A functional solver for word equations with regular constraints

Maximilian David Eipper

April 12, 2023

Outline

1. Foundations
2. Approach
3. Implementation
4. Benchmarks
5. Results

Word equations

Word equations

$abxb = xby$,
where $a, b \in \Sigma$ are symbols and $x, y \in X$ are variables

Word equations

$abxb = xby$,
where $a, b \in \Sigma$ are symbols and $x, y \in X$ are variables

Is there a mapping $[\![\cdot]\!] : X \to \Sigma^*$, s.t. $ab[\![x]\!]b = [\![x]\!]b[\![y]\!]$?

Word equations

$abxb = xby$,
where $a, b \in \Sigma$ are symbols and $x, y \in X$ are variables

Is there a mapping $[\![\cdot]\!] : X \to \Sigma^*$, s.t. $ab[\![x]\!]b = [\![x]\!]b[\![y]\!]$?
$\Rightarrow$ yes, $[\![x]\!] = a$, $[\![y]\!] = ab$

Word equations

$xa = byx$,
where $a, b \in \Sigma$ are symbols and $x, y \in X$ are variables

Word equations

$xa = byx$,
where $a, b \in \Sigma$ are symbols and $x, y \in X$ are variables

Is there a mapping $[\![\cdot]\!] : X \to \Sigma^*$, s.t. $[\![x]\!]a = b[\![y]\!][\![x]\!]$?

Word equations

$xa = byx$,
where $a, b \in \Sigma$ are symbols and $x, y \in X$ are variables

Is there a mapping $[\![\cdot]\!] : X \to \Sigma^*$, s.t. $[\![x]\!]a = b[\![y]\!][\![x]\!]$?
$\Rightarrow$ no

Notation

Notation

- $\mathcal{E}$

Notation

- $\mathcal{E}$
- $\varphi_{x \to y}$

Notation

- ε
- $\varphi_{x \to y}(axbyx)$

Notation

- ε
- $\varphi_{x \to y}(axbyx) = aybyy$

Notation

- ε
- $\varphi_{x \to y}(axbyx) = aybyy$
- φ_{DEL}

Notation

- ε
- $\varphi_{x \to y}(axbyx) = aybyy$
- $\varphi_{DEL}(abb)$

Notation

- ε
- $\varphi_{x \to y}(axbyx) = aybyy$
- $\varphi_{DEL}(abb) = bb$

Nielsen Transformation

Nielsen Transformation

$a, b \in \Sigma, a \neq b, x, y \in X, x \neq y, \alpha, \beta \in (\Sigma \cup X)^*$

Nielsen Transformation

$a, b \in \Sigma, a \neq b, x, y \in X, x \neq y, \alpha, \beta \in (\Sigma \cup X)^*$

1. $a\alpha = a\beta \to \varphi_{DEL}$

Nielsen Transformation

$a, b \in \Sigma, a \neq b, x, y \in X, x \neq y, \alpha, \beta \in (\Sigma \cup X)^*$

1. $a\alpha = a\beta \rightarrow \varphi_{DEL}$
2. $a\alpha = b\beta \rightarrow$ ✗

Nielsen Transformation

$a, b \in \Sigma, a \neq b, x, y \in X, x \neq y, \alpha, \beta \in (\Sigma \cup X)^*$

1. $a\alpha = a\beta \rightarrow \varphi_{DEL}$
2. $a\alpha = b\beta \rightarrow$ ✗
3. $x\alpha = \beta \rightarrow \varphi_{x \rightarrow \varepsilon}$

Nielsen Transformation

$$a, b \in \Sigma, a \neq b, x, y \in X, x \neq y, \alpha, \beta \in (\Sigma \cup X)^*$$

1. $a\alpha = a\beta \rightarrow \varphi_{DEL}$
2. $a\alpha = b\beta \rightarrow$ ✗
3. $x\alpha = \beta \rightarrow \varphi_{x \rightarrow \varepsilon}$
4. $x\alpha = a\beta \rightarrow \varphi_{x \rightarrow ax'}$

Nielsen Transformation

$a, b \in \Sigma, a \neq b, x, y \in X, x \neq y, \alpha, \beta \in (\Sigma \cup X)^*$

1. $a\alpha = a\beta \rightarrow \varphi_{DEL}$
2. $a\alpha = b\beta \rightarrow$ ✗
3. $x\alpha = \beta \rightarrow \varphi_{x \rightarrow \varepsilon}$
4. $x\alpha = a\beta \rightarrow \varphi_{x \rightarrow ax'}$
5. $x\alpha = x\beta \rightarrow \varphi_{DEL}$

Nielsen Transformation

$a, b \in \Sigma, a \neq b, x, y \in X, x \neq y, \alpha, \beta \in (\Sigma \cup X)^*$

1. $a\alpha = a\beta \rightarrow \varphi_{DEL}$
2. $a\alpha = b\beta \rightarrow$ ✗
3. $x\alpha = \beta \rightarrow \varphi_{x \rightarrow \varepsilon}$
4. $x\alpha = a\beta \rightarrow \varphi_{x \rightarrow ax'}$
5. $x\alpha = x\beta \rightarrow \varphi_{DEL}$
6. $x\alpha = y\beta \rightarrow \varphi_{x \rightarrow yx'}$

