $v' = v'_1 \ldots v'_n$ und $w' = w'_1 \ldots w'_m$ mit $v'_i = e(r_i)^{l(r_i)}$ und $w'_i = e(s_i)^{l(s_i)}$
- $\Rightarrow l(r_i) = l(s_i)$ (für untere Grenzen > 0)

Beweis

Beweis für $u(r_i) = u(s_i)$

- wenn $u(r_i) = *$, dann muss v_i von einem Faktor mit oberer Grenze $*$ in s gematcht werden
- $\Rightarrow u(s_i) = *$
- analog für $u(s_i) = * \quad \Rightarrow u(r_i) = *$
- $u(r_i) = * \Leftrightarrow u(s_i) = *$
- analog für Grenzen $< *$
- $\Rightarrow u(r_i) = u(s_i)$

Beweis für $l(r_i) = l(s_i)$

- betrachte $v' = v'_1 \dots v'_n$ und $w' = w'_1 \dots w'_m$ mit $v'_i = e(r_i)^{l(r_i)}$ und $w'_i = e(s_i)^{l(s_i)}$
- $\Rightarrow l(r_i) = l(s_i)$ (für untere Grenzen > 0)

Beweis

Beweis für $u(r_i) = u(s_i)$

- wenn $u(r_i) = *$, dann muss v_i von einem Faktor mit oberer Grenze $*$ in s gematcht werden
- $\Rightarrow u(s_i) = *$
- analog für $u(s_i) = * \quad \Rightarrow u(r_i) = *$
- $u(r_i) = * \Leftrightarrow u(s_i) = *$
- analog für Grenzen $< *$
- $\Rightarrow u(r_i) = u(s_i)$

Beweis für $l(r_i) = l(s_i)$

- betrachte $v' = v'_1 \dots v'_n$ und $w' = w'_1 \dots w'_m$ mit $v'_i = e(r_i)^{l(r_i)}$ und $w'_i = e(s_i)^{l(s_i)}$
- $\Rightarrow l(r_i) = l(s_i)$ (für untere Grenzen > 0)

Beweis

Beweis für $u(r_i) = u(s_i)$

- wenn $u(r_i) = *$, dann muss v_i von einem Faktor mit oberer Grenze $*$ in s gematcht werden
- $\Rightarrow u(s_i) = *$
- analog für $u(s_i) = * \quad \Rightarrow u(r_i) = *$
- $u(r_i) = * \Leftrightarrow u(s_i) = *$
- analog für Grenzen $< *$
- $\Rightarrow u(r_i) = u(s_i)$

Beweis für $l(r_i) = l(s_i)$

- betrachte $v' = v'_1 \dots v'_n$ und $w' = w'_1 \dots w'_m$ mit $v'_i = e(r_i)^{l(r_i)}$ und $w'_i = e(s_i)^{l(s_i)}$
- $\Rightarrow l(r_i) = l(s_i)$ (für untere Grenzen > 0)

Beweis

Gegenbeispiele?

- Versuche Widersprüche zu finden.
- Sei dazu $l(r_i) < l(s_i)$ und i minimal mit dieser Bedingung.

(1) Sei $l(r_i) > 0$ und $v'' := v$.
Ersetze v_i durch $e(r_i)^{l(r_i)}$ in v''. $\Rightarrow s$ matcht v'' nicht $\lightning$

(2) Sei $0 = l(r_i) < l(s_i)$.

Beweis

Gegenbeispiele?

- Versuche Widersprüche zu finden.
- Sei dazu $l(r_i) < l(s_i)$ und i minimal mit dieser Bedingung.

(1) Sei $l(r_i) > 0$ und $v'' := v$.
Ersetze v_i durch $e(r_i)^{l(r_i)}$ in v''. $\Rightarrow s$ matcht v'' nicht $\lightning$

(2) Sei $0 = l(r_i) < l(s_i)$.

Beweis

Gegenbeispiele?

- Versuche Widersprüche zu finden.
- Sei dazu $l(r_i) < l(s_i)$ und i minimal mit dieser Bedingung.