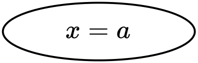
Nielsen Transformation

$a, b \in \Sigma, a \neq b, x, y \in X, x \neq y, \alpha, \beta \in (\Sigma \cup X)^*$

1. $a\alpha = a\beta \rightarrow \varphi_{DEL}$
2. $a\alpha = b\beta \rightarrow$ ✗
3. $x\alpha = \beta \rightarrow \varphi_{x \rightarrow \varepsilon}$
4. $x\alpha = a\beta \rightarrow \varphi_{x \rightarrow ax'}$
5. $x\alpha = x\beta \rightarrow \varphi_{DEL}$
6. $x\alpha = y\beta \rightarrow \varphi_{x \rightarrow yx'}$

Satisfiability: $\alpha = \beta \rightarrow^* \varepsilon = \varepsilon$

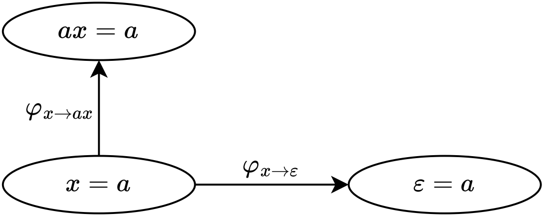
Satisfiable Example $x = a$



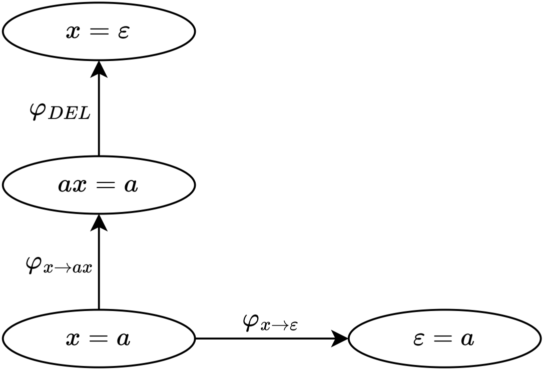
Satisfiable Example $x = a$

$$x = a \xrightarrow{\varphi_{x \to \varepsilon}} \varepsilon = a$$

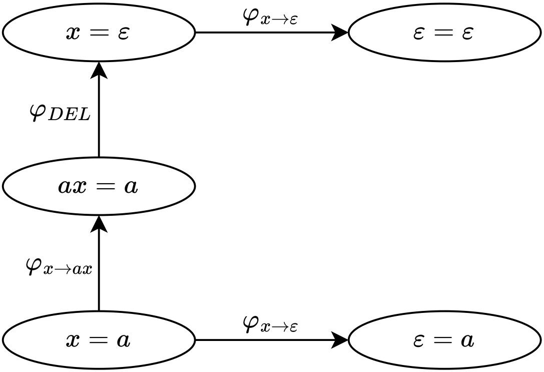
Satisfiable Example $x = a$



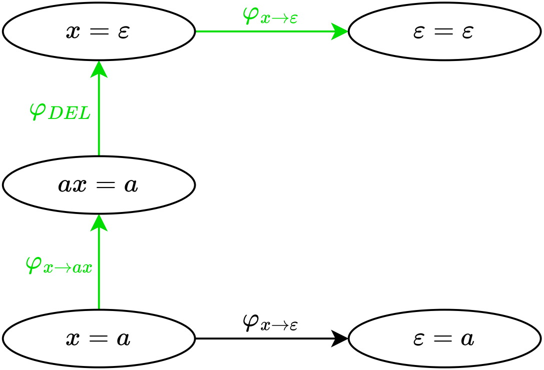
Satisfiable Example $x = a$



Satisfiable Example $x = a$



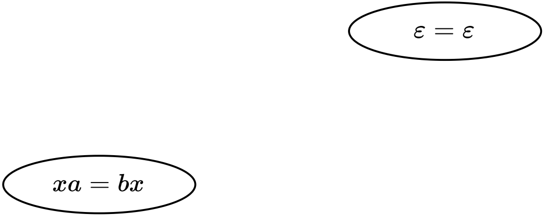
Satisfiable Example $x = a$



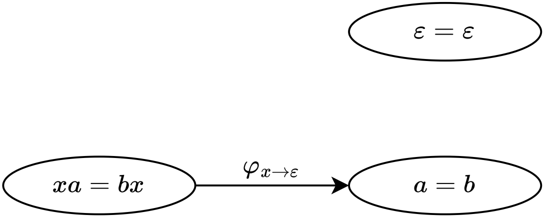
Unsatisfiable Example $xa = bx$

$$xa = bx$$

Unsatisfiable Example $xa = bx$



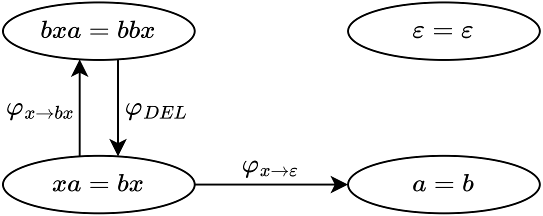
Unsatisfiable Example $xa = bx$



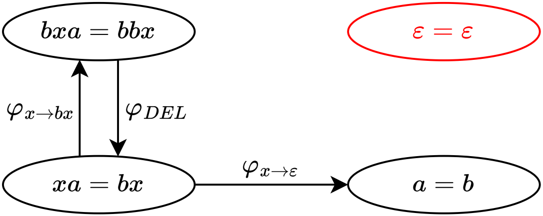
Unsatisfiable Example $xa = bx$

$$
\begin{tikzcd}
bxa = bbx & \varepsilon = \varepsilon \\
xa = bx \arrow[u, "\varphi_{x \to bx}"] \arrow[r, "\varphi_{x \to \varepsilon}"] & a = b
\end{tikzcd}
$$