(1) Sei $l(r_i) > 0$ und $v'' := v$.
Ersetze v_i durch $e(r_i)^{l(r_i)}$ in v''. $\Rightarrow s$ matcht v'' nicht $\lightning$

(2) Sei $0 = l(r_i) < l(s_i)$.

(a) Sei $l(s_i) \geq 2$ und $v'' := v$.
Ersetze w_i durch $e(r_i)$ in v''. $\Rightarrow s$ matcht v'' wegen w_i

Beweis

Gegenbeispiele?

- Versuche Widersprüche zu finden.
- Sei dazu $l(r_i) < l(s_i)$ und i minimal mit dieser Bedingung.

(1) Sei $l(r_i) > 0$ und $v'' := v$.
Ersetze v_i durch $e(r_i)^{l(r_i)}$ in v''. $\Rightarrow$ s matcht v'' nicht $\lightning$

(2) Sei $0 = l(r_i) < l(s_i)$.

(a) Sei $l(s_i) \geq 2$ und $w'' := w$.
Ersetze w_i durch $e(r_i)$ in w''. $\Rightarrow$ s matcht w'' nicht $\lightning$

(b) Sei $l(s_i) = 1$.

Beweis

Gegenbeispiele?

- Versuche Widersprüche zu finden.
- Sei dazu $l(r_i) < l(s_i)$ und i minimal mit dieser Bedingung.

(1) Sei $l(r_i) > 0$ und $v'' := v$.
Ersetze v_i durch $e(r_i)^{l(r_i)}$ in v''. $\Rightarrow$ s matcht v'' nicht $\lightning$

(2) Sei $0 = l(r_i) < l(s_i)$.

(a) Sei $l(s_i) \geq 2$ und $w'' := w$.
Ersetze w_i durch $e(r_i)$ in w''. $\Rightarrow$ s matcht w'' nicht $\lightning$

(b) Sei $l(s_i) = 1$.

Beweis

Gegenbeispiele?

- Versuche Widersprüche zu finden.
- Sei dazu $l(r_i) < l(s_i)$ und i minimal mit dieser Bedingung.

(1) Sei $l(r_i) > 0$ und $v'' := v$.
Ersetze v_i durch $e(r_i)^{l(r_i)}$ in v''. $\Rightarrow$ s matcht v'' nicht $\lightning$

(2) Sei $0 = l(r_i) < l(s_i)$.

(a) Sei $l(s_i) \geq 2$ und $w'' := w$.
Ersetze w_i durch $e(r_i)$ in w''. $\Rightarrow$ s matcht w'' nicht $\lightning$

(b) Sei $l(s_i) = 1$.

Beweis

Gegenbeispiele?

- Versuche Widersprüche zu finden.
- Sei dazu $l(r_i) < l(s_i)$ und i minimal mit dieser Bedingung.

(1) Sei $l(r_i) > 0$ und $v'' := v$.
Ersetze v_i durch $e(r_i)^{l(r_i)}$ in v''. $\Rightarrow$ s matcht v'' nicht $\lightning$

(2) Sei $0 = l(r_i) < l(s_i)$.

(a) Sei $l(s_i) \geq 2$ und $w'' := w$.
Ersetze w_i durch $e(r_i)$ in w''. $\Rightarrow$ s matcht w'' nicht $\lightning$

(b) Sei $l(s_i) = 1$.

Beweis

Gegenbeispiele?

(2)(b) Sei $0 = l(r_i) < l(s_i) = 1$.

- Mit $i = 1$ matcht s $v_2 \dots v_n$ nicht, mit $i = n$ nicht $v_1 \dots v_{n-1}$.
- $\Rightarrow 1 < i < n$
- Sei $x = v_1 \dots v_{i-1} v_{i+1} \dots v_n$ (r (und daher auch s) matcht x).
- Fall 1: $e(r_{i-1}) \neq e(r_{i+1})$
- Fall 2: $e(r_{i-1}) = e(r_{i+1})$

Beweis

Gegenbeispiele?

(2)(b) Sei $0 = l(r_i) < l(s_i) = 1$.