Unsatisfiable Example $xa = bx$



Unsatisfiable Example $xa = bx$



Regular Expressions

Regular Expressions

$ab*$

Regular Expressions

$ab*, L(ab*) = \{a, ab, abb, ...\}$

Regular Expressions

$ab*, L(ab*) = \{a, ab, abb, ...\}$

- $a, \ L(a) = \{a\}$

Regular Expressions

$ab*, L(ab*) = \{a, ab, abb, ...\}$

- $a,\ L(a) = \{a\}$
- $\lambda,\ L(\lambda) = \{\varepsilon\}$

Regular Expressions

$ab*, L(ab*) = \{a, ab, abb, ...\}$

- a, $L(a) = \{a\}$
- λ, $L(\lambda) = \{\varepsilon\}$
- $\emptyset$, $L(\emptyset) = \{\}$

Regular Expressions

$ab*, L(ab*) = \{a, ab, abb, ...\}$

- $a,\ L(a) = \{a\}$
- $\lambda,\ L(\lambda) = \{\varepsilon\}$
- $\emptyset,\ L(\emptyset) = \{\}$

For regular expressions p, q

Regular Expressions

$ab*, L(ab*) = \{a, ab, abb, ...\}$

- $a,\ L(a) = \{a\}$
- $\lambda,\ L(\lambda) = \{\varepsilon\}$
- $\emptyset,\ L(\emptyset) = \{\}$

For regular expressions p, q

- pq

Regular Expressions

$ab*, L(ab*) = \{a, ab, abb, ...\}$

- $a,\ L(a) = \{a\}$
- $\lambda,\ L(\lambda) = \{\varepsilon\}$
- $\emptyset,\ L(\emptyset) = \{\}$

For regular expressions p, q

- pq
- $p*,\ L(p*) = \{\varepsilon, p, pp, ppp, ...\}$

Regular Expressions

$ab*, L(ab*) = \{a, ab, abb, ...\}$

- $a, \ L(a) = \{a\}$
- $\lambda, \ L(\lambda) = \{\varepsilon\}$
- $\emptyset, \ L(\emptyset) = \{\}$

For regular expressions p, q

- pq
- $p*, \ L(p*) = \{\varepsilon, p, pp, ppp, ...\}$
- $p', p\&q, p|q$

Brzozowski Derivatives

Observation: $a\alpha \in L(ap) \Leftrightarrow \alpha \in L(p)$

Brzozowski Derivatives

Observation: $a\alpha \in L(ap) \Leftrightarrow \alpha \in L(p)$
D

Brzozowski Derivatives

Observation: $a\alpha \in L(ap) \Leftrightarrow \alpha \in L(p)$
D_a

Brzozowski Derivatives

Observation: $a\alpha \in L(ap) \Leftrightarrow \alpha \in L(p)$
$D_a(ap)$

Brzozowski Derivatives

Observation: $a\alpha \in L(ap) \Leftrightarrow \alpha \in L(p)$
$D_a(ap) = p$

Brzozowski Derivatives

Observation: $a\alpha \in L(ap) \Leftrightarrow \alpha \in L(p)$
$D_a(ap) = p$
$D_a q = ?$

Brzozowski Derivatives

Observation: $a\alpha \in L(ap) \Leftrightarrow \alpha \in L(p)$
$D_a(ap) = p$
$D_a q = ?$, s.t. $L(D_a q) = \{\alpha \mid \forall a\alpha \in L(q)\}$

Example

$$ab \qquad \in L(a(b|c))$$

Example

$$\begin{aligned} & ab && \in L(a(b|c)) \\ \Leftrightarrow \quad & b && \in L(D_a(a(b|c))) \end{aligned}$$

Example

$$
\begin{aligned}
& ab && \in L(a(b|c)) \\
\Leftrightarrow\quad & b && \in L(D_a(a(b|c))) = L(b|c)
\end{aligned}
$$

Example

$$
\begin{aligned}
& ab && \in L(a(b|c)) \\
\Leftrightarrow\ & b && \in L(D_a(a(b|c))) = L(b|c) \\
\Leftrightarrow\ & \varepsilon && \in L(D_b(b|c))
\end{aligned}
$$

Example

$$
\begin{aligned}
& ab && \in L(a(b|c)) \\
\Leftrightarrow\quad & b && \in L(D_a(a(b|c))) = L(b|c) \\
\Leftrightarrow\quad & \varepsilon && \in L(D_b(b|c)) = L(\lambda|\emptyset)
\end{aligned}
$$

Example

$$
\begin{aligned}
& ab && \in L(a(b|c)) \\
\Leftrightarrow \quad & b && \in L(D_a(a(b|c))) = L(b|c) \\
\Leftrightarrow \quad & \varepsilon && \in L(D_b(b|c)) = L(\lambda|\emptyset) \\
\Leftrightarrow \quad & \mathit{true} &&
\end{aligned}
$$

Example

$$
\begin{array}{lll}
 & ab & \in L(a(b|c)) \\
\Leftrightarrow & b & \in L(D_a(a(b|c))) = L(b|c) \\
\Leftrightarrow & \varepsilon & \in L(D_b(b|c)) = L(\lambda|\emptyset) \\
\Leftrightarrow & true &
\end{array}
$$

$a_1 a_2 ... a_n \in L(p)$

Example

$$
\begin{aligned}
& ab && \in L(a(b|c)) \\
\Leftrightarrow\quad & b && \in L(D_a(a(b|c))) = L(b|c) \\
\Leftrightarrow\quad & \varepsilon && \in L(D_b(b|c)) = L(\lambda|\emptyset) \\
\Leftrightarrow\quad & true &&
\end{aligned}
$$

$$a_1 a_2 \dots a_n \in L(p) \Leftrightarrow \varepsilon \in L(D_{a_n}(\dots(D_{a_2}(D_{a_1} p))))$$

Regex Nullability

Regex Nullability

We define $\nu(p) :\Leftrightarrow \varepsilon \in L(p)$

Regex Nullability

We define $\nu(p) :\Leftrightarrow \varepsilon \in L(p)$

$$\nu(\lambda) = \mathit{true}$$

Regex Nullability

We define $\nu(p) :\Leftrightarrow \varepsilon \in L(p)$

$$
\begin{aligned}
\nu(\lambda) &= \mathit{true} \\
\nu(\emptyset) &= \mathit{false}
\end{aligned}
$$

Regex Nullability

We define $\nu(p) :\Leftrightarrow \varepsilon \in L(p)$

$$\begin{aligned}
\nu(\lambda) &= \mathit{true}\\
\nu(\emptyset) &= \mathit{false}\\
\nu(a) &= \mathit{false}
\end{aligned}$$

Regex Nullability

We define $\nu(p) :\Leftrightarrow \varepsilon \in L(p)$

$$
\begin{aligned}
\nu(\lambda) &= \mathit{true} \\
\nu(\emptyset) &= \mathit{false} \\
\nu(a) &= \mathit{false} \\
\nu(pq) &= \nu(p) \wedge \nu(q)
\end{aligned}
$$

Regex Nullability

We define $\nu(p) :\Leftrightarrow \varepsilon \in L(p)$

$$
\begin{aligned}
\nu(\lambda) &= \mathit{true} \\
\nu(\emptyset) &= \mathit{false} \\
\nu(a) &= \mathit{false} \\
\nu(pq) &= \nu(p) \wedge \nu(q) \\
\nu(p*) &= \mathit{true}
\end{aligned}
$$

Regex Nullability

We define $\nu(p) :\Leftrightarrow \varepsilon \in L(p)$

$$
\begin{aligned}
\nu(\lambda) &= \mathit{true} \\
\nu(\emptyset) &= \mathit{false} \\
\nu(a) &= \mathit{false} \\
\nu(pq) &= \nu(p) \wedge \nu(q) \\
\nu(p*) &= \mathit{true} \\
&\dots
\end{aligned}
$$

Regex Satisfiability

Regex Satisfiability

We define $\sigma(p) :\Leftrightarrow L(p) \neq \{\}$

Regex Satisfiability

We define $\sigma(p) :\Leftrightarrow L(p) \neq \{\}$

$$\sigma(\emptyset) = \mathit{false}$$

Regex Satisfiability

We define $\sigma(p) :\Leftrightarrow L(p) \neq \{\}$

$$\begin{aligned} \sigma(\emptyset) &= \mathit{false} \\ \sigma(\lambda) &= \mathit{true} \end{aligned}$$

Regex Satisfiability

We define $\sigma(p) :\Leftrightarrow L(p) \neq \{\}$

$$
\begin{aligned}
\sigma(\emptyset) &= \mathit{false} \\
\sigma(\lambda) &= \mathit{true} \\
\sigma(a) &= \mathit{true}
\end{aligned}
$$

Regex Satisfiability

We define $\sigma(p) :\Leftrightarrow L(p) \neq \{\}$

$$\begin{aligned}
\sigma(\emptyset) &= \mathit{false} \\
\sigma(\lambda) &= \mathit{true} \\
\sigma(a) &= \mathit{true} \\
\sigma(pq) &= \sigma(p) \wedge \sigma(q)
\end{aligned}$$

Regex Satisfiability

We define $\sigma(p) :\Leftrightarrow L(p) \neq \{\}$

$$\begin{aligned}
\sigma(\emptyset) &= \mathit{false} \\
\sigma(\lambda) &= \mathit{true} \\
\sigma(a) &= \mathit{true} \\
\sigma(pq) &= \sigma(p) \land \sigma(q) \\
&\dots
\end{aligned}$$