- Mit $i = 1$ matcht s $v_2 \dots v_n$ nicht, mit $i = n$ nicht $v_1 \dots v_{n-1}$.
- $\Rightarrow 1 < i < n$
- Sei $x = v_1 \dots v_{i-1} v_{i+1} \dots v_n$ (r (und daher auch s) matcht x).
- Fall 1: $e(r_{i-1}) \neq e(r_{i+1})$
- Fall 2: $e(r_{i-1}) = e(r_{i+1})$

Beweis

Gegenbeispiele?

(2)(b) Sei $0 = l(r_i) < l(s_i) = 1$.

- Mit $i = 1$ matcht s $v_2 \dots v_n$ nicht, mit $i = n$ nicht $v_1 \dots v_{n-1}$.
- $\Rightarrow 1 < i < n$
- Sei $x = v_1 \dots v_{i-1} v_{i+1} \dots v_n$ (r (und daher auch s) matcht x).
- Fall 1: $e(r_{i-1}) \neq e(r_{i+1})$
- Fall 2: $e(r_{i-1}) = e(r_{i+1})$

Beweis

Gegenbeispiele?

(2)(b) Sei $0 = l(r_i) < l(s_i) = 1$.

- Mit $i = 1$ matcht s $v_2 \dots v_n$ nicht, mit $i = n$ nicht $v_1 \dots v_{n-1}$.
- $\Rightarrow 1 < i < n$
- Sei $x = v_1 \dots v_{i-1} v_{i+1} \dots v_n$ (r (und daher auch s) matcht x).
- Fall 1: $e(r_{i-1}) \neq e(r_{i+1})$
  - x hat $n - 1$ Blöcke
  - da $e(s_j) \neq e(s_{j+1})$ und $e(s_j) \neq e(s_{j-1})$, kann somit s maximal j Blöcke (für $j < i - 1$ oder $j \geq i - 1$)
- Fall 2: $e(r_{i-1}) = e(r_{i+1})$

Beweis

Gegenbeispiele?

(2)(b) Sei $0 = l(r_i) < l(s_i) = 1$.

- Mit $i = 1$ matcht s $v_2 \dots v_n$ nicht, mit $i = n$ nicht $v_1 \dots v_{n-1}$.
- $\Rightarrow 1 < i < n$
- Sei $x = v_1 \dots v_{i-1} v_{i+1} \dots v_n$ (r (und daher auch s) matcht x).
- Fall 1: $e(r_{i-1}) \neq e(r_{i+1})$
  - x hat $n-1$ Blöcke
  - da $e(s_i) \neq e(s_{i+1})$ und $e(s_i) \neq e(s_{i-1})$, kann s_i nur v_j matchen (für $j > i+1$ oder $j < i-1$)
  - die $\geq i$ Blöcke vor bzw. hinter v_j müssten von $s_1 \dots s_{i-1}$ oder $s_{i+1} \dots s_n$ gematcht werden $\lightning$
- Fall 2: $e(r_{i-1}) = e(r_{i+1})$

Beweis

Gegenbeispiele?

(2)(b) Sei $0 = l(r_i) < l(s_i) = 1$.

- Mit $i = 1$ matcht s $v_2 \dots v_n$ nicht, mit $i = n$ nicht $v_1 \dots v_{n-1}$.
- $\Rightarrow 1 < i < n$
- Sei $x = v_1 \dots v_{i-1} v_{i+1} \dots v_n$ (r (und daher auch s) matcht x).
- Fall 1: $e(r_{i-1}) \neq e(r_{i+1})$
  - x hat $n - 1$ Blöcke
  - da $e(s_i) \neq e(s_{i+1})$ und $e(s_i) \neq e(s_{i-1})$, kann s_i nur v_j matchen (für $j > i + 1$ oder $j < i - 1$)
  - die $\geq i$ Blöcke vor bzw. hinter v_j müssten von $s_1 \dots s_{i-1}$ oder $s_{i+1} \dots s_n$ gematcht werden $\lightning$
- Fall 2: $e(r_{i-1}) = e(r_{i+1})$

Beweis

Gegenbeispiele?

(2)(b) Sei $0 = l(r_i) < l(s_i) = 1$.

- Mit $i = 1$ matcht s $v_2 \dots v_n$ nicht, mit $i = n$ nicht $v_1 \dots v_{n-1}$.
- $\Rightarrow 1 < i < n$
- Sei $x = v_1 \dots v_{i-1} v_{i+1} \dots v_n$ (r (und daher auch s) matcht x).
- Fall 1: $e(r_{i-1}) \neq e(r_{i+1})$
  - x hat $n - 1$ Blöcke
  - da $e(s_i) \neq e(s_{i+1})$ und $e(s_i) \neq e(s_{i-1})$, kann s_i nur v_j matchen (für $j > i + 1$ oder $j < i - 1$)
  - die $\geq i$ Blöcke vor bzw. hinter v_j müssten von $s_1 \dots s_{i-1}$ oder $s_{i+1} \dots s_n$ gematcht werden $\lightning$
- Fall 2: $e(r_{i-1}) = e(r_{i+1})$

Beweis

Gegenbeispiele?

(2)(b) Sei $0 = l(r_i) < l(s_i) = 1$.

- Mit $i = 1$ matcht s $v_2 \dots v_n$ nicht, mit $i = n$ nicht $v_1 \dots v_{n-1}$.
- $\Rightarrow 1 < i < n$
- Sei $x = v_1 \dots v_{i-1} v_{i+1} \dots v_n$ (r (und daher auch s) matcht x).
- Fall 1: $e(r_{i-1}) \neq e(r_{i+1})$
  - x hat $n - 1$ Blöcke
  - da $e(s_i) \neq e(s_{i+1})$ und $e(s_i) \neq e(s_{i-1})$, kann s_i nur v_j matchen (für $j > i + 1$ oder $j < i - 1$)
  - die $\geq i$ Blöcke vor bzw. hinter v_j müssten von $s_1 \dots s_{i-1}$ oder $s_{i+1} \dots s_n$ gematcht werden $\lightning$
- Fall 2: $e(r_{i-1}) = e(r_{i+1})$
  - Idee: r nicht in starker Sequenz-Normalform, da r $a[i,*]b[0,*]a[0,*]b[1,*]a[i,*]$ enthält $\lightning$

Beweis

Gegenbeispiele?

(2)(b) Sei $0 = l(r_i) < l(s_i) = 1$.

- Mit $i = 1$ matcht s $v_2 \dots v_n$ nicht, mit $i = n$ nicht $v_1 \dots v_{n-1}$.
- $\Rightarrow 1 < i < n$
- Sei $x = v_1 \dots v_{i-1} v_{i+1} \dots v_n$ (r (und daher auch s) matcht x).
- Fall 1: $e(r_{i-1}) \neq e(r_{i+1})$
  - x hat $n - 1$ Blöcke
  - da $e(s_i) \neq e(s_{i+1})$ und $e(s_i) \neq e(s_{i-1})$, kann s_i nur v_j matchen (für $j > i + 1$ oder $j < i - 1$)
  - die $\geq i$ Blöcke vor bzw. hinter v_j müssten von $s_1 \dots s_{i-1}$ oder $s_{i+1} \dots s_n$ gematcht werden $\lightning$
- Fall 2: $e(r_{i-1}) = e(r_{i+1})$
  - Idee: r nicht in starker Sequenz-Normalform, da r $a[i, *]b[0, *]a[0, *]b[1, *]a[l, *]$ enthält $\lightning$

Beweis

Gegenbeispiele?

(2)(b) Sei $0 = l(r_i) < l(s_i) = 1$.