Regex Derivative

We define $D_a p$

Regex Derivative

We define $D_a p$

$$D_a a = \lambda$$

Regex Derivative

We define $D_a p$

$$
\begin{aligned}
D_a a &= \lambda \\
D_a b &= \emptyset
\end{aligned}
$$

Regex Derivative

We define $D_a p$

$$
\begin{aligned}
D_a a &= \lambda \\
D_a b &= \emptyset \\
D_a \lambda &= \emptyset
\end{aligned}
$$

Regex Derivative

We define $D_a p$

$$
\begin{aligned}
D_a a &= \lambda \\
D_a b &= \emptyset \\
D_a \lambda &= \emptyset \\
D_a \emptyset &= \emptyset
\end{aligned}
$$

Regex Derivative

We define $D_a p$

$$
\begin{array}{lcl}
D_a a & = & \lambda \\
D_a b & = & \emptyset \\
D_a \lambda & = & \emptyset \\
D_a \emptyset & = & \emptyset \\
D_a(pq) & = & \begin{cases} (D_a p)q \mid D_a q & \text{if } \nu(p) \\ (D_a p)q & \text{if } \neg\nu(p) \end{cases}
\end{array}
$$

Regex Derivative

We define $D_a p$

$$
\begin{array}{lcl}
D_a a & = & \lambda \\
D_a b & = & \emptyset \\
D_a \lambda & = & \emptyset \\
D_a \emptyset & = & \emptyset \\
D_a(pq) & = & \begin{cases} (D_a p)q \mid D_a q & \text{if } \nu(p) \\ (D_a p)q & \text{if } \neg\nu(p) \end{cases} \\
D_a(p*) & = & (D_a p)p*
\end{array}
$$

Regex Derivative

We define $D_a p$

$$
\begin{array}{lcl}
D_a a & = & \lambda \\
D_a b & = & \emptyset \\
D_a \lambda & = & \emptyset \\
D_a \emptyset & = & \emptyset \\
D_a(pq) & = & \begin{cases} (D_a p)q \mid D_a q & \text{if } \nu(p) \\ (D_a p)q & \text{if } \neg\nu(p) \end{cases} \\
D_a(p*) & = & (D_a p)p* \\
 & \ldots &
\end{array}
$$

Modified Algorithm

Modified Algorithm

$a, b \in \Sigma, a \neq b, x, y \in X, x \neq y, \alpha, \beta \in (\Sigma \cup X)^*$

Modified Algorithm

$a, b \in \Sigma, a \neq b, x, y \in X, x \neq y, \alpha, \beta \in (\Sigma \cup X)^*$, $c(x)$, $c(y)$ regex constraints

Modified Algorithm

$a, b \in \Sigma, a \neq b, x, y \in X, x \neq y, \alpha, \beta \in (\Sigma \cup X)^*$, $c(x), c(y)$ regex constraints

1. $a\alpha = a\beta \rightarrow \varphi_{DEL}$

Modified Algorithm

$a, b \in \Sigma, a \neq b, x, y \in X, x \neq y, \alpha, \beta \in (\Sigma \cup X)^*$, $c(x)$, $c(y)$ regex constraints

1. $a\alpha = a\beta \rightarrow \varphi_{DEL}$
2. $a\alpha = b\beta \rightarrow$ ✗

Modified Algorithm

$a, b \in \Sigma, a \neq b, x, y \in X, x \neq y, \alpha, \beta \in (\Sigma \cup X)^*$, $c(x)$, $c(y)$ regex constraints

1. $a\alpha = a\beta \to \varphi_{DEL}$
2. $a\alpha = b\beta \to$ ✗
3. $x\alpha = \beta \to \varphi_{x \to \varepsilon}$

Modified Algorithm

$a, b \in \Sigma, a \neq b, x, y \in X, x \neq y, \alpha, \beta \in (\Sigma \cup X)^*$, $c(x), c(y)$ regex constraints

1. $a\alpha = a\beta \rightarrow \varphi_{DEL}$
2. $a\alpha = b\beta \rightarrow$ ✗
3. $x\alpha = \beta \rightarrow \varphi_{x \rightarrow \varepsilon}$, if $\nu(c(x))$

Modified Algorithm

$a, b \in \Sigma, a \neq b, x, y \in X, x \neq y, \alpha, \beta \in (\Sigma \cup X)^*$, $c(x)$, $c(y)$ regex constraints

1. $a\alpha = a\beta \rightarrow \varphi_{DEL}$
2. $a\alpha = b\beta \rightarrow$ ✗
3. $x\alpha = \beta \rightarrow \varphi_{x \rightarrow \varepsilon}$, if $\nu(c(x))$
4. $x\alpha = a\beta \rightarrow \varphi_{x \rightarrow ax'}$

Modified Algorithm

$a, b \in \Sigma, a \neq b, x, y \in X, x \neq y, \alpha, \beta \in (\Sigma \cup X)^*$, $c(x), c(y)$ regex constraints

1. $a\alpha = a\beta \rightarrow \varphi_{DEL}$
2. $a\alpha = b\beta \rightarrow$ ✗
3. $x\alpha = \beta \rightarrow \varphi_{x\rightarrow\varepsilon}$, if $\nu(c(x))$
4. $x\alpha = a\beta \rightarrow \varphi_{x\rightarrow ax'}$, if $\sigma(D_a(c(x))$