- Mit $i = 1$ matcht s $v_2 \dots v_n$ nicht, mit $i = n$ nicht $v_1 \dots v_{n-1}$.
- $\Rightarrow 1 < i < n$
- Sei $x = v_1 \dots v_{i-1} v_{i+1} \dots v_n$ (r (und daher auch s) matcht x).
- Fall 1: $e(r_{i-1}) \neq e(r_{i+1})$
  - x hat $n - 1$ Blöcke
  - da $e(s_i) \neq e(s_{i+1})$ und $e(s_i) \neq e(s_{i-1})$, kann s_i nur v_j matchen (für $j > i + 1$ oder $j < i - 1$)
  - die $\geq i$ Blöcke vor bzw. hinter v_j müssten von $s_1 \dots s_{i-1}$ oder $s_{i+1} \dots s_n$ gematcht werden $\lightning$
- Fall 2: $e(r_{i-1}) = e(r_{i+1})$
  - Idee: r nicht in starker Sequenz-Normalform, da r $a[i, *]b[0, *]a[0, *]b[1, *]a[l, *]$ enthält $\lightning$

Übersicht

Schnittproblem für reguläre Ausdrücke

Theorem

Für RE(a, a^*) ist das Schnittproblem NP-vollständig.

Beweisidee

- NP-Härte:
  Reduktion auf Erfüllbarkeitsproblem 3-CNF
- NP-Vollständigkeit:
  finde NP-Algorithmus für das Schnittproblem

Schnittproblem für reguläre Ausdrücke

Theorem

Für RE(a, a^*) ist das Schnittproblem NP-vollständig.

Beweisidee

- NP-Härte:
  Reduktion auf Erfüllbarkeitsproblem 3-CNF
- NP-Vollständigkeit:
  finde NP-Algorithmus für das Schnittproblem

Schnittproblem für reguläre Ausdrücke

Theorem

Für RE(a, a^*) ist das Schnittproblem NP-vollständig.

Beweisidee

- NP-Härte:
  Reduktion auf Erfüllbarkeitsproblem 3-CNF
- NP-Vollständigkeit:
  finde NP-Algorithmus für das Schnittproblem

Erfüllbarkeitsproblem 3-CNF

Definition

Das Problem $3\text{-}\mathrm{CNF}$ ist:

Gegeben: Eine aussagenlogische Formel Φ in konjunktiver Normalform mit genau drei Literalen pro Klausel.

Frage: Ist Φ erfüllbar?

Bemerkung

3-CNF ist NP-vollständig.

Erfüllbarkeitsproblem 3-CNF

Definition

Das Problem $3\text{-}\mathrm{CNF}$ ist:

Gegeben: Eine aussagenlogische Formel Φ in konjunktiver Normalform mit genau drei Literalen pro Klausel.

Frage: Ist Φ erfüllbar?

Bemerkung

3-CNF ist NP-vollständig.

Erfüllbarkeitsproblem 3-CNF

Definition

Das Problem $3\text{-}\mathrm{CNF}$ ist:

Gegeben: Eine aussagenlogische Formel Φ in konjunktiver Normalform mit genau drei Literalen pro Klausel.

Frage: Ist Φ erfüllbar?

Bemerkung

3-CNF ist NP-vollständig.

Erfüllbarkeitsproblem 3-CNF

Definition

Das Problem $3\text{-}\mathrm{CNF}$ ist:

Gegeben: Eine aussagenlogische Formel Φ in konjunktiver Normalform mit genau drei Literalen pro Klausel.

Frage: Ist Φ erfüllbar?

Bemerkung

3-CNF ist NP-vollständig.

Beweis

NP-Härte

(in der Online-Version nicht verfügbar)

Beweis

NP-Vollständigkeit

Seien $r_1, \ldots, r_n$ aus $\mathrm{RE}(a, a^*)$.

- rate für jedes r_i und jedes a^* in r_i, ob a^* durch das leere Wort oder durch a^+ interpretiert wird
- die Ausdrucksstärke ändert sich dadurch nicht
- betrachte daher $r'_1, \ldots, r'_n$ aus $\mathrm{RE}(a, a^+)$

Theorem

Für $\mathrm{RE}(a, a^+)$ liegt das Schnittproblem in PTIME.

Beweis

NP-Vollständigkeit

Seien $r_1, \ldots, r_n$ aus $\mathrm{RE}(a, a^*)$.

- rate für jedes r_i und jedes a^* in r_i, ob a^* durch das leere Wort oder durch a^+ interpretiert wird
- die Ausdrucksstärke ändert sich dadurch nicht
- betrachte daher $r'_1, \ldots, r'_n$ aus $\mathrm{RE}(a, a^+)$

Theorem

Für $\mathrm{RE}(a, a^+)$ liegt das Schnittproblem in PTIME.

Beweis

NP-Vollständigkeit

Seien $r_1, \ldots, r_n$ aus $\mathrm{RE}(a, a^*)$.

- rate für jedes r_i und jedes a^* in r_i, ob a^* durch das leere Wort oder durch a^+ interpretiert wird
- die Ausdrucksstärke ändert sich dadurch nicht
- betrachte daher $r'_1, \ldots, r'_n$ aus $\mathrm{RE}(a, a^+)$

Theorem

Für $\mathrm{RE}(a, a^+)$ liegt das Schnittproblem in PTIME.

Beweis

NP-Vollständigkeit

Seien $r_1, \ldots, r_n$ aus $\mathrm{RE}(a, a^*)$.

- rate für jedes r_i und jedes a^* in r_i, ob a^* durch das leere Wort oder durch a^+ interpretiert wird
- die Ausdrucksstärke ändert sich dadurch nicht
- betrachte daher $r'_1, \ldots, r'_n$ aus $\mathrm{RE}(a, a^+)$

Theorem

Für $\mathrm{RE}(a, a^+)$ liegt das Schnittproblem in PTIME.

Beweis

NP-Vollständigkeit

Seien $r_1, \ldots, r_n$ aus RE(a, a^*).

- rate für jedes r_i und jedes a^* in r_i, ob a^* durch das leere Wort oder durch a^+ interpretiert wird
- die Ausdrucksstärke ändert sich dadurch nicht
- betrachte daher $r'_1, \ldots, r'_n$ aus RE(a, a^+)

Theorem

Für RE(a, a^+) liegt das Schnittproblem in PTIME.

Beweis

Theorem

Für RE(a, a^+) liegt das Schnittproblem in PTIME.

Beweis

Seien $r_1, \ldots, r_n$ aus RE(a, a^+) in Sequenz-Normalform.

- Schnitt $r_1 \cap \cdots \cap r_n$ kann nur $\neq \varnothing$ sein, wenn alle r_i die gleiche Anzahl m an Faktoren haben und wenn, für alle $j \leq n$, der j-te Faktor jedes r_i dasselbe Basissymbol a_j hat
- also: $r_i = e_1^i \ldots e_n^i$ mit $e_j^i = a_j[k_j^i : l_j^i]$ für $k_j^i, l_j^i \in \mathbb{N}$ und $k_j^i \geq 1$
- sei $p_j := \max\{k_j^i \mid i \leq n\}$ und $q_j := \min\{l_j^i \mid a_j = a\}$ für alle $j \leq n$
- $\bigcap_{i=1}^{n} L(r_i) \neq \varnothing \Leftrightarrow p_j \leq q_j$ für alle $j \leq n$

Beweis

Theorem

Für RE(a, a^+) liegt das Schnittproblem in PTIME.

Beweis

Seien $r_1, \ldots, r_n$ aus RE(a, a^+) in Sequenz-Normalform.

- Schnitt $r_1 \cap \cdots \cap r_n$ kann nur $\neq \varnothing$ sein, wenn alle r_i die gleiche Anzahl m an Faktoren haben und wenn, für alle $j \leq n$, der j-te Faktor jedes r_i dasselbe Basissymbol a_j hat
- also: $r_i = e_1^i \ldots e_n^i$ mit $e_j^i = a_j[k_j^i, l_j^i]$ für $k_j^i, l_j^i \in \mathbb{N}$ und $k_j^i \geq 1$
- sei $p_j := \max\{k_j^i \mid i \leq n\}$ und $q_j := \min\{l_j^i \mid i \leq n\}$ für alle $j \leq m$
- $r_1 \cap \cdots \cap r_n \neq \varnothing \Leftrightarrow p_j \leq q_j$ für alle $j \leq m$

Beweis

Theorem

Für RE(a, a^+) liegt das Schnittproblem in PTIME.