Modified Algorithm

$a, b \in \Sigma, a \neq b, x, y \in X, x \neq y, \alpha, \beta \in (\Sigma \cup X)^*$, $c(x), c(y)$ regex constraints

1. $a\alpha = a\beta \rightarrow \varphi_{DEL}$
2. $a\alpha = b\beta \rightarrow$ ✗
3. $x\alpha = \beta \rightarrow \varphi_{x \rightarrow \varepsilon}$, if $\nu(c(x))$
4. $x\alpha = a\beta \rightarrow \varphi_{x \rightarrow ax'}$, if $\sigma(D_a(c(x))$, introduce $c(x') := D_a(c(x))$

Modified Algorithm

$a, b \in \Sigma, a \neq b, x, y \in X, x \neq y, \alpha, \beta \in (\Sigma \cup X)^*$, $c(x), c(y)$ regex constraints

1. $a\alpha = a\beta \to \varphi_{DEL}$
2. $a\alpha = b\beta \to$ ✗
3. $x\alpha = \beta \to \varphi_{x \to \varepsilon}$, if $\nu(c(x))$
4. $x\alpha = a\beta \to \varphi_{x \to ax'}$, if $\sigma(D_a(c(x))$, introduce $c(x') := D_a(c(x))$
5. $x\alpha = x\beta \to \varphi_{DEL}$

Modified Algorithm

$a, b \in \Sigma, a \neq b, x, y \in X, x \neq y, \alpha, \beta \in (\Sigma \cup X)^*$, $c(x)$, $c(y)$ regex constraints

1. $a\alpha = a\beta \rightarrow \varphi_{DEL}$
2. $a\alpha = b\beta \rightarrow$ ✗
3. $x\alpha = \beta \rightarrow \varphi_{x \rightarrow \varepsilon}$, if $\nu(c(x))$
4. $x\alpha = a\beta \rightarrow \varphi_{x \rightarrow ax'}$, if $\sigma(D_a(c(x)))$, introduce $c(x') := D_a(c(x))$
5. $x\alpha = x\beta \rightarrow \varphi_{DEL}$, if $\sigma(c(x))$

Modified Algorithm

$a, b \in \Sigma, a \neq b, x, y \in X, x \neq y, \alpha, \beta \in (\Sigma \cup X)^*$, $c(x), c(y)$ regex constraints

1. $a\alpha = a\beta \rightarrow \varphi_{DEL}$
2. $a\alpha = b\beta \rightarrow$ ✗
3. $x\alpha = \beta \rightarrow \varphi_{x \rightarrow \varepsilon}$, if $\nu(c(x))$
4. $x\alpha = a\beta \rightarrow \varphi_{x \rightarrow ax'}$, if $\sigma(D_a(c(x)))$, introduce $c(x') := D_a(c(x))$
5. $x\alpha = x\beta \rightarrow \varphi_{DEL}$, if $\sigma(c(x))$
6. $x\alpha = y\beta$

Modified Algorithm

$a, b \in \Sigma, a \neq b, x, y \in X, x \neq y, \alpha, \beta \in (\Sigma \cup X)^*$, $c(x), c(y)$ regex constraints

1. $a\alpha = a\beta \rightarrow \varphi_{DEL}$
2. $a\alpha = b\beta \rightarrow$ ✗
3. $x\alpha = \beta \rightarrow \varphi_{x\rightarrow\varepsilon}$, if $\nu(c(x))$
4. $x\alpha = a\beta \rightarrow \varphi_{x\rightarrow ax'}$, if $\sigma(D_a(c(x)))$, introduce $c(x') := D_a(c(x))$
5. $x\alpha = x\beta \rightarrow \varphi_{DEL}$, if $\sigma(c(x))$
6. $x\alpha = y\beta$, ???

Case 6 - Prefix fetching

$x\alpha = y\beta$

Case 6 - Prefix fetching

$x\alpha = y\beta$
common prefix $a \in \Sigma$, s.t. $\sigma(D_a(c(x))), \sigma(D_a(c(y)))$

Case 6 - Prefix fetching

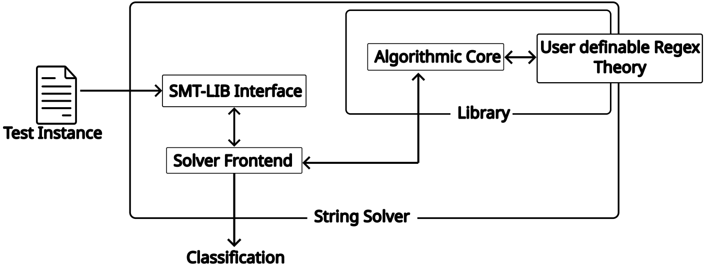
$x\alpha = y\beta$
common prefix $a \in \Sigma$, s.t. $\sigma(D_a(c(x))), \sigma(D_a(c(y)))$
rewrite $\varphi_{y \to ay'}$

Case 6 - Prefix fetching

$x\alpha = y\beta$
common prefix $a \in \Sigma$, s.t. $\sigma(D_a(c(x))), \sigma(D_a(c(y)))$
rewrite $\varphi_{y \to ay'}$, introduce $c(y') := D_a(c(y))$