Beweis

Seien $r_1, \ldots, r_n$ aus RE(a, a^+) in Sequenz-Normalform.

- Schnitt $r_1 \cap \cdots \cap r_n$ kann nur $\neq \varnothing$ sein, wenn alle r_i die gleiche Anzahl m an Faktoren haben und wenn, für alle $j \leq n$, der j-te Faktor jedes r_i dasselbe Basissymbol a_j hat
- also: $r_i = e_1^i \ldots e_n^i$ mit $e_j^i = a_j[k_j^i, l_j^i]$ für $k_j^i, l_j^i \in \mathbb{N}$ und $k_j^i \geq 1$
- sei $p_j := \max\{k_j^i \mid i \leq n\}$ und $q_j := \min\{l_j^i \mid i \leq n\}$ für alle $j \leq m$
- $r_1 \cap \cdots \cap r_n \neq \varnothing \Leftrightarrow p_j \leq q_j$ für alle $j \leq m$

Beweis

Theorem

Für RE(a, a^+) liegt das Schnittproblem in PTIME.

Beweis

Seien $r_1, \ldots, r_n$ aus RE(a, a^+) in Sequenz-Normalform.

- Schnitt $r_1 \cap \cdots \cap r_n$ kann nur $\neq \varnothing$ sein, wenn alle r_i die gleiche Anzahl m an Faktoren haben und wenn, für alle $j \leq n$, der j-te Faktor jedes r_i dasselbe Basissymbol a_j hat
- also: $r_i = e_1^i \ldots e_n^i$ mit $e_j^i = a_j[k_j^i, l_j^i]$ für $k_j^i, l_j^i \in \mathbb{N}$ und $k_j^i \geq 1$
- sei $p_j := \max\{k_j^i \mid i \leq n\}$ und $q_j := \min\{l_j^i \mid i \leq n\}$ für alle $j \leq m$
- $r_1 \cap \cdots \cap r_n \neq \varnothing \Leftrightarrow p_j \leq q_j$ für alle $j \leq m$

Beweis

Theorem

Für RE(a, a^+) liegt das Schnittproblem in PTIME.

Beweis

Seien $r_1, \ldots, r_n$ aus RE(a, a^+) in Sequenz-Normalform.

- Schnitt $r_1 \cap \cdots \cap r_n$ kann nur $\neq \varnothing$ sein, wenn alle r_i die gleiche Anzahl m an Faktoren haben und wenn, für alle $j \leq n$, der j-te Faktor jedes r_i dasselbe Basissymbol a_j hat
- also: $r_i = e_1^i \ldots e_n^i$ mit $e_j^i = a_j[k_j^i, l_j^i]$ für $k_j^i, l_j^i \in \mathbb{N}$ und $k_j^i \geq 1$
- sei $p_j := \max\{k_j^i \mid i \leq n\}$ und $q_j := \min\{l_j^i \mid i \leq n\}$ für alle $j \leq m$
- $r_1 \cap \cdots \cap r_n \neq \varnothing \Leftrightarrow p_j \leq q_j$ für alle $j \leq m$

Beweis

Theorem

Für RE(a, a^+) liegt das Schnittproblem in PTIME.

Beweis

Seien $r_1, \ldots, r_n$ aus RE(a, a^+) in Sequenz-Normalform.

- Schnitt $r_1 \cap \cdots \cap r_n$ kann nur $\neq \varnothing$ sein, wenn alle r_i die gleiche Anzahl m an Faktoren haben und wenn, für alle $j \leq n$, der j-te Faktor jedes r_i dasselbe Basissymbol a_j hat
- also: $r_i = e_1^i \ldots e_n^i$ mit $e_j^i = a_j[k_j^i, l_j^i]$ für $k_j^i, l_j^i \in \mathbb{N}$ und $k_j^i \geq 1$
- sei $p_j := \max\{k_j^i \mid i \leq n\}$ und $q_j := \min\{l_j^i \mid i \leq n\}$ für alle $j \leq m$
- $r_1 \cap \cdots \cap r_n \neq \varnothing \Leftrightarrow p_j \leq q_j$ für alle $j \leq m$

Fragen?