Implementation

Implementation



Implementation

```
class RegexTheory r where
    derive :: Char -> r -> r
    nullable :: r -> Bool
    satisfiable :: r -> Bool
```

Implementation

```haskell
class RegexTheory r where
    derive :: Char -> r -> r
    nullable :: r -> Bool
    satisfiable :: r -> Bool
```

```haskell
nielsen :: RegexTheory r => Equation r -> Bool
```

Benchmarks

Benchmarks

- benchmarkset-1 (quadratic, regex)

Benchmarks

- benchmarkset-1 (quadratic, regex)
- benchmarkset-2 (quadratic)

Benchmarks

- benchmarkset-1 (quadratic, regex)
- benchmarkset-2 (quadratic)
- benchmarkset-2 $\setminus$ benchmarkset-1 (quadratic, non-regex)

Solvers

Solvers

- CVC4, CVC5

Solvers

- CVC4, CVC5
- Z3Seq, Z3str3

Solvers

- CVC4, CVC5
- Z3Seq, Z3str3
- Woorpje, Woorpje-levis

Solvers

- CVC4, CVC5
- Z3Seq, Z3str3
- Woorpje, Woorpje-levis
- Noodler

Solvers

- CVC4, CVC5
- Z3Seq, Z3str3
- Woorpje, Woorpje-levis
- Noodler
- nielsen-transformation (nt)

Solvers

Solver	Version	State of the art	Can solve regex	Combined solving
CVC4	1.8	✓	✓	✗
CVC5	1.0.3	✓	✓	✗
Z3Seq	4.8.10	✓	✓	✗
Z3str3	4.8.10	✓	✓	✗
woorpje	spin22	✗	✓	✗
woorpje-levi	spin22	✗	✗	✗
noodler	f752e79	✗	✓	✓
nielsen-transformation	0.1.0	✗	✓	✓

Results

benchmarkset-1

Solver	Satis	NSatis	Unknown	Timeout	Errors	Total Time	Total Time w/o Timeout
CVC4	747	1400	0	0	0	240297	240297
CVC5	747	1400	0	0	0	247331	247331
Z3seq	748	1398	0	1	55	865382	845382
Z3str3	748	1398	0	1	55	950974	930974
woorpje	393	643	684	427	126	13319883	4779883
noodler	243	824	637	443	0	27816800	18956584
nielsen-transformation	747	1398	0	2	0	251808	211808

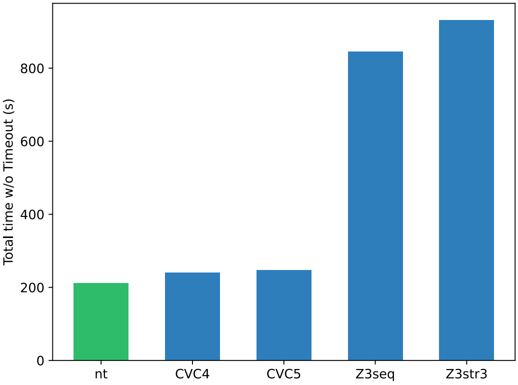
benchmarkset-1

Solver	Satis	NSatis	Unknown	Timeout	Errors	Total Time	Total Time w/o Timeout
CVC4	747	1400	0	0	0	240297	240297
CVC5	747	1400	0	0	0	247331	247331
Z3seq	748	1398	0	1	55	865382	845382
Z3str3	748	1398	0	1	55	950974	930974
woorpje	393	643	684	427	126	13319883	4779883
noodler	243	824	637	443	0	27816800	18956584
nielsen-transformation	747	1398	0	2	0	251808	211808

benchmarkset-1

Solver	Satis	NSatis	Unknown	Timeout	Errors	Total Time	Total Time w/o Timeout
CVC4	747	1400	0	0	0	240297	240297
CVC5	747	1400	0	0	0	247331	247331
Z3seq	748	1398	0	1	55	865382	845382
Z3str3	748	1398	0	1	55	950974	930974
woorpje	393	643	684	427	126	13319883	4779883
noodler	243	824	637	443	0	27816800	18956584
nielsen-transformation	747	1398	0	2	0	251808	211808

benchmarkset-1



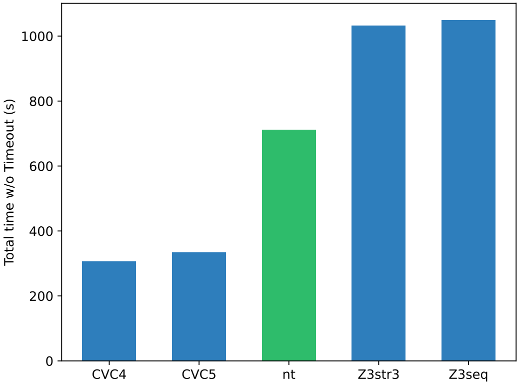
benchmarkset-2

Solver	Satis	NSatis	Unknown	Timeout	Errors	Total Time	Total Time w/o Timeout
CVC4	1055	1491	0	1	0	326543	306543
CVC5	1055	1491	0	1	0	354380	334380
Z3seq	1045	1409	0	93	55	2908422	1048422
Z3str3	1057	1408	5	77	55	2571637	1031637
woorpje	693	651	695	508	127	15410164	5250164
noodler	544	888	665	450	0	32926399	23926181
nielsen-transformation	1011	1453	0	83	0	2371652	711652

benchmarkset-2

Solver	Satis	NSatis	Unknown	Timeout	Errors	Total Time	Total Time w/o Timeout
CVC4	1055	1491	0	1	0	326543	306543
CVC5	1055	1491	0	1	0	354380	334380
Z3seq	1045	1409	0	93	55	2908422	1048422
Z3str3	1057	1408	5	77	55	2571637	1031637
woorpje	693	651	695	508	127	15410164	5250164
noodler	544	888	665	450	0	32926399	23926181
nielsen-transformation	1011	1453	0	83	0	2371652	711652

benchmarkset-2



benchmarkset-2 \ benchmarkset-1

Solver	Satis	NSatis	Unknown	Timeout	Errors	Total Time	Total Time w/o Timeout
CVC4	308	91	0	1	0	90983	70983
CVC5	308	91	0	1	0	110981	90981
Z3seq	297	11	0	92	0	2085124	245124
Z3str3	309	10	5	76	0	1643121	123121
woorpje	299	4	8	89	0	2464010	684010
woorpje-levis	300	90	9	1	1	206817	186817
noodler	281	20	19	80	0	5256067	3656067
nielsen-transformation	264	54	0	82	0	2111959	471959

benchmarkset-2 \ benchmarkset-1

Solver	Satis	NSatis	Unknown	Timeout	Errors	Total Time	Total Time w/o Timeout
CVC4	308	91	0	1	0	90983	70983
CVC5	308	91	0	1	0	110981	90981
Z3seq	297	11	0	92	0	2085124	245124
Z3str3	309	10	5	76	0	1643121	123121
woorpje	299	4	8	89	0	2464010	684010
woorpje-levis	300	90	9	1	1	206817	186817
noodler	281	20	19	80	0	5256067	3656067
nielsen-transformation	264	54	0	82	0	2111959	471959

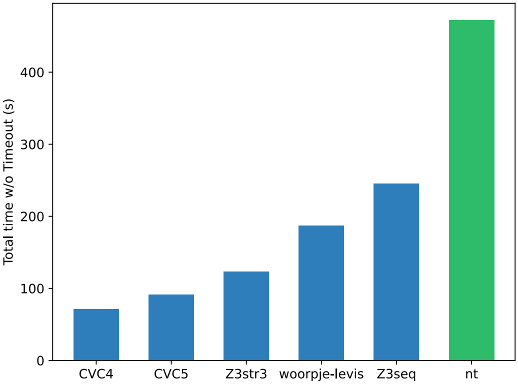
benchmarkset-2 \ benchmarkset-1

Solver	Satis	NSatis	Unknown	Timeout	Errors	Total Time	Total Time w/o Timeout
CVC4	308	91	0	1	0	90983	70983
CVC5	308	91	0	1	0	110981	90981
Z3seq	297	11	0	92	0	2085124	245124
Z3str3	309	10	5	76	0	1643121	123121
woorpje	299	4	8	89	0	2464010	684010
woorpje-levis	300	90	9	1	1	206817	186817
noodler	281	20	19	80	0	5256067	3656067
nielsen-transformation	264	54	0	82	0	2111959	471959

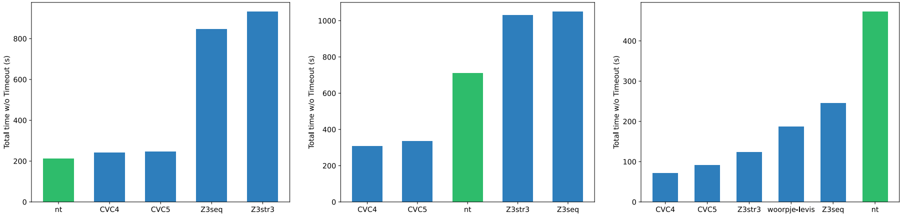
benchmarkset-2 $\setminus$ benchmarkset-1

Solver	Satis	NSatis	Unknown	Timeout	Errors	Total Time	Total Time w/o Timeout
CVC4	308	91	0	1	0	90983	70983
CVC5	308	91	0	1	0	110981	90981
Z3seq	297	11	0	92	0	2085124	245124
Z3str3	309	10	5	76	0	1643121	123121
woorpje	299	4	8	89	0	2464010	684010
woorpje-levis	300	90	9	1	1	206817	186817
noodler	281	20	19	80	0	5256067	3656067
nielsen-transformation	264	54	0	82	0	2111959	471959

benchmarkset-2 \ benchmarkset-1



Summary


Total time w/o Timeout (s)
0
200
400
600
800
nt
CVC4
CVC5
Z3seq
Z3str3
Total time w/o Timeout (s)
0
200
400
600
800
1000
CVC4
CVC5
nt
Z3str3
Z3seq
Total time w/o Timeout (s)
0
100
200
300
400
CVC4
CVC5
Z3str3
woorpje-levis
Z3seq
nt